**Universität Bielefeld**
**Technische Fakultät**

# Development of a Read Mapping Analysis Software and Computational Pan Genome Analysis of 20 *Pseudomonas aeruginosa* Strains

**Zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften an der Technischen Fakultät der Universität Bielefeld vorgelegte Dissertation**

**von**

**Rolf Hilker**

**Gießen, im April 2015**

Rolf Hilker
Hardtallee 29
35398 Gießen
rhilker@cebitec.uni-bielefeld.de
rolf.hilker@mikrobio.med.uni-giessen.de

Supervisors: Prof. Dr. Jens Stoye
Prof. Dr. Alexander Goesmann

Gedruckt auf alterungsbeständigem Papier (ISO 9706).

# Abstract

In times of multi-resistant pathogenic bacteria, their detailed study is of utmost importance. Their comparative analysis can even aid the emerging field of personalized medicine by enabling optimized treatment depending on the presence of virulence factors and antibiotic resistances in the infection concerned. The weaknesses and functionality of these pathogenic bacteria can be investigated using modern computer science and novel sequencing technologies. One of these methods is the bioinformatics evaluation of high-throughput sequencing data.

A pathogenic bacterium posing severe health care issues is the ubiquitous *Pseudomonas aeruginosa*. It is involved in a wide range of infections mainly affecting the pulmonary or urinary tract, open wounds and burns. The prevalence of chronic obstructive pulmonary disease cases with *P. aeruginosa* in Germany alone is ~600,000 per year. Within the framework of this dissertation, computational comparative genomics experiments were conducted with a panel of 20 of the most abundant *Pseudomonas aeruginosa* strains. 15 of these strains were isolated from clinical cases, while the remaining 5 were strains without a known infection history isolated from the environment. This division was chosen to enable direct comparison of the pathogenic potential of clinical and environmental strains and identification of their possible characteristic differences.

When designing the bioinformatics experiments and searching for an efficient visualization and automatic analysis platform for read alignment (mapping) data, it became evident that no adequate solution was available that included all required functionalities. On these grounds, the decision was made to define two main subjects for this dissertation.

Besides the *P. aeruginosa* pan genome analysis, a novel read mapping visualization and analysis software was developed and published in the journal Bioinformatics. This software - ReadXplorer - is partly based upon a prototype, which was developed during a preceding master's thesis at the Center for Biotechnology of the Bielefeld University under the name VAMP. The software was developed into a comprehensive user-friendly platform augmented with several newly developed and implemented automatic bioinformatics read mapping analyses. Two examples of these are the transcription start site detection and the single nucleotide polymorphism detection. Moreover, new intuitive visualizations were added to the existent ones and existing visualizations were greatly enhanced. ReadXplorer is designed to support not only DNA-seq data as accrued in the *P. aeruginosa* experiments, but also any kind of standard read mapping data as obtained from RNA-seq or ChIP-seq experiments. The data management was designed to comply with the latest performance and efficiency needs emerging from the large next generation sequencing data sets. Finally, ReadXplorer was empowered to deal with eukaryotic read mapping data as well.

Amongst other software, ReadXplorer was then used to analyze different comparative genomics aspects of *P. aeruginosa* and to draw conclusions regarding the development of their pathogenicity. The list of conducted experiments includes phylogeny and gene set determination, analysis of regions of genomic plasticity and identification of single nucleotide polymorphisms. The achieved results were published in the journal Environmental Biology.

# Table of Contents

# Table of Figures

# Table of Tables

# Chapter 1

# Introduction

*Pseudomonas aeruginosa* is a ubiquitous, metabolically very versatile gram-negative gammaproteobacterium acting as an opportunistic pathogen. It colonizes plants as well as animals. In the latter, it can be encountered within the skin microbiome and thrives within wounds and in immunocompromised hosts. Such infections can cause life-threatening inflammation and sepsis. (Ramos, 2004-2010a, 2004-2010b)
Additionally, more and more strains acquire resistances against multiple antibiotics, hindering successful healing. Hence, the Infectious Diseases Society of America declared *P. aeruginosa* as one of six "top-priority dangerous, drug-resistant microbes" (Talbot *et al.*, 2006). In the latest threat report from the Centers for Disease Control and Prevention (2013), *P. aeruginosa* is also listed among the serious health concerns. Alone the prevalence of chronic obstructive pulmonary disease (COPD) cases with *P. aeruginosa* in Germany is ~600,000 per year. These facts explicitly show the relevance of researching genetic properties of *P. aeruginosa* to aid identification and development of medical cures.
Present-day microbiology analysis methods allow to closely inspect the building blocks of bacteria like *P. aeruginosa* - their deoxyribonucleic acid (DNA). This is only possible due to the revolutionary discovery of the DNA molecule and its double-helix structure (Watson and Crick, 1953). In this context, the development of modern molecular biology and genome research was enabled by the groundbreaking invention of DNA sequencing (Sanger and Coulson, 1975), the deciphering of the DNA sequence of an organism under investigation. Since then new application areas have evolved and several new sequencing methods have been developed and largely improved. The output of the sequencing machines has been boosted to amounts allowing sequencing of complex eukaryotic genomes like a whole human genome in a single run. Simultaneously, the sequencing costs dropped by several orders of magnitude. Therefore, DNA sequencing is a key technique in molecular biology, evolutionary biology and microbiology. Its applications range from *de novo* whole genome sequencing, genome re-sequencing, transcriptome sequencing, metagenomics and amplicon sequencing to nucleotide modification assays.
The latest developments in the field of DNA sequencing, making whole genome sequencing a routine task, enabled the still young research field of comparative genomics (Bachhawat, 2006). This research field is concerned with differences and communalities of related organisms on the genome level. This relates to the gene content and gene products as well as the genome structure and single nucleotide exchanges, insertions and deletions. Generally, distantly related organisms are commonly compared to identify large scale genome rearrangements, while closely related organisms are often analyzed for smaller, but nonetheless important differences. The reach of comparative genomics goes as far as to aid successful treatment of diseases and personalized medicine.

Meanwhile more than 20,000 genome sequences are available in the National Center for Biotechnology Information (NCBI) genome database for comparative analyses and more and more research projects with the goal of large-scale elucidation of genomic relationships are launched (Grim *et al.*, 2013; Liang *et al.*, 2012).
One of these projects is the *P. aeruginosa* pan genome project discussed in this dissertation. Applying the comparative genomics approach to prevalent innocuous and the most abundant pathogenic *P. aeruginosa* strains, posing the greatest threat among all *P. aeruginosa* strains, will reveal the influence of their genetic blueprint on their level of virulence. Herein, the inclusion of environmental strains allows an unbiased overview of the *P. aeruginosa* pan genome.
To carry out the analyses required for a detailed pan genome study, several bioinformatical tools are needed. Bioinformatics is still a young research area. Thus, not all tasks are already solved optimally and several tasks are just emerging with new findings.
Ready-to-use tools are already available for assembling the investigated genomes and whole genome comparison of their gene sets. For the analysis of small-scale sequence variations and the presence of accessory genomic islands and regions of genomic plasticity though, the manner of analysis can still be refined and simplified by an approach combining automatic analysis with intuitive visualizations. Both tasks can be covered by read mapping data analysis.
Currently, most tools in this field either support visualization or automatic analysis of read mapping data. Additionally, the available analysis tools do not support the full range of analysis functions required for this study. To satisfy the need for an integrative intuitive and precise visualization and automatic analysis tool for read mapping data, a major part of this dissertation deals with the development of a tailored software solution with a broad scope of application. This tool is later applied to analyze the pan genome data. It offers established as well as novel tailored solutions for up-to-date bioinformatics approaches.

## 1.1. Structure of the Dissertation

An expanded overview for the key technologies and approaches mentioned earlier in this chapter is given in Chapter 2. It starts with an introduction to the history of sequencing technologies and continues with downstream analyses. In particular, sequencing data quality, genome assembly, large-scale comparative genomics and read mapping data analyses are covered by this chapter. The state-of-the-art of read mapping visualization tools is closely analyzed in Chapter 3. Chapter 4 presents the motivation and the goals of the dissertation in detail. The two chapters containing the main results are Chapter 5 and Chapter 6. Chapter 5 explains the design, architecture and automatic analysis methods of the here developed read mapping visualization and analysis software ReadXplorer. Chapter 6 presents the pan genome analysis of *P. aeruginosa*. It starts with a detailed introduction to the current stand of the comparative genomics research of *P. aeruginosa* and continues with the establishment of the analysis workflow and the results generated by applying the developed workflow. Finally, Chapter 7 and Chapter 8 conclude the results of both main chapters and give a future perspective on enhancements of ReadXplorer and the comparative analysis of *P. aeruginosa* strains.

# Chapter 2

# NGS and Comparative Genomics

As mentioned in the previous chapter, the development of new high-throughput sequencing technologies in the past 15 years paved the way for the research field of comparative genomics. These sequencing technologies are described in Section 2.1. Section 2.2 is concerned with assessing the virtual DNA sequences ("**reads**") obtained from the sequencing machines. In Section 2.3, the major approaches for assembling genomic sequences from these reads are introduced. The next step on the path to gain insight into the functional parts of the DNA, the automatic genome annotation is described in Section 2.4. Nowadays, not only one, but dozens or more bacteria strains are sequenced in one experiment. These experiments aim at elucidating the evolutionary relationships and the shared genome content. The corresponding field of comparative genomics is introduced in Section 2.5. In Section 2.6, a second approach for handling sequencing reads is explained: mapping of the reads on reference sequences. This is done to enable downstream analyses, explicated in Sections 2.7 and 2.8.

## 2.1. Sequencing Technologies

In this section, the development, functionality and field of application of the different sequencing technologies is elucidated.
Determining the precise nucleotide sequence of an arbitrary DNA molecule is called DNA sequencing. It is a key technique in molecular biology, evolutionary biology and microbiology. On the one hand, it allows molecular biologists to identify functional regions of a previously unknown DNA molecule. Further, knowing the DNA sequence of an organism is inevitable for unravelling the molecular functions hidden in its genome. On the other hand, DNA sequencing can be used for large scale comparisons, e.g. of whole genome sequences or relevant genomic regions, in evolutionary biology and microbiology.
The application range of DNA sequencing includes *de novo* (R. Li *et al.*, 2010) and re-sequencing of whole genomes, ribonucleic acid sequencing (RNA-seq) (Wang *et al.*, 2009), full isoform sequencing (Sharon *et al.*, 2013), amplicon sequencing (Thomas *et al.*, 2006), metagenomics sequencing (Handelsman, 2004), metatranscriptomics sequencing (Simon and Daniel, 2011), TAG sequencing (Porter *et al.*, 2006), large scale chromatin immuno-precipitation sequencing (ChIP-Seq) (Johnson *et al.*, 2007) and nucleotide modification assays including DNA methylation analysis (Bock *et al.*, 2010; Flusberg *et al.*, 2010).
Whole genomes are generally not sequenced in one piece. They are rather fractionized into smaller fragments which serve as input for all sequencing methods. When RNA sequences are analyzed, the RNA is reversely transcribed into complementary DNA (cDNA) prior to sequencing. The process of assigning a nucleotide character to a sequencing signal is called

**base calling**. Not all signals show the same distinctiveness. Therefore, today the quality of a virtual nucleotide base obtained from sequencing is declared as **PHRED** quality score (Ewing and Green, 1998). The formula of the PHRED score $Q$ is $Q = -10 \log_{10} P$, where $P$ is the error probability of the base call. Thus, this formula logarithmically relates the probability of a false base call to the corresponding PHRED score value. For clarification: $Q = 20$ serves as an often used default cut-off value and equates to a 99% accuracy of a correct base call.

The first attempts to sequence DNA have been made in the mid 1970s by (Sanger *et al.*, 1973; Sanger and Coulson, 1975; Maxam and Gilbert, 1977). Directly after its publication, Maxam-Gilbert sequencing became more popular than Sanger sequencing (see Section 2.1.1), since it does not require cloning steps. Maxam-Gilbert sequencing utilizes different chemical treatments of copies of the same DNA template. Each of these chemicals introduces breaks at different points within the template. Size separation of each reaction in an electrophoresis then allows deducting the nucleotide sequence of the DNA template. Due to scalability problems and extensive use of hazardous chemicals, Maxam-Gilbert sequencing lost its market share to Sanger's "Sequencing with chain-terminating inhibitors", which soon became the *de-facto* gold standard for sequencing. Both Frederick Sanger and Walter Gilbert were awarded the Nobel Prize in Chemistry in 1980 for their major contributions to the field of DNA sequencing.

Since the mid 1990s, second generation sequencing technologies were developed. Three technologies reached marketability between 2005 and 2007: *454 Life Sciences*[1] introduced "Pyrosequencing" in 2005 (see Section 2.1.2), *Illumina*[2] introduced "Sequencing by synthesis" in 2006 (see Section 2.1.3) and *Life Technologies*[3] introduced "Sequencing by oligonucleotide ligation and detection" (SOLiD) in 2007 (see Section 2.1.4). These technologies outperformed Moore's law (Moore, 1998) by a drastic decrease of sequencing costs especially in late 2007 and 2008 (see Figure 1). Moore's law mainly states that computer chip performance is expected to double every two years and thus costs of the same task decrease at the same rate.
All three second generation technologies have constantly been improved in terms of output and cost effectiveness and are still present in the field of next generation sequencing (NGS) technologies. Between 2009 and 2011 three additional NGS technologies became commercially available. The first single molecule sequencing technology "Single molecule fluorescent sequencing" by Helicos BioSciences[4] was introduced with the HeliScope system in 2009 (Bowers *et al.*, 2009). It is based on a sequencing by synthesis approach utilizing so-called "virtual terminator" nucleotides without the need for prior amplification of the template DNA. This technology was not successful at the market and Helicos BioSciences filed for chapter 11 bankruptcy in November 2012. Since Helicos sequencing is not available anymore and was not used during this dissertation, further details of this technology are omitted here.
The other two technologies are "Single molecule real-time (SMRT) sequencing" by *Pacific Biosciences*[5] introduced in 2011 (see Section 2.1.6), and "Semiconductor sequencing" by *Life Technologies*[3] introduced in 2011 (see Section 2.1.5).

[1] 454 Life Sciences, a Roche company, Branford, CT, USA
[2] Illumina, Inc., San Diego, CA, USA
[3] Life Technologies Corporation, Carlsbad, CA, USA
[4] Helicos BioSciences Corporation, Cambridge, MA, USA
[5] Pacific Biosciences, Menlo Park, CA, USA

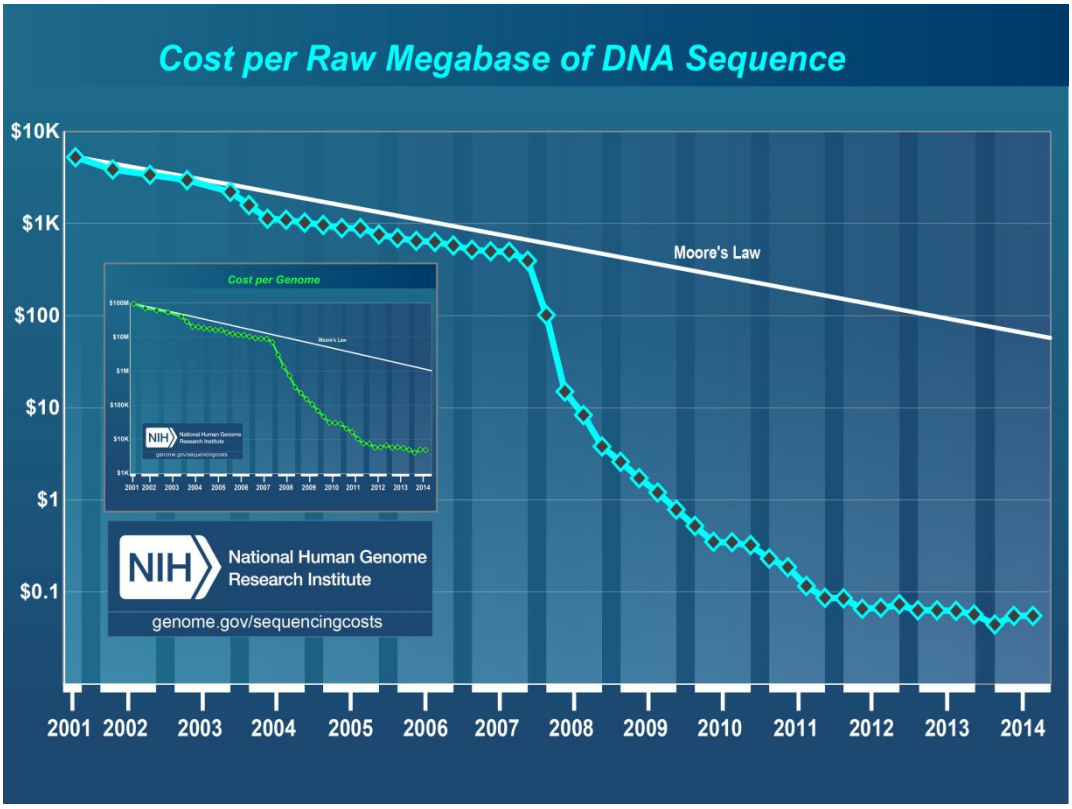



**Figure 1: Development of sequencing costs per raw megabase (mb) of DNA sequence**. Since late 2007 sequencing costs per raw megabase of DNA sequence as well as per genome dropped much faster than suggested by Moore's law (Wetterstrand, 2015). The development of the sequencing costs per human sized genome is shown in the inlet, which is the second figure from (Wetterstrand, 2015).

Another novel sequencing technology is under development by Oxford *Nanopore Technologies*[6] for almost two decades. Nanopore sequencing is based on the eponymous nanopore (Kasianowicz *et al.*, 1996) and a voltage set across the nanopore containing membrane. Molecules, such as nucleotides, passing through the nanopore create a distinctive, measurable change in current. The technology is yet only available to a limited number of selected partners in their "Access Programme" and has not been used during the course of this dissertation. Therefore, it is excluded from further description here.

Another trend that emerged in the last few years is the development of relatively cheap and small "benchtop" sequencers (e.g. Illumina MiSeq and Ion Torrent PGM). Their throughput is much smaller in comparison to their larger relatives, but they can be used in almost every laboratory, despite of the laboratory size, and still enable whole genome sequencing of several bacteria in one run. An overview of the different up-to-date sequencing machines and their costs is given in Table 1.

An important approach common to all available sequencing technologies except SMRT sequencing is the sequencing of pairs of reads with different insert sizes. This technique was first introduced by (Roach *et al.*, 1995). They generated a read pair by sequencing both ends of a clonal DNA template of known length. The distance between both fragment ends is called "**insert size**". Note that some programs alternatively define fragment length minus read length as insert size. In this work, the term **read pair** will be used throughout for all techniques involving pairs of reads. The most common techniques are *paired end* and *mate pair* sequencing.

[6] Oxford Nanopore Technologies, Oxford, OX4 4GA, UK

**Table 1: Comparison of sequencing platforms from (Glenn, 2011) updated in 2014.** The table lists the currently available commercial sequencing technologies including a few upcoming technologies (marked by "fc" = "forecast"). Sequencing run time is less than a day for most technologies and the cheapest technology with the highest throughput is the Illumina HiSeq X. The abbreviation "rr" stands for "rapid run", "ho" denotes "high output", "PCR" denotes "polymerase chain reaction" and "gb" denotes "gigabase". Technologies from different companies are separated by a grey line.

| Instrument | Amplification | Run time | Millions of Reads/run | gb/run | Bases / read | Reagent Cost/run | Reagent Cost/gb |
|---|---|---|---|---|---|---|---|
| **Applied Biosystems** 3730 (capillary) | PCR, cloning | 2 hrs. | 0.000096 | 0.000 | 650 | $144 | $2,307,692.31 |
| **454** | | | | | | | |
| GS Jr. Titanium | emPCR | 10 hrs. | 0.1 | 0.050 | 400 | $977 | $19,540.00 |
| FLX Titanium | emPCR | 10 hrs. | 1 | 0.400 | 400 | $6,200 | $15,500.00 |
| FLX+ | emPCR | 20 hrs. | 1 | 0.650 | 650 | $6,200 | $9,538.46 |
| **Illumina** | | | | | | | |
| GA IIx - v5 SE | BridgePCR | 2 days | 640 | 23.040 | 36 | $4,842 | $210.16 |
| GA IIx - v5 PE | BridgePCR | 14 days | 640 | 184.320 | 288 | $17,978 | $97.54 |
| MiSeq v2 Nano | BridgePCR | 17 hrs. | 1 | 0.300 | 300 | $530 | $1,766.67 |
| MiSeq v2 Nano | BridgePCR | 28 hrs. | 1 | 0.500 | 500 | $639 | $1,278.00 |
| MiSeq v2 Micro | BridgePCR | 19 hrs. | 4 | 1.200 | 300 | $798 | $665.00 |
| MiSeq v2 | BridgePCR | 5 hrs. | 15 | 0.750 | 50 | $747 | $996.00 |
| MiSeq v2 | BridgePCR | 24 hrs. | 15 | 4.500 | 300 | $958 | $212.89 |
| MiSeq v2 | BridgePCR | 39 hrs. | 15 | 7.500 | 500 | $1,066 | $142.13 |
| MiSeq v3 | BridgePCR | 20 hrs. | 22 | 3.300 | 150 | $824 | $249.70 |
| MiSeq v3 | BridgePCR | 55 hrs. | 22 | 13.200 | 600 | $1,442 | $109.24 |
| NextSeq 500 | BridgePCR | 15 hrs. | 130 | 19.500 | 150 | $975 | $50.00 |
| NextSeq 500 | BridgePCR | 26 hrs. | 130 | 39.000 | 300 | $1,560 | $40.00 |
| NextSeq 500 | BridgePCR | 11 hrs. | 400 | 30.000 | 75 | $1,300 | $43.33 |
| NextSeq 500 | BridgePCR | 18 hrs. | 400 | 60.000 | 150 | $2,500 | $41.67 |
| NextSeq 500 | BridgePCR | 30 hrs. | 400 | 120.000 | 300 | $4,000 | $33.33 |
| HiSeq 2500 rr | BridgePCR | 10 hrs. | 300 | 15.000 | 50 | $1,350 | $90.00 |
| HiSeq 2500 rr | BridgePCR | 27 hrs. | 300 | 60.000 | 200 | $3,126 | $52.10 |
| HiSeq 2500 rr | BridgePCR | 40 hrs. | 300 | 90.000 | 300 | $4,126 | $45.84 |
| HiSeq 2500 ho v3 | BridgePCR | 2 days | 1500 | 75.000 | 50 | $5,866 | $78.21 |
| HiSeq 2500 ho v3 | BridgePCR | 11 days | 1500 | 300.000 | 200 | $13,580 | $45.27 |
| HiSeq 2500 ho v4 | BridgePCR | 40 hrs. | 2000 | 100.000 | 50 | $5,866 | $58.66 |
| HiSeq 2500 ho v4 | BridgePCR | 6 days | 2000 | 500.000 | 250 | $14,950 | $29.90 |
| HiSeq X (2 flow cells) | BridgePCR | 3 days | 6000 | 1,800.000 | 300 | $12,750 | $7.08 |
| **Ion Torrent** | | | | | | | |
| PGM 314 chip | emPCR | 2.3 hrs. | 0.475 | 0.095 | 200 | $349 | $3,673.68 |
| PGM 314 chip | emPCR | 3.7 hrs. | 0.475 | 0.190 | 400 | $474 | $2,494.74 |
| PGM 316 chip | emPCR | 3 hrs. | 2.5 | 0.500 | 200 | $549 | $1,098.00 |
| PGM 316 chip | emPCR | 4.9 hrs. | 2.5 | 1.000 | 400 | $674 | $674.00 |
| PGM 318 chip | emPCR | 4.4 hrs. | 4.75 | 0.950 | 200 | $749 | $788.42 |
| PGM 318 chip | emPCR | 7.3 hrs. | 4.75 | 1.900 | 400 | $874 | $460.00 |
| Proton I | emPCR | 4 hrs. | 70 | 12.250 | 175 | $1,000 | $81.63 |
| Proton II (fc) | emPCR | 5 hrs. | 280 | 49.000 | 175 | $1,000 | $20.41 |
| Proton III (fc) | emPCR | 6 hrs. | 500 | 87.500 | 175 | $1,000 | $11.43 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Life Technologies SOLiD** – 5500xl | emPCR | 8 days | 1410 | 155.100 | 110 | $10,503 | $67.72 |
| **Pacific Biosciences** RS II | None - SMS | 2 hrs. | 0.03 | 0.090 | 3000 | $100 | $1,111.11 |
| **Oxford Nanopore** | | | | | | | |
| MinION (fc) | None - SMS | ≤6 hrs. | 0.1 | 0.900 | 9000 | $900 | $1,000.00 |
| GridION 2000 (fc) | None - SMS | varies | 4 | 40.000 | 10000 | $1,500 | $37.50 |
| GridION 8000 (fc) | None - SMS | varies | 10 | 100.000 | 10000 | $1,000 | $10.00 |

The main differences between both techniques are the different fragment size ranges (a few hundred nucleotides for paired end and several kilobases (kb) for mate pair sequencing), different library preparation protocols optimized for the respective fragment length and a different orientation of the paired reads.
Paired end read orientation points inward of the fragment (The first read is from the forward strand and the second from the reverse strand), while mate pair read orientation points outward of the fragment (The first read is from the reverse strand and the second from the forward strand). A single read of a pair is referred to as **mate** throughout the dissertation.

### 2.1.1. Sanger Sequencing

"Sequencing with chain-terminating inhibitors", mostly called Sanger sequencing, was the first widely spread sequencing technique developed by (Sanger and Coulson, 1975).

**Initial Method**
After a first DNA sequencing approach (Sanger *et al.*, 1973) involving a specific primer, radioactively labelled nucleotides and a ribosubstitution technique, Sanger and Coulson developed the more rapid and simpler Sanger sequencing method. This method became the *de-facto* gold standard for sequencing for more than three decades.
The initial Sanger sequencing method uses a single stranded DNA (ssDNA) template, a DNA polymerase, a specific DNA primer, deoxy nucleosidetriphosphates (dNTPs) and radioactively or fluorescently labelled dideoxy NTPs (ddNTPs) to synthesize new DNA fragments of varying length complementary to the given ssDNA strand. The ddNTPs, also called chain-termination inhibitors, are the main component of this method, because the incorporation of a ddNTP terminates the DNA strand elongation due to a lack of the 3'-hydroxyl(-OH) group. This hydroxyl group is essential for establishing a connection between sugar and phosphate of two neighbouring nucleotides. Four different DNA replication reactions are needed, each containing only one of the four ddNTPs.

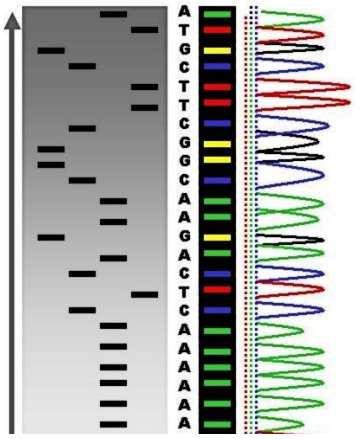



**Figure 2: Sequence ladder and electropherogram peaks.** The same DNA sequence is shown as radioactive sequence ladder on the left, as fluorescent sequence ladder in the middle and as electropherogram peaks on the right. The reading direction of the sequenced fragment is from bottom to top. Source: commons.wikimedia.org/wiki/File:Radioactive_Fluorescent_Seq.jpg, accessed on 05.01.2014

At first, the primer is annealed to the template DNA in each of these reactions. Then the elongation is started by adding the DNA polymerase, all four types of dNTPs and one of the four ddNTPs. The newly synthesized DNA fragments differ in length, since the ddNTPs are only incorporated at each appropriate position by chance, competing with the corresponding dNTP. After several rounds of DNA extension, the resulting DNA fragments are separated by size in a denaturing polyacrylamide-urea gel-electrophoresis with one lane for each reaction. The bands of the DNA fragments are visualized by UV-light or autoradiography and show the sequential arrangement of chain-termination events. Ultimately, the DNA sequence can be composed of the relative positions of all bands among all four lanes by reading them against the electrophoresis direction (Figure 2).

**Enhancement: Automated Dye-Terminator Sequencing**

The initial method was enhanced by using fluorescent dye-labelled ddNTPs and capillary electrophoresis for automation in a DNA sequencer.

The use of fluorescent dye-labelled ddNTPs permits the addition of all four ddNTPs into a single sequencing reaction, since each ddNTP emits light at a different wavelength. The number of necessary reactions is thus reduced from four to one. The gel electrophoresis is replaced by the capillary electrophoresis, which separates the DNA fragments by their electrophoretic mobility. After primer elongation the DNA fragments are placed in a source vial connected to an anode. The negatively charged DNA molecules are then pulled through a capillary into the destination vial by electroosmotic flow of the used electrolyte buffer and separated by fragment size due to their electrophoretic mobility. A detector near the end of the capillary detects the fluorescence of the labelled ddNTPs as a function of time. This result is visualized as an electropherogram (Figure 3).

High throughput capillary sequencing systems currently in use like the *ABI3730xl* can generate 2.1mb of sequence data per day with an average length of 900bp per sequence exceeding the QV20 value (PHRED score of 20, see Section 2.1).
Until the second generation sequencing technologies emerged in 2005, the capillary electrophoresis was utilized for all genome sequencing projects including the Human Genome Project (Lander *et al.*, 2001).

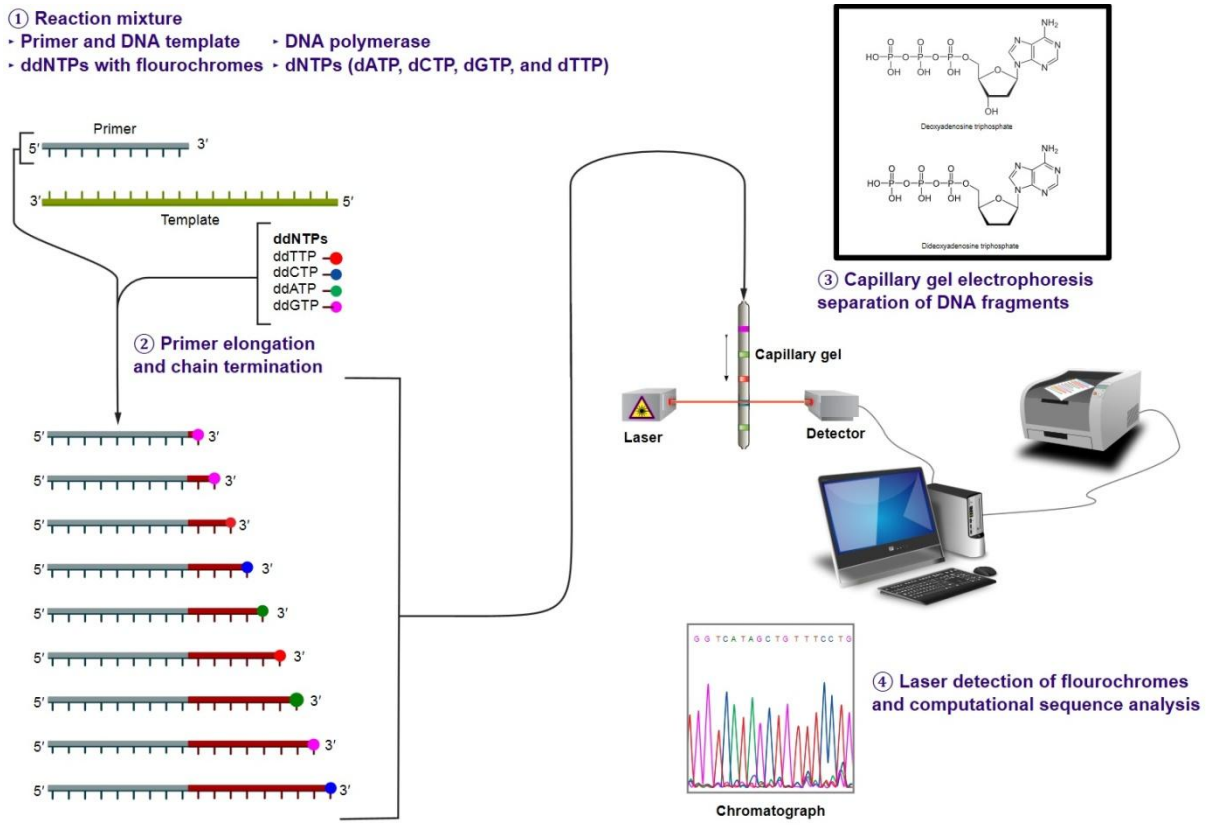



**Figure 3: Automated Sanger sequencing steps**. **(1)** List of the ingredients of the sequencing reaction. **(2)** Visualization of DNA fragments obtained during primer elongation. **(3)** The DNA fragments from (2) are used as input for a capillary electrophoresis, which separates the DNA fragments by electrophoretic mobility. **(4)** After laser induced detection of the used flourochromes, the wavelengths are translated into electropherograms and the DNA bases are called. Source: `commons.wikimedia.org/wiki/File:Sanger-sequencing.svg`, accessed on 05.01.2014

### 2.1.2. Roche 454: Pyrosequencing

In 2005 the first second generation sequencing technique reached marketability: The pyrosequencing method developed by 454 Life Sciences. It is based on a massive parallelization of the real-time sequencing technique developed by Ronaghi *et al.* (1996). The technique is based on emulsion PCR (emPCR) of DNA coated beads and pyrosequencing. Pyrosequencing utilizes a luciferase and sulfurylase reaction for light emission, which was first described in (Nyrén, 1987).

In the first step, the template DNA is fragmented into sequences of appropriate length by nebulization (Sambrook and Russell, 2006). The target length depends on the selected sequencer and chemistry version. Specific adapters are ligated to the ends of the DNA fragments after the ends are blunted by polishing. Two different adapters are used, both consisting of a PCR amplification primer followed by a sequencing primer. One of them contains a 5'biotin tag (Figure 4, A). This tag is needed, to bind the ligated fragments to streptavidin-coated beads, since streptavidin has a very high affinity for binding biotin. Only fragments containing both adapters are kept and the non-biotinylated strand is released to get single stranded template DNA. Optimally, each fragment from the template library is then bound to its own primer-coated magnetic bead using the complementary adapters (Figure 4, B). These beads are encapsulated in droplets of a water-oil emulsion, trapping each bead in its own microreactor. Now the fragments are ready for clonal amplification by an emPCR, which creates millions of clonal copies of the distinct template DNA fragment on each bead (Figure 4, C). After the emPCR, the DNA fragments bound to the bead surface are denatured to single stranded DNA.

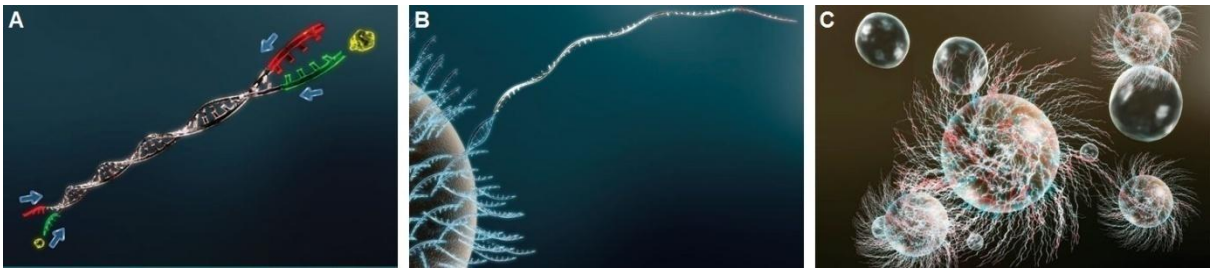



**Figure 4: 454 template preparation. (A)** Two specific adapters, one including a biotin tag, are ligated to both ends of the double stranded DNA templates. **(B)** One DNA fragment is bound to one DNA capture bead utilizing the biotin tag. **(C)** Each bead is covered by millions of clonal copies of the same DNA fragment after the emPCR. Source: www.454.com, accessed on 05.01.2014

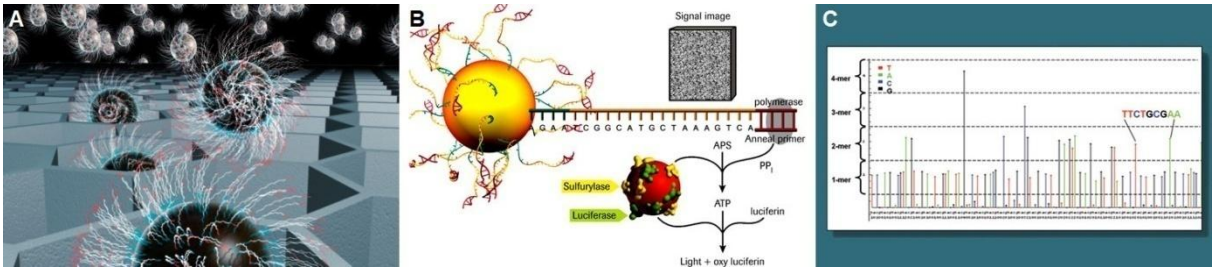



**Figure 5: 454 sequencing. (A)** Deposition of DNA loaded beads onto the PicoTiterPlate (PTP) prior to sequencing. One bead fits into one well, creating a micro reaction chamber. **(B)** dNTPs are flowed across the PTP in cycles. On incorporation of a base, pyrophosphate is released and the sulfurylase-luciferase reaction generates a light signal utilizing luciferin. The signal is optically recorded by a CCD camera. **(C)** The flowgram of a sequencing read shows the intensity of each recorded light signal from one well on the PTP over time. The signals are proportional to the number of incorporated bases. Source: www.454.com, accessed on 05.01.2014

Next, the actual sequencing steps are performed. Only DNA-covered beads are deposited into reaction wells of a so-called PicoTiterPlate (PTP) by centrifugation. The PTP contains millions of reaction wells. Their diameter is 44µm, which is adjusted to the average bead size, which is approximately 28µm. This ensures that only one bead is deposited in each reaction well (Figure 5, A). Before the pyrosequencing reaction starts, small, enzyme carrying beads are added to the reaction wells. They are loaded with the pyrosequencing enzymes luciferase and ATP sulfurylase. Additionally, the nucleotide-degrading apyrase and DNA polymerase are added. Sequencing is then carried out in cycles of fixed order of nucleotide addition. All dNTPs except dATP are added in separate cycles. dATP is not used, because it also is a substrate to luciferase. Instead, a-thio-dATP (dATPαS) is used, which is not a substrate for luciferase, but can be incorporated into a newly synthesized DNA strand. The time-lag between two cycles is chosen such that all remaining dNTPs of the previous cycle are completely degraded by the apyrase. Then, the next dNTP type is added and another sequencing strand elongation is triggered. Upon incorporation of a base, pyrophosphate (PPi) is released stoichiometrically from the incorporated dNTP(s). PPi is quantitatively converted to ATP in the presence of adenosine 5' phosphosulfate by the ATP sulfurylase. A detectable chemiluminescent light signal is then generated by the luciferase (Figure 5, B) which converts luciferin to oxyluciferin in the presence of ATP. The light emission is in amounts that are proportional to the amount of available ATP and detected by a CCD (charged-coupled device) camera. Therefore, the strength of a detected signal is proportional to the number of incorporated nucleotides (Figure 5, C). If more than 8 nucleotides are incorporated in a cycle, the proportionality of the light signal is not given anymore. Due to this fact, 454 pyrosequencing is not suitable for sequencing of homopolymer regions (Margulies *et al.*, 2005). Since each bead is covered with clonal copies of the same DNA fragment, one sequencing read is generated per analyzed bead.

The introduction of pyrosequencing was a groundbreaking step to make the cost of whole genome sequencing affordable for many institutions and research projects. The main advantage of pyrosequencing is that it has the longest read length among all second generation sequencing techniques with currently 600-1000bp, almost reaching the read length of Sanger sequencing. Repetitive regions shorter than pyrosequencing reads can thus be spanned by this technology. Disadvantages of the technique are the lower throughput and higher costs per base in comparison to Illumina (see Section 2.1.3) and SOLiD (see Section 2.1.4) sequencing technologies. Additionally, pyrosequencing had problems with sequencing of high GC-content DNA fragments due to self-annealing issues in the emPCR step until 2011. The problem was overcome by addition of trehalose (Schwientek *et al.*, 2011).

### 2.1.3. Illumina (Solexa): Sequencing by Synthesis

In 2006 the next second generation sequencing technique became available on the market: "Sequencing by synthesis" (SBS) (Bennett, 2004). The technique was initially developed by Solexa since 1998. In early 2007 Solexa was acquired by Illumina. The SBS technique is designed for massively parallel sequencing of short reads and based on the so-called reversible dye-terminator technology applied to DNA clusters on a glass surface. The release of this technique led to a rapid decrease of sequencing costs (see Figure 1).
Most of the sequencing data generated and analyzed for this thesis was generated using the SBS sequencing technique.
In order to use the SBS technology, at first the template DNA has to be fragmented by tagmentation (Adey *et al.*, 2010), nebulization (Sambrook and Russell, 2006) or other suitable approaches. Specific adapters are ligated to both ends of the fragmented DNA (Figure 6, 1). The DNA double strands are then separated and immobilized on the surface of a glass flow cell. The flow cell surface is also coated with primers, which are complementary to the adapter sequences (Figure 6, 2). These primers allow the adapter-ligated fragments to form bridges with the complementary primer. This is the starting point of the so-called bridge amplification, which is divided into three steps:

1. Both sides of each single stranded DNA fragment are fixed on the flow cell surface.
2. The solid-phase PCR is started by addition of unlabeled nucleotides and an engineered DNA-polymerase (Figure 6, 3), synthesizing the second strand of each fragment (Figure 6, 4).
3. The resulting double strands are denatured again (Figure 6, 5) and the next PCR cycle is initiated.

This bridge amplification process creates dense clusters of both strands of the same DNA fragment (Figure 6, 6). Before sequencing, all reverse strand sequences are cleaved and washed away.
The sequencing procedure takes place millions of times in parallel on a flow cell, once for every single DNA template in each cluster, and consists of one cycle for each base in the template. A sequencing cycle consists of the following steps:

1. Primers complementary to the ligated adapters are hybridized to the remaining forward DNA strands, the templates of the sequencing reaction.
2. An engineered DNA polymerase and fluorescently labelled reversible terminator bases (RT-bases) are added (Figure 7, 7). An RT-base is a 3'-O-azidomethyl-2'-deoxynucleoside triphosphate, which acts as terminator of the DNA elongation (Bentley *et al.*, 2008). Each of the four bases is additionally labelled with a different

removable fluorophore, emitting light at a distinct wavelength. The RT-bases guarantee, that the newly synthesized DNA strand of each template in each cluster is only elongated by one base in each sequencing cycle.

3. The fluorophores are excited by a laser and an image is taken by a camera, which reveals the identity of the incorporated nucleotide for each cluster (Figure 7, 8).
4. The fluorescent dye is chemically removed
5. A new 3'-OH is added to the RT-bases to allow for the next base incorporation.

The sequencing cycles are repeated until the predetermined read length is reached and the complete sequence of the bases of each cluster on the flow cell can be uncovered by the sequential images (Figure 7, 9-11).

When the first Illumina sequencer, the *Genome Analyzer*, was released in 2006, the read length was very short with up to 35 bases due to the decreasing base quality in later sequencing cycles. The sequence information received from one flow cell after three days was 1-2gb. During the next years, the enzymes, sequencing chemistry and software for the Illumina sequencers have been improved significantly. In 2013 the Illumina SBS technique is the state of the art sequencing technology for large sequencing projects. Currently, the *HiSeq* and *Genome Analyzer IIx* can generate sequencing reads up to 150bp length, while the benchtop sequencer *MiSeq* is capable of generating 300bp long reads with the latest reagent kit (V3). For production scale sequencing, the *HiSeq X*, only sold within the *HiSeq X Ten* package, has been developed with an output of 1.6 - 1.8tb per dual-flow cell, generating up to 6 billion 150bp reads passing the quality filter in less than 3 days. For very high throughput sequencing the *HiSeq 2500* offers an output of up to 1tb per dual-flow cell after 6 days, while the *NextSeq 500* has a maximum output of up to 120gb per run after 29 hours and the benchtop sequencer *MiSeq* of up to 15gb per flow cell after 65 hours.

Since the beginning in 2006, the most important advantage of the Illumina SBS technique is its versatility based upon the sheer amount of generated sequence data at a comparably low cost (see Table 1), especially in experiments where a detailed resolution is preferred over the read length. The application range of SBS is very broad. Besides the standard applications (see Section 2.1), it also enables DNA methylation assays (Bock *et al.*, 2010).
Disadvantages of the technique are problems with the assembly of repetitive regions which are longer than single reads and the need of powerful and automated data storage and management systems to handle the huge amount of generated sequence data.

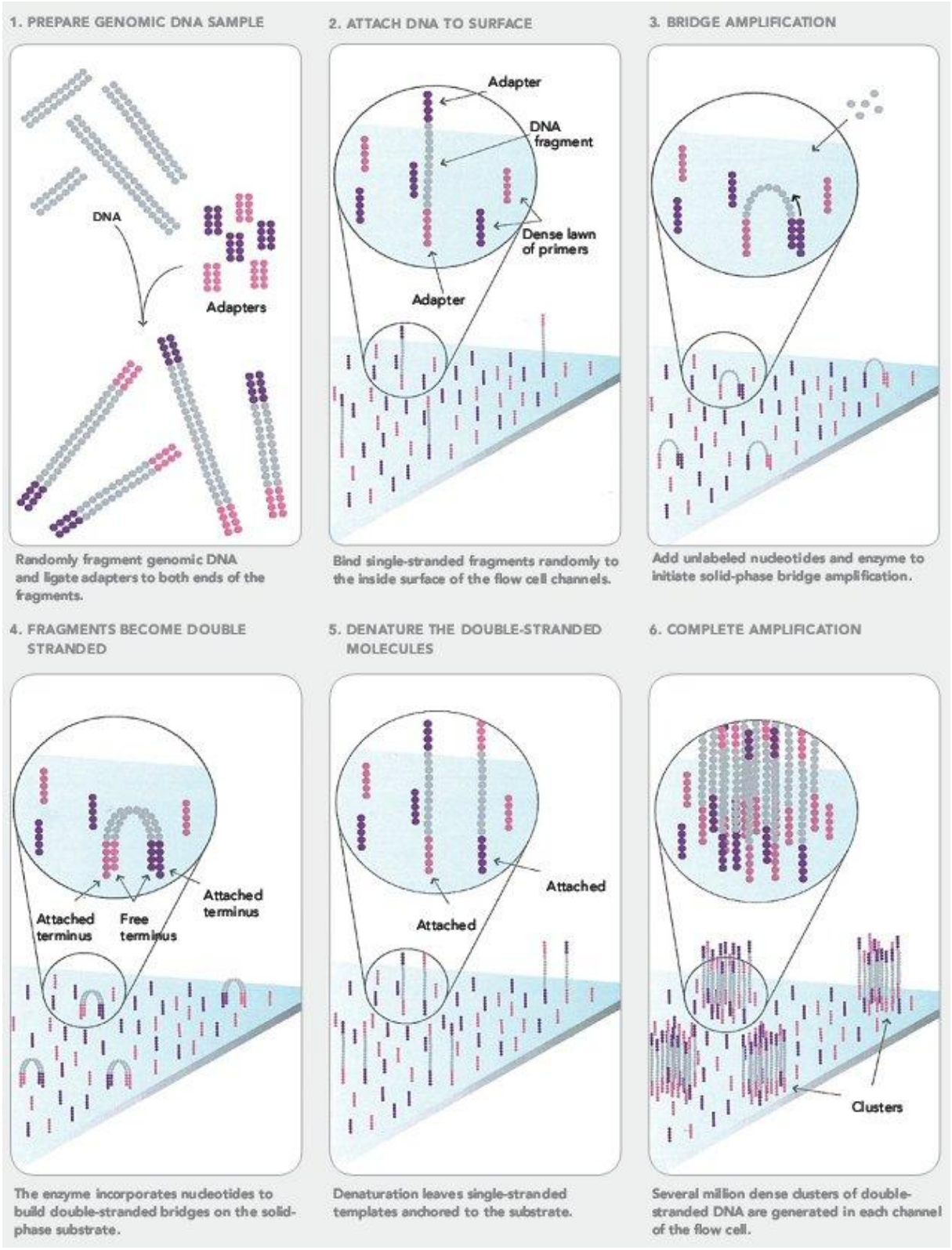



**Figure 6: Sequencing by synthesis steps 1-6.** The figures describe the sample preparation and bridge amplification steps used in SBS. Source: `seqanswers.com/forums/showthread.php?t=21`, accessed on 05.01.2014

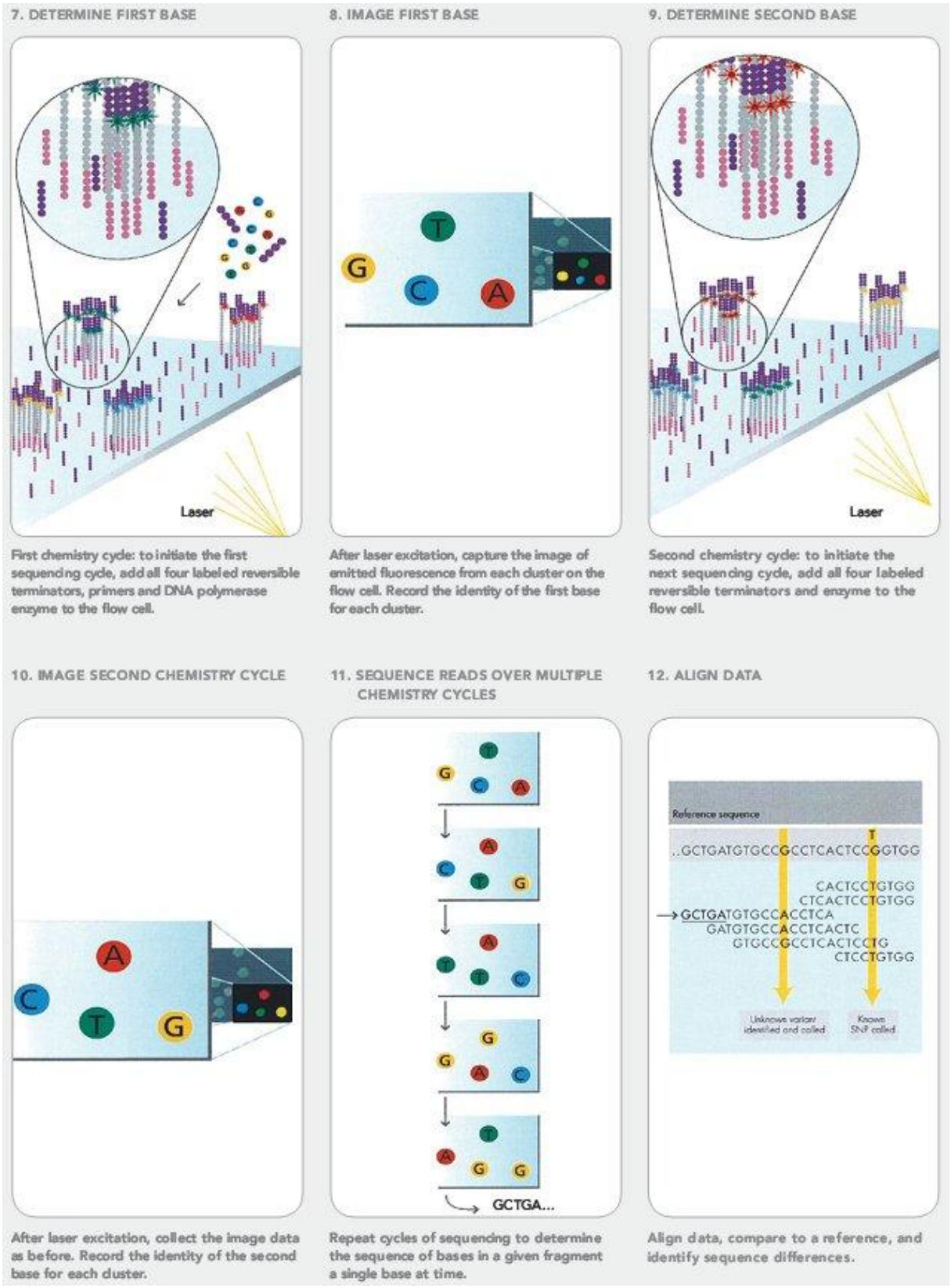



**Figure 7: Sequencing by synthesis steps 7-12.** The figures describe the Sequencing procedure of the SBS technique in detail. Source: `seqanswers.com/forums/showthread.php?t=21`, accessed on 05.01.2014.

### 2.1.4. SOLiD Sequencing

Sequencing by Oligonucleotide Ligation and Detection (SOLiD) (Pandey *et al.*, 2008) is the second short read NGS technique, which became commercially available approximately one year after SBS (see Section 2.1.3) in late 2007 (Liu *et al.*, 2012). It is marketed by Life Technologies. In contrast to SBS, SOLiD is based on a sequencing by ligation approach involving fluorescently labelled di-base probes. Other distinct features of this technique are the two base encoding stored in colour space and the twofold sequencing of each template base, yielding a high sequence accuracy of 99.94%.

As in all other techniques, at first the template DNA has to be fragmented. For SOLiD DNA shearing is recommended. At both ends of the fragmented template DNA a specific adapter is ligated. For mate-pairs a third adapter connects the first mate with the second to a single template. Next, an amplification step is carried out to amplify the template DNA fragments into larger clonal colonies. SOLiD first relied on the emPCR technique already mentioned in Section 2.1.2., but today the newest machine, the *5500 W Series Genetic Analyzer*, adopts a more convenient isothermal PCR technique: The FlowChip itself is covered by a dense lawn of identical primers P1 and the template DNA is randomly immobilized on the FlowChip by hybridization to them. By utilizing a DNA polymerase with strand displacement and proof-reading activity, a so called "template walking" is achieved: After elongation of the template strand, the original second strand is displaced to a surface-bound primer nearby. A complementary primer then allows finishing the second copy of the original template. These amplification steps are repeated until sufficient template DNA colonies have formed on the FlowChip.

For ligation sequencing, ligase, a primer complementary to P1, and fluorescently labelled di-base 8-mer probes are used (see Figure 8, a). Only the first two bases of the probes need to be complementary to the template in order to be ligated. Bases three to five are arbitrary, but also bind the template as degenerated bases. They are followed by a site for cleaving bases six to eight, which carry one of four possible fluorescent dyes. The four dyes representing all sixteen possible di-base pairings are chosen such that they can be decoded unambiguously (see Figure 8, b). In each sequencing round all probes compete for ligation. After incorporation of a suitable probe, the fluorescence is measured, the last three bases of the probe including the fluorophore are cleaved off, and unextended strands are capped. Thus, when starting sequencing at base n, bases n and n+1 have been uncovered in the current sequencing round. Since the degenerated three bases in between are not known yet, the next round reveals bases n+5 and n+6. Therefore, a primer reset is conducted after complete elongation of the ligated strand. In total four rounds of primer reset are performed for five different primers starting at five neighbouring positions (see Figure 8, a). This procedure ensures that in the end each base is covered and actually sequenced twice independently.

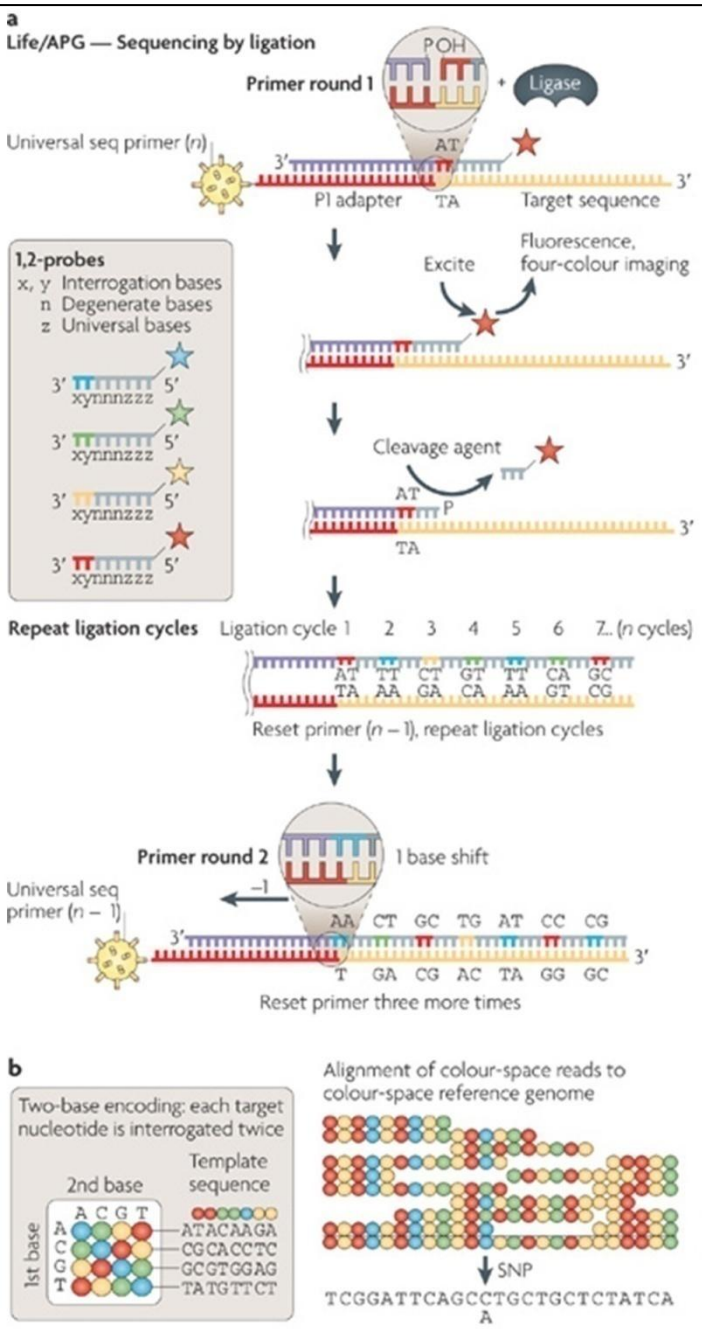



**Figure 8: SOLiD sequencing technique (Metzker, 2009).** The figure describes the ligation sequencing process using two base encoding.

In the first years the read length was very short with 25-35bp, but the current 5500 W Series Genetic Analyzer is capable of producing 75bp single end and two times 50bp mate pair reads. This is still less than the other techniques, but the disadvantage is compensated by the high base accuracy. SOLiD covers the same application range than SBS (see Section 2.1.3), but facilitates very good single nucleotide polymorphism detection possibilities due to the high read quality assured by the internal read quality filter of the system. On the other hand, a high quality is essential for this technique, because if only one base in a SOLiD read has a sequencing error, the decoding of all following bases is corrupted. Another disadvantage is the problem of SOLiD to deal with palindromic sequences (Huang *et al.*, 2012).

## 2.1.5. Ion Torrent

The "Ion Torrent" sequencing technology offered by Life Technologies (Rothberg *et al.*, 2011) is another real-time sequencing by synthesis approach and an adaption of the 454 pyrosequencing described in Section 2.1.2. It is based on a high density array of micro wells, acting as micro reactors, for single DNA templates. An ion sensitive layer and a proprietary ion-sensitive field-effect transistor (ISFET) sensor are located below each micro well. During incorporation of each DNA base in a newly synthesized strand an $H^+$ ion is released and the ion sensor allows measuring this event (Figure 9).

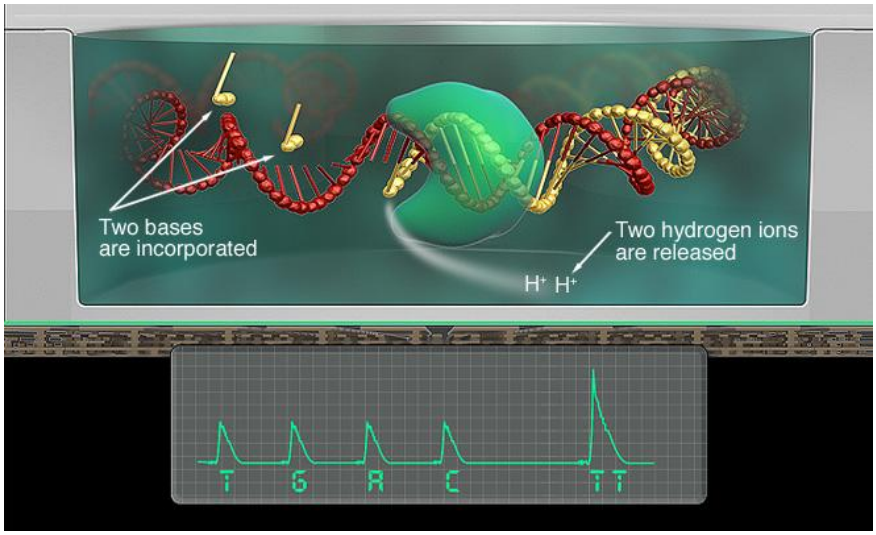



**Figure 9: Ion semiconductor sequencing procedure.** Two $H^+$ ions are released due to incorporation of two dTTPs. This event is directly recorded as an electrical signal, resulting in the corresponding peak in the lower part of the figure. Source: `www.lifetechnologies.com`, accessed on 05.01.2014.

As already mentioned, this technology is based on 454 pyrosequencing. Therefore, the preparation of the template DNA is the same described in Section 2.1.2 and Ion Torrent sequencing also utilizes an emPCR with one template DNA fragment located on one bead to clonally amplify the fragment for sequencing. Afterwards, the loaded beads are flowed across a complementary metal-oxide-semiconductor (CMOS) chip (Figure 11) containing the micro reactors for the loaded beads with the ion sensitive layer and the ISFET sensor below (Figure 10, a-c). The sequencing reagents including the DNA-polymerase are then added to the solution.

The sequencing itself is carried out in cycles - one cycle for each of the four dNTP species. Whenever a complementary dNTP is incorporated to the leading unpaired base of the template strand, this process involves the formation of a hydrogen bond and the release of pyrophosphate and a positively charged hydrogen ion (Figure 9). Each $H^+$ release changes the pH of the solution in the micro well. This triggers a voltage between substrate and oxide surfaces due to an ions sheath in the ISFET sensor. The strength of the voltage is proportional to the number of incorporated dNTPs. The electrical voltage signals of each micro reactor on the CMOS chip can directly be transmitted to and recorded by a computer without further conversion. The species of the incorporated nucleotide is set depending on the current sequencing cycle. After each cycle, a washing step has to be conducted in order to remove all free remaining dNTPs.

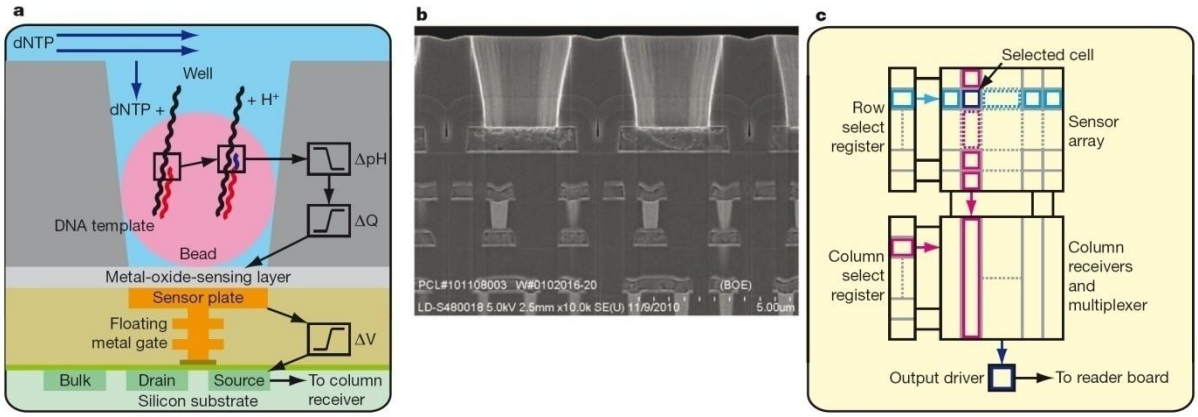



**Figure 10: (a)** Depicts a well in the CMOS chip, containing a DNA loaded bead, and the underlying ISFET sensor. When a dNTP is incorporated in the template DNA, an $H^+$ ion is released and changes the pH (ΔpH) of the solution in the well. This induces a change in surface potential of the metal-oxide-sensing layer, and a change in the potential (ΔV) of the source terminal of the underlying field-effect transistor. **(b)** Electron micrograph showing the alignment of the wells over the ISFET metal sensor plate and the underlying electronic layers. **(c)** The sensors are arranged in a two-dimensional array. A row select register enables one row of sensors at a time, causing each sensor to drive its source voltage onto a column. A column select register selects one of the columns for output to external electronics. Figure and description are adapted from (Rothberg *et al.*, 2011).

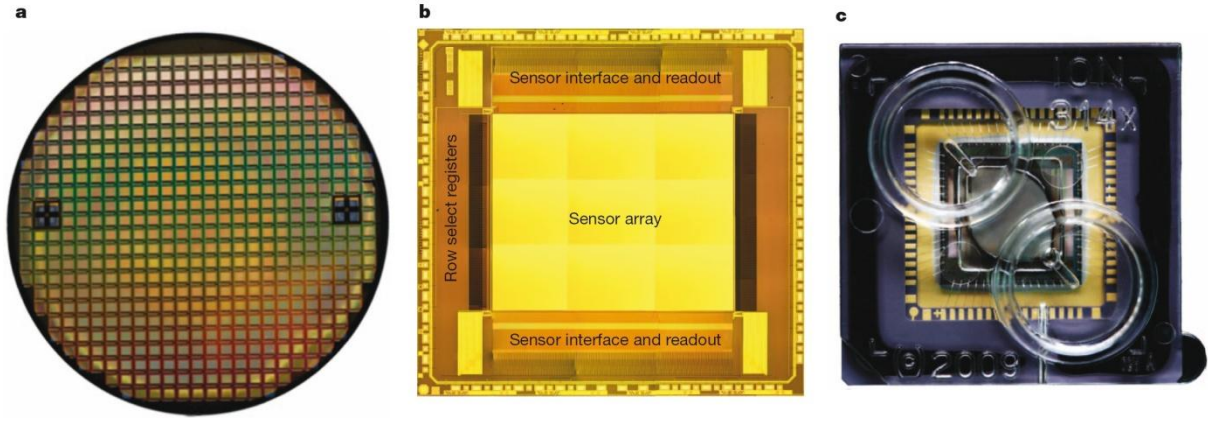



**Figure 11: (a)** Fabricated CMOS 8´´ wafer containing approximately 200 individual functional ion sensor dies. **(b)** Unpackaged die, after automated dicing of wafer, with functional regions indicated. **(c)** Die in ceramic package wire bonded for electrical connection, shown with moulded fluidic lid to allow addition of sequencing reagents. Figure and description are adapted from (Rothberg *et al.*, 2011).

The advantages of this sequencing technology are the simple sequencing chemistry, which does neither involve modified nucleotides, nor an enzymatic cascade, fluorescence or chemiluminescence. In addition, no optical signal measurement is necessary. Therefore, the time required for a sequencing run is short (2-7h, depending on machine and chip) and the costs of an Ion Torrent run are low (see Table 1). Here, the speed limiting factor is the cycling of substrate nucleotides through the CMOS chip and the throughput of one chip can be increased by further miniaturization of the chip components, which would allow for more micro reactors on one CMOS chip.

A limitation of Ion Torrent sequencing is the correct sequencing of long homopolymer repeats. Like in 454 pyrosequencing, long homopolymer stretches create ambiguous signals.
Currently, Life Technology offers two different sequencing systems. The *Ion Personal Genome Machine* (PGM) is designed for smaller scale sequencing with an output of 400,000 up to 5.5 million reads, depending on the sequencing chip. The average read length is 200 bases. The *Ion Proton* is designed for larger scale sequencing with an output of 60-80 million reads per run and a read length of up to 200 bases.

## 2.1.6. Pacific Biosciences: Real-Time Sequencing

The second single molecule sequencing technology offered by Pacific Biosciences entered the market in 2011. "Single molecule real-time (SMRT) sequencing" (Eid *et al.*, 2009) uses a nanoscale-optical approach utilizing so called zero-mode waveguides (ZMW) (Levene *et al.*, 2003; Foquet *et al.*, 2008) to detect DNA strand elongation of a single DNA molecule by a single DNA polymerase using fluorescently labelled dNTPs.
SMRT sequencing allows choosing between various DNA fragment sizes ranging from 500bp up to more than 20kb, depending on the application. For appropriate library fragmentation, a shearing approach for specific size selections is recommended. Afterwards, the fragment ends and other DNA damages have to be repaired using appropriate kits. Specific blunt-end hairpin adapters, called *SMRTbells* are ligated to both ends of the DNA fragments. The sequencing primers are complementary to the hairpin adapter (Figure 12, A). After primer annealing a single DNA polymerase is bound to one of the hairpins of each DNA fragment either by diffusion or proprietary magnetic beads. The prepared polymerase-template complexes are then loaded by diffusion or magnetic beads onto a silica slide of only a few nanometers in diameter containing the ZMWs - the *SMRT cell* (Figure 12, C). A ZMW is a tiny circular cavity of 70nm diameter, which is smaller than the approximately 390-700nm wavelength of visible light. Thus, the light of the laser in the sequencing machine, illuminating the SMRT cell from below, decays by entering the ZMW. Hereby, a nanophotonic confinement structure is created, which acts as confocal light microscope with an observation volume of ~20 zeptoliters. Statistically, one polymerase-template complex enters a ZMW, but in some ZMWs more than one or none can be found. Due to the surface structure, the complexes only bind to the ZMW surface (Figure 13, A). Each species of dNTPs is phospholinked with a different fluorescent dye. Thus, all dNTP species can be added to the sequencing solution at once without cycling.

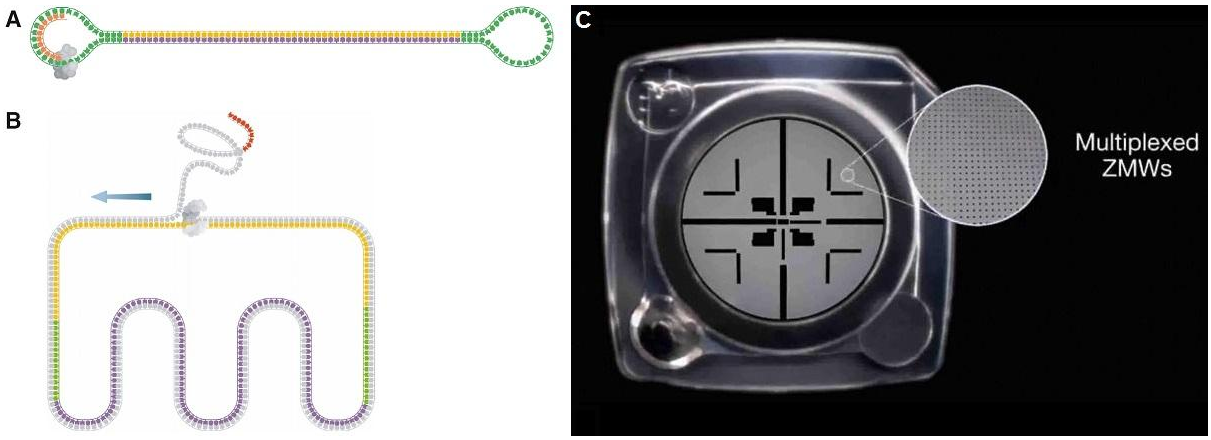



**Figure 12: SMRT sequencing template and SMRT cell. (A)** A sequencing template consists of a double-stranded DNA fragment flanked by two hairpin loops. The hairpin loops present a single-stranded region to which a sequencing primer can bind (orange). **(B)** As a strand displacing polymerase (gray) extends a primer from one of the hairpin loops, it uses one strand as template strand and displaces the other. When the polymerase returns to the 5'-end of the primer, it begins strand displacement of the primer and continues to synthesize DNA (moving in the direction of the blue arrow). Therefore, the length of sequence obtained from these templates is not limited by the insert length. Furthermore, the resulting sequence is derived from both sense and anti-sense strands. **(C)** Shows a SMRT cell containing thousands of ZMWs. Figure (A), (B) and their description are adapted from (Travers *et al.*, 2010), Figure (C) is taken from `www.pacificbiosciences.com`, accessed on 05.01.2014.

The sequencing reaction is then started by injection of missing sequencing reagents such as metal ions. Whenever a matching nucleotide is incorporated during DNA synthesis, pyrophosphate is naturally released from the dNTP and thereby also the fluorescent dye

(Figure 13, B). The dye is excited by a laser and the fluorescence is detected for a few milliseconds - as long as the incorporation of a nucleotide takes and until it diffuses away from the confined illumination area of the laser. Only 30nm of the ZMW surface are illuminated due to the light decay. A prism dispersive element then records and discriminates the wavelength of the dye. A computer records the intensities of the fluorescent signals of incorporated bases over time and directly deduces the read sequence of the DNA molecule anchored in each ZMW (Figure 14).

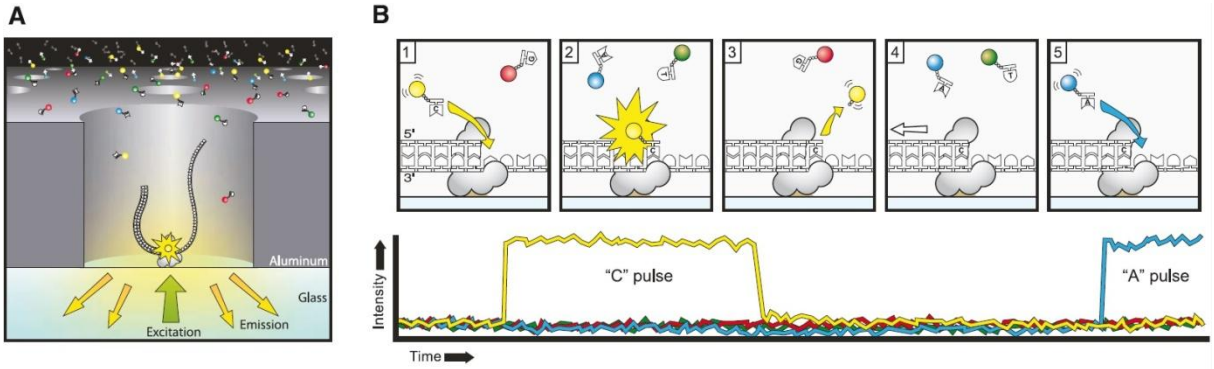



**Figure 13: Principle of single-molecule real-time DNA sequencing. (A)** Experimental geometry. A single molecule of DNA template-bound DNA polymerase is immobilized at the bottom of a ZMW, which is illuminated from below by laser light. The ZMW nanostructure provides excitation confinement in the zeptoliter (10−21 liter) regime, enabling detection of individual phospholinked nucleotide substrates against the bulk solution background as they are incorporated into the DNA strand by the polymerase. **(B)** Schematic event sequence of the phospholinked dNTP incorporation cycle, with a corresponding expected time trace of detected fluorescence intensity from the ZMW. (1) A phospholinked nucleotide forms a cognate association with the template in the polymerase active site, (2) causing an elevation of the fluorescence output on the corresponding color channel. (3) Phosphodiester bond formation liberates the dye-linker-pyrophosphate product, which diffuses out of the ZMW, thus ending the fluorescence pulse. (4) The polymerase translocates to the next position, and (5) the next cognate nucleotide binds the active site beginning the subsequent pulse. Figure and description are adapted from (Eid *et al.*, 2009).

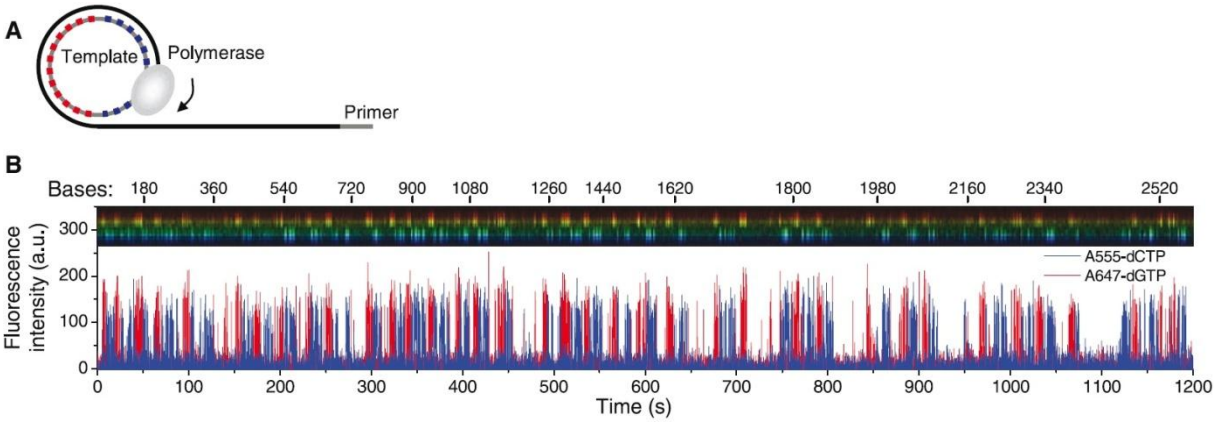



**Figure 14: Long read length activity of DNA polymerase. (A)** A circular DNA template for continuous incorporation via strand-displacement DNA synthesis. **(B)** Time-resolved intensity spectrum of fluorescence emission from a single ZMW synthesizing alternating blocks of two phospholinked nucleotides (A555-dCTP and A647-dGTP), interspersed with the other two unmodified dNTPs. The corresponding total length of synthesized DNA is indicated by the top axis. The figure and description are adapted from (Eid *et al.*, 2009).

The sequencing machine currently offered by Pacific Biosciences is the *PacBio RS II*. One SMRT cell (Figure 12) has 150,000 ZMWs and 1-16 SMRT cells can be used per sequencing run. Typically ~50,000 reads can be generated per SMRT cell with an average read length of 5.5-8.5kb, resulting in 275-375mb per SMRT cell and 4.4-6gb of sequence data per run.
The application range of SMRT sequencing is very broad. It offers two different sequencing variants: either Continuous Long Reads (CLRs) or Circular Consensus Sequencing (CCS) reads. In CLR mode, the polymerase elongates the complementary strand of the template until it falls of the DNA. The result is a single-pass long read with an average read length of ~10kb

and a low single base accuracy of currently 82.1-84.4%. This mode e.g. allows scaffolding of longer repetitive regions in *de novo* sequencing projects. Optimally, other high quality short read data sets should be available to correct sequencing errors of the SMRT sequencing reads. In CCS mode, strand-displacement DNA synthesis is used to elongate the complementary strand of the template circularly due to the SMRTbell hairpin. In this mode, DNA templates with an average length of ~2kb can be sequenced multiple times in one reaction by the polymerase, walking along the template circularly. The base quality of the resulting read sequence is competitive to the other sequencing technologies and can be used in both whole genome *de novo*- and re-sequencing experiments. Additionally, SMRT sequencing can be used to detect over 25 base modifications including DNA methylation by measuring the kinetics of base incorporation (Flusberg *et al.*, 2010). The long reads of SMRT sequencing also enable full isoform sequencing (Sharon *et al.*, 2013). In 2013, a partnership between Pacific Biosciences and Roche Diagnostics has been announced to develop *in vitro* diagnostics relying on SMRT sequencing.

## 2.2. Assessing Sequencing Read Quality

Assessing the consistency of sequencing libraries and performing quality control are crucial steps prior to downstream analyses (Del Fabbro *et al.*, 2013). This task embraces trimming reads by base quality (see Figure 15, left) and identification and removal of vector, adapter, tag and primer sequences. Inconsistencies can arise e.g. when the data shows an unexpected GC-content distribution not fitting the analyzed organism or vastly varying at certain read positions (see Figure 15, right).

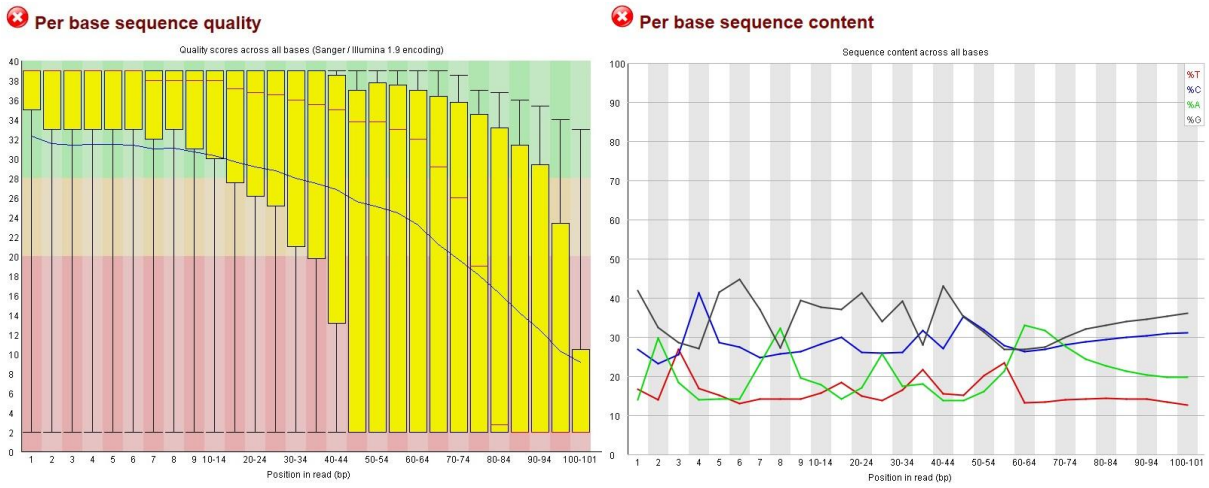



**Figure 15: Quality control of sequencing reads.** These two figures demonstrate the importance of quality control with the help of FastQC (Andrews, 2010). In the analyzed Illumina (see Section 2.1.3) raw read data set, the second half of many reads is hallmarked by low base quality values (histogram on the left), indicated by the red area in the lower part of the histogram. The second issue with the presented data set is the position specific discrepancy of the base content of the reads (line chart on the right). If reads are uniformly sampled from the whole genome, the data set should feature a steady base content in accordance with the expected GC-content of the analyzed organism.

A low GC-content bias can indicate that only genomic regions up to a certain GC-content could be sequenced. Low base quality reads are prone to false base calls. They disturb genome assembly by preventing overlaps between biologically overlapping reads with many mismatches and render mapping impossible due to their discrepancies to the reference. For analyzing and dealing with these issues, several tools have been developed and are freely available. Del Fabbro *et al.* (2013) compared 8 of them in their publication. Instead of recommending a single read trimming tool, they rather advise researchers to make a decision

depending on their individual data set quality, biological research question and their parameter choice including the requested trade-off between data set size and read quality.

In the following, an overview of these and additional trimming tools and their main features is presented:

- **FastQC** (Andrews, 2010) is not a trimming tool, but rather a quality control tool giving insight into sequencing data sets implemented in Java. FastQC shows various statistics and properties of the analyzed data set, e.g. per base sequence quality, per sequence GC content, overrepresented sequences and k-mer content.

- **Trimmomatic** (Lohse *et al.*, 2012) is a Java tool for trimming reads by base quality and removing adapter sequences. Trimmomatic offers different trimming options and allows converting read base qualities.

- **cutadapt** (Martin, 2011) is a tool for removing adapter sequences and trimming read sequences on the basis of the BWA trimming algorithm (Li and Durbin, 2009). Cutadapt is implemented in Python and C.

- **btrim** (Kong, 2011) is a tool for both, removing adapter sequences and trimming low quality regions from reads written in C.

- **SeqTrim** (Falgueras *et al.*, 2010) is a whole pipeline for processing sequencing data sets and implemented in Perl. Like most of the other tools, SeqTrim can trim low quality regions and adapter sequences. Additionally, it can detect and remove or mask low complexity, contaminant and chimeric regions from reads.

- **PRINSEQ** (Schmieder and Edwards, 2011) is a comprehensive read trimming tool implemented in Perl, which includes around 30 different options for adapter and quality trimming. Among them e.g. removal of polyA or polyT stretches and independent quality thresholds for left and right read ends. Additionally, it returns output statistics.

- **SolexaQA** (Cox *et al.*, 2010) is a quality control and read trimming tool implemented in Perl, producing visual output of read quality statistics. Besides their own trimming algorithm, the BWA trimming algorithm (Li and Durbin, 2009) can be used.

- **TagCleaner** (Schmieder *et al.*, 2010) is a tool for detection and removal of adapter sequences implemented in Perl. The adapter sequences can be unknown and are then estimated by the software.

- **ConDeTri** (Smeds and Künstner, 2011) is a 3' end read quality trimming tool for Illumina sequencing data implemented in Perl.

- **ERNE-FILTER** (Vezzi *et al.*, 2012) is a read quality trimming tool implemented in C++ and shipped within the *Extended Randomized Numerical alignEr* (ERNE) package. Contaminant sequences are identified and removed by mapping on a set of contamination reference sequences.

- **FASTX-Toolkit** (Gordon, 2009) is a collection of multiple C++ tools including read quality trimming applying a sliding window approach, trimming reads to a fixed length, adapter and contaminant sequences removal.

- **Sickle** (Joshi, 2011) is a read quality trimming tool relying on a sliding window approach, implemented in C.

- **Conveyor** (Linke *et al.*, 2011) is a general bioinformatics workflow engine. One of the available workflows performs trimming of single or read pair data by PHRED quality. The implemented approach takes the length of a low quality stretch as input along with a PHRED quality value and trims the read ends when a low quality stretch is encountered satisfying both parameters.

In conclusion, FastQC is the perfect starting point for read quality assessment. It provides a first insight into the data set quality and reveals possible contaminations and problems with the data. On this basis, a tool of choice can be applied to trim the reads. For many cases, the FASTX-Toolkit or cutadapt are suitable solutions because they offer a wide range of trimming options and contaminant removal. For trimming reads by base quality, the Conveyor trimming workflow is a reasonable choice. A clear advantage is the configuration of the low quality stretch. Thereby not each single low quality base leads to a loss of the following bases if their quality is above the configured PHRED quality threshold.

## 2.3. Assembling Genomes

As already mentioned in Section 2.1, all sequencing technologies require fractionation of large DNA sequences, such as whole eukaryotic chromosomes or complete circular bacterial genomes. After sequencing, the main goal is to reconstruct the original contiguous input sequence from a data set of short sequencing reads. This process is called "sequence assembly".

A DNA sequence is assembled by comparing all sequencing reads of the data set with one another and identifying overlaps in order to form contiguous sequences (**contigs**). Further information about the contig ordering can be obtained by utilizing scaffolding information from read pairs as introduced by Roach *et al.* (1995) (see Section 2.1).
After contig formation, the read pair information can be used to order adjacent contigs and estimate the gap between them. Two contigs can be considered as adjacent if one read of the same pair is located in each of the contigs. Most assemblers fill remaining gaps by a number of 'N's, estimated by the known insert size. The resulting sequences of merged contigs are called **scaffolds** and this process is called **scaffolding**. If the read pair information alone fails to establish the correct ordering of all contigs and as verification of scaffolding, a computational alignment to an available reference genome (see Section 2.3.3) and an optical map (Zhou *et al.*, 2007) of globally ordered restriction site locations can be used as genome wide scaffold. Integration of an additional SMRT sequencing data set can also drastically reduce the number of gaps in the assembly (English *et al.*, 2012).

Besides genome assembly, modern RNA-seq technology (Wang *et al.*, 2009) also enables transcriptome assembly (Lu *et al.*, 2013). It is important to note that transcriptome assembly captures the transcript presence and levels at a single time point in the investigated tissue or cell under given environmental conditions. Thus, the complete transcriptome of an organism cannot be recovered from a single RNA-seq run as at least some genes will not be expressed at the current development stage and environmental conditions. Two different approaches can be applied: *De novo* or reference based transcript assembly. For further reading on this topic and transcriptome assembly software, Lu *et al.* (2013) is recommended.

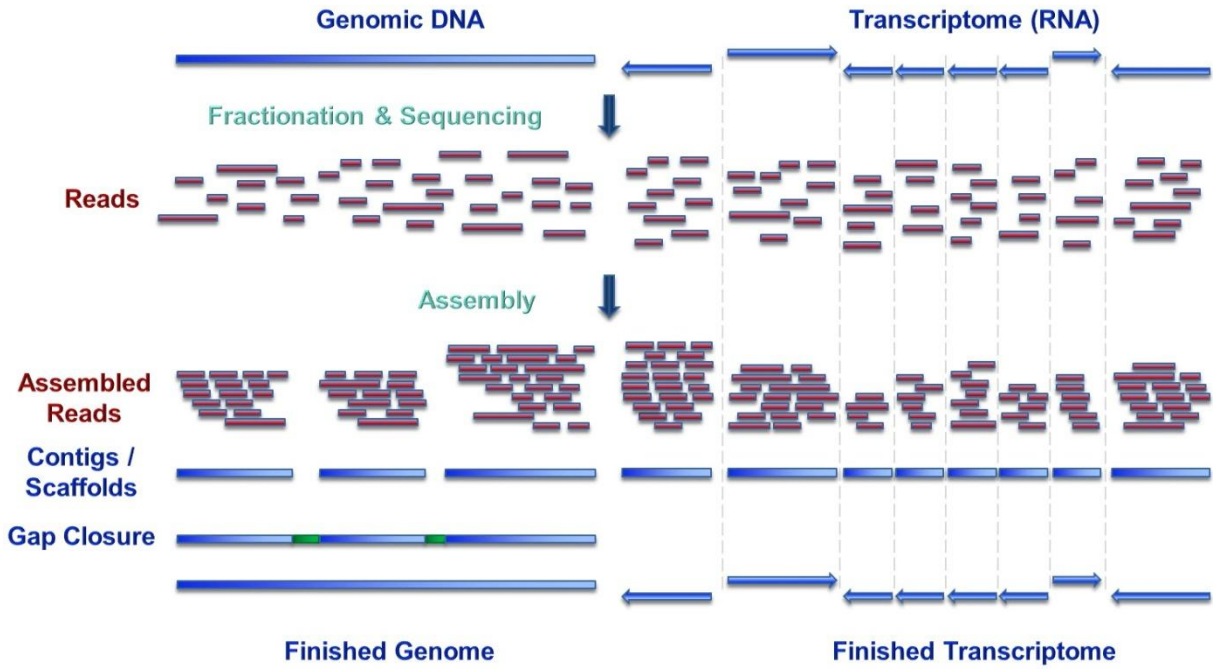



**Figure 16: Schematic representation of genome (left) and prokaryotic transcriptome assembly (right).** At first the DNA or RNA is extracted, fractionated and sequenced. The sequencing reads (red) represent small, unsorted pieces of DNA (RNA). During the assembly process, overlapping reads are merged into contigs or scaffolds. For finishing genomes, a gap closure (green) and polishing phase has to be conducted. The result is a textual representation of the genome or prokaryotic transcriptome under investigation (bottom line). A finished and complete prokaryotic transcriptome consists of as many sequences, as genes in that genome and does not contain any non-transcribed sequences.

Even though dozens of tools exist for solving the task of sequence assembly (see Section 2.3.1), it still poses a very complicated research challenge. Additionally, no perfect assembly metrics for assessing assembly quality could be developed to date (Baker, 2012). One of the major assembly challenges are almost indistinguishable highly-repetitive regions. Sequencing errors are another source of missassemblies and problems during genome assembly. Further, the genome coverage of sequencing data sets is not uniform and some regions may be underrepresented while others are overrepresented. E.g. pyrosequencing (see Section 2.1.2) and semiconductor sequencing (see Section 2.1.5) struggle with correct sequencing of homopolymer stretches. Also vector or primer artefacts of the sequencing process can confuse the assembly. Therefore, a quality assessment of sequencing data sets (described in Section 2.2) is more crucial for the quality of an assembly than the assembler itself (Salzberg *et al.*, 2012).

Combining multiple sequencing technologies with different insert sizes and long SMRT sequencing reads and producing sufficient coverage to overcome general sequencing errors can also greatly improve the quality of an assembly (Koren *et al.*, 2013).

### 2.3.1. Genome Assembly Software

Two different approaches are used by up-to-date assembly tools: They are either based on overlap graphs or De Bruijn graphs (Li *et al.*, 2012). Algorithms based on overlap graphs mainly consist of three phases forming the name of this approach: First an overlap phase, second a layout phase and third a consensus phase (overlap-layout-consensus algorithms) (see Figure 17) are conducted.

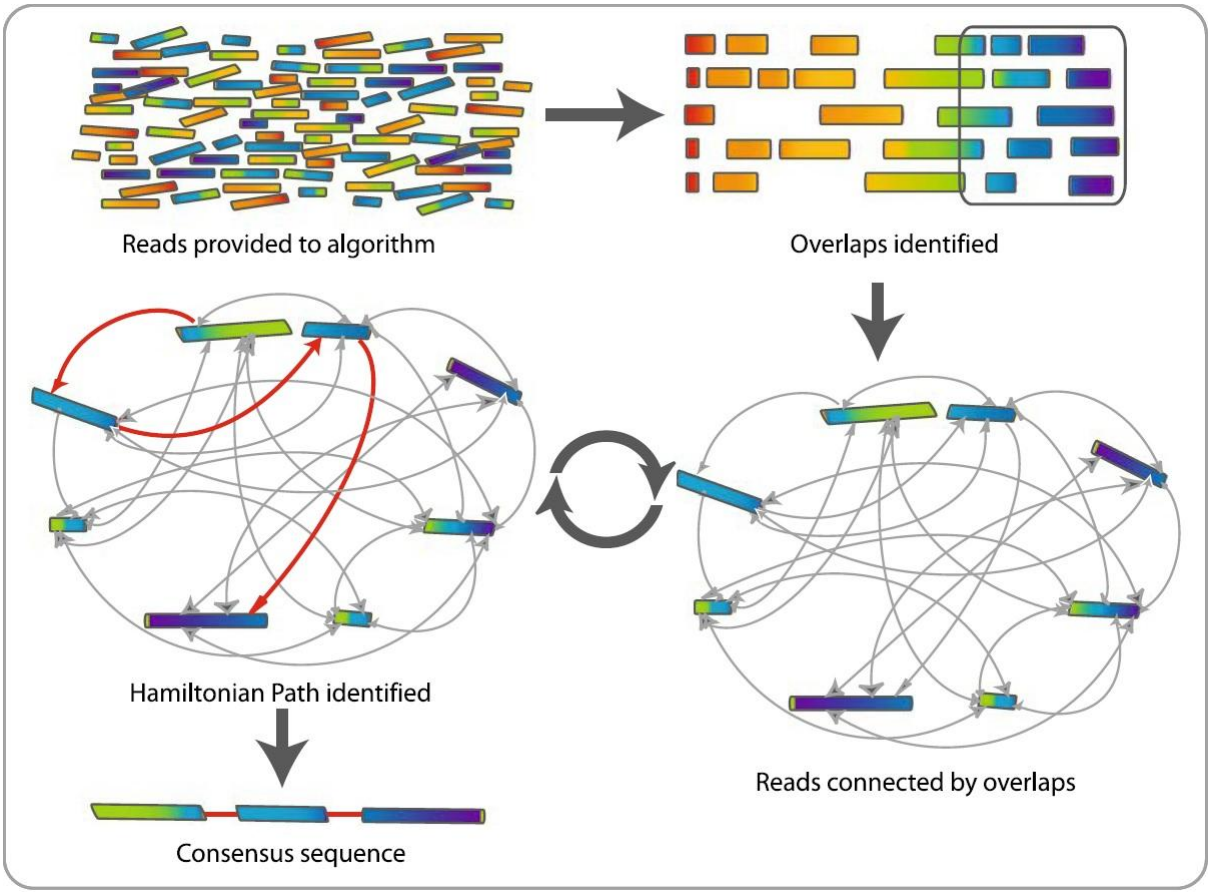



**Figure 17: Overlap–layout–consensus approach.** First, reads are provided to the algorithm. Overlapping regions are identified. Each read is graphed as a node and the overlaps are represented as edges joining the two nodes. The algorithm determines the best path through the graph (Hamiltonian path). Redundant information (i.e., unused nodes and edges) is discarded. This process is carried out multiple times and resulting sequences are combined to give the final consensus sequence that represents the genome. The figure and description are adapted from (Commins *et al.*, 2009).

In contrast, algorithms based on De Bruijn graphs first cut the reads into smaller pieces of equal size, the **k-mers**. These k-mers are used to construct the corresponding De Bruijn graph. This type of graph was discovered independently by Nicolas Govert de Bruijn (de Bruijn, 1946) and Irving John Good (Good, 1946). A De Bruijn graph is a directed graph representing overlaps of consecutive character sequences. A De Bruijn graph for genome assembly is created by forming a node for each k-mer observed in the reads. Their length is chosen to be odd to avoid palindromic k-mers which would create a more complex and harder to resolve graph. All nodes overlapping by all except the last base are connected by directed arcs. Nodes forming a unique path can already be merged, but each k-mer occurring multiple times in the genome creates ambiguities which have to be solved by the assembler (see Figure 18).

In 2015 numerous commercial as well as free assembly tools exist. Thus, it is not an easy task to identify the most suitable assembler for a *de novo* genome assembly project. Multiple studies have been carried out to evaluate different assemblers and to ascertain the best programs (Earl *et al.*, 2011; Salzberg *et al.*, 2012; Magoc *et al.*, 2013; Bradnam *et al.*, 2013). Assembly results can further benefit from combination of assemblies produced by different programs for the same data set, since most assemblers introduce mainly unique errors to the assembly (Salzberg *et al.*, 2012).

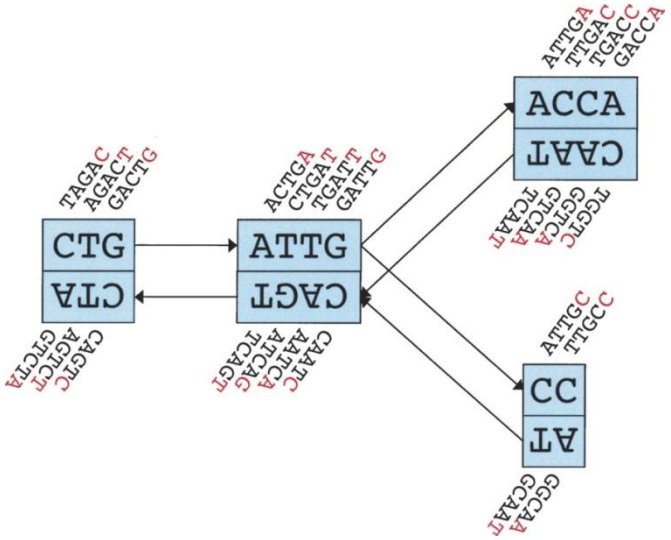



**Figure 18: Schematic De Bruijn graph representation as used in Velvet (Zerbino and Birney, 2008).** Each node, represented by a single rectangle, represents a series of overlapping *k*-mers (in this case, *k* = 5) from the De Bruijn graph, listed directly *above* or *below*. (Red) The last nucleotide of each *k*-mer. The sequence of those final nucleotides, copied in large letters in the rectangle, is the sequence of the node. The twin node, directly attached to the node, either *below* or *above*, represents the reverse series of reverse complement *k*-mers. Arcs are represented as arrows *between* nodes. The last *k*-mer of an arc's origin overlaps with the first of its destination. Each arc has a symmetric arc. Note that the two nodes on the *left* could be merged into one without loss of information, because they form a chain. The figure and description are adapted from (Zerbino and Birney, 2008).

The most important metrics obtainable for each assembly are introduced in the following:

- **Contig count**: It reveals the number of breaks and gaps in the assembly.
- **Scaffold count**: In connection to the contig count it reveals how many contigs can be scaffolded and how many unresolved breaks and gaps remain in the assembly. This metric is only available if a read pair data set has been included in the assembly.
- The **average alignment depth**: This measure gives an insight in the overall quality of the assembly. A low average alignment depth indicates that too less reads were available or could be integrated in any contig of the assembly. Regions of the contigs with low alignment depth should be examined carefully. They might be more prone to assembly errors.
- **N50**: This value represents the length *N* of the shortest contig contributing to more than 50% of the total assembled bases when summing the assembled bases starting from the largest contig and continuing with the next smaller contig (Earl *et al.*, 2011). The N50 gives an insight in the potential contiguity and extent of an assembly.

More metrics, especially related to analyses in connection with a reference genome and for eukaryotic genomes, are discussed in the assembly evaluation publications listed above.
In the following, relevant assembly programs and their main features are presented. If not stated otherwise, they support read pair data. A more complete list of assembly tools can be found on http://en.wikipedia.org/wiki/Sequence_assembly.

Assemblers based on overlap graphs:

- **Newbler** (Margulies *et al.*, 2005) is the proprietary assembler of 454 Life Sciences and was mainly designed for 454 pyrosequencing data. Later, it was enhanced to work with Illumina and other sequencing data as well.

- **CABOG** (Celera Assembler with Best Overlap Graph) (Miller *et al.*, 2008) is an extension of one of the older assembly tools, developed since 1999: The Celera

Assembler (Myers *et al.*, 2000). CABOG supports hybrid assemblies of Sanger, pyrosequencing, SBS and SMRT sequencing data sets.

- **SGA** (String Graph Assembler) (Simpson and Durbin, 2012) is a parallelizable assembly tool based on the concept of string graphs, which yields a reduced memory footprint. SGA assembles solely single end data sets.

Assemblers based on De Bruijn graphs:

- **Velvet** (Zerbino and Birney, 2008) assembles a genome based on a De Bruijn graph for a given k-mer. It is fast due to multithreading and supports reads from different sequencing platforms like SBS and SOLiD colorspace reads.

- **SPAdes** (Bankevich *et al.*, 2012): Uses paired assembly graphs, special cases of the A-Bruijn graph (Pevzner *et al.*, 2004), to include read pair distances in the assembly process. The k-mers are only used for graph construction. Afterwards, solely graph-theoretical operations are performed based on the graph topology, coverage and sequence length to restore the original genome sequence. A specialty of SPAdes is also that it automatically combines the results from iterations with varying k-mer lengths.

- **SOAPdenovo2** (Short Oligonucleotide Analysis Package) (Luo *et al.*, 2012) automatically combines results for multiple k-mers and was especially designed for fast assembly of large genomes.

- **ABySS** (Assembly By Short Sequences) (Simpson *et al.*, 2009) is a parallelizable assembler whose latest version 1.3.7 allows re-scaffolding after an initial assembly by adding additional data sets.

- **ALLPATHS-LG** (Gnerre *et al.*, 2011) mandatorily requires the combination of multiple data sets with different insert sizes or from different sequencing platforms in order to improve assembly quality.

- **Meraculous** (Chapman *et al.*, 2011) is a conservative, memory-efficient De Bruijn graph assembler, which has performed well in the Assemblathon 2 (Bradnam *et al.*, 2013).

- **Ray** (Boisvert *et al.*, 2010) can combine multiple data sets with different insert sizes and from different sequencing platforms to yield high quality assemblies. Running time is decreased by multithreading.

Assemblers combining overlap and De Bruijn graph approaches:

- **MaSuRCA** (Maryland Super-Read Celera Assembler) (Zimin *et al.*, 2013) is the only assembler combining a De Bruijn graph approach with an overlap-layout-consensus approach. It is designed for assembling hybrid data sets from multiple sequencing technologies.

For scaffolding of an initial assembly with complementary data sets or combination of multiple assemblies specialized standalone programs are available. The implemented algorithms are either greedy (e.g. SSPACE (Boetzer *et al.*, 2010)) or based on the concept of a

contig graph (e.g. BESST (Sahlin, 2013), SOPRA (Dayarian *et al.*, 2010), Bambus2 (Koren *et al.*, 2011), and Opera (Gao *et al.*, 2011)).
In a nutshell, there is not one best assembler to recommend. It rather depends on the sequenced organism, the used sequencing technology, if a combination of data sets is used and the metrics applied to rate the assembly. In the latest assembler evaluations the best scoring assemblers were Newbler, ALLPATH-LG and SGA in the Assemblathon 2 (Bradnam *et al.*, 2013), while in GAGE-B (Magoc *et al.*, 2013) SPAdes and MaSuRCA scored best. Among these assemblers, only SGA was included in both studies, but outperformed in GAGE-B. To generate the most reliable assembly, the authors of the Assemblathon 2 recommend testing different assemblers and ranking their assemblies against each other.

### 2.3.2. Assembly Quality and Genome Finishing

A single high quality consensus sequence is rarely achieved solely by computational methods. Instead, the result of initial assemblies is mostly a draft genome containing gaps. To close remaining gaps in the assembly, genome finishing is required. The cost and time requirements for generating a finished high quality genome are considerably higher than for generating a draft genome (Mardis *et al.*, 2002), since it involves additional manual laboratory work and more expensive Sanger sequencing. Since 2012 also SMRT sequencing can be used for gap closure before other techniques are employed (English *et al.*, 2012), but until today the disadvantage of this technique are the sequencing costs.

Finishing can generally be divided into two steps: The first step is gap closure and the second is validation and refinement. The most well-established gap closure methods are either directed PCRs or primer walking. Directed gap closure PCRs are useful for gaps smaller than reads generated by Sanger sequencing (~1000bp).

For this method, unique primers flanking a gap between adjacent contigs ordered according to a reference are designed. A PCR on the complete genomic DNA using these primers amplifies the missing DNA fragment, which spans the assembly gap (Figure 19, A). If the contig ordering is unknown, unique primers for all contig ends have to be created and tested pairwise for compatibility. This process can be accelerated by multiplex PCRs (Tettelin *et al.*, 1999). Resulting PCR fragments are typically sequenced using the Sanger sequencing technology. For longer gaps, enhanced primer walking can be used. The original approach was introduced in (Kieleczawa *et al.*, 1992). Today, this method either uses fosmids (Hall, 2004) or bacterial artificial chromosomes (BACs) (Shizuya *et al.*, 1992) for scaffolding and gap closure. A fosmid is a bacterial F-plasmid with an insert size of ~40kb, while a BAC originates from a bacterial F-plasmid, but has a much larger insert size of up to 300kb. The first step is to create a fosmid or BAC library or a combination of both for the complete genome sequence. Next, the insert ends from all plasmids are sequenced using Sanger technology and the reads are mapped onto the assembled contigs. Scaffolding can be done, if both ends of one plasmid map to different contigs. In this case, primer walking is conducted: New primers are designed for the respective contig ends and the PCR products which are spanning the whole gap are sequenced by Sanger technology from both ends. Hereby, both contigs are extended by ~1kb into the gap. This primer walking is repeated, until the PCR products from both contig ends overlap and the whole gap is closed (Figure 19, B).

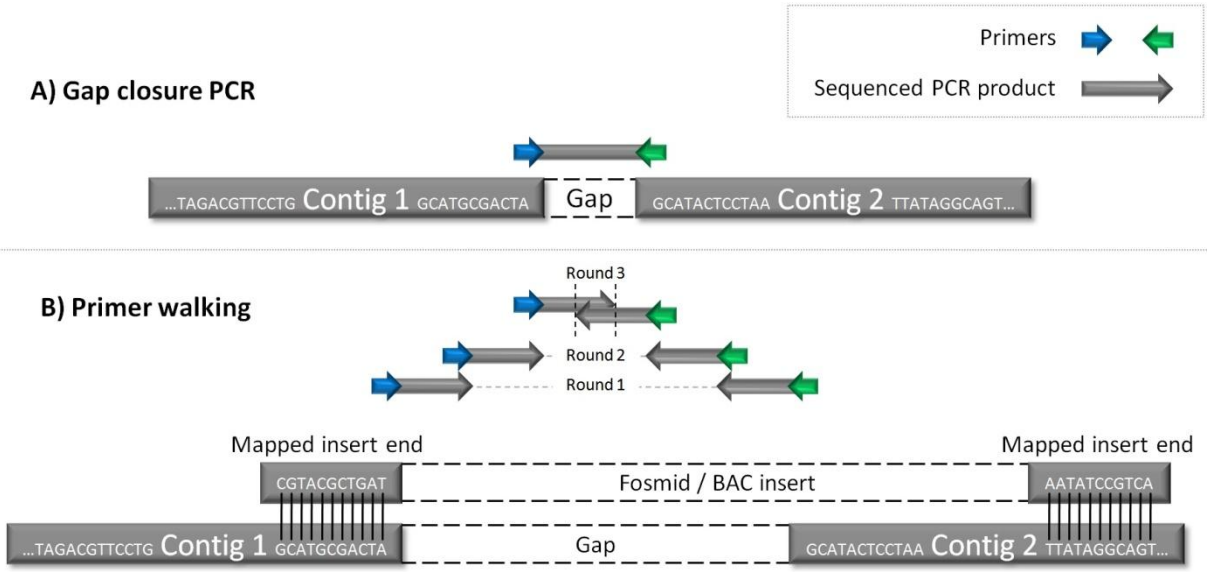



**Figure 19: PCR genome finishing methods.** The figure shows a schematic representation explaining the principle of gap closure PCRs (A) and primer walking (B). **A)** If the contig ordering is known, a gap shorter than a Sanger sequencing read (~1kb) can be closed by a single PCR with specific primers for both contig ends. **B)** In this example a gap is closed by three rounds of primer walking. The plasmid (fosmid (~40kb) or BAC (~300kb)) insert contains the genomic sequence of the gap, missing in the assembly. Once unique primers are designed for the adjacent contig ends (blue and green), a PCR and Sanger sequencing of the PCR product from both strands reveals ~1kb of the gap sequence. New primers are created for the new contig ends and another PCR is carried out. By iteratively repeating the primer walking three times the showcase gap can be closed, since the sequenced PCR products of round 3 contain a complementary region. An overlap has been found and the gap is closed. This method gets more laborious and time consuming in dependency of the gap size.

The question if finishing is necessary for all sequenced genomes and all genomic regions keeps the scientific community occupied since Sanger sequencing was automated and the first NGS technologies emerged (Fraser *et al.*, 2002; Branscomb and Predki, 2002). Depending on the research question, a draft genome of good quality can be sufficient if money and time are limited resources. Especially, when a finished genome sequence of a closely related organism is available, a draft genome is sufficient for many genomic studies. The sequence of the close relative can be employed to enhance the draft assembly. Additionally, most genomic features are contained in a draft genome of good quality which can already be generated within less than one week. Whereas, due to laborious work, finishing can take several weeks or even month (Nagarajan *et al.*, 2010; Aury *et al.*, 2008).
For high-priority genomes, a finished assembly on the other hand provides a valuable permanent resource, allowing more detailed analyses of the data. Identifying the correct operon structures or patterns of gene regulation are only two examples (Nagarajan *et al.*, 2010). Genomic rearrangements pose another problem to draft assemblies. Rearrangements might get lost due to gaps between contigs and when contigs are aligned to a reference sequence with another genomic structure. Additionally, genes overlapping contig borders are often destroyed or contain at least a frame-shift. These destroyed genes and also missing DNA regions in a draft assembly lead to a bias in gene content analyses and do not fully reflect the real genome content (Alkan *et al.*, 2009).

A much less time consuming and with further decreasing sequencing costs probably cheaper method to generate finished genomes has been established by the long reads offered by the SMRT sequencing technology as of now (English *et al.*, 2012; Koren *et al.*, 2013).
In fact, more than 50% of the complete genome projects listed in the Genomes Online Database (GOLD)[7] only provide draft genomes. Thus, draft genomes are not the exception and genomic analysis methods should take their wise treatment into consideration.

[7] http://genomesonline.org/cgi-bin/GOLD/sequencing_status_distribution.cgi (accessed 26.03.2014)

## 2.3.3. Genome Layout and Finishing Software

Many software tools for genome layout and finishing have been developed since whole genome sequencing became feasible. One group of tools focuses on contig and scaffold alignment to an already known reference genome after the initial assembly. These programs should be used before starting with the manual finishing phase in the laboratory, since the knowledge of the contig order is crucial to the effort needed for closing the remaining gaps. A few relevant tools are:

- **r2cat** (Related Reference Contig Arrangement Tool) (Husemann and Stoye, 2010) automatically arranges contigs or scaffolds to another reference by first applying a q-gram filter (Rasmussen *et al.*, 2006) and then ordering the contigs according to their matches by a sliding window approach. Ordering can be changed manually by drag and drop.

- **ABACAS** (Algorithm-Based Automatic Contiguation of Assembled Sequences) (Assefa *et al.*, 2009) identifies regions of synteny between contigs and reference with MUMmer (Kurtz *et al.*, 2004) and arranges the contigs respectively. Primers flanking remaining gaps are created by an integrated version of Primer3 (Koressaar and Remm, 2007).

- **OSLay** (Optimal Syntenic Layouter) (Richter *et al.*, 2007) does not compute matches between assembly and reference itself. It rather takes BLAST or MUMmer results as input. Its method is based on maximum weight matching within a so called layout graph. OSLay can also handle a reference, which consists of multiple sequences.

- **Projector2** (van Hijum *et al.*, 2005) uses BLAST for synteny detection and orders the contigs according to the three criteria order, orientation and spacing. Further, it includes repeat masking for repetitive regions.

All of these tools, except OSLay, can automatically design primers for the contig ends.
For manual refinement of genome assemblies basically two widely used tools are available:

- **Consed** (Gordon *et al.*, 1998) is one of the first finishing tools and has just undergone a modernization (Gordon and Green, 2013). It automatically identifies problematic regions in assemblies and offers different visual editors, e.g. for the reference consensus sequence and the reads.

- **Hawkeye** (Schatz *et al.*, 2013) enables several perspectives for genome assembly data. It starts with basic statistics and overview of the assembly. On the next levels, the user is empowered to zoom into the data to inspect scaffolds, contigs, reads and nucleotides. Hawkeye integrates the AMOS assembly forensics pipeline (Phillippy *et al.*, 2008) in order to automatically detect erroneous regions within assemblies. They are highlighted for easy location.

The finishing methods explicated in the previous section need to be applied to gaps which still remain after computational arrangement and finishing contigs. A special finishing tool supporting the laboratory work with fosmid and BAC libraries is **BACcardi** (Bartels *et al.*, 2005). It visually displays mappings of fosmid or BAC end sequences on the contigs. Thus, BACcardi aids identification of missassemblies and creation of a scaffold for the contigs based on the plasmid library.

# 2.4. Automatic Genome Annotation

Genome annotation is the identification of genomic features and the subsequent assignment of their biological function to a known genomic sequence (Stein, 2001). Annotation can be performed either manually or automatically. Manual genome annotation always involves tedious manual work handling each potential gene region separately up to generation and integration of experimental data as verification. Therefore, genome annotation is normally started by automatic gene and function prediction, which is afterwards refined by manual annotation. Due to high-throughput generation of sequence data, unrevised automatic annotation data is often directly kept as end result in many genome projects. The cost and time requirements for complete manual refinement are much too expensive for most projects analyzing dozens or even hundreds of genomes (Fox and Kling, 2010; Hilker *et al.*, 2014). Thus, extensive manual refinement is only possible for a few high-priority genomes.

The process of automatic genome annotation (Yandell and Ence, 2012) starts with repeat masking and filtering of non-coding regions. Available tools for this step are also listed in the publication of Yandell and Ence (2012). Three different approaches exist for gene prediction. The first method is the alignment of existent evidence to the yet unannotated novel genome sequence. This evidence can be represented by coding sequences (CDS), protein, expressed sequence tag (EST) or RNA-seq data from available databases (e.g. from the UniProt database (UniProt Consortium, 2011)) or parallel experiments. The second method is *ab initio* gene prediction. This method utilizes statistical models to predict the genes. The third method combines *ab initio* gene prediction with incorporation of evidence. For prokaryotic genomes, the third approach produces the most reliable output and is used in state-of-the-art annotation platforms (see below). Following gene prediction, the annotation phase assigns biological functions to the predicted genes by selecting the most probable ortholog from existing databases.
Automatic genome annotation requires extensive compute power to efficiently analyze genomic data and compare it to several huge databases. Web-based platforms running on appropriate computing infrastructure can comply with the demand of the scientific community.
In the following, the focus is placed on software platforms for microbial genome annotation. Such applications can be subdivided into three classes. First, simply automatic annotation services, secondly, annotation services including result visualizations and thirdly, more comprehensive annotation platforms also supporting manual annotation editing and visualization of the genome. Common automatic annotation services belonging to the first class include the NCBI Prokaryotic Genome Annotation Pipeline based on the Prokaryotic Genomes Automatic Annotation Pipeline (PGAAP) mentioned in (Angiuoli *et al.*, 2008) and Xbase2 (Chaudhuri *et al.*, 2008). Also pipelines for user servers (DIYA (Do-It-Yourself Annotator) (Stewart *et al.*, 2009)) and desktop computers (EUGENE-PP (Sallet *et al.*, 2014) and Prokka (Seemann, 2014)) have been made publicly available. A widely used platform belonging to the second class is RAST (Aziz *et al.*, 2008). However, the focus here is placed upon commonly used comprehensive annotation web-platforms belonging to the third class:

- **GenDB** (Meyer *et al.*, 2003) is an open-source project developed as a global web service using Perl and common gateway interface (CGI). It offers public as well as private projects with restricted access and a collaborative web interface for manual annotation refinement after automatic annotation. The GenDB annotation pipeline is modular and extensible and starts with a gene prediction tool, such as Prodigal (Hyatt *et al.*, 2010) or Glimmer3 (Delcher *et al.*, 2007). Afterwards, several tools for

automatic annotation refinement are used (e.g. tRNAscan-SE (Lowe and Eddy, 1997) and SignalP (Nielsen *et al.*, 1997)). GenDB also includes visualizations of the results and several functions for whole genome and metabolic pathway analysis.

- **SABIA** (System for Automated Bacterial Integrated Annotation) (Almeida *et al.*, 2004) is a Perl and CGI based web service combining automatic assembly and annotation of private genome data. Assembly is performed by Consed (Gordon and Green, 2013). For gene prediction also Glimmer3 (Delcher *et al.*, 2007) is employed. Further, SABIA supports prediction of tRNA genes with tRNAscan-SE (Lowe and Eddy, 1997). Results can be viewed and edited manually in the web interface.
- **AGMIAL** (Bryson *et al.*, 2006) is a web service focusing on manual expert annotators within private genome projects. Its annotation strategy is modular and extensible. AGMIAL is a two component system of one manager for the gene prediction and contig visualization and one for protein annotation. They use their own Hidden Markov Model based *ab initio* gene prediction tool named SHOW. AGMIAL also includes a few more prediction tools including tRNAscan-SE (Lowe and Eddy, 1997).
- **MicroScope** (Vallenet *et al.*, 2013) started solely as microbial annotation platform MaGe (Vallenet *et al.*, 2006), employing a forest of tools for identification and refinement of genes (e.g. tRNAscan-SE (Lowe and Eddy, 1997) and MICheck (Cruveiller *et al.*, 2005)). Afterwards the annotation can be manually modified. Today MicroScope has been extended by metabolic pathway and comparative genome analysis functions (see Section 2.5.3). MicroScope supports private as well as public projects.

A more comprehensive overview of annotation platforms is given in (Bryson *et al.*, 2006).
When annotating a prokaryotic genome, the tool of choice depends on time restriction and quality requirements. When time is the critical factor, it might be useful to set up client software like Prokka which allows annotation of a complete bacterial genome within ~10 minutes (min) on an average desktop computer. When more time can be spent, a comprehensive annotation platform facilitating subsequent manual annotation like GenDB or MicroScope can be recommended. These two platforms are favored, because they contain the broadest sets of annotation tools and thus will generate the most detailed annotation results.

## 2.5. Large-Scale Comparative Genomics

One of the applications of annotated genomes is to use them in comparative genome analyses. The comparative genomics research field targets the comparative analysis of multiple genomes and their features. Depending on the study, the analyzed genomes can originate from closely or more distantly related organisms. Closely related organisms are often analyzed for smaller, but important differences, e.g. a pathogenic and a non-pathogenic strain of the same bacterium. More distantly related genomes can shed light on the overall phylogenetic tree like for the "tree of life" (Roger and Simpson, 2009).
Prior to the discovery of DNA, the most accurate definitions available for taxonomic division were in general phenotypic traits and for microbial organisms gram-staining and shape. Already a decade after DNA discovery, it's importance for the taxonomic classification was realized by (Zuckerkandl and Pauling, 1965). The era of comparative whole genome studies based on their genomic features began at the same time when the proposition to utilize measurable criteria like DNA reassociation in combination with phenotypic traits for phylogenetic relationships was made by (Wayne *et al.*, 1987). One of the first comparative studies of whole genomes compared the genomic content of two viruses (McGeoch and Davison, 1986).

Since the Sanger sequencing era, whole prokaryotic as well as eukaryotic organisms are subject to comparative studies (Tettelin *et al.*, 2008; Ureta-Vidal *et al.*, 2003). The continuous tremendous growth of available genomic sequence data enabled by NGS techniques has further revolutionized the research field of comparative genomics. Especially studies of multiple small prokaryotic genomes profit from the advent of high-throughput sequencing technologies. An appropriate sequencer can reveal the complete DNA sequences of many bacteria strains in a single run.
Several research questions can be addressed by comparative genomics. One question is the evaluation of evolutionary relationships between different species, strains and all organisms in order to understand the driving force behind evolution. In particular, the focus is placed on bacterial comparative genomics in the following sections due to the research topic of this work. Correct classification and knowledge about the genomic content of bacterial strains can play an important role for successful treatment of diseases caused by bacteria (Veesenmeyer *et al.*, 2009). In personalized medicine, treatments can optimally be varied depending on the presence of virulence factors and antibiotic resistances in the current infestation. In this regard another relevant question is how many specimen of an organism need to be sequenced in order to establish a substantiated data source of its genomic information and variability. Among several studies (Hiller *et al.*, 2007; Klockgether *et al.*, 2011) have exposed that one specimen is clearly not sufficient. Variable elements like accessory genomic islands or plasmids demand sequencing of multiple specimens per bacterial strain. One example for a universal vaccine which could only be created due to sequencing of multiple entities of the same bacteria strain was presented by (Kung *et al.*, 2010).

### 2.5.1. Homology and Genomic Subsets

When it comes to phylogenetic comparison of genomes, several terms are important to distinguish different evolutionary relationships. These terms where mainly shaped by (Fitch, 1970, 2000).
DNA, RNA and amino acid comparisons are predicated on homology. **Homology** denotes evolution and divergence of biological features, such as physical attributes, proteins or genes, from the same ancestor. In this context, Fitch defined several terms related to homology which are here taken from the review of (Bachhawat, 2006):

- **Orthologues** are homologous genes that have evolved from a common ancestral gene by speciation. They usually have similar functions.

- **Paralogues** are homologues that are related or produced by duplication within a genome. They often have evolved to perform different functions.

- **Xenologues** are homologues that are related by an interspecies (horizontal) transfer of the genetic material for one of the homologues. The functions of the xenologues are quite often similar.

- **Analogues** are non-homologous genes/proteins that have descended convergently from an unrelated ancestor (this is also referred to as "homoplasy"). They have similar functions although they are unrelated in either sequence or structure. This is a case of "non-orthologous gene displacement".

- **Horizontal (lateral) gene transfer** is the movement of genetic material between species (or genus) other than by vertical descent. In bacteria this process occurs by either natural transformation, conjugation, or transduction (through viruses).

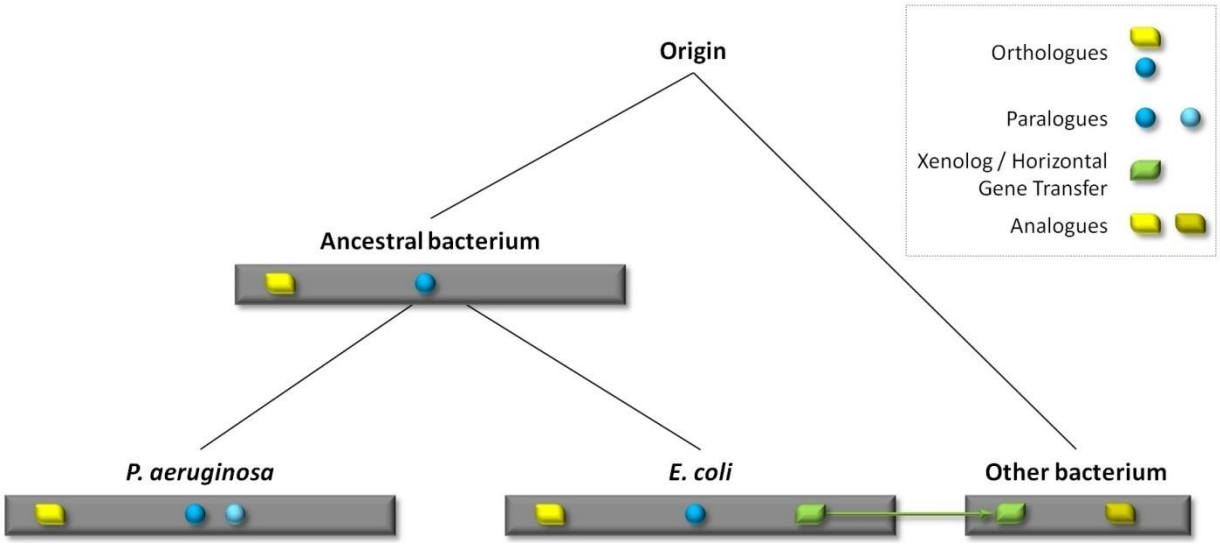



**Figure 20: Gene homology.** A top-down phylogenetic tree is shown, representing the origin of life and the genomes of four bacteria (grey rectangles) at two given time points during evolution. The ancestral genome, *P. aeruginosa* and *E. coli* share the **orthologous** yellow and dark blue genes. Note that the sequence of both genes will most probably not be identical in the three bacteria, but they originate from the same genes of the ancestral bacterium. A duplication event in *P. aeruginosa* has created a **paralogous** gene (light blue) of the dark blue gene. The green gene from *E. coli* has been **horizontally transferred** to the other bacterium, resulting in a **xenologue** gene. The dark yellow gene from the other bacterium fulfills the same biological function as the yellow gene in the other three bacteria. Thus, both genes are **analogues** of each other.

From this list, only orthologous genes allow phylogenetic conclusions (Fitch, 1970): "Where the homology is the result of speciation so that the history of the gene reflects the history of the species (for example α hemoglobin in man and mouse) the genes should be called orthologous (ortho = exact). Phylogenies require orthologous, not paralogous, genes." Thus, the most important task of comparative genome analyses is the identification of orthologous genes.
The entirety of all genes found in compared organisms is categorized in gene sets based on their homology as introduced in the groundbreaking publication of (Tettelin *et al.*, 2005):
"A bacterial species can be described by its "**pan-genome**" (pan, from the Greek word παν, meaning whole), which includes a **core genome** containing genes present in all strains and a **dispensable genome** composed of genes absent from one or more strains and **genes** that are **unique** to each strain."
These definitions were further refined in their subsequent publication (Medini *et al.*, 2005). Accordingly, the core genome of a species contains all genes essential for survival and genes coding for general phenotypic traits. The dispensable genome (sometimes also called **accessory genome**) consists of genes encoding niche functions or secondary pathways, which are advantageous in certain environments (e.g. antibiotic resistances). The dispensable genome adds to the diversity of a species as well as the unique genes. In bacteria, large parts of the dispensable genome are often organized in regions of genomic plasticity (**RGP**) which either contain **genomic islands** (GI) (> 10kb) or **genomic islets** (< 10kb) (Kung *et al.*, 2010). According to (Mathee *et al.*, 2008) an RGP is defined as a section of at least four contiguous CDSs not belonging to the core genome. Kung *et al.* (2010) further define a genomic island as a "horizontally acquired genetic element present in the chromosome of some strains but absent from closely related strains." Thus, the location of an RGP is the same in different strains of the same bacterium, but can contain different genetic material (Kung *et al.*, 2010).

For pan genomes, a distinction is made between open and closed (Medini *et al.*, 2005). The whole closed pan genome can be identified by only sequencing a few members of a species. E.g. (Tamas *et al.*, 2002) discovered an "extreme genome stability" among two *Buchnera aphidicola* strains which did not undergo any genome rearrangements or gene acquisitions.

Thus, the pan genome of *Buchnera aphidicola* seems to be closed. In contrast, the analysis of (Medini *et al.*, 2005) proposed an open pan genome for *Streptococcus agalactiae*. Each newly sequenced strain added many new genes to the pan genome. The real-world pan genome of a species with a closed pan genome can thus be estimated after sequencing sufficient representatives of that species. Sequencing of more representatives from species with an open pan genome on the other hand will always reveal a certain amount of new genes. The number of new genes per genome varies vastly depending on the examined species and its general genome size.
Besides comparisons on the level of whole genes, also variation on the nucleotide level in the form of single nucleotide and deletion-insertion polymorphisms (SNPs and DIPs) can be used to deduct accurate phylogenetic relationships and identify the origin of pathogenicity factors (Pandya *et al.*, 2009) (see Sections 2.7.1). A combination of global genome rearrangements and local nucleotide mutations enables an in-depth picture of evolutionary changes in an organism.

### 2.5.2. Application Areas

The wide application area of comparative genomics comprises medical, industrial and fundamental research.
In medical microbiology research, the main interest lies in the comparison of pathogenic and non-pathogenic organisms. This enables the identification of virulence factors and antibiotic resistances, which, in turn, aids targeted drug design and reverse vaccinology. In this favor, genomic comparisons of related bacterial strains in order to identify pathogenic and virulent traits have been carried out as soon as enough sequence material was available (Bolotin *et al.*, 2004; Eppinger *et al.*, 2004; Brzuszkiewicz *et al.*, 2006). Due to the rapidly decreasing sequencing costs, analyses on the single nucleotide level for point mutations within many *de facto* identical outbreak strains became affordable (Niemann *et al.*, 2009; Ford *et al.*, 2011). They allow revealing the mutations responsible for enhanced or decreased fitness and prevalence. Access to multiple genomes of a pathogen unfolds all its potential antigens and enables prioritization in reverse vaccinology. Potential widely spread and potent antigen targets are selected via comparative bioinformatic analysis, engineered in the laboratory, and subsequently tested in preclinical and clinical studies (Tettelin, 2009; Sette and Rappuoli, 2010). Several *in silico* tools are available for reverse vaccinology, e.g. the online databases VIOLIN (Vaccine Investigation and Online Information Network) (Xiang *et al.*, 2008) and Vaxign (He *et al.*, 2010). The benefit of this relatively new approach is evident in several studies (De Groot and Rappuoli, 2004; Giuliani *et al.*, 2006; Liu *et al.*, 2009).

For the industry, bacteria play an important role in producing valuable biological compounds. They are genetically engineered either to be able of producing a certain compound or to increase their yield (S. Y. Lee *et al.*, 2005). Two important examples are insulin producing *Escherichia coli* (Goeddel *et al.*, 1979) and amino acid producing *Corynebacterium glutamicum* (Rückert *et al.*, 2003). Researchers study entire strains for beneficial genes or compare production strains on the single nucleotide level to identify SNPs leading to desired behavior (Ohnishi *et al.*, 2002). Studies often aim to identify candidate genes of a product relevant pathway for modification (Yukawa *et al.*, 2007; S. J. Lee *et al.*, 2005). These genes are then engineered for an increasing yield of the desired substance.
As already mentioned at the beginning of Section 2.5, the most important research targets in fundamental research are the reconstruction of phylogenetic relationships and the understanding of the mechanisms driving evolution.

### 2.5.3. Comparative Genomics Software

Several web portals and applications are available for carrying out different comparative genomics analyses. Among them are VISTA tools (Frazer *et al.*, 2004), Panseq (Laing *et al.*, 2010), CoGe (Lyons *et al.*, 2008), and SEED (Overbeek *et al.*, 2005). These tools are not suited for the context of this work, since it focuses on the pan genome analysis of 20 yet unsequenced *P. aeruginosa* strains. All these tools either do not allow submission of custom genomes, or the amount of genomes is too limited in number and size. Additional software solutions for extensive pan genome analyses exist and are shortly introduced and their feature set is compared in this section.

- The **Microbial Genomes Database (MBGD)** (Uchiyama, 2003) is a web service mainly providing analyses for integrated genomes, but it allows uploading custom genomes as well. Currently, 2823 genomes from 742 microbial organisms are maintained in the MBGD[8]. Supported comparative analysis functions for selected genomes are orthologous gene clustering and multiple alignment of these gene clusters, circular genome plots, a region viewer displaying the neighborhood of a selected gene in multiple genomes and a pan genome table with different filters. The filters e.g. enable access to the list of core and singleton genes. Unfortunately, the MBGD user interface is quite circumstantial. Several features are hidden away from the main menu. They only become available on result pages after other analyses have been started first and are often not found intuitively.

- **Sybil** (Crabtree *et al.*, 2007) is different from the other tools, because it is a software package for creating a comparative genomics web server. Thus, using Sybil requires knowledge of how to set up a web server, databases and install all of the several required software libraries. Sybil is script based and offers functionality for protein or gene cluster search within multiple genomes, statistical pan genome analysis listing e.g. the expected sizes of the core genome and unique genes per genome, a region viewer showing the neighborhood of a gene for multiple genomes, synteny gradient plots, a linear genome plot including genomic features, and a shared cluster matrix for the core genome of a project.

- **MicroScope** (Vallenet *et al.*, 2013) is a web service for automatic genome annotation (see Section 2.4), as well as comparative genomics. It offers publicly available genomes as well as private projects for combined analysis of private and public data. The set of features consists of six main features:

    1. Gene phyloprofile: A feature to list unique and dispensable genes of max. 60 selected genomes

    2. Regions of genomic plasticity: A detection of RGPs within a single selected genome in comparison to other genomes via three steps.

    3. Line plot: A synteny plot for comparison of two genomes.

    4. Gene fusion/fission: An analysis to detect gene fusions and fissions within a single selected genome. It calculates a list of candidate genes and from the synteny results of all genomes contained in the MicroScope database for these two events.

[8] Last accessed on 13.04.2015

5. PkGDB synteny statistics: Comparison of a single genome to all other genomes in their prokaryotic genome database (PkGDB). The result lists all available genomes and e.g. contains the number and percentage of genes with bidirectional best hits and in synteny groups.

6. Pan and core genome: Calculation of the pan and core genome for a set of genomes. This analysis is based upon gene families computed with SiLiX (Miele *et al.*, 2011) beforehand.

- **EDGAR** (Blom *et al.*, 2009) is a web service offering public or private comparative genomics projects. EDGAR employs an automatically calculated global orthology threshold: Score ratio values (SRVs) (Lerat *et al.*, 2003) are calculated for all genes of each genome and an SRV master cutoff is automatically derived for the whole project. Afterwards, all genomes included in the project can be analyzed with 11 main features:

  1. Score ratio value plots: This feature shows the plot for the SRV master cutoff of the current project and the SRV plot of two selected genomes.

  2. Pan and core genome: Shows a list of the pan or core genome genes of an arbitrary number of selected genomes from the current project. These lists can be exported as gene tables, multiple DNA fasta or multiple amino acid fasta files.

  3. Singletons: Shows a list of all unique genes of a single genome in comparison to all other selected genomes.

  4. Venn diagrams: This feature enables visual comparison of up to 5 genomes using Venn diagrams. The Venn diagram shows the distribution of genes in core, unique and dispensable genes.

  5. Calculate gene sets: It calculates the gene sets of multiple genomes. Each genome of the current project can either be included or all its genes can explicitly be excluded from the list.

  6. Core and singleton development plots: This feature plots the development of the core genome or the number of unique genes for a selection of genomes.

  7. Synteny plots: A synteny plot can be drawn for multiple genomes, enabling simultaneous synteny analysis of multiple genomes.

  8. Comparative viewer: A region viewer to compare the neighborhood of either a gene of interest or the same genomic position of multiple selected genomes.

  9. Create AAI matrix: An average amino acid identity (AAI**)** matrix is calculated for the selected genomes.

  10. Phylogenetic tree: The phylogenetic tree of all genomes included in the current project is shown by default. Custom sub-trees can be calculated and all trees can be exported into newick[9] or nexus (Maddison *et al.*, 1997) tree files. The

[9] http://evolution.genetics.washington.edu/phylip/newicktree.html, last visited on 22.04.2014

phylogenetic tree calculation is either based on the amino acid or nucleotide sequences of all annotated genes of the core genome.

11. Define replicon group or metacontig: This feature enables creation of either a replicon group, e.g. for combining a bacterial chromosome and the corresponding plasmids, or a metacontig. Separate metacontigs are stored for the pan and the core genome. A core genome metacontig contains all core genes taken from a selected reference genome. A pan genome metacontig contains all core, dispensable and unique genes from a selected reference and all additional dispensable and unique genes of the other included genomes. The replicon groups and metacontigs can subsequently be used in analysis functions.

Features offered by all tools are multiple alignments and multi genome comparisons. Another popular feature supported by three of the four tools (Sybil, MicroScope and EDGAR) are synteny plots. However, EDGAR is the only tool with a synteny plot capable of plotting multiple genomes. All tools except Sybil feature analysis of genomic subsets. A viewer for orthologous regions on the other hand is contained in all tools, except MicroScope. Statistical development of genomic subsets is only featured by Sybil and EDGAR, while gene clustering analyses are exclusive to MBGD and Sybil. Collaborative work of multiple scientists from different geographical locations is only supported by MicroScope and EDGAR. Two unique features of MicroScope are RGP and fusion and fission detection. Nonetheless, EDGAR is the tool providing the most analysis functions from one source, such as genomic subsets including their statistical development, phylogenetic trees, visual genome comparisons especially of orthologous regions and collaborative work for all involved scientists. Additionally, the use of a global cutoff value for the assignment of orthologous genes via SRVs is another advantage of EDGAR facilitating tailored treatment of single genomes as well as uniform treatment of the collectivity of genomes in a project.

## 2.6. Mapping of Short NGS Reads

Short read mapping is the process of aligning a set of sequencing reads to an already known reference sequence with a given error threshold. Errors comprise mismatching, inserted and deleted bases between the reference and the mapping data set. Each error class is assigned a certain cost and maximum error thresholds for accepted alignments are set before starting the read mapping process. Mainly two approaches are used in modern read mapping software: Either seed based algorithms utilizing hash tables or algorithms based on the Burrows-Wheeler transform (BWT) (Burrows and Wheeler, 1994). The output of a read mapper is called **read mappings**, or in short **mappings**. This section gives an insight into the application areas of short read mapping, both mapping approaches, the variety of available mapping tools and commonly used data formats.

### 2.6.1. Application Areas

Short read mapping has extensive application areas and is an inherent part of NGS data analysis pipelines. Short read mapping aids detection of small genetic variants (SNPs and DIPs), genome re-sequencing, RNA-seq (Wang *et al.*, 2009), ChIP-Seq (Johnson *et al.*, 2007), TAG sequencing (Porter *et al.*, 2006), metagenomics (Kunin *et al.*, 2008), and metatranscriptomics (Simon and Daniel, 2011). It has also proven useful for detection of

genome rearrangements, especially when read pairs have been sequenced (Skovgaard *et al.*, 2011).
Before starting a mapping, the allowed mismatch and insertion and deletion (indel) rates have to be defined. They should be set according to the expected error rate of the employed sequencing method and the expected similarity between sample and reference. The error rate of Illumina and 454 sequencing is around 1% (Minoche *et al.*, 2011; Gilles *et al.*, 2011). The highest error rate of single reads is observed in SMRT sequencing with ~15% (Carneiro *et al.*, 2012).
One approach in re-sequencing is to determine the consensus sequence of a yet unknown genome by mapping reads onto an available reference genome of a close relative. All positions with a sufficient amount of reads exhibiting the same mismatch or indel define the part of the consensus sequence which is characteristic for the novel genome. Further, an analysis of the coverage reveals the intervals and genes present and absent in the novel genome. Naturally, additional genes, genomic islands and RGPs contained in the novel genome cannot be identified by mapping to a reference missing these elements. For already known genomic islands and RGPs with existing DNA sequence data, an additional mapping to these references can be created and their coverage can be analyzed for completeness and presence of genes. Additionally, genome rearrangements of the re-sequenced genome can be identified when read pairs are used and both reads of a pair map to other regions of the reference than expected. SNPs and DIPs (see Section 2.7.1) are called in the same way as the consensus sequence is constructed (Nielsen *et al.*, 2011). Just that in this case only the positions with a sufficient amount of deviating bases are of interest. Further, many experiments can benefit from closely analyzing the coverage of the reference genome. It is often useful to examine genomic regions showing certain coverage characteristics.
For RNA-seq, mapping facilitates the detection of gene expression levels under different environmental conditions (see Sections 2.8.3 and Section 2.8.1), transcription start sites (see Section 2.8.2), novel transcripts (see Section 2.8.2), transcripts of small RNAs (see Section 2.8.2) and operon structures (Passalacqua *et al.*, 2009) (see Section 2.8.4). Beforehand of sequencing, the RNA needs to be reversely transcribed into cDNA. Subsequently, the sequenced reads can be used for the above mentioned applications. A distinction has to be drawn between prokaryotic and eukaryotic RNA-seq. Mapping eukaryotic RNA-seq data back on the genome sequence demands a splice aware alignment tool. The sequenced RNA only contains the exon sequences of the genes. Hence, the alignment software needs to be capable of mapping reads partly if the read spans an exon junction.
For the identification of genome wide protein binding sites, mapping is a vital component of chromatin immunoprecipitation sequencing (ChIP-seq) (Johnson *et al.*, 2007) - a combination of ChIP with NGS. Protein-DNA interactions are an essential process for gene expression and regulation and epigenetic chromatin modifications, i.e. histone modifications and methylation (O'Geen *et al.*, 2011). ChIP-seq enables analysis of all these functions and is composed of several steps: At first, the sample DNA is sheared by sonication *in vivo*, and then all protein-DNA complexes contained in the cell lysate are isolated. Next, selective immunoprecipitation for a protein of interest and its bound DNA is performed by corresponding antibody carrying beads. This step yields all DNA sequences to which the protein can bind. At this point, high-throughput sequencing is employed in ChIP-seq to sequence the DNA samples. The obtained reads are mapped to the corresponding genome.
Thus, if a reference genome is available, this technique is more sensitive than the older alternative ChIP-on-chip method (Aparicio *et al.*, 2004), because it is not bound to predefined probes. Additionally, ChIP-seq indicates the affinity of a binding site (Jothi *et al.*, 2008).

### 2.6.2. Seed-And-Extend Approach

Seed-and-extend based alignment algorithms have already been used at the beginning of the 1990s for DNA alignment in BLAST (Altschul *et al.*, 1990) and are composed of two phases. In the first phase, the seeding phase, (nearly) exact matches of short substrings - the seeds - of sequencing reads are detected within the reference. In the second phase, the extension phase, the seed matches are extended to the complete read sequence. Only matches complying with the given error thresholds are kept.
Two different techniques for seeding exist: The naive approach makes use of continuous seeds, while the more sophisticated and sensitive approach uses spaced seeds. A spaced seed allows a given number of internal mismatches. Parameters varying in the implementations of both techniques are the number of seed matches required and the choice of the seed length. The extension of candidate matches in the second phase up to the total length of the original read, while adhering to the predefined error threshold, is often done by dynamic programming approaches. Three techniques significantly increasing the performance of read extension are worth noting. One technique is vectorization of the standard Smith-Waterman algorithm (Farrar, 2007). This is achieved by utilizing widely spread CPUs capable of "single instructions, multiple data" (SIMD), which allows parallelizing the alignment step. The second technique is the confinement of dynamic programming around already detected seeds from the seeding phase (Eppstein *et al.*, 1990; Slater and Birney, 2005). The third technique is to utilize the observation of (Myers, 1986) that a string of length $a$ can be aligned to a substring of a target with length $b$ with a maximum of $D$ errors in $O(ND)$ time and space, where $N := a + b$.
Inherently, both seeding techniques do not support gaps in the seed. But gaps play an important role, especially in variant discovery (Krawitz *et al.*, 2010). Three techniques are available to overcome this problem. Gaps can be identified during the extension phase by dynamic programming, by introducing small gaps at each read position (R. Li *et al.*, 2008), or - the most popular approach - by utilizing a different, very efficient technique combining seeding and extension in just one phase, the *q-gram filter* (Rasmussen *et al.*, 2006).
A *q-gram* is simply a string of length $q$. The q-gram filter is based on the *q-gram lemma* (Jokinen and Ukkonen, 1991):

**q-gram lemma:** For a query sequence (e.g. a read) and a template sequence (e.g. a substring of a reference) of length $l$ with at most $e$ errors exist at least $(l + 1) - (e + 1)q$ valid matches of length $q$.

Thus, one error depletes the total number of valid q-grams by $e*q$. When only the $\frac{l}{q}$ non-overlapping q-grams are considered, the *pigeonhole principle* can be applied, because one error can only decrease the number of valid q-grams by 1. The pigeonhole principle, first mentioned by the German mathematician Dirichlet in 1834, states that, when $n$ items are placed in $m$ containers, with $n > m$, then at least one container holds more than one item. In terms of read mapping, this observation permits the fast identification of candidate regions for a valid match of a read. Only a reference region in which enough valid q-grams complying with the given error thresholds have been identified for a read can constitute a valid match to the reference. A solution for reads which have a non-empty remainder $r$, when dividing them by a given $q$, is to shift the start of all seeds by $r$ and start another query against the reference. Correlating both queries of the read reveals if a valid match is possible or not (Blom *et al.*, 2011).

## 2.6.3. Burrows-Wheeler Transform Approach

Mapping algorithms based on the BWT (Burrows and Wheeler, 1994) are mainly split into two phases: The first phase is an alternative seeding technique. Exact mappings of seeds are identified in this phase. In the second phase, the inexact alignments with mismatches and indels are calculated.

The basis of algorithms applying BWT in the first phase is a representation of a prefix or suffix trie (see Figure 21). An important advantage of a trie is that all identical substrings of the input string are collapsed into a single branch of the trie. Thus, identical substrings only need to be aligned once to identify all potential alignment positions. The most widely used algorithms all make use of the "Full text index in Minute space" (FM-index) (Ferragina and Manzini, 2000, 2005) based on the BWT.

The BWT is generally a permutation algorithm for the characters of a string, which was mainly applied in the field of text compression. Text transformed by BWT can be compressed very efficiently due to the fact that duplicated strings are easier to compress than unique strings. BWT permutes the input string and places identical - duplicated - runs of characters next to each other in the output string. Another very important property of BWT is that it is reversible without the need to store additional data structures. This fact makes it very useful for exact string matching, because a query string contained in the BWT transformed template can be discovered by a backward search.

As already mentioned, BWT alone is not a compression algorithm and thus not memory efficient enough for a complete human genome. Therefore it has been coupled with the FM index which enables a compressed full-text substring index. The advantage of this dovetailing is that it allows fast access to substrings of the text, although the text is efficiently compressed after applying BWT. The basis of an FM index is formed by a *compressed suffix array* (see Figure 21) combining the text compressed using BWT and auxiliary data structures permitting full-text index search.

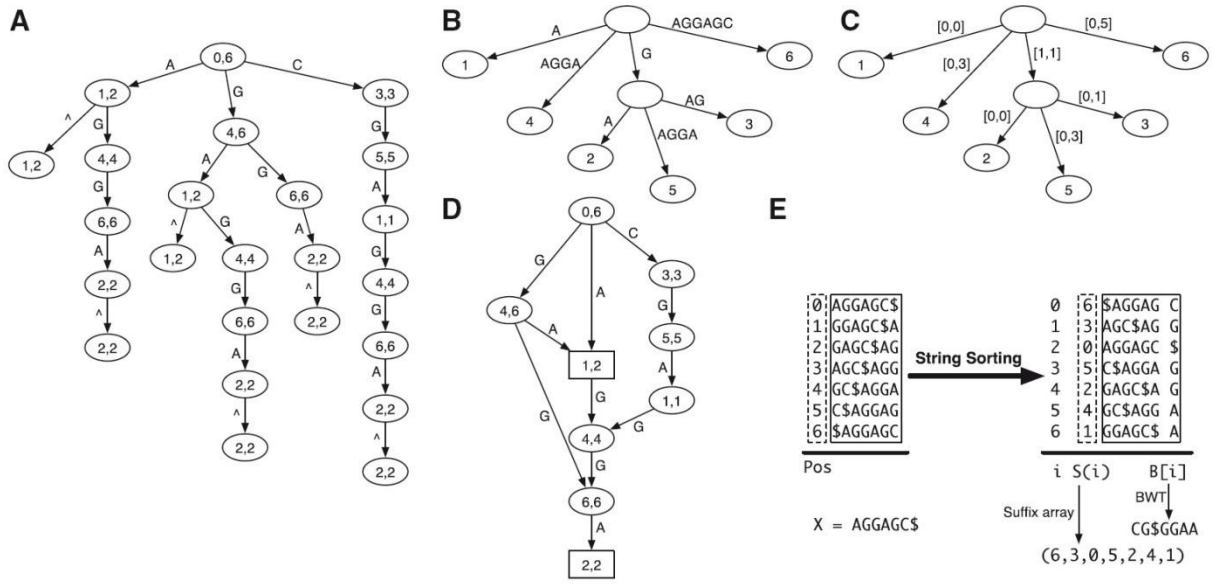



**Figure 21: Data structures based on a prefix trie**. (A) Prefix trie of string AGGAGC where symbol OE marks the start of the string. The two numbers in each node give the suffix array interval of the substring represented by the node, which is the string concatenation of edge symbols from the node to the root. (B) Compressed prefix trie by contracting nodes with in- and out-degree both being one. (C) Prefix tree by representing the substring on each edge as the interval on the original string. (D) Prefix directed word graph (prefix DAWG) created by collapsing nodes of the prefix trie with identical suffix array interval. (E) Constructing the suffix array and Burrows-Wheeler transform of AGGAGC. The dollar symbol marks the end of the string and is lexicographically smaller than all the other symbols. The suffix array interval of a substring W is the maximal interval in the suffix array with all suffixes in the interval having W as prefix. For example, the suffix array interval of AG is [1, 2].The two suffixes in the interval are AGC$ and AGGAGC$, starting at position 3 and 0, respectively. They are the only suffixes that have AG as prefix. Figure and description are taken from (Li and Homer, 2010).

The time complexity for identifying an exact match of a query to a trie or its FM-index representation is linear in time and space in terms of the query length. Indexing the whole human genome of ~3.2gb only uses 1.3 gigabyte (GB) of RAM in an efficient implementation (Langmead *et al.*, 2009).

### 2.6.4. The Variety of Mapping Tools

More than 60 different short read mapping tools have been developed to date. A list containing many of these tools can be found on Wikipedia[10]. The software diversity arose due to the indispensable need for reliable and accurate read alignments for all downstream tasks. A further contribution to this development is the fact that efficient mapping of many millions of reads in a reasonable amount of time and space is not a trivial task. Most of the mappers output their results in the *de facto* standard for read mapping data, the sequence alignment/map (SAM) format (H. Li *et al.*, 2009). However, only a handful of mapping tools are widely in use around the globe. In the context of this work, the two most important of these mappers are BWA and Bowtie 2. Both aligners employ heuristics and thus do not find all possible alignments under the given parameters (Blom *et al.*, 2011). To be able to identify all possible alignments, the exact and efficient mapping tool SARUMAN (Blom *et al.*, 2011) has been chosen. All three mappers support mismatches and gaps and the output of multiple alignments per read. In the following a brief description of the three aligners is given:

- **BWA** (Burrows-Wheeler aligner) (Li and Durbin, 2009) is a heuristic mapping tool based on backward search after BWT on an FM-index. BWA samples all distinct substrings from the mimicked prefix trie with an edit distance $\leq$ the maximal allowed edit distance. The main features of BWA include support of read pairs and mapping quality output for each read. The output is written in SAM format.

- **Bowtie 2** (Langmead and Salzberg, 2012) is a heuristic mapping tool combining backward search in the FM-index based on BWT with dynamic programming approaches for extending the exact matches. The extension phase is implemented to benefit from parallel processing via SIMD (see Section 2.6.2). The main features of Bowtie 2 include support of read pairs and mapping quality for each read. The output is written in SAM format.

- **SARUMAN** (Semiglobal alignment of short reads using CUDA and Needleman-Wunsch) (Blom *et al.*, 2011) is an exact and complete mapping tool returning all possible alignments of a read to the reference complying with a given error threshold. The phase for identifying exact matches is driven by a q-gram and pigeonhole principle based filter algorithm. The extension phase massively parallelizes the alignment step on the graphics processing unit (GPU) of CUDA (Compute unified device architecture) graphic cards to speed up the overall alignment process of a data set. The alignments in this phase are computed by a modified Needleman-Wunsch algorithm (Needleman and Wunsch, 1970). The main advantage of SARUMAN is its completeness while running in time competitive to the other tools. Read pairs are not yet supported, but read mappings can be post-processed (e.g. by SAMtools (H. Li *et al.*, 2009) or as described in Section 5.2.2, Read Pair Classification). It is noteworthy that SARUMAN requires all input reads to have the same length. This is inevitable for the parallelization step on the CUDA graphic card.

[10] http://en.wikipedia.org/wiki/List_of_sequence_alignment_software

The biological case study in this work deals with bacteria, but as already mentioned in the Application Areas Section 2.6.1, mapping of eukaryotic RNA-seq data requires specialized, splice aware alignment tools. For the sake of completeness, three widely used tools are pointed out here. The most famous tool is **TopHat** (Kim *et al.*, 2013, 2), a splice junction mapper based on the aforementioned Bowtie 2 (Langmead and Salzberg, 2012). Two further favorable splice junction mappers are **GSNAP** (Wu and Nacu, 2010) and **STAR** (Dobin *et al.*, 2013), both introducing their own novel mapping algorithm.

## 2.7. Re-sequencing Mapping Analyses with NGS

Genome re-sequencing often aims at tracing the course of genome evolution on both local and global level. SNPs and DIPs on the one hand are small scale variations mutating a single or a handful of base pairs. Genome rearrangements or structural variations on the other hand are global changes of the genome sequence. Besides revealing evolutionary relationships, the analysis of genome mutations can shed light on presence or absence of important gene variants, whole genes or genomic regions. In general, mutations can be classified according to their fitness effect. They can reduce the fitness of the organism, be lethal, neutral or beneficial. The distribution of these types of mutations, however, varies largely among different organisms (Eyre-Walker and Keightley, 2007). In research concerned with bacterial pathogenicity, these analyses play an important role for the classification of the hazardousness of a strain. The following section introduces methods for analyzing both scales of genome evolution and available analysis tools.

### 2.7.1. Single Nucleotide and Deletion-Insertion Polymorphisms

Despite their size, small DNA sequence variations can have huge impact on functions of genes as well as regulatory elements. Three types of single nucleotide variation can occur: Substitution of a single DNA base by another, insertion of a new base or deletion of an existing base. A single nucleotide alteration can lead to a completely different gene product or regulatory RNA or even loss-of-function. Thus, different phenotypes may be caused by single nucleotide polymorphisms (SNPs) or deletion-insertion polymorphisms (DIPs). Since acquired mutations are passed on to progeny, they provide an insight into evolutionary relationships, allowing the reconstruction of phylogenetic trees based on a SNP and DIP analysis (Pandya *et al.*, 2009).
The mentioned characteristics of small sequence variations facilitate identification of the origin of pathogenicity factors (Pandya *et al.*, 2009) as well as mutations leading to desired behaviour or phenotypes (Ohnishi *et al.*, 2002).
In the following, the shorter "SNP detection" is used synonymously to "SNP and DIP detection".
Reference coverage is decisive for reliable SNP detection. Deep sequencing with a coverage > 15-20 provides clearly more reliable results than shallow coverage. In the latter case sequencing errors or wrongly mapped reads can easily lead to false positive SNPs (Nielsen *et al.*, 2011; Kosugi *et al.*, 2013). For eukaryotes with a huge genome, such as human, deep sequencing of the complete genome is still expensive. Thus, researchers have to face the shallow coverage problem along the whole genome. Bacterial genomes are much smaller, and thus nowadays they are mostly sequenced at higher depth; just as the *P. aeruginosa* strains examined in this work (see Section 6.2.4).

Besides, this work focuses on "SNP detection" instead of "genotype calling". The first describes the identification of variable sites in a genome, while the latter denotes the assignment of explicit genotypes of individuals to the identified sites. The standard format for bioinformatical SNP and genotype calling used by most tools working with this kind of data is the Variant Call Format (VCF) (Danecek *et al.*, 2011).
SNP detection is influenced by many different factors: Base calling, base quality, applied mapping algorithm, mapping quality, allele frequencies in case of polyploid organisms and proximity to indels and clustering of SNPs leading to alternative mapping possibilities at the same genomic position. Base calling errors should be reflected by a lower base quality and can be filtered by appropriate thresholds. Mapping quality varies among different mapping tools, but should generally adhere to the meaning of PHRED scores (see Section 2.1 and (Ewing and Green, 1998)). The most probable mapping of a read at a possible variant site can be analyzed by local realignment. Especially around possible indels and clusters of SNPs, a local realignment can clearly reduce false positive SNPs (Kosugi *et al.*, 2013).
If the error rate of the sequencing technology and the read mapping is known and additional information such as allele frequency and patterns of linkage disequilibrium (LD) are available, this information can be used for a probabilistic Bayesian approach. In this case, prior probabilities are established, which can represent for instance the expected frequency of SNPs or the sequencing error probability for a given base quality. Subsequently, posterior probabilities are calculated for all possible variant sites. Only sites indicating a SNP by a significant posterior probability are kept.
Several tools for the computational identification of SNPs and DIPs in NGS data have been developed in the past years. They mainly apply three different algorithmic approaches:

1. **Empirical filter parameter estimation** including coverage, base quality, neighbourhood quality standard (NQS) (Altshuler *et al.*, 2000), mapping quality and allele frequency in case of polyploid organisms. Tools based on this approach include: Coval (Kosugi *et al.*, 2013), VarScan (Koboldt *et al.*, 2009) and GenomeComb (Reumers *et al.*, 2012).
2. **Probabilistic Bayesian approach** determines genotype likelihoods based on prior probabilities incorporating knowledge about errors from base calling and read mapping, allele frequency and patterns of LD. SNP results enclose a statistically derived quality score. Tools based on this approach include: MAQ (H. Li *et al.*, 2008), SOAPsnp (R. Li *et al.*, 2009), GATK (McKenna *et al.*, 2010), SeqEM (Martin *et al.*, 2010), Slider II (Malhis and Jones, 2010), SAMtools (Li, 2011) and SAVI (Trifonov *et al.*, 2013). An alternative probabilistic approach is employed by SNPseeker (Druley *et al.*, 2009). This tool uses large deviations theory to probabilistically detect SNPs.
3. **Machine learning approach** based on already available reliable training data. Fitting of training data can be done by logistic regression. Tools based on this approach include: GATK (alternative algorithm) (DePristo *et al.*, 2011), Atlas-SNP2 (Shen *et al.*, 2010) and ProbHD (Hoberman *et al.*, 2009).

Many tools additionally perform local realignment at possible variant sites regardless of the employed algorithm family to filter or improve the quality of SNP calls (Kosugi *et al.*, 2013; DePristo *et al.*, 2011; H. Li *et al.*, 2009).

### 2.7.2. Genome Rearrangements

In contrast to the small scale of SNPs, large scale evolution on the genomic level is driven by structural variations (SVs) rearranging parts of the genome. Genome rearrangements can insert or delete genomic regions or change their order, orientation and position. The possible

rearrangement events comprise insertions, duplications, deletions, inversions, transpositions, block interchanges, fusions and fissions (see Figure 22). The genomic content of regions undergoing rearrangements can be classified as described in Section 2.5.1. For example the content of an insertion belongs to the dispensable genome, as it is not present in all genomes under investigation.

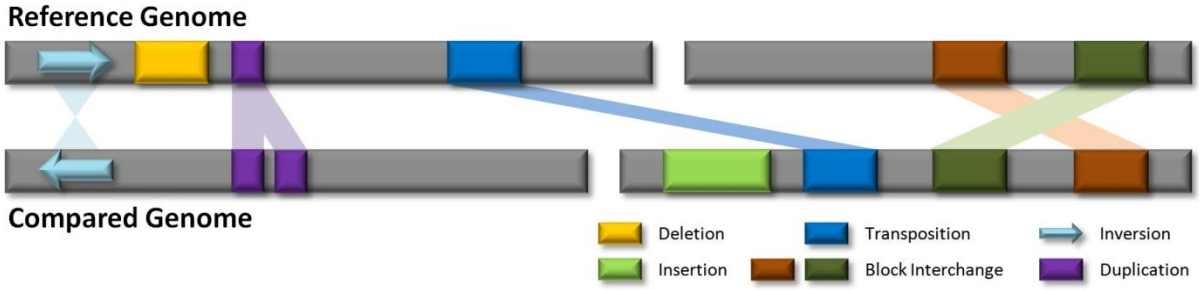



**Figure 22: Genome Rearrangements.** Possible genome rearrangement events are illustrated in a schematic comparison of two genomes, both consisting of two chromosomes (grey bars). The event types are explicated in the legend. Conserved genomic regions present in both genomes are interconnected by shaded colored shapes.

An alternative to synteny analysis on the gene level is to utilize read pair data for the detection of genome rearrangements via read mapping. This technique became affordable due to the drastic decrease of sequencing costs (see Section 2.1). Read pair mapping of sufficiently large NGS data sets enables high-resolution insight into the exact break points of genome rearrangements (Fullwood *et al.*, 2009). One of the first research areas in which this technique was applied is human cancer research (Campbell *et al.*, 2008). At best, rearrangement sites between two organisms (which can be two human individuals, bacterial strains or other sufficiently related organisms) can exactly be determined by read pair or split read mapping with sufficient coverage.
Read pairs from paired end or mate pair data possess an expected distance and orientation. Observed values for these properties violating the expected values for the given library indicate genome rearrangements. Generally speaking, the higher the genome coverage and the more read pairs support the same rearrangement event, the more reliable a rearrangement can be identified correctly. Thus, rearrangements are found in genomic regions with a significant number of discordant read pairs. The type of the rearrangement event depends on the characteristics of the discordance (compare Figure 22).

- **Insertion:** This event inserts a new region into the mapped genome. The read pair distance is too small (for sufficiently long pairs) and/or sufficient pairs with only one mapped mate are observed in the respective region. In this case, a *de-novo* assembly (see Section 2.3) can reveal the sequence of the insertion.
- **Deletion:** This event deletes a region from the reference genome in the mapped genome. The read pair distance is too large and no reads can be mapped to the interjacent region deleted in the mapped genome.
- **Transposition:** This event moves a region of the reference genome to another position/chromosome in the mapped genome. The distance of the pairs around the breakpoints is too large. Depending on the size of the transposed region, concordant pairs might be found in between the boundaries of the region. Transpositions can be classified as intra- or inter-chromosomal, depending on the chromosome on which the mapped mate is located.
- **Block interchange:** It is an event exchanging two regions of the reference genome with each other in the mapped genome. Read pairs with a too large distance will be observed around all four breakpoints of a block interchange (see Figure 22). Just as transpositions, block interchanges can be classified as intra- or inter-chromosomal.

- **Inversion:** This event inverts a region of the reference in the mapped genome. The read pair distance is too large and the orientation of the reads towards each other is discordant. Depending on the size of the inversed region, concordant pairs might be found in between the boundaries of the region.
- **Duplication:** This event duplicates a region of the reference genome in the mapped genome. One part of the read pairs is concordant also around the breakpoints of the duplication, while the other part of pairs around the breakpoints is discordant. The latter fraction originates from a different genomic region, in which the duplicated region has been inserted. Thus, the region duplicated in the mapped genome is marked by significantly more reads mapping uniquely to the region.

Several software tools adopting different approaches are available for identifying SV events using read pair data. Popular examples are: GASV (Sindi *et al.*, 2009), SVMiner (Hayes *et al.*, 2012), BreakDancer (Chen *et al.*, 2009) and SVDetect (Zeitouni *et al.*, 2010). GASV (Geometric Analysis of Structural Variants) uses a geometric approach representing uncertainties in the prediction of a SV as a polygon in the plane. Supporting measurements for a variation are identified by computing intersections of these polygons by a computational geometry algorithm. SVMiner at first generates candidate SVs and then validates them by a model-based clustering approach. BreakDancer uses a different clustering approach based on user defined thresholds. SVDetect includes two approaches: A clustering and a sliding window approach. Apart from the four mentioned tools other programs specialized only on single SV types are also available (e.g. MoDIL (Lee *et al.*, 2009) and AggloIndel (Wittler, 2013) for insertions and deletions only). Results of these tools are mainly the type of rearrangement, its coordinates and the estimated size of the rearranged genomic regions.

## 2.8. RNA-seq Mapping Analyses using NGS

The transcriptome of an organism or cell denotes its complete set of transcripts and their abundances at a fixed time point under given environmental conditions including all mRNAs, rRNAs, tRNA, miRNAs and other small regulatory RNAs.
The state of the art approach to analyze an entire transcriptome is called **RNA-seq**. It is a cost effective technique utilizing deep high-throughput sequencing after reverse transcribing all isolated RNA into cDNA. This method offers various deep insights into the transcriptional landscape (Wang *et al.*, 2009).
Before the age of massively parallel RNA-seq, microarrays and low throughput sequence based approaches were established. Microarrays containing DNA probes from known genes of interest are incubated for hybridization with fluorescently labelled cDNA in order to determine genes present in the sample (Bertone *et al.*, 2004). The sequence based approaches started with relatively expensive Sanger sequencing before tag-based methods (e.g. SAGE (Velculescu *et al.*, 1995), MPSS (Brenner *et al.*, 2000)) were developed. Several limitations of these methods are largely overcome by RNA-seq and the increasing read length of the employed sequencing technologies: Short probes or tags often hybridize or map to multiple locations of the reference genome. Additionally, microarrays tend to relatively high noise rates, only already known gene sequences can be found, isoforms cannot be distinguished and they have an upper limit of sensitivity for transcript quantification. Another disadvantage of the older methods are their high costs in comparison to RNA-seq.
RNA-seq sheds light on transcriptome complexity including isoforms in eukaryotes and operons in prokaryotes. It allows detailed insights into transcript abundances, offers single base resolution and low background noise. Additionally, no upper quantification limit exists.

Instead, the number of reads correlates with the number of transcripts. A high reproducibility of biological and technical replicates can also be achieved (Mortazavi *et al.*, 2008).
When using RNA-seq, the transcriptional landscape can be analyzed either by mapping the reads to an already known reference genome (see Section 2.6) or by creating a *de novo* transcriptome assembly (see Section 2.3).
Afterwards, the main goals are to catalogue all types of transcripts (see Section 2.8.1) and to quantify transcript abundances under different environmental conditions at fixed time points or development stages. Subsequently, the different conditions are compared and differences and commonalities are identified (see Section 2.8.3). Another goal is the determination of transcript structures. Therefore, transcription start sites (TSS), 3'-ends and UTRs (see Section 2.8.2), operon structures in prokaryotes (see Section 2.8.4) and isoforms in eukaryotes can be identified. Isoforms are not further addressed here, since this work focuses on bacterial research. A valuable introduction to isoform analysis is given in the publication of (Breitbart *et al.*, 1987) about alternative splicing.
Just like in re-sequencing experiments, SNP and DIP detection (see Section 2.7.1) can also be conducted with RNA-seq data for all expressed transcript regions.

### 2.8.1. Read Count & Normalization Calculations

Before calculating normalized read count values or differential gene expression (see Section 2.8.3), the number of reads associated with specific transcripts has to be determined. For unannotated references or when a *de novo* transcriptome assembly is desired for other reasons, this is a challenging task approached by *de novo* transcriptome assembly tools like Velvet/Oases (Schulz *et al.*, 2012), SOAPdenovo-trans (Xie *et al.*, 2014) and Trinity (Grabherr *et al.*, 2011). Details are not discussed here, since *de novo* transcriptome assembly is beyond the scope of this work. But even for already annotated reference genomes this is not a trivial task, requiring insights in the data and the planned downstream analyses.
At first, the transcript boundaries have to be determined and the feature overlap model has to be chosen. The transcript boundaries are mostly chosen as the exact genomic positions of annotated genes of coding and non-coding RNAs. The feature overlap model choice depends on the application and available data. Three different models (see Figure 23) have been proposed by the authors of the popular HTSeq-count tool (Anders *et al.*, 2014) for generating read count data:

1. The *union* model collects the union of all genes overlapping the read.
2. The *intersection_strict* model collects all genes which are overlapping exactly each position of the read.
3. The *intersection_nonempty* model collects all genes which are covering all positions of the read, but also allows partial overlaps as long as no other gene fully overlaps the read.

In HTSeq-count all reads with an empty feature set or with a set containing more than one gene with respect to the chosen model are discarded. Discarding the reads with multiple assigned genes leads to an underestimation of the real read counts. In the worst case, this decision can lead to false negatives if not sufficient other valid reads are mapped e.g. for genes shorter than the read length. Alternative to the three proposed models, a read could be either assigned randomly to one of the genes, to all genes, or its single read count is divided by the number of genes overlapped by the read. The drawback of this model is that it might cause false positives in a subsequent analysis.
Reads mapping multiple times to the reference pose a similar problem. They can be handled by one of the above mentioned models, too. The only difference for the model including

multiple mapped reads is the division by the number of mappings of that read instead of the number of genes.

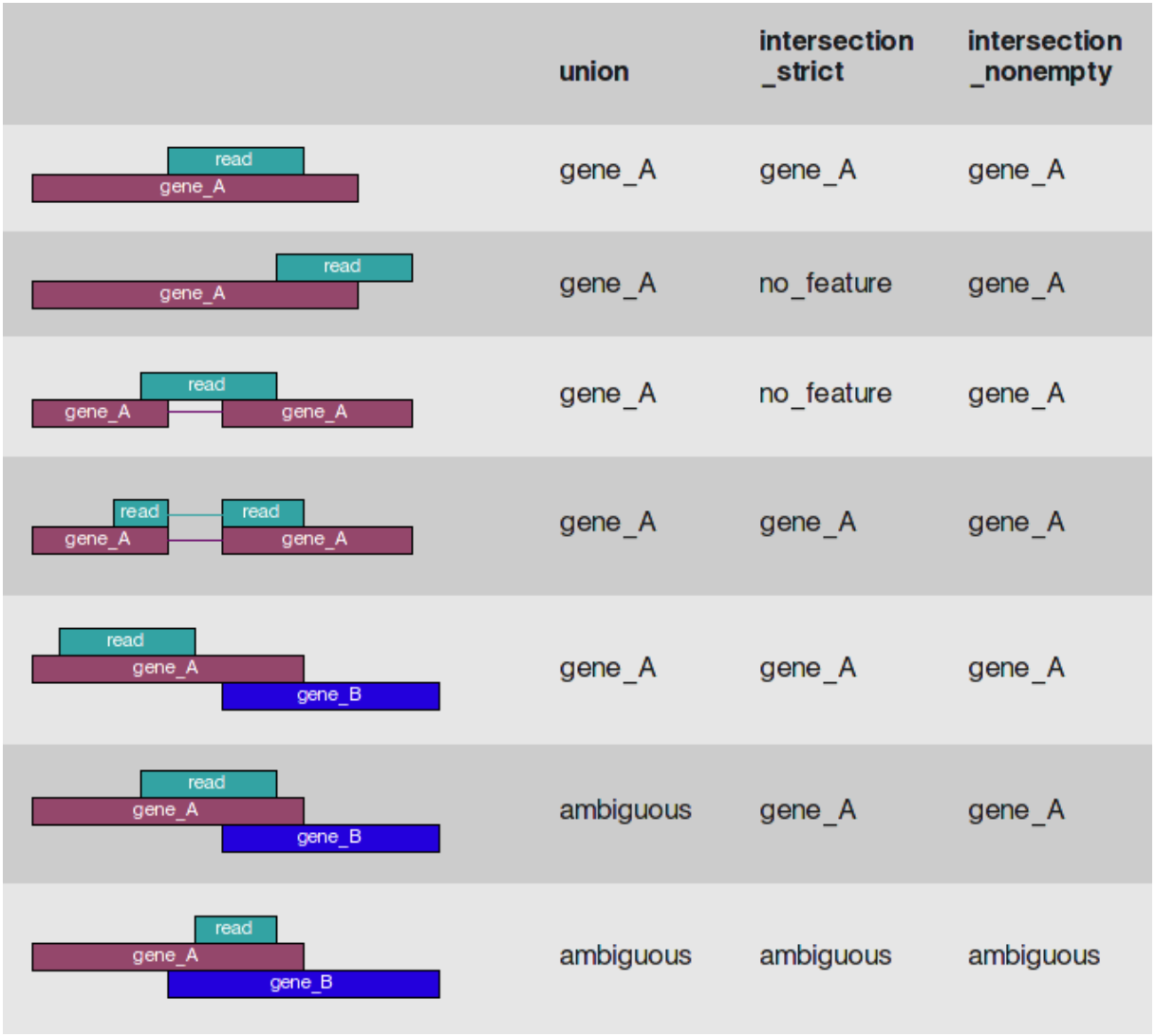



**Figure 23: Effect of three read overlapping models.** The three models are described in the text above. In HTSeq-count (Anders *et al.*, 2014), all reads with more than one gene assigned (*ambiguous*) or with *no_feature* assigned are not counted for any gene. Source: `http://www-huber.embl.de/HTSeq`, accessed on 17.01.2014.

In parallel to this work, an alternative approach to determine transcript boundaries has been proposed by McClure *et al.* (2013). They also start with positions provided by annotated genes and novel transcript seeds. The latter are regions of sufficient coverage of a certain minimum length. Then, the transcript boundaries are extended by a Bayesian approach applied to the mapped reads connected to the seed at both transcript ends. In the final step, their procedure merges adjacent and overlapping extended transcripts if their coverage characteristics are significantly concordant. The advantage of using an approach always starting with transcript boundary estimation is that novel transcripts are directly available for all other downstream analyses like the two methods explicated in the last paragraph of this section.

Normalization of read counts is an important pre-processing step for many RNA-seq data analyses. RNA-seq is less biased than microarrays, but nonetheless there are other biases and variability in the data (Robinson and Smyth, 2007). Additionally, reads are mostly shorter than gene transcripts and can map to multiple genes and isoforms. The same issue can be observed for small ncRNAs whose corresponding short reads are more likely to map to multiple regions of the reference genome than longer reads. It is important to review the

requirements of planned downstream analyses and decide on that basis if pre-processing in the form of sample normalization is needed. E.g. all differential gene expression analysis tools introduced in Section 2.8.3 require raw read counts as input, because they include normalization in their statistical model.
Two widely used and straight forward normalization methods are transcripts per million (**TPM**) (B. Li *et al.*, 2010) and reads per kb of exon model per million mapped reads - short **RPKM** - introduced by (Mortazavi *et al.*, 2008). RPKM is meant to reflect molar concentration of transcripts by normalizing for transcript length and library size. This type of normalization is necessary for reasonable comparisons of transcript abundances both within one and among multiple samples. TPM has been proposed as an improvement to RPKM, because TPM has the advantage of being invariant between samples and species (B. Li *et al.*, 2010) while RPKM values may change when the mean expressed transcript length changes due to different sets of active genes in two samples. To take into account that reads of a given length cannot start at each position in a transcript, the effective length of transcripts $\tilde{l}$ is used for the normalization instead of the whole transcript length *l*. The effective length of transcript *i* has first been defined by (Trapnell *et al.*, 2010) as

$$\widetilde{l_i} = \sum_{x \leq l_i} \lambda_F(x) * l_i - x + 1,$$

where *x* is one of the observed read length values in transcript *i* of length $l_i$, and $\lambda_F$ is the fraction of reads for *i* with length *x*.
The formula for the RPKM value of gene *i* is:

$$RPKM_i = 10^9 * \frac{C_i}{N * \widetilde{l_i}}, \text{ where}$$

*C* is the number of mappable reads for genomic feature *i* (e.g. a gene) and *N* is the total number of mappable reads for the experiment/data set.

$$TPM_i = 10^6 * \left(\frac{c_i}{\widetilde{l}_i}\right) * \left(1 \Big/ \sum_j \frac{c_j}{\widetilde{l_j}}\right), \text{ where}$$

*c* is the number of mappable reads for genomic feature *i* and *j* is an integer ranging from 1 to the number of genomic features of the same type (e.g. genes).

### 2.8.2. Transcription Start Site & Novel Transcript Detection

Transcription initiation at the transcription start site (TSS) is one of the biological key mechanisms defining the function of the cell. Besides mRNA transcripts which are translated into proteins, numerous non-coding regulatory RNAs exist. Also these RNAs are subject to regulation and mostly regulate the expression of other genes as well. In general, gene regions can be classified into a TSS, 5' untranslated region (UTR), translation start site, coding sequence, stop of the CDS and 3'UTR. The identification of TSS is of high importance, as it aids identification of the correct translation start site, the 5'UTR and allows prediction of the respective promoter sequence. Knowing the promoter sequence, more reliable conclusions regarding transcript regulation can be made. Gene analysis also benefits from TSS detection, because it relies on accurate gene annotation. Additionally, genome-wide identification of TSS reveals not only protein-coding genes, but also novel miRNA, small regulatory RNA and antisense genes.

TSSs can be classified into **primary TSSs** - the most prominent TSS for a gene, **alternative TSSs** of a gene, **internal TSSs** - TSSs found inside annotated genes, **antisense TSSs** - TSSs found on the antisense strand of genes and **orphan TSSs** - TSSs indicating novel transcripts with no annotated features within reach.

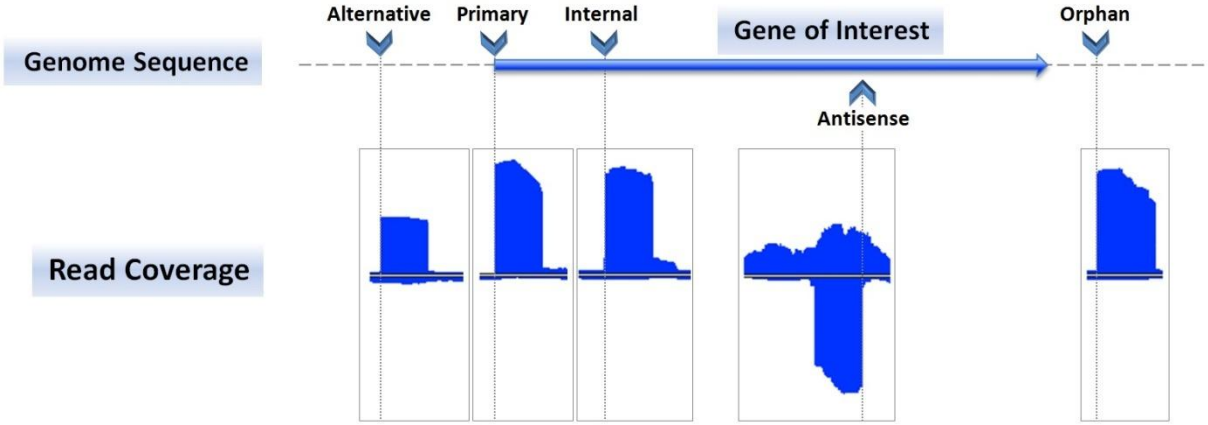



**Figure 24: Transcription start site classification.** A schematic genome sequence is shown (discontinuous horizontal grey line) with a sector containing only a single annotated gene (blue arrow) on the forward strand. Below, read coverage sections of a corresponding RNA-seq data set are shown (blue). Coverage above the center line belongs to the forward strand, while coverage below the center line belongs to the reverse strand. The sharp peak starts depicting a TSS are marked on their respective strand by vertical dotted grey lines connected to the corresponding TSS description. The leftmost TSS is classified as alternative TSS, because the primary TSS is more abundant in the examined data set, i.e. the primary TSS has a higher coverage. The figure was created with the aid of ReadXplorer (see Chapter 5).

In the recent years, the rapidly decreasing sequencing costs enabled NGS as a key tool for analyses regarding transcriptional mechanisms in both eukaryotes and prokaryotes. This research revealed that also prokaryotic transcriptomes exhibit a complex structure (Sharma *et al.*, 2010; Mendoza-Vargas *et al.*, 2009).

For TSS analysis in eukaryotes, the capped analysis of gene expression (CAGE) technique (Shiraki *et al.*, 2003) has been developed more than 10 years ago, which is heavily used and has been further improved over time (Kodzius *et al.*, 2006). CAGE extracts 5' tag sequences of the currently expressed transcripts in the sample utilizing the capping structure of unprocessed RNAs. Unfortunately, this method cannot be applied to prokaryotic organisms because of differences in mRNA processing, i.e. prokaryotic mRNAs do not have a cap.

With the era of NGS and RNA-seq the entire transcriptome of a prokaryote can be sequenced. To identify TSSs within this data, approaches for peak detection from methods like ChIP-seq (Johnson *et al.*, 2007) could be employed, but these algorithms are not suited for the noise of RNA-seq data. In normal RNA-seq data, an enrichment of read starts can occur apart from TSS locations at processing sites, secondary structures influencing RNA degradation and sites for chemical modification (Amman *et al.*, 2014). Especially weak TSS signals are hardly interpretable in such an environment.

To overcome these problems, the differential RNA-seq (**dRNA-seq**) approach (Sharma *et al.*, 2010) was proposed for generating large-scale experimental evidence for complete transcriptomes in prokaryotes. This technique generates two - differential - data sets. One is treated with 5' mono-phosphate-dependent terminator exonuclease (TEX) and the other is not. Since processed RNAs exhibit a 5' mono-phosphate, they are degraded by the TEX treatment while primary transcripts required for TSS detection are protected by a 5' tri-phosphate.

The first tools for automatic prokaryotic TSS detection from dRNA-seq data became available during the course of this thesis and their number is still very limited. **TSSPredator** (Dugar *et al.*, 2013) detects TSS for TEX-treated (TEX+) versus untreated (TEX-) data sets based on the evaluation of read start peaks enriched in the TEX+ sample by fixed thresholds. **Rockhopper** (McClure *et al.*, 2013) follows a Bayesian approach to identify transcript boundaries, but has not been evaluated on dRNA-seq data. Two more sophisticated statistical

models have just been published last year: TSSAR (Amman *et al.*, 2014) and TSSer (Jorjani and Zavolan, 2014). **TSSAR** models the number of read starts at a genomic position as Poisson distributed. They presume that the difference of the TEX+ versus the TEX- sample follows a Skellam distribution and detect the TSS of enriched primary transcripts on this basis. **TSSer** models the number of read starts at a genomic position by a hypergeometric distribution, which is approximated by a binomial distribution. TSSer allows for replicates and includes them in the statistical model.
Note that only TSSs of expressed genes can be identified by dRNA-seq. In order to get experimental evidence for as many genes of an organism as possible it is advisable to analyze data sets generated from different environmental conditions including stress conditions.

### 2.8.3. Differential Gene Expression Analysis

The adaptation of an organism to different environmental conditions, developmental stages or genetic variants is reflected in changes of gene regulation, leading to different transcripts and transcript levels. This effect can be measured on the molecular level by comparison of NGS count data, i.e. read counts per genes in RNA-seq data sets. A gene is accepted to be differentially expressed between different conditions, if it can be shown that the transcript levels for that gene differ significantly among the conditions (see Figure 25). An important observation which is a premise for differential gene expression (abbreviated as DE from here on) analysis by RNA-seq was made by (Mortazavi *et al.*, 2008): Read counts per gene are approximately linearly related to the respective transcript abundance. In eukaryotes the gene models are more complicated due to splicing and the necessity to distinguish different splice variants. Thus, in prokaryotes the RNA-seq reads can be completely mapped to the reference (see Section 2.6), while eukaryotic reads should be mapped using a split read mapper like TopHat (Trapnell *et al.*, 2009) to be able to map reads across splice junctions.

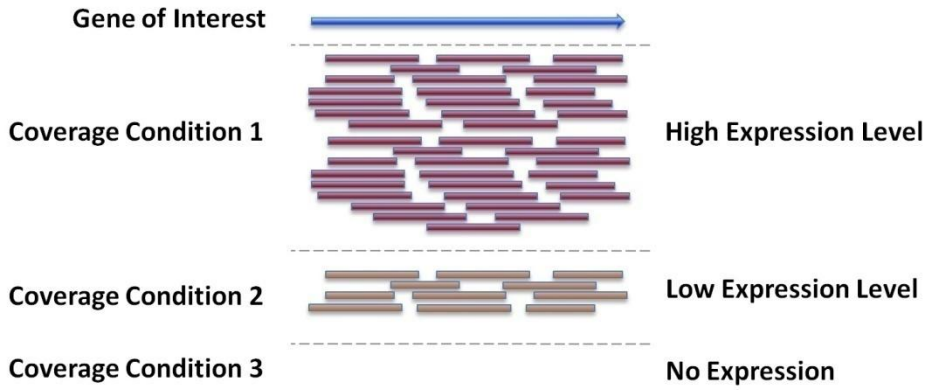



**Figure 25: Differential Gene Expression.** A schematic example of differential gene expression among three samples, each obtained from a different environmental condition, is shown in the figure for a single gene of interest. The first condition exhibits a high expression level compared to the significantly lower expression level observed in condition two and no expression in condition three. Such observations show that gene regulation of the gene of interest changes with the environmental setting.

DE evaluation methods require suitable statistical models for normalization and DE testing. The chosen models should account for the extremely high dynamic range of RNA-seq data and incorporate suitable error modelling. Reliable normalization across conditions is especially important to prevent false positive results when the library sizes of the compared conditions vary significantly. The first methods for DE analysis were based on the assumption that the Poisson distribution is a suitable model for RNA-seq data, but it has been pointed out by (Robinson and Smyth, 2007; Nagalakshmi *et al.*, 2008) that the Poisson distribution does not account for the amount of variability in RNA-seq data. The single parameter of the

Poisson distribution determining both variance and mean is thus too tight for the observed data and does not control type-I errors (false positive results). This behaviour is referred to as the overdispersion problem. Due to these findings, most newer approaches (including all mentioned approaches below) employ a Negative Binomial (NB) distribution (Whitaker, 1914) to model the number of reads assigned to a specific gene in a specific sample. Note that statistical DE models address a more general problem and can also be applied to ChIP-seq (Johnson *et al.*, 2007), Tag-seq (Morrissy *et al.*, 2009), 4C-seq (van de Werken *et al.*, 2012), Hi-C (van Berkum *et al.*, 2010) data sets or counts of observed taxa in metagenomic studies. Available software packages for DE analysis all implemented in the statistical programming language R[11] as R-packages within the Bioconductor project (Gentleman *et al.*, 2004) include

- **baySeq** (Hardcastle and Kelly, 2010): It is determining the prior distribution of read counts by a computationally intensive empirical Bayes approach. DE is measured by calculating the posterior probabilities. baySeq starts by storing the read counts and library size scaling factors together as

$$D_c = \{(u_{1,c}, \dots, u_{n,c}), (l_1, \dots, l_n)\},$$

where $c$ is a gene, $n$ is the number of samples, $u$ is the read count of a gene and $l$ is the scaling factor of a sample. A distinctive feature of baySeq is that it supports designing multi-condition experiments by entering multiple models for the same data sets. All models are then tested for DE. baySeq acts on the assumption that replicates share the same prior distribution on the underlying parameters of each gene. The posterior probability of a model $M$ given the data $D_c$ is then calculated according to Bayes:

$$\mathbb{P}(M|D_c) = \frac{\mathbb{P}(D_c|M) * \mathbb{P}(M)}{\mathbb{P}(D_c)}$$

To solve this equation, the replicate sets of each $M$ are considered by baySeq. $\theta_i$ are the parameters of the underlying distribution of one replicate set and all replicate sets are represented by $K = \{\theta_1, \dots, \theta_m\}$. Now, $\mathbb{P}(D_c|M)$ is calculated as the marginal likelihood

$$\mathbb{P}(D_c|M) = \int \mathbb{P}(D_c|K, M) * \mathbb{P}(K|M)\, dK$$

The NB distribution is used in this equation to calculate $D_c|K, M$ and $K|M$. The prior probability of a model $\mathbb{P}(M)$ is estimated by first choosing some prior probability $p$ for the model $M$. $p$ is then used to estimate the posterior probability of $\mathbb{P}(M|D_c)$, generate a new estimate $p'$ for the prior probability of $M$. This process is iterated until convergence to obtain an estimate for the prior probability of $M$.

- **DESeq** (Anders and Huber, 2010): It is a widely used tool linking variance and mean by local regression as error model for the count data. Thus, normalization is included in the statistical model. DESeq includes modelling and evaluation of multi-factor designs. The NB distribution for the non-negative count data is given as $K_{ij} \sim NB(\mu_{ij}, \sigma_{ij}^2)$, where $i = (1, \dots, n)$ is the gene index and $j = (1, \dots, m)$ is the sample index. Both true mean and variance are unknown and have to be fit to the data.

[11] http://www.r-project.org

DESeq calculates the mean as

$$\mu_{ij} = q_{i,p(j)} * s_j,$$

$q_{i,p(j)}$ is a per-gene value dependent on the experimental condition $p(j)$ and estimated as the average of the counts of all samples in condition $j$:

$$\hat{q}_{ip} = \frac{1}{m_p} * \sum_{j:p(j)=p} \frac{k_{ij}}{\hat{s}_j},$$

where $m_p$ is the number of replicates in condition $p$ and $k_{ij}$ is the read count of gene $i$ in condition $j$. $s_j$ is a library size parameter and estimated as

$$\hat{s}_j = median_i * \frac{k_{ij}}{(\prod_{v=1}^{m} k_{iv})^{1/m}}$$

The variance is calculated as sum of a shot noise and a raw variance term in DESeq:

$$\sigma_{ij}^2 = \mu_{ij} + s_j^2 * v_p(q_{i,p(j)}),$$

where $v_p(q_{i,p(j)})$ is a smooth function of the per gene abundance $q_{i,p(j)}$. The test for DE is carried out by assuming $q_{i,A} = q_{i,B}$ for two conditions $A$ and $B$ as null hypothesis. The sum of all read counts for gene $i$ is denoted as $K_{iA}$ for condition $A$ and as $K_{iB}$ for condition $B$. Given that the overall sum of all counts for gene $i$ is $K_{iS} = K_{iA} + K_{iB}$, the probability $p(a, b)$ of the event $K_{iA} = a$ and $K_{iB} = b$ can be computed for any pair of numbers $a$ and $b$ by utilizing the above presented equations. A p-value for DE of gene $i$ can then be computed using $p(a, b)$:

$$p_i = \frac{\sum_{\substack{a+b=K_{iS} \\ p(a,b) \leq p(K_{iA},K_{iB})}} p(a,b)}{\sum_{a+b=K_{iS}} p(a,b)}$$

- **DESeq2** (Love *et al.*, 2014): It is an enhanced alternative to DESeq focusing on the DE strength instead of the presence or absence of DE. Thus it provides a more quantitative DE analysis than DESeq and enables gene ranking. The methodology of DESeq2 starts by fitting a generalized linear model (GLM) (McCullagh and Nelder, 1989) to each gene. Like in DESeq, normalization is included in the process. The dispersion estimates of each gene are shrunk by an empirical Bayes approach. A Wald-test (Wald, 1943) is used to test for differential expression.
- **DSS** (Dispersion Shrinkage for Sequencing) (Wu *et al.*, 2013): This tool focuses on adequately capturing the heterogeneity of gene-specific dispersion across biological replicates by an empirical Bayes shrinkage more accurate than the method implemented in edgeR.
- **edgeR** (Robinson *et al.*, 2010): In edgeR the overdispersion is shrunk by conditional weighted likelihood. For statistical testing of DE two tests are employed: The Wald-test (Wald, 1943) and their own test developed earlier (Robinson and Smyth, 2007). An update of edgeR enabled evaluating multi-factor designs and includes an empirical Bayes shrinkage estimate for the dispersion (McCarthy *et al.*, 2012). The dispersion is first estimated gene-wise and subsequently the empirical Bayes approach shares information between genes.

- **ShrinkSeq** (Van De Wiel *et al.*, 2013): This tool focuses on improved shrinkage of dispersion-related parameters. Consequently, it offers joint shrinkage of parameters. Further, it uses the zero-inflated NB distribution, which treats zero counts differently than positive counts.

Especially for the analysis of eukaryotic genomes, the Cufflinks package (Trapnell *et al.*, 2012) has been developed, which contains a comprehensive DE analysis tools called **Cuffdiff** (Trapnell *et al.*, 2013).

### 2.8.4. Operon Detection

Operons are units of co-regulated neighbouring genes located on the same strand. Classical operons, as coined by Jacob and Monod (Jacob *et al.*, 1960), consist of a promoter and operator(s) regulating several structural genes (for details see Figure 26).

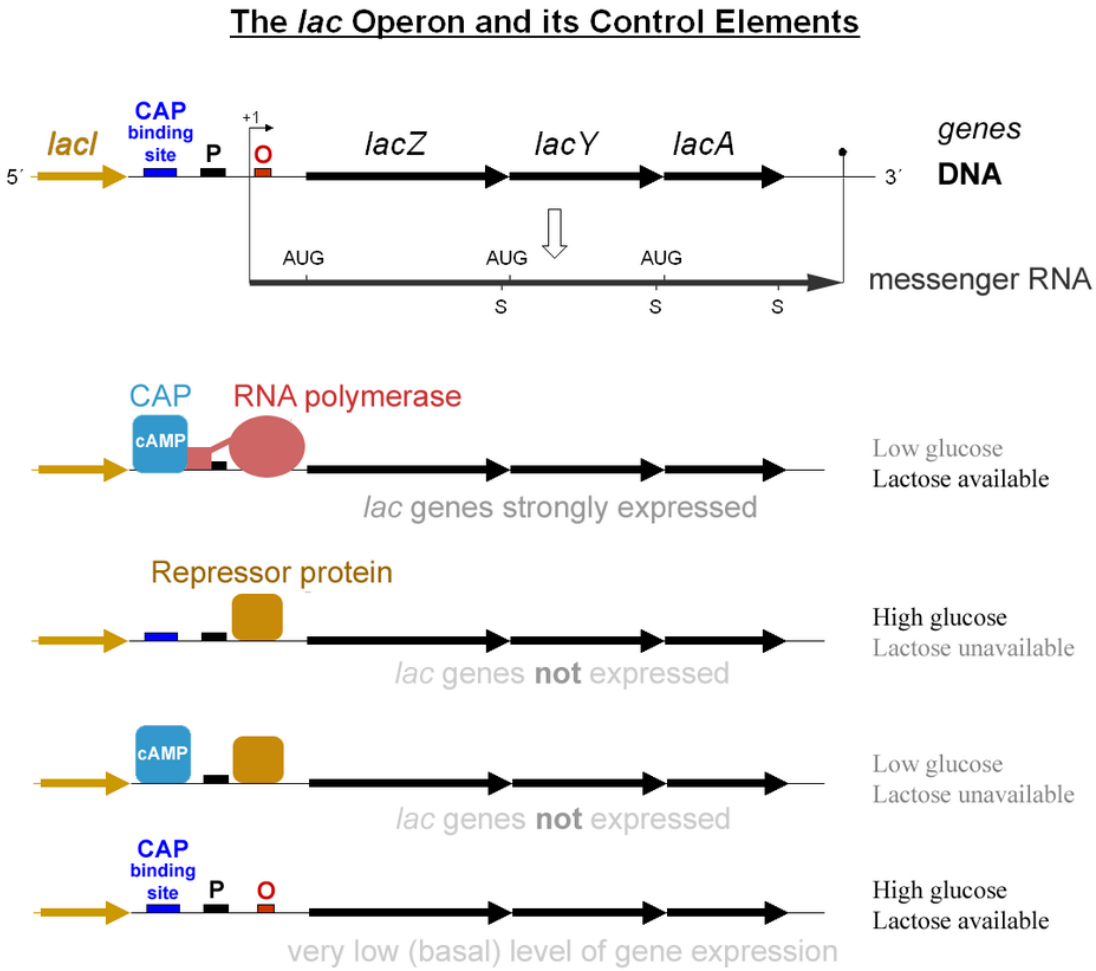



**Figure 26: The *lac* operon and its control elements.** The figure illustrates regulation of the lactose operon in *Escherichia coli* K12. This was the first operon analyzed by Jacob and Monod (Jacob *et al.*, 1960). The three structural genes (*lacZ*, *lacY*, *lacA*) of this operon control the transport and metabolism of lactose. The first regulation is constituted by the *repressor protein* binding the operator (*O*). It is generated from the *lacI* gene in the absence of lactose. The repressor ensures that all *lac* operon genes are only transcribed in the presence of lactose. A second regulation is constituted by *CAP* (catabolite activator protein). In absence of glucose *CAP* is loaded with cAMP (cyclic adenosine monophosphate) which in turn allows *CAP* to bind upstream of the promoter (*P*). Binding of *RNA-polymerase* to the promoter is thereby alleviated and increases transcription of the *lac* operon. The figure is adapted from: `commons.wikimedia.org/wiki/File:Lac_operon-2010-21-01.png`, accessed on 03.02.2014.

These genes are transcribed into a single polycistronic mRNA and mostly encode for similar functions or belong to the same metabolic pathway. However, operons can also contain

functionally unrelated genes (Price *et al.*, 2006). Operons are mainly occurring in prokaryotes, but can also be found in eukaryotes (Hurst *et al.*, 2004). A notable difference of operon-like gene clusters in eukaryotes is that their structural genes are transcribed separately in most cases instead of polycistronic.
In the past few years it became evident that operons are not static entities. They are rather composed dynamically. Depending on the environmental conditions and development stages, the product of an operon can vary in terms of alternative transcription start sites, the number of expressed genes and even monocistronic mRNAs can be produced (Güell *et al.*, 2009; Li *et al.*, 2013).

Several computational methods for predicting operons have been suggested since the first whole genome sequences of bacteria became available. Before the era of RNA-seq, the majority of approaches for operon detection relied on models trained on experimentally well-characterized operons (Craven *et al.*, 2000; Tran *et al.*, 2009). These tools often achieve low sensitivity when they are applied to yet uncharacterized genomes with very little or no available training data. Alternative approaches rely on typical characteristics of operons, like intergenic distance, conservation of gene order among multiple reference genomes and functional relation of genes within an operon (Dam *et al.*, 2007). Recent developments are now either based on experimental data from RNA-seq experiments or combine RNA-seq with computational methods using trained classifiers. RNA-seq data sets only contain data for expressed genes under the given conditions. Therefore, the use of RNA-seq enables identification of expression patterns of operons and elucidation of expression dynamics within an operon under different environmental conditions (e.g. an alternative transcription start site can lead to expression of only a sub-part of an operon). The most recent RNA-seq based approaches supporting these features include:

- **Rockhopper** (McClure *et al.*, 2013) is an open-source software implemented in Java enabling several RNA-seq data analyses, like detecting novel sRNAs, transcription start sites and operons. Their approach solely relies on RNA-seq data. A naive Bayes classifier is fed with prior operon probabilities. These probabilities are derived from two properties: Intergenic distance specific for the genome and expression correlation over all available corresponding samples.
- **RNAseg** (Bischler *et al.*, 2014) is a C++ software designed for identification of transcription units solely based on dRNA-seq data (see Section 2.8.2). RNAseg divides the genome in transcribed and untranscribed segments. For this purpose they extended the SCM segmentation algorithm introduced by Huber *et al.* (2006) to make use of dRNA-seq data. Afterwards, all genes overlapping the same segment are assigned to an operon.
- (Fortino *et al.*, 2014) proposed an integrative approach combining experimental dRNA-seq data with a classifier for genomic properties trained on a small set of experimentally validated operons. The RNA-seq data is analyzed for transcript segments utilizing RPKM (see Section 2.8.1), intergenic distance within the genome and the distinct transcript level of intra- and intergenic regions. All significant evidence found within the RNA-seq data is then cross-validated by linking the results to confirmed operons from the DOOR (Mao *et al.*, 2009) database. Putative novel operons are also reported, but classified accordingly. This approach is implemented in R.

# Chapter 3

# Read Mapping Analysis - State of the Art

Specialized viewers for visualizing and dealing with high throughput NGS mapping data sets, called **tracks**, in connection with a genomic reference sequence were developed in the past few years. The list of available viewers already comprises more than 20 tools (Abeel *et al.*, 2012). Note that not all of these tools are maintained and still capable of efficiently handling huge data sets from today's NGS platforms. To provide an overview of the spearhead of genome browsers capable of visualizing and analyzing read mapping data (see Section 2.6) five popular state of the art open source software packages are reviewed in this chapter. All five tools are implemented in the platform independent programming language Java. A common functionality of all presented tools is the ability to visually explore genome sequences and their annotations in connection to corresponding RNA-seq, DNA-seq and ChIP-seq data. For a side-by-side comparison of the main features of all tools please visit Table 9 in the results section. The chapter concludes with an evaluation of available read mapping quality and quantity information in SAM formatted mapping files and their availability in the five presented tools.

## 3.1. Savant

Savant (Sequence Annotation, Visualization and Analysis Tool) (Fiume *et al.*, 2010, 2012) aims at dynamically visualizing up to human sized genomes along with corresponding read mapping, point, interval and continuous data sets (see Figure 27). Additionally, it defines a plug-in interface to make the software extensible for external programmers. Some automatic analysis function, mentioned below, has already been incorporated in Savant through this plug-in interface.
Multiple visualization modes are available for read mapping data in Savant (see Figure 27). They address manual exploration of SNPs, structural and copy number variants, peaks and other characteristics visible from the genome coverage. Read pairs can be visualized as arcs, which colour-code the distance, orientation and concordance of the reads (see Figure 28).

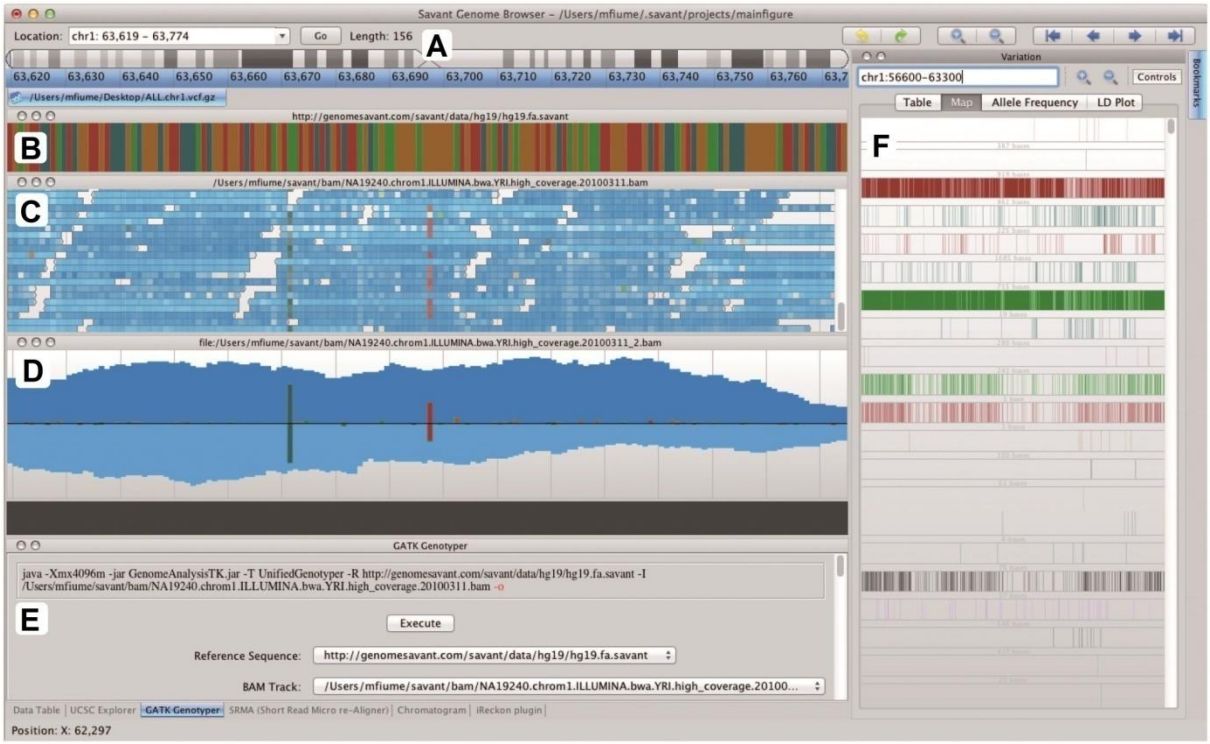



**Figure 27: Savant 2 user interface**. **(A)** The visible chromosome can be selected and is shown in the row below. **(B)** Below the chromosome, the chromosome sequence can be visualized by base specific colors. **(C)** A read alignment track assigning each read position an intensity proportional to its base quality. Mismatches within reads are denoted by colors. **(D)** The same read alignment track in Strand-SNP mode. This mode shows coverage and allele support partitioned by strand, with positive strand support above the black line and negative strand support below. **(E)** Plug-in panel. The opened GATK (McKenna *et al.*, 2010) plug-in can be used to compute genotypes from read alignment tracks within the browser. **(F)** Variant Navigator panel. The Variant Navigator visualizes and guides the navigation of genetic variant data. The map page of the Variant Navigator displays a matrix where each column represents an individual or sample from the file and each row represents a variant position; each cell in the matrix is coloured according to the allele possessed by the corresponding sample and position, or is transparent if no allele is predicted there. The genomic range displayed in the Variant Navigator is a superset of the range for tracks, and users can click within the Variant Navigator to navigate to sub-ranges of the variant range. Figure and description are adapted from (Fiume *et al.*, 2012).

Automatic SNP detection can be performed with a plug-in for the GATK package (McKenna *et al.*, 2010), allowing users which have installed GATK to run it from within Savant. GATK outputs VCF (see Section 2.7.1), which can be visualized by Savant in several ways: The data can be represented in a table, as allele frequencies for each SNP, assigned to two cohorts in order to compare allele frequencies and as an LD (Linkage Disequilibrium) plot (see Figure 28).

Two plug-ins are available for automatic analysis of RNA-seq data. The first plug-in allows to perform differential gene expression analysis using edgeR (Robinson *et al.*, 2010) (see Section 2.8.3) from within Savant, if an edgeR installation is present. The second plug-in supports import or calculation of isoforms and their abundances. Other plug-ins adding notable functionality to the software are:

- A plug-in for exploring protein-protein interaction partners of a query gene.
- A wrapper plug-in for the local re-alignment tool SRMA (Homer and Nelson, 2010). Other than read mappers, SRMA shares information across mapped reads to locally realign them based on their local consensus sequence.
- A plug-in for WikiPathways (Pico *et al.*, 2008) is available to browse the genome based on functional annotation.
- A plug-in is available allowing the integration of remote genomic content from the UCSC genome database (Karolchik *et al.*, 2014).

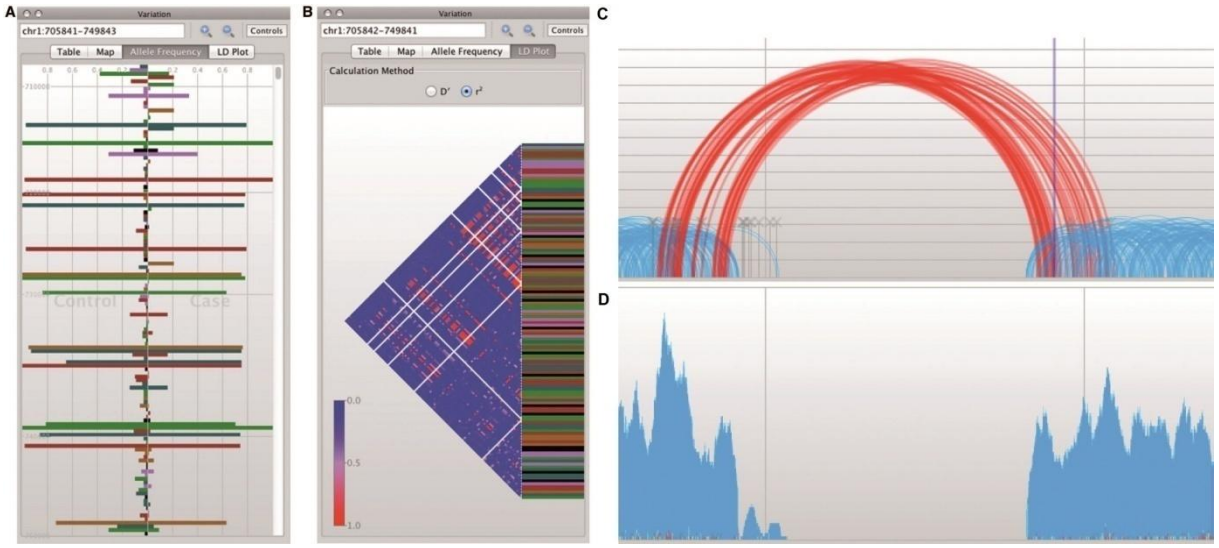



**Figure 28: Savant SNP and read pair visualizations. (A)** The allele frequency view enabling comparison of allele frequencies of genetic variants from samples assigned to two cohorts is shown. **(B)** An LD plot of variants in the same range as in (A). Blue and red cells represent low and high correlation between variant positions, respectively. **(C)** Paired reads displayed as vertically scaled arcs depending on their insert size. Red arcs represent pairs that are identified as being discordant. All blue pairs are classified as concordant. The example indicates a deletion event in the mapped genome. **(D)** The same data as shown in (C) displayed as coverage plot. Figure and description are adapted from (Fiume *et al.*, 2012).

## 3.2. GenomeView

The dynamic GenomeView software (Abeel *et al.*, 2012) facilitates visualization of genomes and read mapping data (see Figure 29). This browser does not offer any automatic analysis functions, but a distinct feature of GenomeView in comparison to other genome browsers is its functionality for editing and curating gene annotations. Like Savant (see Section 3.1), GenomeView is extensible through a plug-in interface. Here, the Java Plug-in Framework (JPF) is used. To demonstrate the extensibility, a plug-in has been developed by the authors for visualization of DNA properties like GC-content.

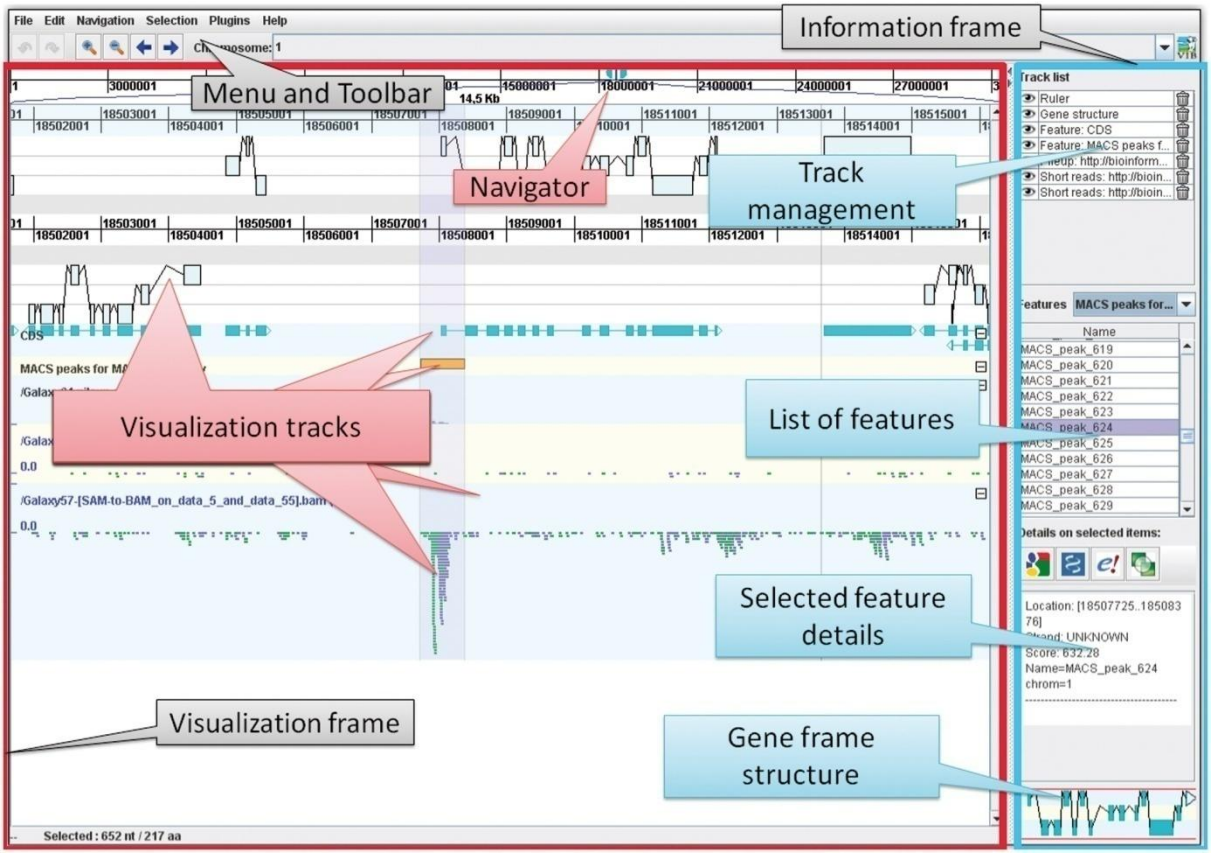



**Figure 29: GenomeView graphical user interface.** The reference is displayed at the top along with all six reading frames. Genomic annotations are shown in their respective frame. Below, a track with CDS annotations is shown (blue). Further below, two read alignment data sets are shown. The information frame on the right hand side displays global information and context sensitive information as described in the blue balloons. Figure and description are adapted from (Abeel *et al.*, 2012).

Input data can be loaded from both, local files or URLs. Secure data transfer by encrypting the data via Secure Socket Layer (SSL) is also supported. Since genome annotations can be edited in GenomeView, it allows sending updated data back to a server via http-post. The authors claim that it is thus straight forward to integrate their software as visualization front-end into an existing web page.

Besides standard visualizations for genomic annotations and read alignments from RNA-seq, DNA-seq or ChIP-seq, GenomeView has the ability to import multiple (whole) genome alignments from multiple fasta or MAF (Multiple Alignment Format) files (see Figure 30). These data tracks are visualized in a specialized view including sequence logos at the highest zoom level. Zooming out on the other hand switches to overall conservation of the data.

Visual SNP inspection is supported by emphasizing mismatching bases by a colour coded consensus base histogram and supporting import of VCF files (see Section 2.7.1) for pre-computed SNPs.

Database searches are available when a genomic annotation is selected. It can then be blasted against the NCBI database.

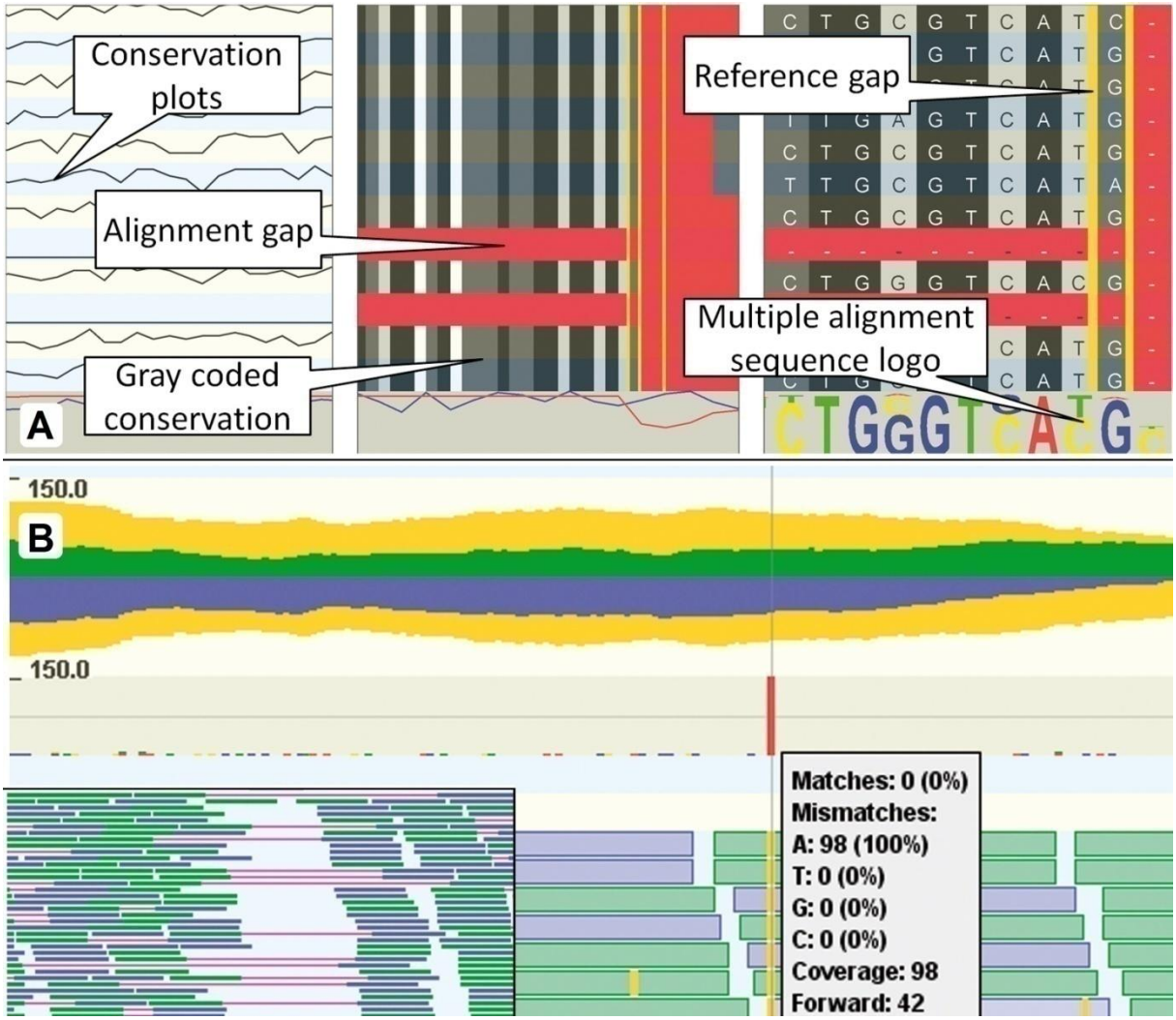



**Figure 30: GenomeView multiple alignment and read mapping visualizations. (A)** On higher zoom levels, the multiple whole genome alignments are shown as conservation plots (left). Zooming in further switches to a colour-coded view (middle), which reveals the underlying bases in the lowest zoom level (right). Alignment gaps in the histogram are shown in red and reference gaps in yellow. The nucleotide zoom level also contains a sequence logo visualization. **(B)** Coverage is visualized in a colour-coded plot. Coverage above (green) or below (purple) the central axis originates from the respective strand (forward, reverse). Coverage depicted in yellow is the total coverage mirrored above and below the central axis. The lower part shows the alignment view, representing the single read mappings. The strand of the alignments is coded by the same colors as in the coverage plot. The inset shows the visualization for read pairs or spliced reads, which are both connected by red lines. The histogram in the middle of (B) visually reveals one SNP (red bar). All reads mapping to that reference position show an 'A' instead of the reference base, which is also shown in the tool tip. Figure and description are adapted from (Abeel *et al.*, 2012).

## 3.3. IGV & Rockhopper

The Integrative Genomics Viewer (IGV) (Robinson *et al.*, 2011; Thorvaldsdóttir *et al.*, 2013) focuses on visualization and data integration (see Figure 31), but does not offer any automatic analysis capability itself.

Visualizations in IGV include views for RNA-seq of pro- and eukaryotes including splice junctions (see "RNA-seq" in Figure 32) and for DNA-seq facilitating visual variation detection. Distinct features of IGV are the ability to view multiple regions of the same genome simultaneously - either by selecting multiple genome loci of interest or by showing both regions of mapped read pairs - and their own data representation. Two relevant features for paired reads are offered: Color coding mates by size, orientation and strandedness and grouping by strand and mate chromosome.

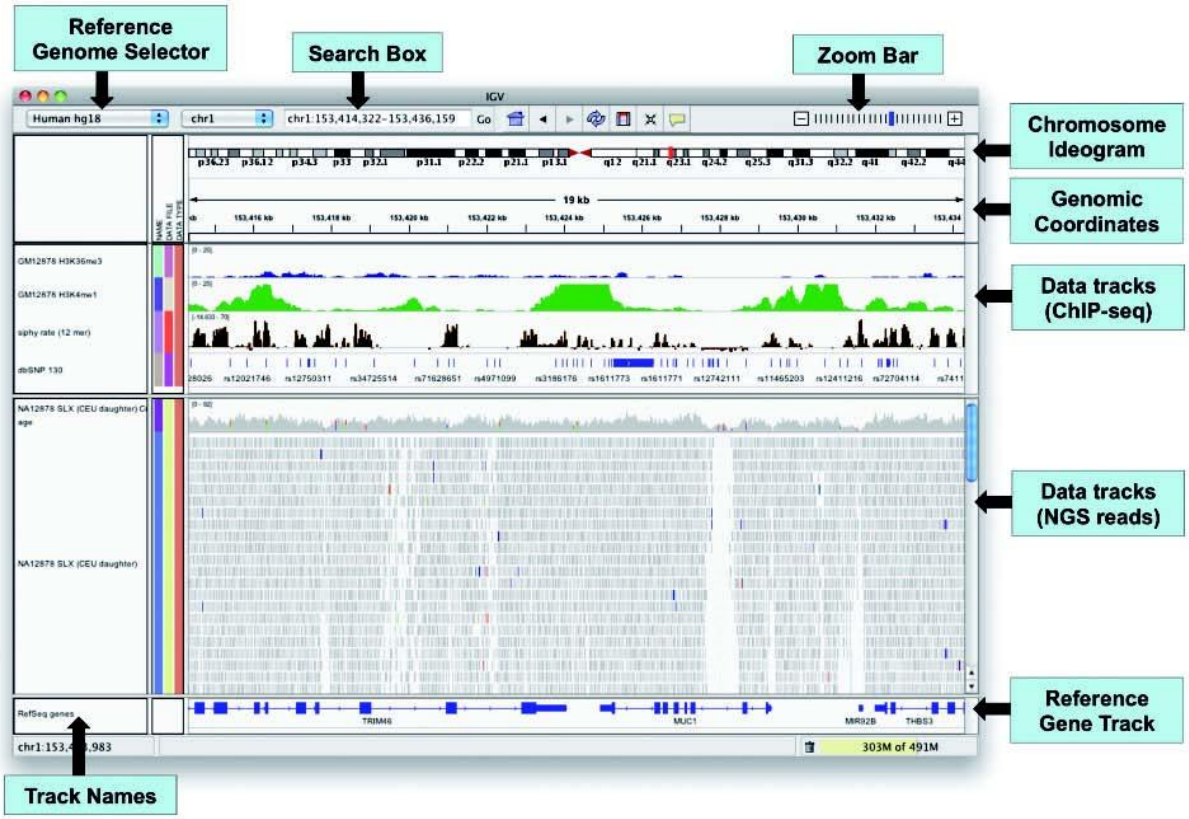



**Figure 31: IGV user interface.** IGV showing a region of a eukaryotic chromosome at a medium zoom level. The view includes the genomic region and annotations, three ChIP-seq tracks and a NGS data track visualized as unstranded coverage and read alignments with mismatches. Figure and description are adapted from (Thorvaldsdóttir *et al.*, 2013).

Input data, such as read mapping tracks, can be preprocessed and converted into a multi-resolution format called Tiled Data Format (TDF). This format is employed for a hybrid data-loading approach: Depending on the current zoom level, data is either loaded from the pre-computed TDF files or calculated on the fly from the original data file. Since 2013 methylation patterns obtained from bisulfite sequencing (Bock *et al.*, 2010) can be visually explored in IGV (see "Bisulfite-seq" in Figure 32). Data integration in IGV comprises several data formats including formats for genome annotations (GFF, GTF2, BED and PSL), read mappings (SAM, BAM), genetic variants (VCF, see Section 2.7.1) and more. The data can be presented in both non-indexed (slower access) or indexed (faster access) format and contain multiple resolutions (TDF, bigWig, bigBed (Kent *et al.*, 2010)). Additionally, copy number or expression data can be imported and viewed in heat maps. IGV also supports saving a current session state and reopening it or sending the stored session to other collaborators. A command line interface enables other software or command line users to communicate with IGV. They support command line, socket port and HTTP access.

A first step in the direction of a combination of automatic read mapping analysis and visualization has been made in parallel to the work presented here by the authors of the bacterial RNA-seq analysis tool Rockhopper (McClure *et al.*, 2013). They proposed to use IGV for the visualization of their command-line analysis tool results. Combining Rockhopper and IGV enables automatic detection of bacterial transcript boundaries and normalizing their read count (see Section 2.8.1) for analysis and subsequent visualization of TSS (see Section 2.8.2), differential gene expression (see Section 2.8.3), operons (see Section 2.8.4), RPKM and read count values (see Section 2.8.1). Still, Rockhopper lacks support for eukaryotic data, TPM values (see Section 2.8.1) and automatic DNA-seq analysis functions, like SNP detection (see Section 2.7.1) and detection of genome rearrangements (see Section 2.7.2). Likewise, basic analysis functions such as the detection of covered or uncovered intervals of the genome are not included.

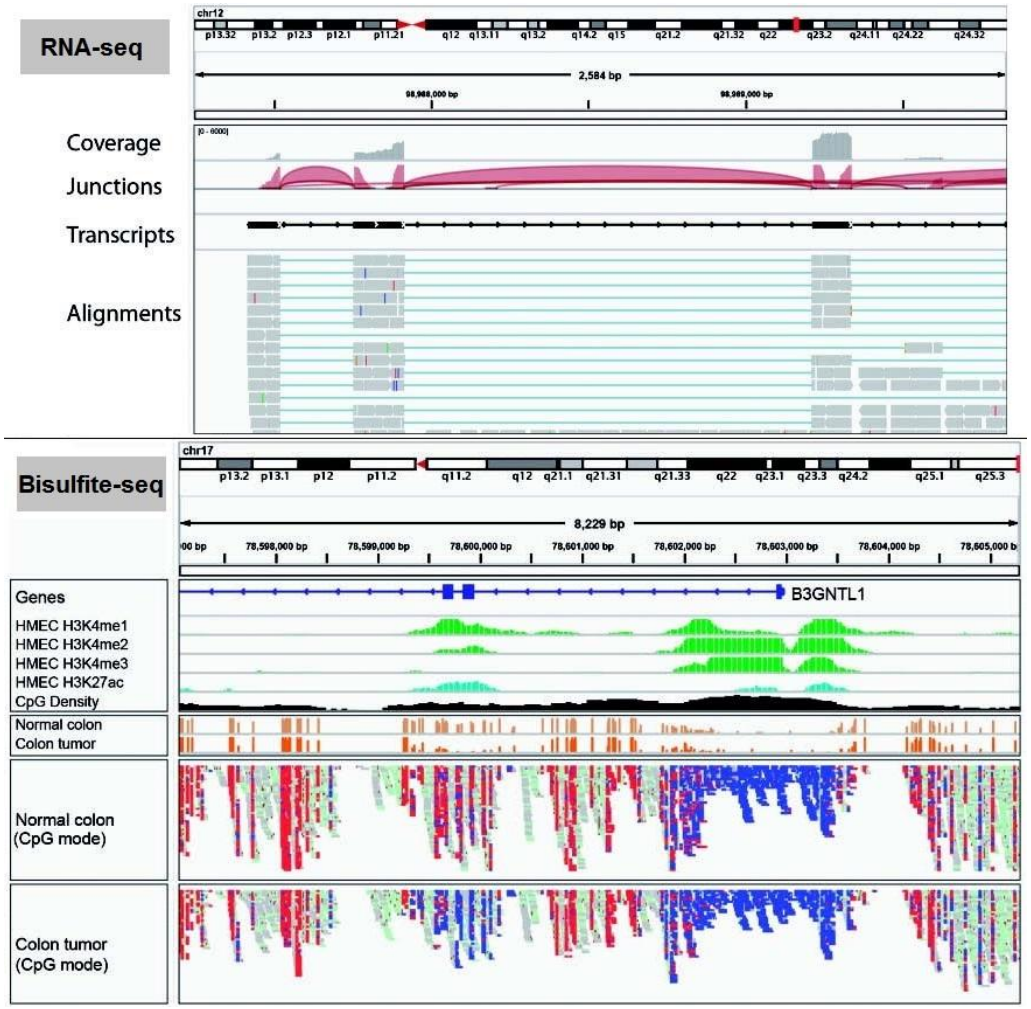



**Figure 32: Visualizations for eukaryotic RNA-seq and methylation in IGV. RNA-seq:** The view includes tracks for total coverage, junction coverage, predicted transcripts and read alignments. Reads spanning junctions are connected by thin blue lines. In the junction track, the height of each arc is proportional to the total number of reads spanning the junction. **Bisulfite-seq:** Read alignments with methylated bases are displayed in red, while un-methylated read alignments are displayed in blue. The coloring enables visual exploration of the methylation patterns. Figure and description are adapted from (Thorvaldsdóttir *et al.*, 2013).

## 3.4. IGB

The Integrated Genome Browser (IGB) (Nicol *et al.*, 2009) focuses on customization of visualizations, but does not contain embedded automatic analysis capabilities. Thus, many colour and layout adjustments can be made in IGB, like drag and drop of data tracks to the preferred position on the screen. Additional to the standard genome and read mapping views, IGB contains a viewer for genome graphs which can display numerical data. Integrated functionality for graphs of read coverage, read starts and mismatch density is available. Data tracks can either be separated by strand, or both strands can be combined (see Figure 33).

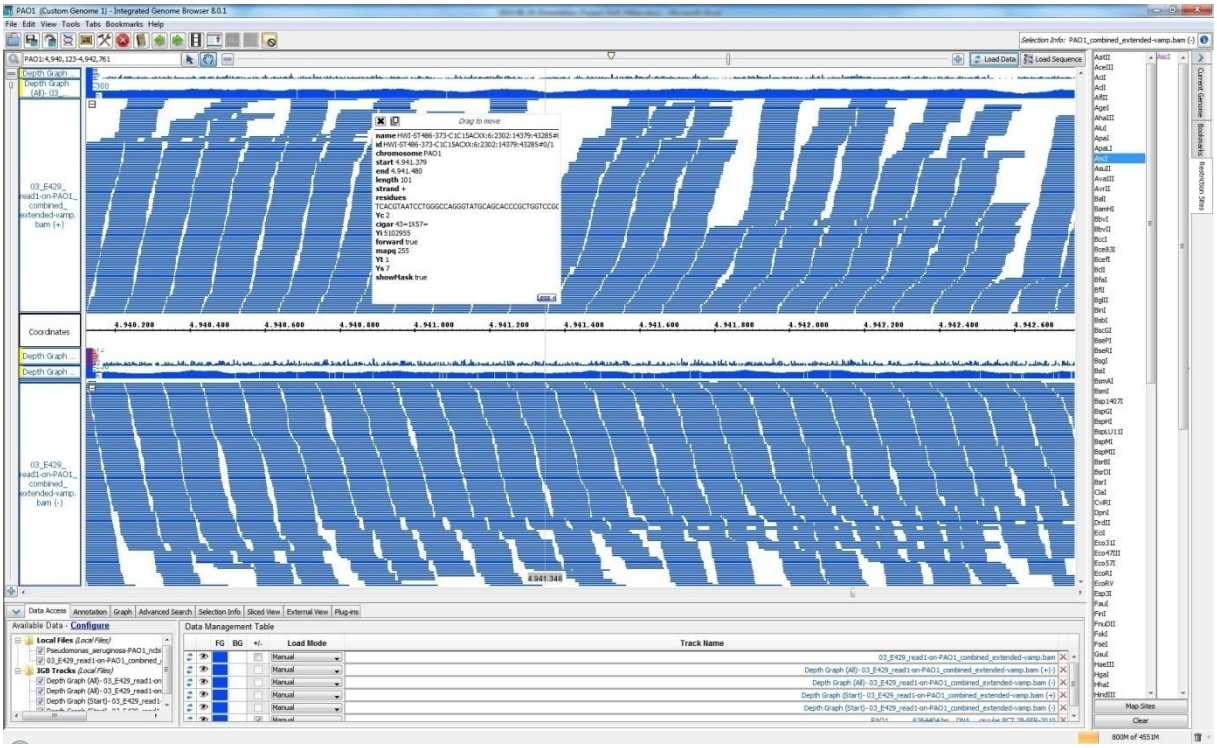


**Figure 33: IGB user interface showing strand specific read mapping data.** In this example bacterial read mapping data has been split by strand. The forward mappings are shown in the upper half, the reverse mappings are shown in the lower half. Additionally, coverage graphs (immediately above the alignments) and read start graphs (above the coverage graphs) are shown. The read start graph is especially interesting for visual inspection of RNA-seq data. In the right panel, the functionality to select restriction sites for highlighting in the reference sequence is shown.

Reads of eukaryotes mapping across exon junctions are shown by exon blocks intercepted by thin lines throughout the introns (see Figure 34), but a specialized visualization for read pairs is not available. The introns can also be trimmed from the display by a switching on a sliced view. For expression data from microarrays, heatmap visualizations are available. A unique feature of IGB among the presented tools is the ability to show restriction sites for selected restriction enzymes from a predefined list (see Figure 33).
IGB can be controlled from the command line by a simple IGB-specific scripting language, introduced by the IGB authors. Therefore, IGB can be embedded in web pages. It can be controlled via HTTP requests from these web pages or other software by their own mechanism QuickLoad or by the Distributed Annotation System (DAS) (Jenkinson *et al.*, 2008).

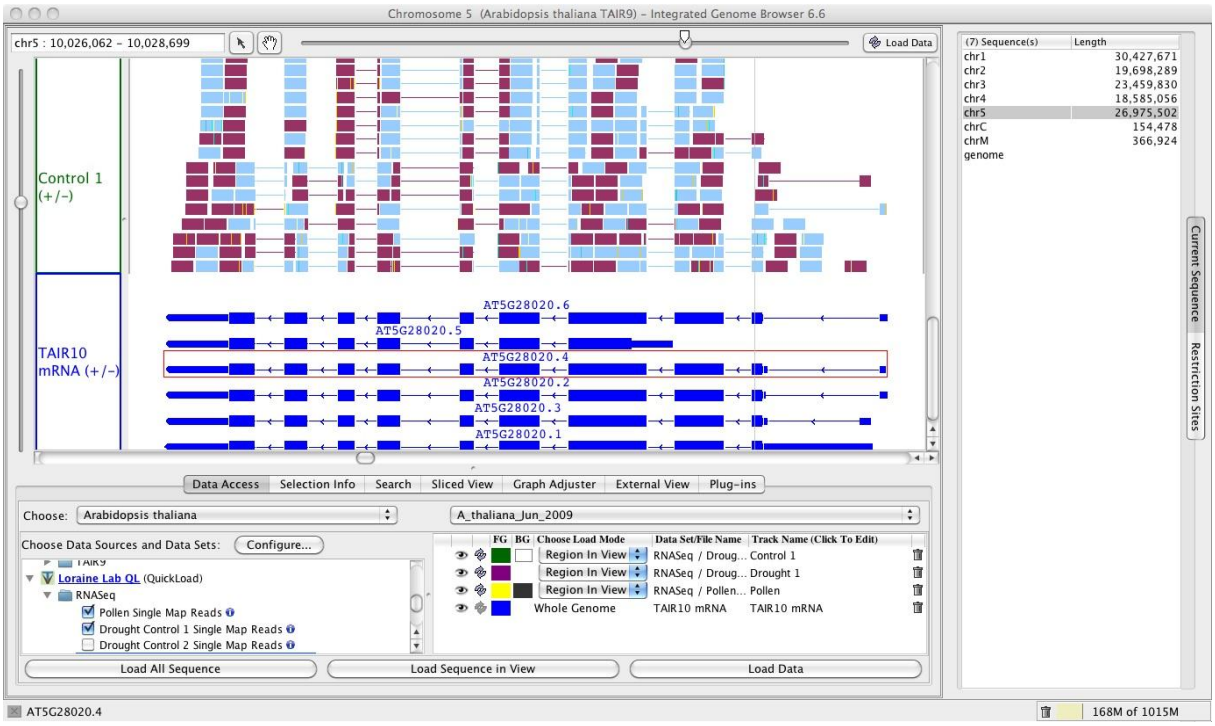



**Figure 34: IGB visualization of RNA-seq reads and a reference.** In this example data from Arabidopsis is shown. The gene models are displayed in the blue track below the read mappings in cyan and red. Overlapping alternative transcripts stored in the reference annotation track are expanded (blue). Split read mappings across splice junctions are connected by thin lines. The reads have been colored by strand. The available chromosome sequences are listed on the right hand side. Further customization can be carried out in the lower options panel. The figure is adapted from the user's guide on: `http://bioviz.org/igb/`, accessed on 15.02.2014.

## 3.5. Artemis

14 years ago, Artemis started as a genome visualization and annotation tool (Rutherford *et al.*, 2000). Its basic capabilities were the visualization of editable annotations next to the genomic sequence (see Figure 35, (e), (f) and (g)), some graph plots e.g. for GC-content and a table listing the annotations. Lately, Artemis has been extended to be able to visualize read mapping data next to the genomic sequence (Carver *et al.*, 2012). This functionality is provided by an integrated version of the simple read mapping viewer BamView (Carver *et al.*, 2010, 2013). With this update five different visualizations for read mapping data became available (see Figure 35, (a) - (e)) and can be shown simultaneously for the same data set. To facilitate comparison of data sets, the viewer panel can be cloned for side-by-side comparisons of two data sets.
Additional to read mapping data, Artemis can import and visualize variants from VCF files (see Figure 35, (f) and section 2.7.1). This functionality includes a SNP density plot. Variants and mappings can be filtered by different constraints by Artemis. For mappings SAM-flag and mapping and base quality filters are available.

Data shown in Artemis can come from local or remote files. For remote file access, HTTP and FTP connections are offered. Artemis is also capable of preprocessing input data for fast access. Therefore, it can create index files and compress formats like VCF and SAM. Other user-defined numerical input coming from external analyses can be imported and visualized as graphs or heat maps.
Artemis now has a built-in automatic analysis capability for SNP density, read counts and RPKM (see Section 2.8.1) limited to selected genes. Thus, this analysis capability is not designed to scan a complete genome and summarize the results.

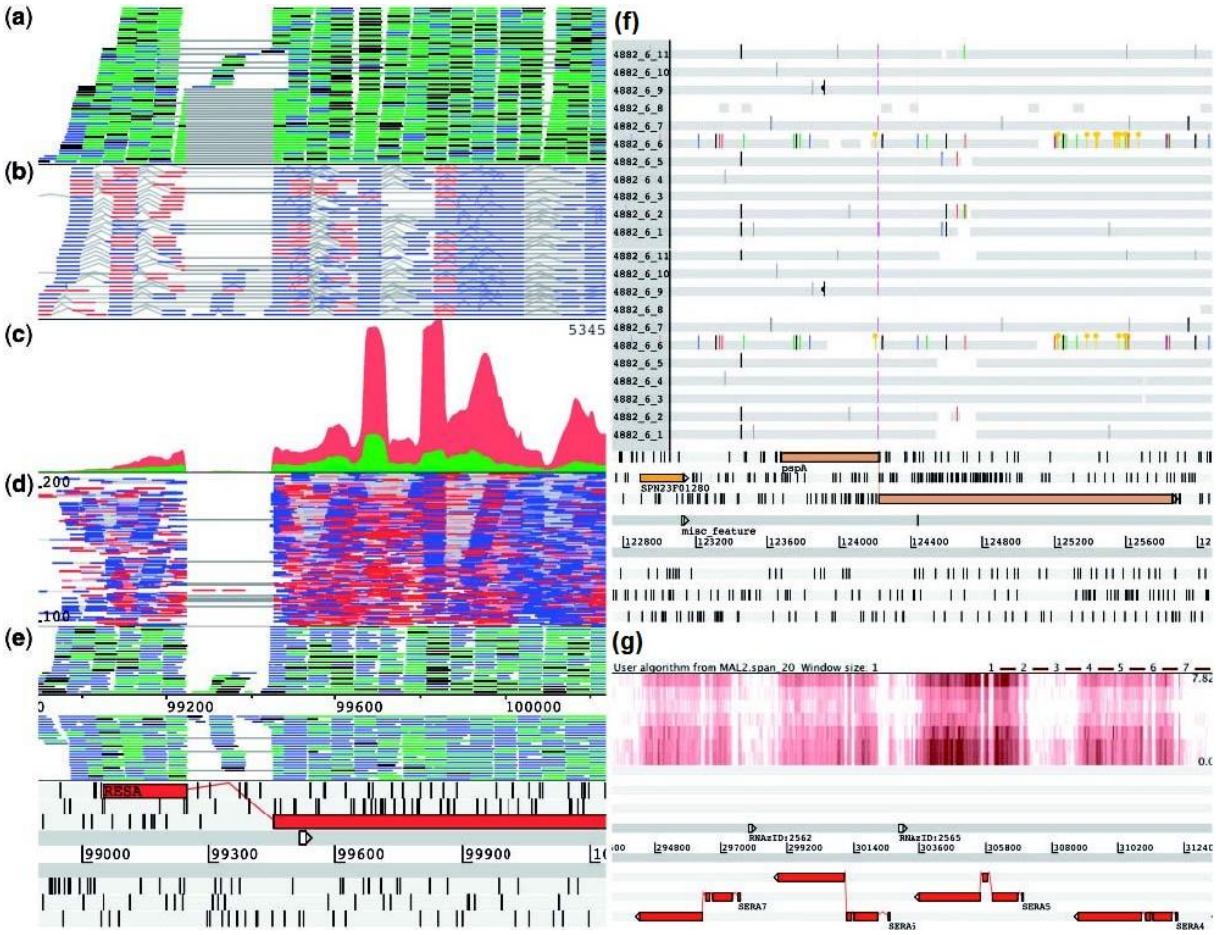



**Figure 35: Genomic and read mapping visualizations in Artemis. (a)** The 'stack' view (paired reads are blue, multiple reads spanning the same region are green, single reads or reads with an unmapped pair are black). Alignment blocks across splice junctions are joined with a grey line. (**b**) The 'paired-stack' view (inverted reads are red). (**c**) The coverage view with a separate plot for each BAM. (**d**) The 'inferred size' view, plotting read pairs along the y-axis by their inferred insert size (or optionally the log of this). (**e**) The 'strand stack' view, with the forward and reverse strand reads above and below the scale, respectively. At the bottom of **(e)**, **(f)** and **(g)** the six frame visualization of the genomic annotations are shown. **(f)** Shows an unfiltered (upper) and a filtered (lower) VCF panel above the reference. Insertions are shown in magenta. Premature stop codons are depicted by yellow circles. **(g)** Shows a user-defined input from a wiggle file, which is visualized as heat map. In this case, the wiggle heat map is used to visualize expression levels of a sample over seven time points (top-down). Figure and description are adapted from (Carver *et al.*, 2012).

## 3.6. Mapping quality and quantity measures in SAM format

The SAM format (H. Li *et al.*, 2009) is the *de-facto* standard for read mapping data. It contains several measures related to mapping quality and quantity. In the following, an overview is given on these measures and their availability and applicability in the previously presented visualization software (see Sections 3.1 - 3.5).
Data stored in SAM records may contain the following quality and quantity information:

- An optional PHRED (see Section 2.1 and (Ewing and Green, 1998)) based **mapping quality** as set by the alignment software if the software supports mapping quality calculation. Otherwise, the mapping quality is set to "255". Thus, mapping quality information is not necessarily given for each mapping data set. Only Savant has the ability to shade mappings by mapping quality and Savant, Artemis and IGB can filter mappings by their mapping quality.
- A flag for mappings **not passing quality controls**: This is a mapper specific flag which also does not have a unique definition. Generally, it should be set by the mapper if a read in the input data set is flagged as not passing the quality controls from the sequencer. But again, mappers might use this flag also for other internal quality controls or not at all. Thus, users cannot rely on this flag for every data set, especially, if it is not clear which mapper produced the alignments or how exactly the mapper implements handling of this flag. Savant and Artemis support filtering mappings by this flag.
- An optional **base quality** field which is labeled by "*" if the quality is not stored. Otherwise, the field contains the original base qualities obtained from the sequencer output file. If available, these qualities can be used for position specific quality checks. Savant, Artemis and IGB are able to color mappings by position specific base quality, while only IGB allows filtering of mappings by base quality.
- A **secondary alignment** flag: By convention, only one of the mappings of a read mapped multiple times to the reference should have the secondary alignment flag unset - flagging it as primary alignment. But, as this is a mapper specific flag, it does not contain information on why an alignment has been selected as primary alignment. As an illustrative example, consider a read mapping twice to the reference without mismatches. Which of the mappings should gain the primary alignment flag? Only Artemis supports filtering mappings by this flag.
- Additional optional **predefined tags** from the SAM-specification (H. Li *et al.*, 2009) can be used, but again, there is no guarantee that the fields have been filled by the read mapper as expected. The terminology of these tags is their composition from two characters, a capital letter and a second capital letter or number. Among the presented tools, only IGV has the capability of coloring mappings by any of the SAM-tags. Filtering mappings based on a SAM-tag is not implemented in any of the genome browsers. In the following table, tags among the list of predefined tags concerning quality and quantity information of a mapping are listed.

**Table 2: Predefined SAM tags concerning mapping quality and quantity.**

| SAM Tag | Description |
|---|---|
| **NM** | It can be used to store the edit distance of a mapping to the reference, excluding clipped bases. |
| **MQ** | It is a quick access point to learn the mapping quality of the mate for read pair data sets. |
| **NH** | It can be used by the mapping tool to store the number of reported alignments for a read. Not all of these alignments might be stored in the resulting SAM-formatted file. Due to this fact, the next tag is important here. |
| **IH** | It can be used to store the number of mappings of a read actually stored in the SAM-formatted file by the mapper. In combination with the "NH" tag, these two tags enable distinction of reads mapped multiple times to the reference and uniquely mapped reads when used. |
| **H0, H1, H2** | They are designed to store the number of mappings of a read with an edit distance of 0, 1 and 2, respectively. While it is beneficial to know how many perfect matches a read has, the combination of these flags does not enable identifying the total number of mappings of a read. All mappings with more than two mismatches cannot be represented here. |
| **CP, CC** | They allow the localization of the next mapping for reads mapped multiple times to the reference. **CP** depicts the leftmost coordinate of the next mapping for a read, while **CC** contains the name of the reference sequence on which the next mapping is located. This can be different if the read is mapped to another chromosome, contig or scaffold from the same overall reference. If these tags are available, they allow fast enumeration of all mappings of a read. |

In summary, several mapping quality and quantity related fields are available in the SAM format, but none of them is mandatory. Hence, users cannot rely on the presence of the needed flag. None of the presented tools has the capability of either visualizing or filtering mappings based on all mentioned measures. Artemis has the widest range of mapping filters and IGV is the only tool supporting coloring of mappings by one user-definable SAM tag at a time.

# Chapter 4

# Motivation and Goals of the Dissertation

*Pseudomonas aeruginosa* is a ubiquitous pathogenic bacterium involved in several prevalent diseases including chronic as well as perilous diseases (Ramos, 2004-2010a, 2004-2010b). *P. aeruginosa* is able to infect a wide range of hosts including humans, mammals, insects and plants. Thus, it is of great importance to learn more about this pathogen.
On this account, representatives of the 15 most abundant *P. aeruginosa* strains (Wiehlmann *et al.*, 2007) and 5 strains from common clones without any known infection history have been chosen for a state-of-the-art pan genome analysis (Hilker *et al.*, 2014), in which this dissertation is involved. The study aims at a better understanding of the causes, developments and the distribution of pathogenicity within bacteria and especially *P. aeruginosa*.
The NGS techniques described in the previous two chapters are well suited to elucidate the pan genome of the 20 *Pseudomonas aeruginosa* strains from a genomics point of view and to analyze their phylogeny in relation to their pathogenicity. Ultimately, these analyses will help to fathom if there is a relationship between pathogenic genome content and the local origin of a strain.
The following workflow was developed for the analysis (see Figure 36):
The first step in the analysis of the raw reads from the sequencer (see Section 2.1) is the read quality assessment (see Section 2.2). Afterwards, two main analysis approaches should be adopted:

1. *De novo* genome assembly (see Section 2.3) and automatic genome annotation (see Section 2.4) followed by comparative analyses of the pan genome (see Section 2.5).
2. Mapping of the sequencing reads onto an already finished and well annotated *P. aeruginosa* reference strain and common genomic islands (see Section 2.6), followed by automatic phylogenetic analyses of SNPs and DIPs (see Section 2.7.1) and the coverage of the genomic islands, revealing the absence, total or partial presence of the island in question (see Section 2.6.1).

For workflow 1, specialized tools can be chosen from a variety of established tools (compare Sections 2.2 - 2.5). After the mapping process in workflow 2 on the other hand, several different tools and formats need to be combined and not all solutions are optimal (compare Sections 2.7 and 2.8 and Sections 3.1 - 3.5). To make important regions of a genome or transcriptome instantly accessible, automatic analysis functions are required. Nowadays, these functions are mostly implemented separately from the context visualization (compare tools mentioned throughout Sections 2.7 and 2.8).
Visual exploration is clearly a major benefit to textual comparison of data, but still, each position of huge genomes has to be inspected manually for events of interest in simple genome browsers. This is a tedious and interminable task which is not feasible for whole genome analyses due to its huge time consumption and high personnel costs.

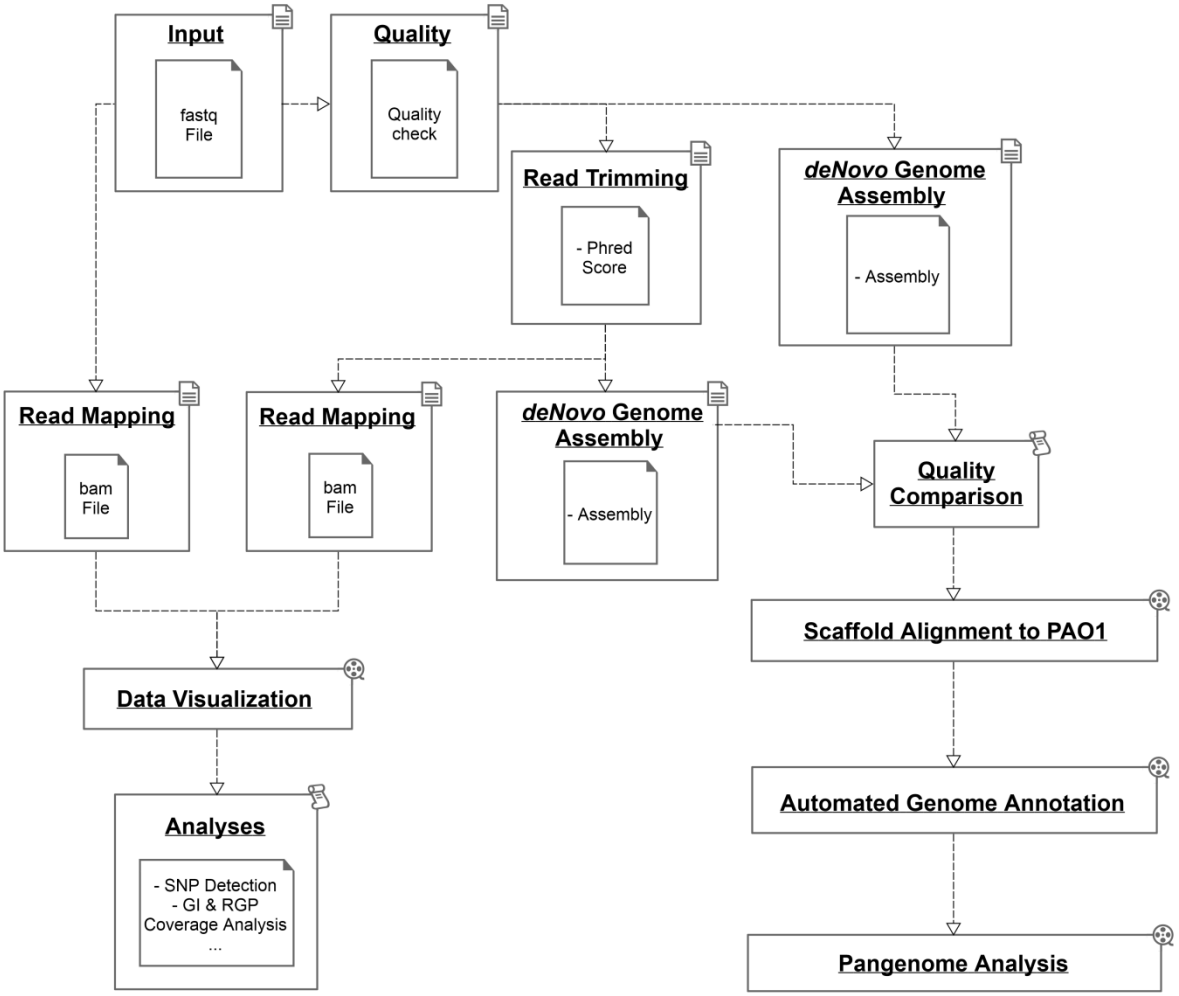



**Figure 36: *P. aeruginosa* pan genome analysis workflow.** The figure visualizes the processing steps required to analyze the pan genome of the 20 *P. aeruginosa* strains of this study. It starts with the input of raw sequencing reads in fastq format and then follows two routes: 1. *de novo* assembly and subsequent pan genome analysis of the assembled and annotated genome sequences (right path), 2. Mapping of the sequencing reads onto the genome sequence of the reference strain PAO1 followed by subsequent analyses (left path).

The identification of this problematic gap gave the motivation to overcome it in the presented dissertation by developing the software ReadXplorer. It combines user-friendly visualization with automatic analysis functions. In addition, the software was meant to carve out comprehensive detailed knowledge about the read mapping data and offer novel features easing their evaluation. The main goal was to largely reduce manual effort for the user and prevent spending time on correctly installing about ten different tools and their dependencies by offering all relevant functionality from one source. Naturally, the *P. aeruginosa* pan genome analysis can greatly benefit from such a novel tool.

The SAM format (H. Li *et al.*, 2009) is the *de-facto* standard for read mapping data. It contains several measures related to mapping quality (see Section 3.6), but none of them is mandatory and thus, data availability depends on the implementation in the read mapping tool employed.

For reliable analyses the read mapping classification is a crucial step, because sequencing reads often map to more than one region of the reference sequence with different amounts of mismatches, for instance in repetitive regions. It is not an option to simply configure the read mapping software (see Section 2.6) to only output one of the possible alignments of a read, because there is no guarantee that the returned mapping is the optimal or the only optimal mapping. Accordingly, not all mapped reads are equally trustworthy for each analysis, but these reads might still be useful for other analyses. At least, all read mappings with a predefined error threshold should be visible to the analyst. Thus, a highly beneficial feature is

a clear classification of mapped reads by mapping quality and quantity even after the mapping process and independent of the mapper. Incorporating this classification into user-friendly and intuitive visualizations is a major improvement in contrast to available mapping visualizations, facilitating a clear visual distinction of mappings with different quality and quantity properties.
With current applications, users requiring a combination of certain mapping properties are forced to use multiple tools and elaborately compare their visual output. Further, it is not possible to select mappings based on any of the criteria mentioned in Section 3.6 for automated analyses in the presented genome browsers. Thus, making mappings selectable for each single automatic analysis function based on their classification will ease the execution of multiple analyses and analyses with a different scope. In addition, it will enable much more specific, dynamic and comparable analyses. Another advantage of this approach is the ability to run the same analysis with different mapping classifications. Afterwards, the impact of the selection of different classes can be investigated.
A genome browser offering an optimal solution for this issue needs to bundle the most relevant quality and quantity measures in an intuitive and clear classification of the read mappings. Bundling interrelated properties has the advantage of superseding tedious manual configuration of several properties and increasing usability. This classification should be presented in visualizations as well as be selectable for performing analyses. Additionally, further fine tuning of the properties should be possible for advanced users.
Since RNA-seq is still a relatively new technique, no tool was available to automatically detect TSS and novel transcripts from RNA-seq read mapping data when this work was started. Naturally, a tool automatically detecting TSS is of great value for the scientific community. Therefore, one goal for the development of ReadXplorer was to be able to detect TSS, novel, antisense and small RNA transcripts. This feature enables correction of gene annotations, new gene annotation and identification and confirmation of small RNAs.
None of the programs presented in Chapter 3 is capable of automatically identifying possible genome rearrangement events in mapping data sets (see Section 2.7.2) supported by a certain number of discordant read pairs. Integrating this functionality in a platform independent read mapping browser could significantly alleviate fast processing of such events by a researcher and overcomes the obstacle of being forced to learn handling of command-line tools or being restricted to a UNIX-like operating system. The most suitable solutions for genome rearrangement detection are either to integrate a suitable tool mentioned in Section 2.7.2 or to re-implement one of their algorithms.
Visualization of read pairs is another important feature of a read mapping visualization tool. Among the five presented tools (see Sections 3.1 - 3.5), Savant offers the best in depth classification of read pairs, but there is no tool available distinctively visualizing discordant pairs being too large and pairs being too small. The latter classification is especially relevant when large mate pairs (e.g. with an expected insert size of 10kb) are analyzed. Highlighting pairs, which have a much smaller distance, indicating an insertion in the analyzed mapping data set, in a distinct color could further enhance fast perception of such events and improve the user experience.
In today's high-throughput experiments mostly multiple data sets need to be analyzed, often requiring direct comparison of two or multiple data sets. None of the presented viewers offers such a comparative visualization for read mapping data, apart from visualizing mapping data sets one below the other. This issue should also be addressed by ReadXplorer.
Another task for working across multiple data sets is their combination while retaining the separate original data sets. When multiple RNA-seq replicates are available for a certain environmental condition, it is useful to be able to combine them for analyses other than differential gene expression. Two examples are SNP detection within transcripts and operon detection across replicates. Artemis and IGV both have the ability to combine multiple data

sets in their visualizations, but none of the presented tools can utilize this function for automatic analyses.
Appropriate selection of the genetic code for the organism of interest is important for all visualizations and analyses involving DNA translation. For instance, a six frame view, showing all possible translation frames of a DNA sequence or amino acids affected by a SNP in a SNP detection should be able to adapt their underlying translation table according to the organism under investigation. Such functionality is only implemented in IGV for the visualization of the three reading frames of the currently selected DNA strand. Therefore, a more comprehensive integration of this feature in results of automatic analyses is desirable to provide interpretation-ready results.
Another helpful visualization is the display of start and stop codons of the active genetic code. Especially, when analyzing RNA-seq data for complete transcripts, displaying interactive start and stop codons facilitates their fast perception, assessment and transcript assignment. Codon interactivity enhancing the user experience could include highlighting of the sequence from a start to the next in-frame stop codon, exporting and translating the corresponding sequence. A simple visualization of start and stop codons is only implemented in GenomeView, but they cannot be adapted to fit different genetic codes and they are not interactive.
In summary, the main goals for the development of ReadXplorer were:

- A user-friendly, platform independent application with minimal installation requirements, minimal external dependencies and functioning without an internet connection
- Extensible and modular programming structure
- Efficient and intelligible visual exploration of read mapping data and reference genomes
- Detailed classification of read mapping data by mapping quality and quantity and its distinct visualization
- Use and configurability of the classification for all automatic analysis functions
- Development and implementation of an automatic TSS and novel transcript detection method
- Enable calculation of normalized read counts for RNA-seq data
- Combine existing DE methods with immediate read mapping visualization
- Enable detection of operons from RNA-seq data
- Enable SNP and DIP detection
- Enable analysis of coverage
- Combine genome rearrangement detection with immediate read mapping visualization
- Possibility to automatically fold RNA sequences and visualize the result
- Read pair visualization distinguishing all different read pair configurations (concordant, too small, too large and wrongly oriented pairs)
- Enable comparative visualization of two data sets
- Enable combination of multiple data sets in visualization and analysis while retaining the original data sets
- Enable selection of the genetic code of the investigated organism taken into consideration in visualizations and analyses

# Chapter 5

# Software Development for Read Mapping Data Analysis - ReadXplorer

In Chapter 4 the capabilities of current read mapping visualization tools are closely analyzed and a problematic gap between visualization and automatic analysis functions for read mapping data is identified. Additionally, several visualization and analysis features can be enhanced and improved for higher specificity, user-friendliness and more details.
This chapter addresses these matters by presenting ReadXplorer (Hilker *et al.*, 2014), a novel software for user-friendly visualization and comprehensive analysis of read mapping data (see Section 2.6 for read mapping details).
The content of this chapter is based on the aforementioned ReadXplorer publication and divided into three sections, starting with ReadXplorer's system design in Section 5.1, followed by the stream-driven data model and modular software architecture in Section 5.2. In Section 5.3, the implementation of automatic analysis functions is explicated. Section 0 harbours an evaluation of ReadXplorer, comparing it against the other genome browsers introduced in Chapter 3 and revealing its performance on large NGS data sets.

## 5.1. System Design

The aim of the software development project for ReadXplorer was to provide a comprehensive open source desktop client application attaining a wide range of users. Therefore, it is crucial for the software to run on all major operating systems such as Windows, Linux or Mac OS. Especially for inexperienced users, the installation effort has to be kept minimal. Overcoming both of these barriers simplifies the work in multi-national teams at different locations around the globe with different operating system preferences and experience levels. Additionally, a high level of flexibility and independence shall be granted to the users by developing a software usable on almost any ordinary desktop computer or laptop and even without continuous internet access.
Adhering to these requirements, ReadXplorer (see Figure 37) is implemented as a desktop client in the platform independent programming language Java 1.7 using the Netbeans IDE (Integrated Development Environment). Using this IDE enables implementation of software based on the Netbeans rich client platform[12].
The modular programming structure of a Netbeans rich client application significantly simplifies maintaining and extending the software. Additionally, it eases programming

[12] http://netbeans.org/features/platform

flexible graphical user interfaces (GUIs). One virtue is the Netbeans modular docking framework for all GUI components. All panels can always be maximized, minimized and restructured in various ways, enabling the user to perform simultaneous comparisons of different data sets or browse different genomic regions of the same data set (see Figure 43). Furthermore, the last user-defined layout of the visual components is always restored after restarting the software.

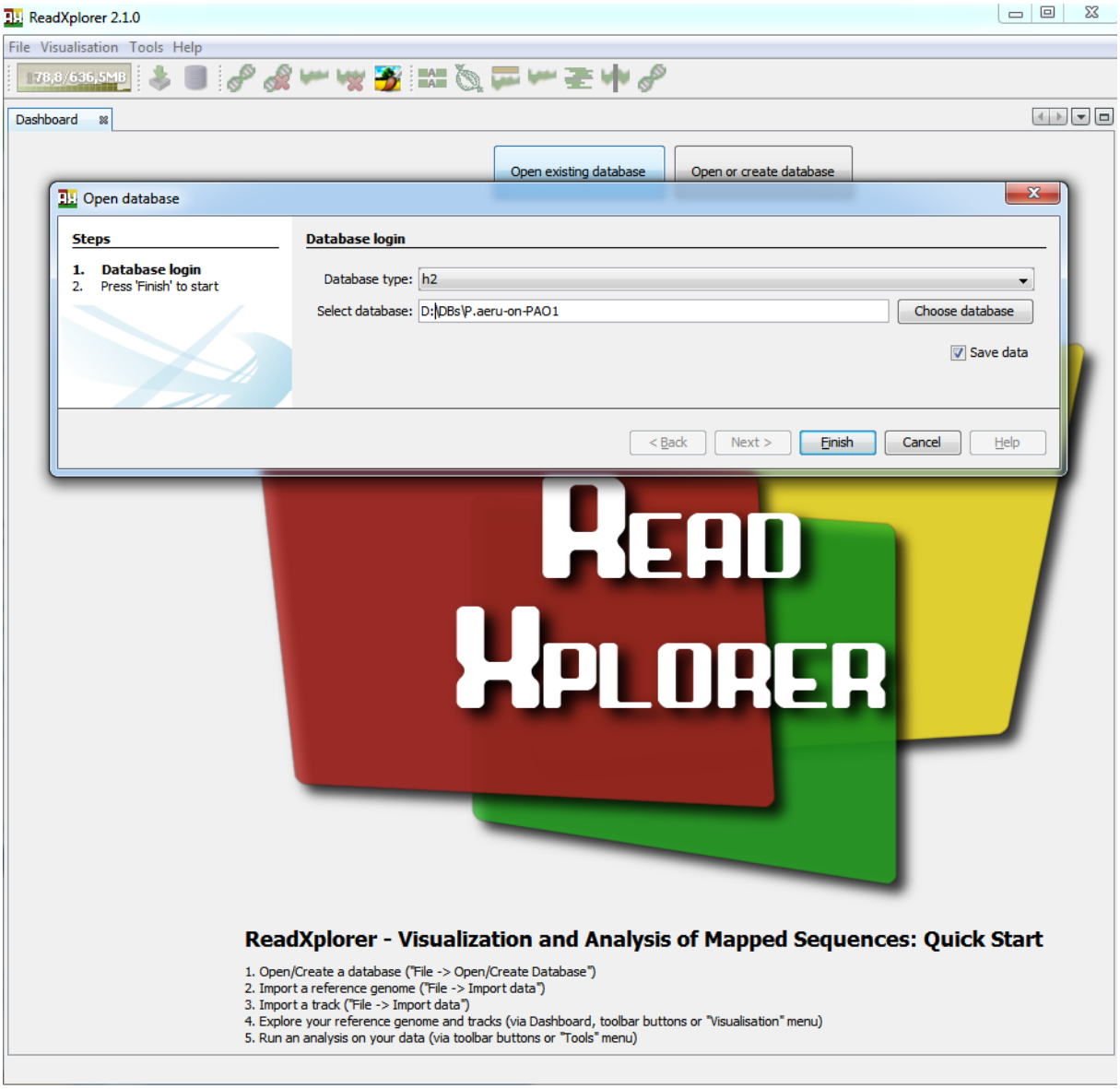



**Figure 37: ReadXplorer start-up screen.** By default, ReadXplorer starts with its dashboard, displaying the ReadXplorer logo, buttons to open or create a project database and useful quick start hints. Here, "Open existing database" was clicked and the corresponding wizard is displayed in the foreground. When a database has been chosen, the wizard page conveniently displays the selected database path. Clicking finish will open the database or create a new and empty database at the specified location.

All features of ReadXplorer are bundled in the form of individual Netbeans modules (NBMs) in a Netbeans module suite. This modular structure can be further enhanced by new module suites and modules on demand.
Each new module is embedded into ReadXplorer as plug-in via the integrated Netbeans plug-in manager. This allows for easy extension of the rich client application: new NBMs developed by native or external programmers only have to be placed into ReadXplorer's update folder. This can be done either by automatic update checks of the central ReadXplorer

update center[13] from within the software or by manually placing own modules in that folder. During the next start of the software, they are immediately integrated in the current ReadXplorer version. This behaviour guarantees easy distribution and integration of new plug-ins, which fit highly specialized user needs.
Further, the software is composed of a classical three-tier architecture, of which a high-level overview is given in Figure 38. User requests for any kind of data are received by the application tier and handled by the business tier, whereas the persistency tier is subsequently responsible for executing necessary data queries. When a query succeeds, the persistency tier passes the result in generic data containers to the business tier. After appropriate processing, the data is displayed by the application tier. This behaviour guarantees independency and exchangeability of each tier and its components.
To make ReadXplorer available and extensible for everyone, it is released under the GNU General Public License (GPL) version 3[14].
ReadXplorer provides a well-defined application programming interface featuring classes for the development of specialized data viewers and analysis functions (see Appendix, Section 10.2).

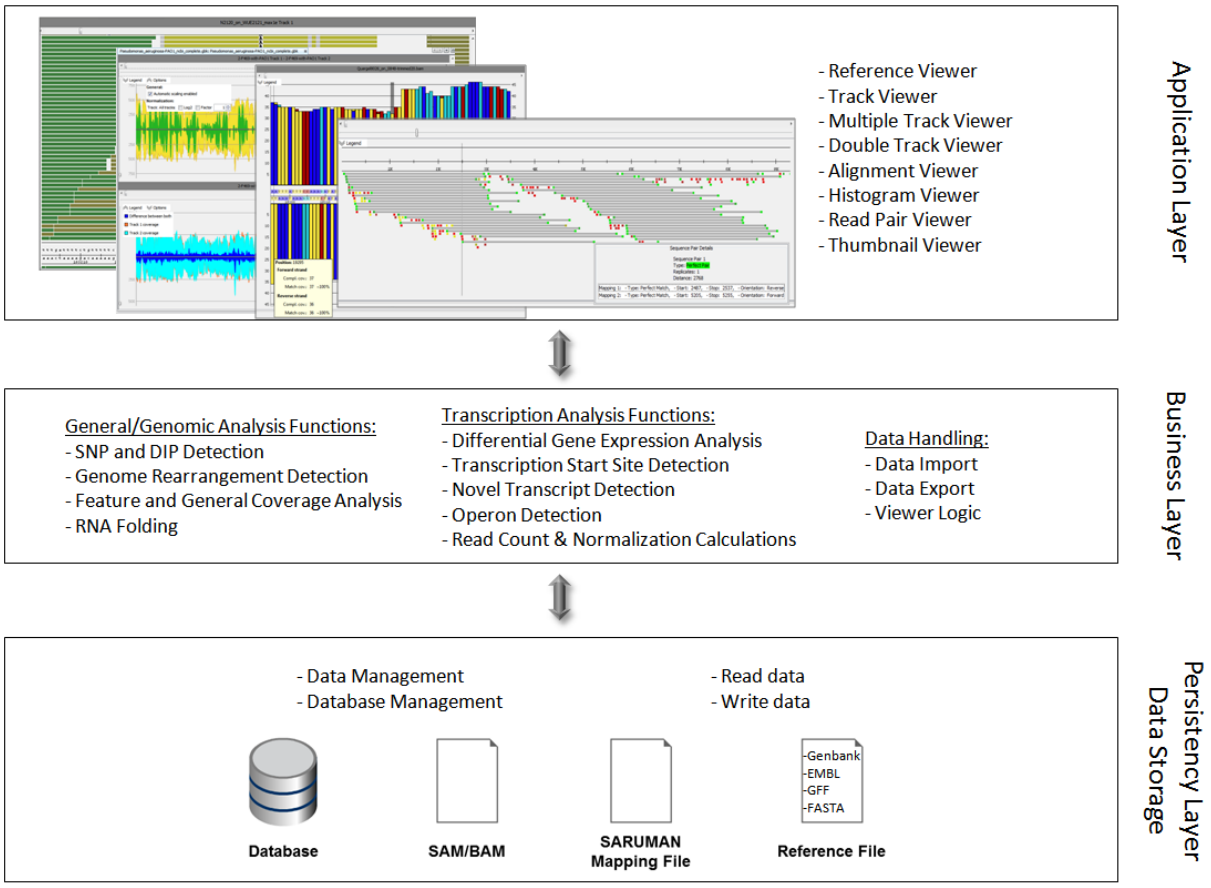



**Figure 38: ReadXplorer tiers.** This figure displays the global three-tier architecture of ReadXplorer. The application tier is responsible for data visualization and user input, whereas the business tier contains all logic associated with analysis functions and data import and export. The NGS data are maintained, written and read by the persistency tier. It either interacts with a database, a file or both.

Most parts of the basic viewer and data storage concept have been severely adopted, purged and updated from the first ReadXplorer prototype - called VAMP - described in the master's thesis of Daniel Doppmeier (Doppmeier, 2009).

[13] ftp://ftp.cebitec.uni-bielefeld.de/pub/readxplorer_repo/update/updates.xml
[14] www.gnu.org/licenses/licenses.html

The reference data formats currently supported by ReadXplorer are EMBL[15] (Stoesser *et al.*, 2002), Genbank[15] (Benson *et al.*, 2014), GFF2[16]/GTF[17], GFF3[18] and FASTA[19]. Read mapping data has to be presented in SAM, BAM (H. Li *et al.*, 2009) or in SARUMAN output format JOK (Blom *et al.*, 2011). Additionally, ReadXplorer supports importing tabular data in CSV[20] or Microsoft's XLS[21] format and specialized nucleotide polymorphism data in VCF (see Section 2.7.1). Section 5.2.1 describes how this data is integrated and treated within the software. The contents of a database are organized in and quickly accessible via a dashboard in the GUI (see Figure 39).

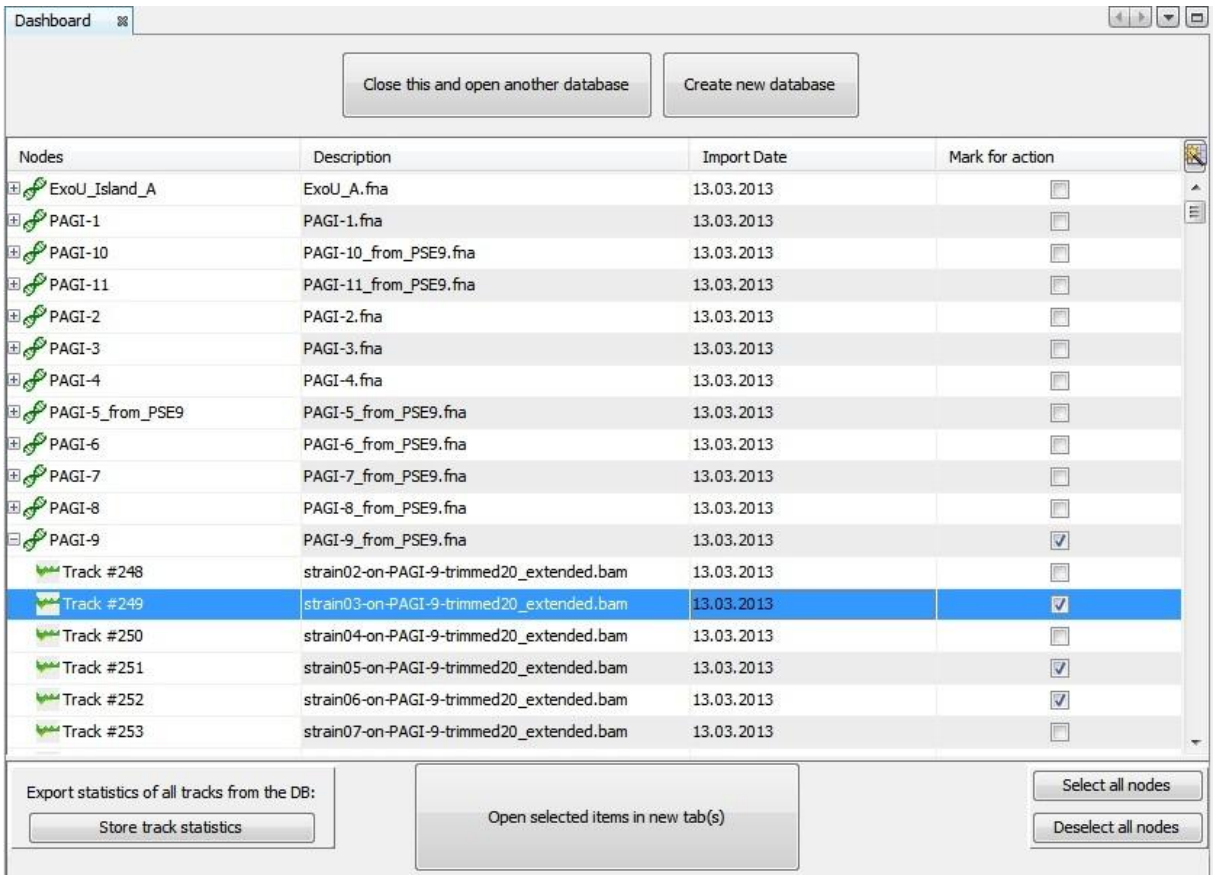



**Figure 39: ReadXplorer dashboard.** The dashboard hierarchically displays the references and tracks stored in the database. To select a data set, a click anywhere on the corresponding data row suffices. Afterwards, all selected data sets are opened by clicking the prominent "Open selected items in new tab(s)" button.

[15] www.insdc.org/files/feature_table.html
[16] www.sanger.ac.uk/resources/software/gff/spec.html
[17] mblab.wustl.edu/GTF22.html
[18] www.sequenceontology.org/gff3.shtml
[19] fasta.bioch.virginia.edu/fasta_www2/fasta_guide.pdf
[20] tools.ietf.org/html/rfc4180
[21] msdn.microsoft.com/en-us/library/cc313154%28v=office.12%29.aspx

## 5.2. Data Model and Software Architecture

### 5.2.1. The Persistency Tier

Redundant calculations for the same experiment have to be avoided. Following this paradigm, all data which shall be analyzed with ReadXplorer undergoes a nonrecurring preprocessing step, producing a valid data basis for all users (see Section 5.2.2, Data Import, Read Mapping Classification and Read Pair Classification). Delimitation of experiments in ReadXplorer is realized by a project based data management approach, storing all relevant data in a database which is a purged version of the concept introduced in (Doppmeier, 2009). In VAMP, all read mapping data was stored in the database, while ReadXplorer directly reads all reference sequences and mappings from indexed files. This change results in a much better performance: i.e. it took a whole day and 80GB of RAM to import the *P. aeruginosa* strain F469 mapping data set (see Table 15), while now the same task is accomplished on an average laptop in ~45min with ~900 megabyte (MB) RAM (see Table 10).
Based on the general three tier architecture of ReadXplorer (see Figure 38), input data first has to be introduced to the NBMs associated with the persistency tier (see Figure 41) for adequate processing and storage in the database of the current project. The data import process starts with the reference annotations and sequences needed for the current experiment. After storing them in the database, tracks can be imported for each reference. This data is made persistent and can be retrieved by any user of the database for viewing and analyzing the data.
Two open source relational database management systems (RDBMS) are supported: H2[22] and MySQL[23]. During this work, H2 has been used almost exclusively, because it is file-based, thus guaranteeing flexibility for users. An H2 database can easily be shared with other collaborators around the globe by copying and opening it anywhere where the ReadXplorer software is available, without the necessity of running a database server.
In the Java Platform Standard Edition (Java SE) relational databases can be handled and accessed natively by the standardized Java Database Connectivity (JDBC) interface[24]. Corresponding open source JDBC drivers for H2 and MySQL are available on the respective homepages and have been used to realize this software project.
The schema of the ReadXplorer database is shown in Figure 40. The underlying concept divides the database into a reference and a track domain, containing reference specific and track specific data (see description of Figure 40 for details).
The responsibilities within the persistency tier are clearly assigned to maximize maintenance, provide application wide access to this main component and an intelligible application programming interface (API). The core of the persistency tier is the singleton `ProjectConnector` (see Figure 41). The responsibilities of this class comprise handling of the central database connection, storing and managing data.

[22] www.h2database.com
[23] www.mysql.com
[24] www.oracle.com/technetwork/java/javase/jdbc/index.html


**Figure 40: ReadXplorer database schema.** The tables can be separated into a reference and a track domain. The first concerns the reference sequences. The REFERENCE table stores the main data of the reference including the corresponding sequence file (in FASTA format). The CHROMOSOME table contains information about all reference chromosomes or sequences. The FEATURE table stores the most relevant information of all annotated reference features. The track domain concerns the tracks. The TRACK table associates each track to one of the references in the database and identifies its corresponding read mapping file (in BAM format). The COUNT_DISTRIBUTION table holds pre-calculated binned distributions for each track which can be used by analysis functions and for displaying statistics. The STATISTICS table contains pre-calculated global statistics for each track (see Figure 43, l for the visual representation). A database version table containing the current database version aids tracking the version number of the database and performing version specific updates for downward compatibility, if necessary.

The `ProjectConnector` grants access to the reference and track domains via a `ReferenceConnector` and a `TrackConnector`, respectively (see Figure 41). Each reference and track possesses its own connector instance. A `ReferenceConnector` grants access to the sequence, all features and properties of the associated reference, while the `TrackConnector` grants access to all properties and the read mapping data of the associated track.

Prior to storing data, it is parsed by the corresponding parser from the parser module. To guarantee exchangeability and extensibility, all parsers in ReadXplorer implement the `ParserI` interface and its corresponding sub-interfaces (see Figure 41).

ReadXplorer stores all sequences belonging to a single reference in an indexed (multiple) FASTA file. All read mapping data is stored in a sorted and indexed BAM file (see Section 5.2.2, Data Import and Read Mapping Classification). For creating these files during or after parsing, specialized methods and writers have been implemented in the parser module (see Figure 41).

Both the sequences belonging to a reference and the read mappings are directly read by their connectors from their indexed file, whose location is stored in the project database. E.g. read mapping data is obtained using either the `CoverageThread` or the `MappingThread` classes. These two threads have been introduced to ensure responsiveness of ReadXplorer during data queries and encapsulate queries to the actual data reader class, the `SamBamFileReader` (see Figure 41). The queries itself are standardized by the highly configurable `IntervalRequest` class. All methods requesting data have to create such an object and, if necessary, assign filter criteria (e.g. to only return data from certain read mapping classes (see Section 5.2.2, Read Mapping Classification)) for the data to obtain.

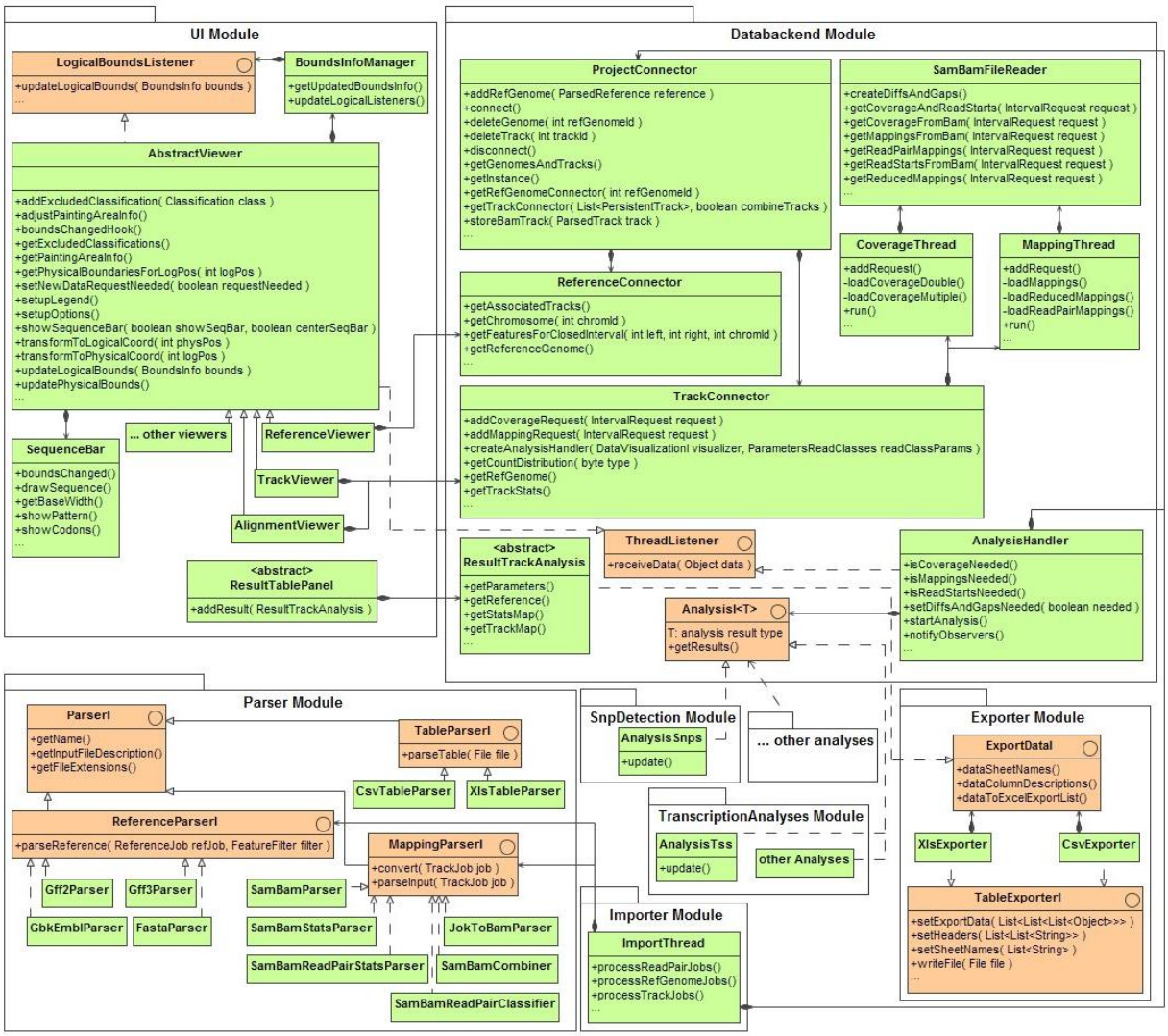



**Figure 41: UML class diagram of ReadXplorer core features.** The most relevant classes and methods of the ReadXplorer core are shown. The importer module is the starting point for importing new data into the software and requires the persistency tier modules parser and data back end. The application and business tier UI module takes care of the visualizations and receives the data via the data back end module. Analysis modules like the SnpDetection module receive the data to analyze from the data back end module as well. To visualize the results, analysis modules implement a `ResultTrackAnalysis` (connection and classes not shown to preserve a clear arrangement), which directly supports export of the results.

The `CoverageThread` is designed for retrieval of coverage or read starts, while the `MappingThread` is designed for retrieval of complete alignments. The corresponding data holder classes for coverage, read starts (`Coverage`) and complete alignments (`Mapping`) provide information on the quality classification of the contained data, easing further filtering of the data.

Using the `IntervalRequest`, the two threads and data holders in combination with the `SamBamFileReader` is a powerful concept to grant fast and generalized access to a genomic interval of interest and its associated data filtered by desired parameters. For fast access, the interval size has to be constricted to a reasonable size (e.g. 100bp-100kb). The maximum viewable interval size in a viewer is between 50 and 100kb, depending on the screen size.

### 5.2.2. Business and Application Tiers

The business tier takes care of four main tasks: Data import, data export, providing the logic for data viewers and analysis functions. The application tier consists of the GUI implementations using the logic provided by the business tier. Since these two tiers work hand in hand, their description and illustration is combined in this section.
A central aspect and major feature of ReadXplorer is the quality and quantity classification of read mapping and read pair data. This classification is carried out once during import of the mapping data. Therefore, this section starts with the import process, followed by sections defining the read mapping and read pair classification. Afterwards, the implementation of each of the remaining three main tasks is presented in their respective sub-sections.

#### Data Import

The central data import actions and functionality are bundled in the importer module (see Figure 41). They become selectable within ReadXplorer's toolbar and "File" menu (see Figure 37), when the user is connected to a database. A well-structured wizard guides the user to a successful import of reference or read mapping data, while a second wizard provides the option to import any kind of table (in CSV or XLS format, as mentioned in Section 5.1) into ReadXplorer. A third wizard is available for VCF (see Section 2.7.1) import of single nucleotide polymorphism data.
The import of references in EMBL, Genbank, GFF2/GTF or GFF3 format is realized by parsers, which use the Biojava 1.8.1 library[25], bundled with ReadXplorer, to get quick access to all sequence features.
Within ReadXplorer, mapping data sets are called **tracks**. The import of read mapping data is divided in "Track" and "Read Pair Track" import. The read pair track import is designed for mate pair and paired end data and includes specification of fragment length and allowed variation of the insert length (see Figure 42). Single end data sets are imported using the standard "Track" import. To be able to handle mapping data sets in SARUMAN output format, a converter has been implemented which converts JOK files into BAM formatted files. However, it is not only available as integrated version for ReadXplorer, but has also been compiled in a stand-alone application for end-users. During import of read mapping data, the additional read mapping classification (see subsequent sections "Read Mapping Classification" and "Read Pair Classification") calculated by ReadXplorer is stored in a BAM file (Li *et al.*, 2009) which is an extended copy of the original import file (see Table 4 for an example).
After completion of the reference and track import wizard, the `ImportThread` class (see Figure 41) executes the import process in a new thread to guarantee responsiveness during the computational intensive import. Parsing is delegated to the respective parsers and initiates subsequent storage of the data by calling the appropriate methods in the `ProjectConnector`. The read mapping parsers also create the extended BAM file with ReadXplorer's quality and quantity classification, which is explicated in the following two subsections.

[25] http://biojava.org/wiki/BioJava:Download_1.8

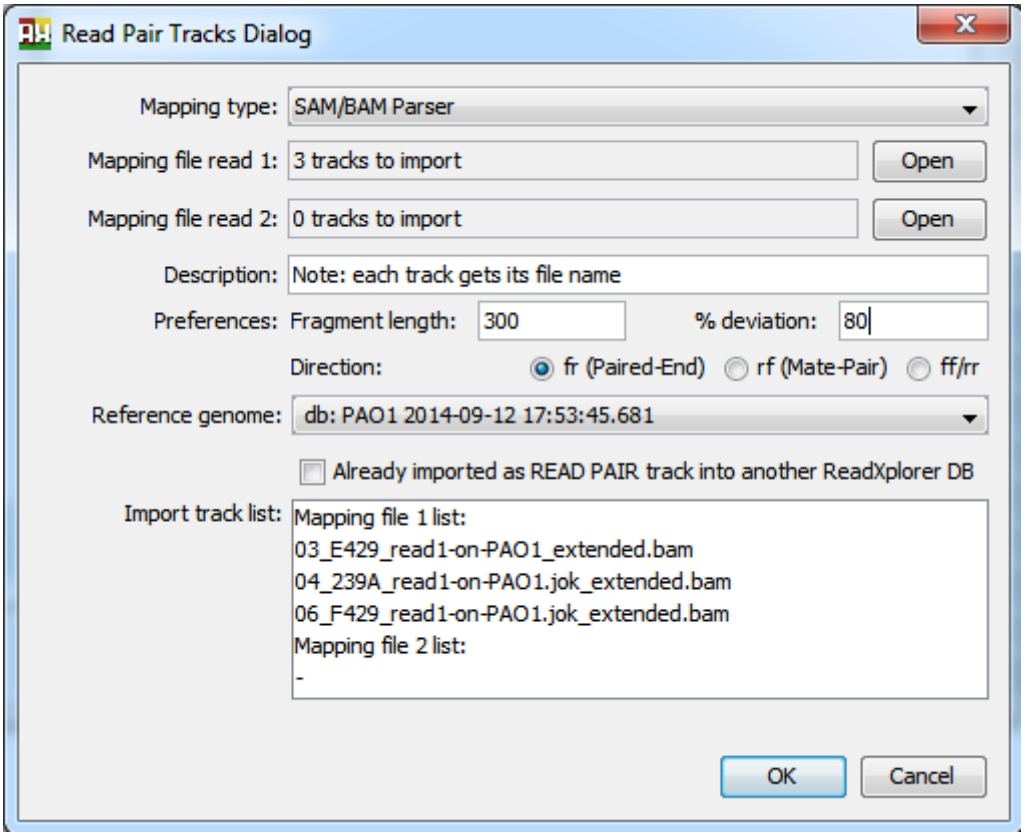



**Figure 42: Read Pair track import dialog.** Multiple tracks can be imported simultaneously and ReadXplorer supports both single file (mostly interleaved read pairs, but the read order does not affect a successful import) and data sets where the first and the second read of the pairs are stored in two separate files. Since the read mapping and read pair classification only has to be carried out once per track, the check box for tracks which have already been imported in another ReadXplorer database on the same reference can be checked and merely the mapping statistics are calculated. This results in a much faster import.

## Read Mapping Classification

As already mentioned, read mapping data has to be provided in SAM, BAM (Li *et al.*, 2009) or SARUMAN (Blom *et al.*, 2011) output format. ReadXplorer classifies the read mapping data during the import process (see Section 5.2.2, Data Import) by mapping quantity and quality measures and read pair concordance.
Therefore, it is important to configure the upstream read mapping tool to output multiple mappings of a read up to a certain error threshold (see Appendix, Section 10.4 for appropriate calls of bwa, Bowtie 2 and SARUMAN). This is of utmost importance, because otherwise one cannot decide whether a read has truly only one valid mapping, or if more mappings exist which were just not calculated.
A read mapped to a certain reference position is called **mapping** in the following. For each read, the number of mappings on the reference is counted along with the lowest number of mismatches found among the various mappings of that read during the import process. Uniquely mapped reads (called **unique mappings**) or reads with a certain amount of mismatches can thus easily be queried. Note that the same read can map to different reference positions with the same number of mismatches. The mappings are classified into one of five classes:

- The **Single Perfect Match** class contains all reads with exactly one (single) perfect match.
- The **Perfect Match** class contains all reads mapped multiple times without any mismatches.

- The **Single Best Match** class contains all mappings that cannot be placed to another position in the reference with the same number or fewer mismatches than at the current position - they exactly have one (single) best match.
- The **Best Match** class contains all reads with multiple best match mappings. I.e. they have multiple mappings to different genomic positions with the same number of mismatches.
- The **Common Match** class contains all remaining mappings.

Distinguishing Single Perfect and Single Best Match mappings from Perfect and Best Match mappings enables quick access to reads with only one Perfect or Best Match mapping from mappings with multiple equally scoring mappings while not requiring the read to be uniquely mapped. Reads from both Single Match classes can have more mappings with lower quality, falling in the Common Match class. Thus, this classification is not as strict as only considering uniquely mapped reads with exactly one valid mapping to the reference.
By showing the mapping classes in the GUI (see Figure 43), the user has the ability to select for each analysis which mapping classes are important and should be included in the analysis. Additionally, this fine grained classification approach allows incorporation of mappings in analyses which would be excluded by other programs.

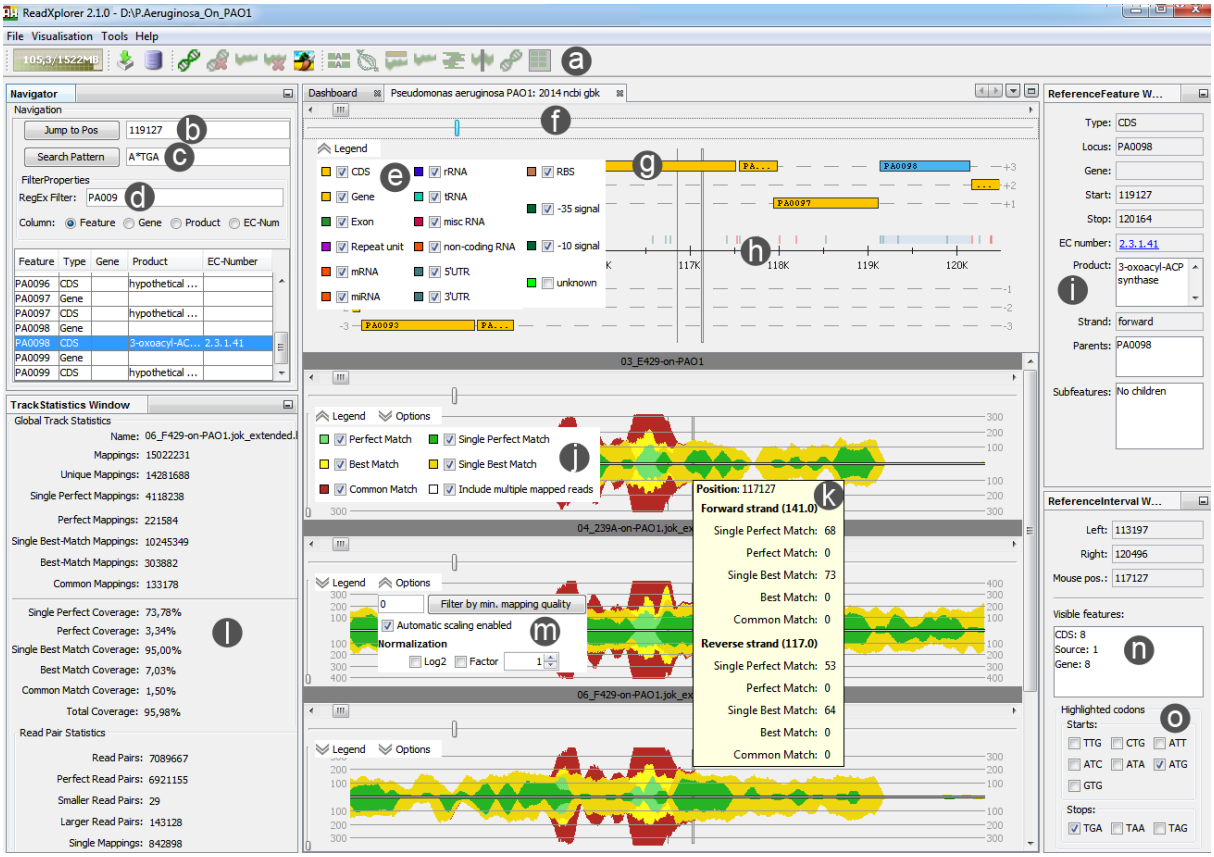



**Figure 43: ReadXplorer main window including mapping classifications.** a) global actions toolbar, b) position navigator, c) pattern filter, d) reference feature filter, e) reference viewer with opened legend panel, f) scroll bar and zoom slider, g) reference features and selected feature (blue), h) sequence bar and interactive start codons, i) reference feature details panel including EC-number link to an enzyme database of choice, j) track viewer with opened legend panel showing the mappings of all different mapping classes in distinct colors, k) track viewer tooltip, l) track statistics panel including mapping, coverage and read pair statistics, m) track viewer options panel, n) reference interval details panel, o) start and stop codon selection panel configurable via the genetic code options (see Figure 69 in Appendix).

The classification data are stored in a BAM file (Li *et al.*, 2009) which is an extended copy of the original file or a converted JOK file (Blom *et al.*, 2011). Subsequently, ReadXplorer only works on these extended files. The additional SAM tags introduced by ReadXplorer in a globally accessible `MappingClass` enumeration enable quick access to the mappings of a

given mapping class and are listed in Table 3. The classification is carried out by the corresponding mapping parser (see Figure 41).

**Table 3: List of ReadXplorer classification SAM tags.** According to the SAM specification, SAM tags starting with 'X', 'Y' or 'Z' or tags containing lower case letters are reserved for end-users. Therefore, tags starting with 'Y' followed by a lowercase letter have been chosen to store augmented mapping information in an extended version of the input file during the import.

| SAM Tag | Value/Range | Explanation |
|---|---|---|
| Yc | 1, 2, 3,4 or 5 | Mapping classification in one of the five mapping classes<br>1= Perfect Match, 2 = Best Match, 3 = Common Match, 4 = Single Perfect Match, 5 = Single Best Match |
| Yt | Number | Number of valid mappings present in the track |
| Yi | Number | Id of the read pair, unique within each track |
| Ys | 0 - 12 | Read pair classification in one of the 13 read pair classes (see Read Pair Classification) |

**Table 4: Example SAM format row of a mapping classified by ReadXplorer.** The four ReadXplorer tags are listed at the end and identify the mapping as uniquely mapped (Yt:i:1) Single Perfect Match (Yc:i:4), being paired with the given id (Yi:i:15323752) in a perfect read pair (Ys:i:7). The read and base quality sequences have been shortened to 15bp for clarity.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| HWI-ST486_0090:5:2205:11613:134447#TAGCTT | 83 | PAO1 | 1019 | 60 | 90M | = | 926 | -183 | CAACCGTACCTTCAC... | @B=EECA9CCDEC@E... | Yc:i:4 Yi:i:15323752 Ys:i:7 Yt:i:1 |

## Read Pair Classification

For paired end or mate pair mappings a read pair classification algorithm was developed which takes into account all occurrences of each read. In the following, a distinction is made between all mappings of the two reads of a pair, which are called **mapping pair**, and two mappings of a mapping pair, classified as **read pair**. In general, the algorithm (see Algorithm 8 and Algorithm 9 in the Appendix) classifies the mappings into three different classes: **Perfect Read Pairs** with correct orientation and a pair distance within a certain range defined by the user, **Distorted Read Pairs**, whose distance deviates from the perfect distance interval and/or whose orientation is incorrect, and **Single Mappings**, whose partner could not be mapped on the reference, or multiple mapped reads of which a mapping cannot be assigned to a read pair (for details see Table 5).
A special case of Single Mappings occurs when one or both reads of a mapping pair also map to other regions of the reference, but they cannot be associated with a Perfect or Distorted Read Pair. There might be more than one Perfect Read Pair for the same mapping pair. Further, the five mapping classes Single Perfect, Perfect, Single Best, Best and Common Match are considered in the read pair identification algorithm implemented in the `SamBamReadPairClassifier` class to improve the accuracy of correct predictions. For each mapping pair, Perfect Match read pairs are preferred to Best Match and both are preferred to Common Match read pairs. The read pairs are visualized in a special Read Pair Viewer (see Figure 44).
All possible read pair classification types available in ReadXplorer through the `ReadPairType` enumeration are listed in Table 6. An exemplary paired read mapping is shown in Table 4.

**Table 5: Read Pair Classification.** This is a list of possible classifications of all mappings of the two reads that belong to a mapping pair according to their respective mapping count on a reference sequence. The first column shows the cumulative mapping count of both reads and additional pairing properties. The second column depicts the classification of the mappings in the classes read pair or single mapping. The affiliation of each mapping to its mapping pair is always preserved to enable retrieval of all mappings of a mapping pair. This table and the description correspond to Table 1 from (Hilker *et al.*, 2014).

| Number of Mappings | Read Pair Classification |
|---|---|
| 1 | Single Mapping |
| 2 | Pair |
| >2, including at least one Perfect Pair | Perfect Pairs are stored, remaining mappings are stored as Single Mappings |
| >2, including at least one Smaller Distance Pair. May also contain perfect pairs | Perfect Pairs are stored, largest Smaller Distance Pair for each region is stored, remaining mappings are stored as Single Mappings |
| >2, including only larger distance mappings | All mappings stored as Single Mappings |

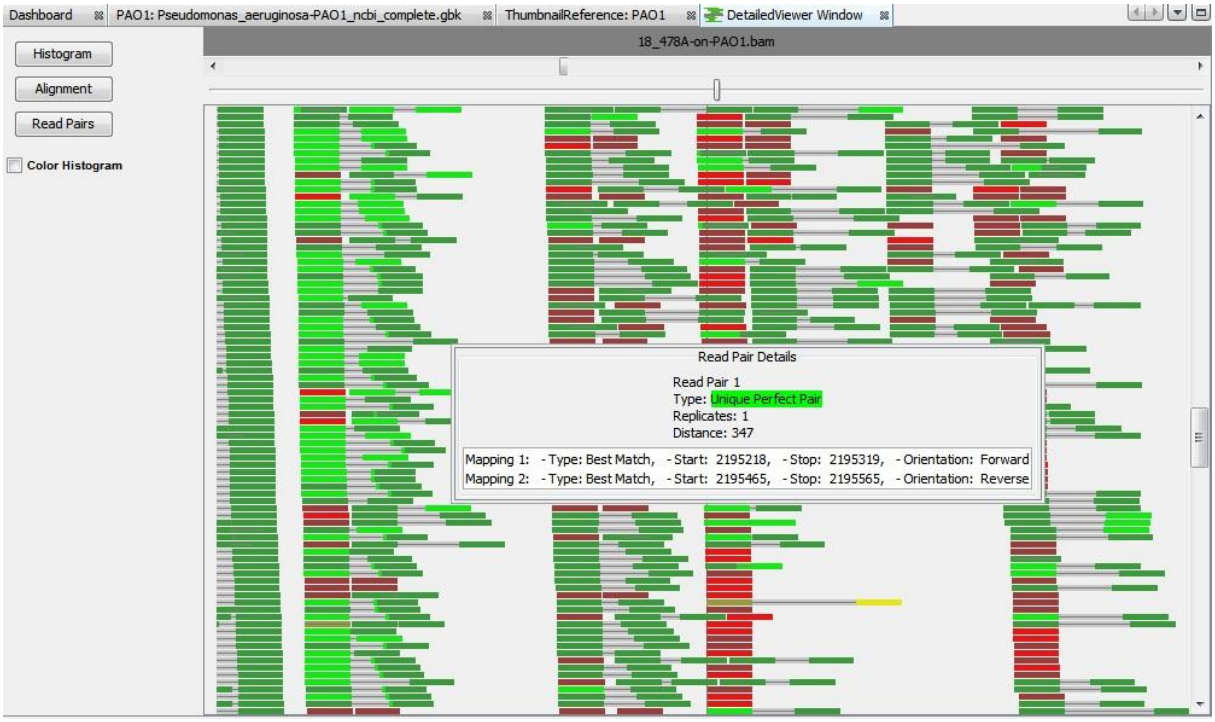



**Figure 44: Read Pair viewer with default coloration.** Perfect read pairs are colored green, Distorted read pairs are colored yellow (there is one in the lower middle part) and Single Mappings are shown in red. The color shade enables visual distinction of Perfect, Best and Common Match mappings from light to dark. By clicking on a read pair, a pop-up window with the read pair details is shown (middle) and repetitive regions can easily be scanned by selecting other mappings of the pair from the list of mappings in the pop-up window. This invokes an action that jumps to the selected mapping.

**Table 6: Possible read pair classifications in ReadXplorer.** This is a list of all 13 built-in read pair types of ReadXplorer, available through the `ReadPairType` enumeration. Using the classification enables direct access to read pairs with selected read pair properties.

| Read Pair Type | ID | Explanation |
|---|---|---|
| PERFECT_PAIR | 0 | Perfect read pair (distance and orientation correct) |
| DIST_LARGE_PAIR | 1 | Read pair with too large distance |
| DIST_SMALL_PAIR | 2 | Read pair with too small distance |
| ORIENT_WRONG_PAIR | 3 | Read pair with wrong orientation (distance is correct) |
| OR_DIST_LARGE_PAIR | 4 | Read pair with too large distance and wrong orientation |
| OR_DIST_SMALL_PAIR | 5 | Read pair with too small distance and wrong orientation |
| UNPAIRED_PAIR | 6 | Single mapping whose mate did not map on the reference or where a pair assignment fails due to too many unambiguous mappings of both reads |
| PERFECT_UNQ_PAIR | 7 | Unique perfect read pair (distance and orientation correct, both reads only mapped once) |
| DIST_LARGE_UNQ_PAIR | 8 | Uniquely mapped read pair with too large distance |
| DIST_SMALL_UNQ_PAIR | 9 | Uniquely mapped read pair with too small distance |
| ORIENT_WRONG_UNQ_PAIR | 10 | Uniquely mapped read pair with wrong orientation (distance is correct) |
| OR_DIST_LARGE_UNQ_PAIR | 11 | Uniquely mapped read pair with too large distance and wrong orientation |
| OR_DIST_SMALL_UNQ_PAIR | 12 | Uniquely mapped read pair with too small distance and wrong orientation |

## Data Export

Software providing analysis functions is in need of an export function for analysis results. For this purpose, ReadXplorer contains an exporter module (see Figure 41) that supports export of tabular data in CSV or XLS format. To make the export functionality available program-wide, the interface `ExportDataI` is provided by the module. Each data container which should have export capability has to implement this interface. Given this implementation, such objects can be passed to any `TableExporterI` implementation. To enable comprehensive storage of data relating to an analysis, this interface also includes methods to store a summary statistic and analysis parameters for a result in an extra sheet. When the results are too long for one XLS data sheet, another data sheet is added to the file, until all data is stored in it. Thus, all results obtained with ReadXplorer are treated in a unified way, reducing code duplication and sharing the same basic format. The recognition value is increased for the users by offering the same button with the same export functionality program-wide for all analysis functions.
Figures are another important type of results, e.g. for publications and documentation of the performed work. High resolution screenshots from any component or the whole program window are provided by a generic general toolbar action throughout ReadXplorer. This action opens a wizard offering the export of publication ready figures in scalable vector graphics (SVG) format.

## Viewer Logic

ReadXplorer is designed to offer different visual perspectives on the data to highlight their particular properties and facilitate quick exploration of interesting regions. This feature has been realized by implementing several data viewers. To reduce code duplication and simplify extension of ReadXplorer by specialized data viewers, an abstract base class for all viewers has been implemented: The `AbstractViewer` (see Figure 41). This class belongs to the application tier and aims at unifying all general commonalities of the viewers. The

implementation provides methods for handling and synchronization of the associated reference interval and zoom level, scaling of the data, capabilities to show the reference sequence and a legend, abilities to show viewer specific options, a unified technique to handle the visible elements within the viewer and a performance related method to assure that data is only read from files or the database when it is essentially needed.
Visualization of the reference sequence is implemented by the `SequenceBar` class and can be enabled in any `AbstractViewer`.
The viewers always refer to data associated to a certain reference interval which makes synchronization of the visible interval between all viewers belonging to the same reference an essential feature. It facilitates direct comparison and correlation of data between different data sets and viewers. The business tier classes controlling the visible reference interval and zoom have been adopted from (Doppmeier, 2009). The administration of the interval is performed by the `BoundsInfoManager` class. It implements the observer design pattern and takes the role of the observable. Its counterparts, the observers listening for updates, have to implement the `LogicalBoundsListener` interface. Listeners registered in the `BoundsInfoManager` always receive an update, if any of its properties has been changed. The `AbstractViewer` is one of the listeners implementing this interface, thus all its implementations can register to the `BoundsInfoManager`. In addition, the current interval data is further tailored for each viewer to its available pixels on the screen.
During this work, ReadXplorer received the ability to control the visible elements in a viewer via a list containing instances of implementations of the `Classification` interface. Thus, all visible data in ReadXplorer implements the `Classification` interface. Two examples are the `FeatureType` and the already mentioned `MappingClass` enumerations. The first provides all genomic feature types of the reference (e.g. gene and CDS).
Extension of ReadXplorer's visualization capabilities by additional viewers is straightforward, due to the functionality already provided by the `AbstractViewer`. Only the new data representation and potentially new data classifications have to be implemented (see `DotViewer` example in the Appendix, Section 10.2).
The mapping classification introduced above in the Read Mapping Classification paragraph is visualized by colour coding the base coverage and read alignments according to their `MappingClass` in the data viewers described below. The default colour code for the mapping classes is shown in Figure 43, j) and can be freely adapted by the user.
In the following, the data viewers natively provided by ReadXplorer are explicated: The **Reference Viewer** displays all six reading frames besides the sequence of both strands (see Figure 43, e)). The **Track Viewer** shows a coverage plot (see Figure 43, j)). The **Double Track Viewer** visualizes the coverage differences between exactly two tracks (see Figure 45) and the **Multiple Track Viewer** combines the coverage of a number of selected tracks in a single data set (see Figure 43, bottom). To enhance comparability of tracks, these track viewers are able to normalize the coverage plot separately for each included track.
In accordance with the model-view-controller pattern all viewers for the same reference are controlled by a single `ViewController` instance.
The **Histogram Viewer** supplies intuitive exploration of position-specific coverage information (see Figure 47). The interactive **Alignment Viewer** displays each computed read alignment and colors the mappings according to their mapping quality (see Figure 47). Both Histogram and Alignment Viewer simplify the visual identification of SNPs (see Section 2.7.1) or variation in the data.

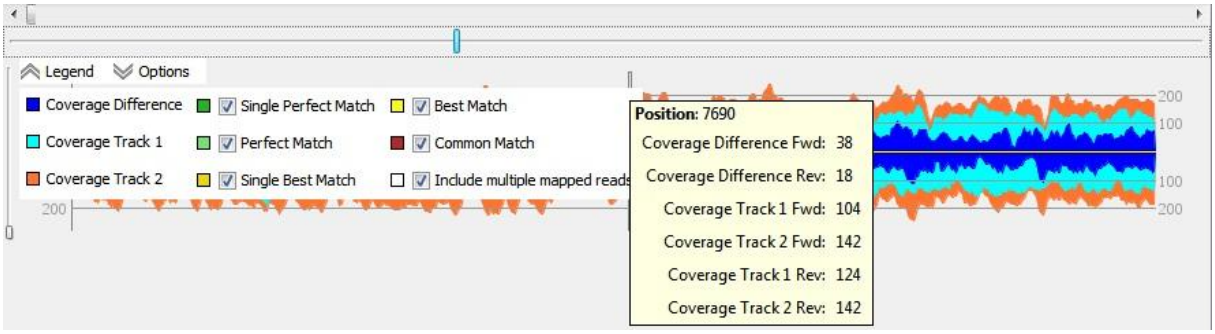



**Figure 45: ReadXplorer Double Track Viewer.** This viewer enables direct comparison of exactly two tracks by overlaying their coverage. The most important characteristic is the blue area around the center line, showing the coverage difference between both tracks. Behind, at first the coverage of track 1 is drawn in cyan and subsequently the coverage of track 2 in orange. Also this color code can be freely adapted by the user.

For paired end or mate pair data, the specialized **Read Pair Viewer** shows the pair configuration of all aligned reads (see Figure 44). Each pair is colour coded in the same manner as the mapping classes: Perfect pairs in green, Distorted pairs in yellow and Single Mappings in red. As described above in the Read Pair Classification paragraph, this viewer displays read pair configurations with the maximum level of details. Thus, it offers a specialized pair classification for all possible pairing configurations. The **Thumbnail Viewer** is available for direct comparison of the coverage of multiple genomic annotations from multiple data sets at a glance (see Figure 46). It is another special feature of ReadXplorer significantly simplifying comparison of selected genomic features of interest.


**Figure 46: Thumbnail Viewer.** The coverage of the four genes pflB, tdcE, ybiW and pflD (rows) from four RNA-seq mapping data sets (first four columns) (Srinivasan *et al.*, 2013) involved in the propanoate metabolism in *Escherichia coli* is compared using the Thumbnail Viewer of ReadXplorer. Their coverage can be compared at a glance, especially when overlaying two tracks using a thumbnail version of the Double Track Viewer (rightmost column). All four genes catalyze the same reaction (EC:2.3.1.54), but apparently only two of them are highly expressed (pflB and tdcE). It also directly catches the eye that tdcE (second row) has a much lower expression level in both wild type (WT, left two columns) replicates in comparison to both H-NS repressor deletion mutant replicates (hns, third and fourth column). Thus, this viewer is optimally suitable for multi-gene comparisons and figure creation, e.g. as shown here for genes from the same metabolic pathway.

# Analysis Framework

During the development of ReadXplorer, the implementation of automated analysis functions was focused enabling users to perform laborious tasks for a selected list of tracks in virtually no time. The implementation of all analyses supported by ReadXplorer is described in the following Section 5.3. It is noteworthy, that the analyses for transcription start sites and novel transcripts, read counts and normalization calculations and operons have been combined in one wizard. Thereby, read mapping data only has to be read once from the file and all analyses can be run in parallel. This design drastically reduces the computation time for these analyses, when they are run in parallel.
All integrated analysis methods, in particular, rely on the mapping quality classification of the data. All parameters are user-adjustable and configured via user-friendly wizards which share common components and methods. Each analysis wizard starts with a panel to select the tracks to include in the analysis (see Figure 66 in Appendix). ReadXplorer supports simultaneous analysis of multiple data sets by selecting all tracks to include in this panel. This panel also offers the option to combine all selected data sets in the subsequent analysis. In that case, all their mapping data is combined and they are treated as a single data set. The next common wizard panel is a panel to select the included mapping classes and set a mapping quality threshold for the analysis. For all analyses running on genomic features it also shows analysis strand selection options (see Figure 67 in Appendix). They allow combining the mappings of both strands for unstranded sequencing libraries or switch the strand whose mappings are included in the analysis. All analyses relying on the presence of genomic features also contain a panel to configure the genomic feature types to include in the analysis (see Figure 68 in Appendix). Hence, it determines the output in the result table, e.g. a prokaryotic SNP detection is oftentimes run only on CDS features.
The whole reference genome is analyzed for references with multiple chromosomes, contigs or other sequences. The general `AnalysisI` interface is provided by the analysis framework for all actual implementations of analysis algorithms (see Figure 41).
Since all results are based on analyses of tracks, they all share common functionality and it is convenient to be able to treat them consistently. Therefore, the abstract class `ResultTrackAnalysis` has been introduced and is intended to be implemented by all track analysis results (see Figure 41). Since analysis results shall be ready for export, the `ResultTrackAnalysis` class also implements the `ExportDataI` interface, mentioned in the previous Data Export paragraph. Thereby, it is guaranteed, that each analysis result can be exported effortlessly. Common functionality for visualizing analysis results has been bundled in the abstract `ResultTablePanel` class (see Figure 41). Another important benefit of the `AnalysisI` interface and the two abstract result classes is that they additionally enable exchangeability of the algorithms, their results and visualizations.
In general, results are displayed to the user in the form of tables which can be sorted by each column, filtered by column values and directly exported into XLS or CSV files (see previous Data Export paragraph). ReadXplorer does not limit the user to run analyses on single data sets. Instead, it allows running analyses for multiple tracks to either combine or compare their data. When multiple tracks are analyzed, an additional filter is provided. It enables viewing all results which have been detected either in at least or at most a given number of tracks. This feature enables quick identification of results present in multiple tracks. For an effortless visual assessment, results are additionally highlighted in their corresponding data viewers. Furthermore, the reference position of the currently selected result is centred in each corresponding data viewer. Below the result table the used parameters of the current analysis are shown and some statistics can be viewed by clicking on the "Show Statistics" button (see Figure 47).

The implementation of ReadXplorer's analysis framework has been designed to simplify addition of new analyses by extracting common tasks. The class serving this purpose is the `AnalysisHandler` (see Figure 41). This handler coordinates all data requests and the transmission of readout data to the respective analysis via the observer design pattern. This behaviour clearly separates data queries from analysis functions.
The `AnalysisHandler` is capable of handling all three basic data types available within ReadXplorer: Coverage, read starts, mapping and read pair data (see Algorithm 1). Afterwards, the particular analysis function is responsible for appropriate treatment of the data. To reduce the memory footprint, the reference genome is split into intervals of 200kb length. A separate data request is sent to the corresponding data query thread for each interval. Thus, the software does not stall during operation of analyses.

**Algorithm 1: Data extraction for analyses**

Note: Only retrieval of coverage and mappings is shown here, as read starts and read pairs are retrieved in the same manner.

```
1:  nd ← the type of required data (coverage | mappings)
2:  mg ← the flag for calculation of mismatches and gaps (true | false)
3:  mc ← the list of included read mapping classifications
4:  bamFileReader ← a reader for BAM files
5:  iLength ← 200000
6:  for each chromosome c do
7:      for each iLength bp interval i of c do
8:          cl ← map of each mapping class to a coverage array with capacity iLength
9:          m ← empty list for mappings
10:         bamRecordIterator ← query bamFileReader for i from c in BAM file
11:         while bamRecordIterator has next do
12:             r ← next mapping record from bamRecordIterator
13:             rmc ← mapping class of r
14:             if r is mapped and mc contains rmc do
15:                 if nd = coverage then
16:                     (* increase coverage in corresponding array *)
17:                     arrayPos ← start of r - start of i
18:                     for arrayPos, ..., arrayPos + length of r do
19:                         cl(rmc)[arrayPos]++
20:                     end for
21:                 else if nd = mappings then
22:                     add r to m
23:                 end if
24:                 if mg = true do
25:                     scan cigar of r for mismatches and gaps and store them
26:                 end if
27:             end if
28:         end while
29:         return cl or m to data requesting object
30:     end for
31: end for
```

# 5.3. Automatic Analysis Functions

In this Section, the implementation of all analyses supported by ReadXplorer is described. The general analysis framework on which all analyses are based is described in the previous Section.
ReadXplorer offers single nucleotide and insertion-deletion polymorphism (SNP and DIP) detection (see Section 5.3.1), reference feature and coverage analyses (see Section 5.3.8), genome rearrangement detection (see Section 5.3.2) and RNA secondary structure prediction (see Section 5.3.7). Especially for RNA-Seq experiments, ReadXplorer offers differential gene expression analysis (see Section 5.3.5), transcription start site (TSS) (see Section 5.3.4), novel transcript (see Section 5.3.4), and operon detection (see Section 5.3.6) as well as read count and normalization calculations for each reference feature (see Section 5.3.3).

## 5.3.1. Single Nucleotide & Deletion-Insertion Polymorphisms

SNP and DIP detection is one of the key analyses when sequenced reads are mapped on a closely related reference sequence (see Section 2.7.1).
The presented SNP and DIP detection algorithm directly reads the data from indexed BAM files (see Section 5.2.2) within a few minutes (see Table 10) and is based upon the following list of parameters:

- A user-definable **minimum percentage** of variation has to be set.
- A **minimum count of mismatching bases** in the mappings at the examined position has to be set.
- A check box for distinguished treatment of organisms with different ploidy is offered. This box complements the minimum count of mismatching bases in the mappings at the examined position. It switches the minimum mismatching bases count between the **single most frequent base** at the current position for haploid organisms and the **sum of all mismatching bases** for di- and polyploid organisms.
- A **minimum base quality** filter is offered which requires the PHRED scaled quality value (see Section 2.1 and (Ewing and Green, 1998)) at the current position to exceed the given minimum value to increase the mismatch coverage. It is only applied if the data set contains base qualities.
- A **minimum average PHRED base quality** parameter takes care that only positions are considered where the average base quality of all mappings exceeds the given base quality value if the data set contains base qualities.
- The **read mapping classification** filter (see Section 5.2.2, Read Mapping Classification) incorporated in all analysis functions is offered.

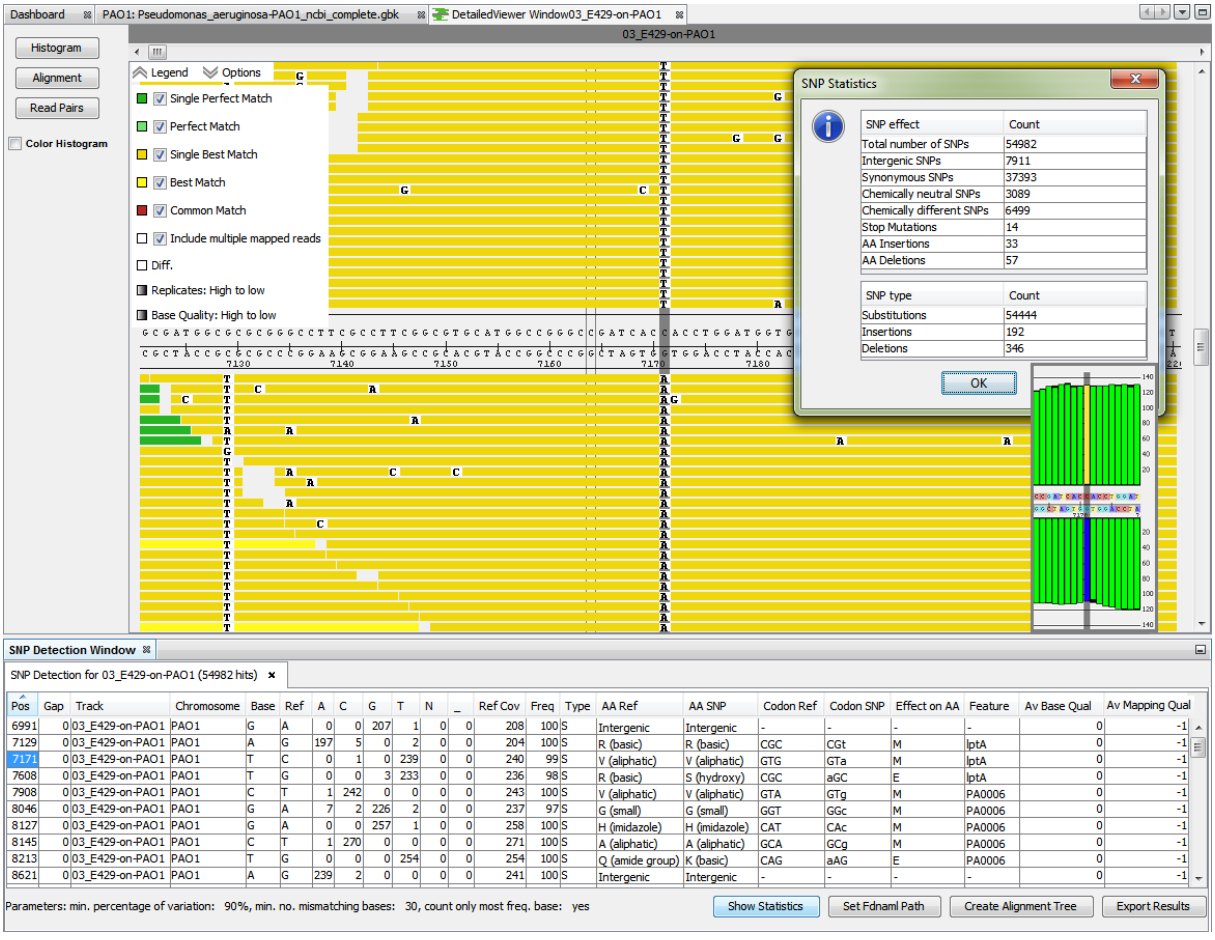
The detection algorithm first queries the coverage including mismatch and gap counts via Algorithm 1. Afterwards, the computation proceeds as described in Algorithm 2 in linear time.

**Algorithm 2: SNP and DIP detection algorithm**

```
m ← list of mismatches for current chromosome interval obtained by Algorithm 1
g ← list of gaps for current chromosome interval obtained by Algorithm 1
coverage ← list of coverage arrays for each included mapping class for current chromosome interval obtained by Algorithm 1
baseArray[][][] ← array of genome position on first level, a base index (0-5 for A,C,G,T,N,gap) on the second level and the base coverage count, average base and average mapping quality (0-2) on the third level
gapCounts[][][][] ← first, third and fourth dimension of the array correspond to the three dimensions of the baseArray. The second dimension represents the gap index of successive gaps in the same mapping

for each gap from g do
    bq ← base quality of gap
    mq← mapping quality of gap
    if bq > min. base quality or bq = -1 do
        gapCounts[gap position][gap index][gap base][0]++
        if bq ≠ -1 do
            gapCounts[gap position][gap index][gap base][1] + bq
        else do
            hasBaseQualities = false
        end if
        if mq > 0 do
            gapCounts[gap position][gap index][gap base][2] + mq
        else do
            hasMappingQualities = false
        end if
    end if
end for

(* Increase mismatch counts obtained from m in baseArray in the same manner than for gaps excluding the second array dimension *)

(* After gathering data for all intervals of all chromosomes *)
for each gapCount from gapCounts do
    for each gapIndex from gapCount do
        largestCount ← largest base count among all gapIndex[gap base][0]
        accumulativeCount ← sum of gap counts among all gapIndex[gap base][0]
        if (use sum of all mismatching bases = true and largestCount > min. # mismatching bases) or (use sum of all mismatching bases = false and accumulativeCount > min. # mismatching bases) do
            averagebq ← Calculate average base quality from all gapIndex[gap base][1]
            averagemq ← Calculate av. mapping quality from all gapIndex[gap base][2]
            if (hasBaseQualities = false or averagebq > min. average base quality) and (hasMappingQualities = false or averagemq > min. avg. mapping quality) do
                cov ← get total coverage for current gap position from coverage
                percentage ← accumulativeCount * 100 / cov
                if percentage > min. percentage do
```

```
                                add DIP to result list
                            end if
                        end if
                    end if
                end for
            end for

            (* Calculate SNPs from baseArray in the same manner than DIPs by leaving out the
            gapIndex dimension *)
```

The results can be examined in detail not only in the alignment or histogram viewer (see Figure 47), but also in the track viewer (see Figure 43). The result table not only reveals small-scale evolutionary differences, but also allows analysis of the resulting functional differences emerging from the polymorphisms. The detected SNPs and DIPs are classified according to their type (substitution, insertion or deletion), their location (intra- or intergenic), and their effect on coding sequences (match, frame shift +1 (insertion), frame shift -1 (deletion), chemically neutral and chemically different substitution). Therefore, associated codons and amino acids are shown for intragenic SNPs (see Figure 47).




| Pos | Gap | Track | Chromosome | Base | Ref | A | C | G | T | N | _ | Ref Cov | Freq | Type | AA Ref | AA SNP | Codon Ref | Codon SNP | Effect on AA | Feature | Av Base Qual | Av Mapping Qual |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6991 | 0 | 03_E429-on-PAO1 | PAO1 | G | A | 0 | 0 | 207 | 1 | 0 | 0 | 208 | 100 | S | Intergenic | Intergenic | - | - | - | - | 0 | -1 |
| 7129 | 0 | 03_E429-on-PAO1 | PAO1 | A | G | 197 | 5 | 0 | 2 | 0 | 0 | 204 | 100 | S | R (basic) | R (basic) | CGC | CGt | M | lptA | 0 | -1 |
| 7171 | 0 | 03_E429-on-PAO1 | PAO1 | T | C | 0 | 1 | 0 | 239 | 0 | 0 | 240 | 99 | S | V (aliphatic) | V (aliphatic) | GTG | GTa | M | lptA | 0 | -1 |
| 7608 | 0 | 03_E429-on-PAO1 | PAO1 | T | G | 0 | 0 | 3 | 233 | 0 | 0 | 236 | 98 | S | R (basic) | S (hydroxy) | CGC | aGC | E | lptA | 0 | -1 |
| 7908 | 0 | 03_E429-on-PAO1 | PAO1 | C | T | 1 | 242 | 0 | 0 | 0 | 0 | 243 | 100 | S | V (aliphatic) | V (aliphatic) | GTA | GTg | M | PA0006 | 0 | -1 |
| 8046 | 0 | 03_E429-on-PAO1 | PAO1 | G | A | 7 | 2 | 226 | 2 | 0 | 0 | 237 | 97 | S | G (small) | G (small) | GGT | GGc | M | PA0006 | 0 | -1 |
| 8127 | 0 | 03_E429-on-PAO1 | PAO1 | G | A | 0 | 0 | 257 | 1 | 0 | 0 | 258 | 100 | S | H (imidazole) | H (imidazole) | CAT | CAc | M | PA0006 | 0 | -1 |
| 8145 | 0 | 03_E429-on-PAO1 | PAO1 | C | T | 1 | 270 | 0 | 0 | 0 | 0 | 271 | 100 | S | A (aliphatic) | A (aliphatic) | GCA | GCg | M | PA0006 | 0 | -1 |
| 8213 | 0 | 03_E429-on-PAO1 | PAO1 | T | G | 0 | 0 | 0 | 254 | 0 | 0 | 254 | 100 | S | Q (amide group) | K (basic) | CAG | aAG | E | PA0006 | 0 | -1 |
| 8621 | 0 | 03_E429-on-PAO1 | PAO1 | A | G | 239 | 2 | 0 | 0 | 0 | 0 | 241 | 100 | S | Intergenic | Intergenic | - | - | - | - | 0 | -1 |

**Figure 47: SNP and DIP detection result with alignment and histogram viewers.** The result and used parameters of an automatic SNP and DIP detection for *P. aeruginosa* strain E429 are shown in the table at the bottom. The SNP selected in the table is automatically centered in the data viewers. The alignment viewer shows almost only Single Best Match mappings in the visible region and all of them contain the predicted SNP in their alignment. The inlet shows a section of the histogram viewer for the same position. This viewer reveals the coverage deviating from the reference by base specific coloring (green = match coverage). Both viewers are well-suited for a quick visual inspection of observed alignment events. The SNP detection result table contains comprehensive information including the precise coverage of each nucleotide and several other details as mentioned in the text. The average base and mapping quality in the last two columns are not available in this mapping data set from SARUMAN, thus they are set to 0 and -1.

The chemical classification of amino acids has been chosen according to Knippers (2006) and Barnes and Gray (2003) and is shown in Table 7. A classification of "chemically neutral" neither means that the protein will definitely be functional, nor does a classification of "chemically different" imply that the protein will definitely not be functional. The amino acid classification is rather meant as guidance to assist quick perception of positions with possibly devastating mutations in contrast to harmless neutral mutations. The true effect of such mutations still has to be reviewed in detail and examined by experiments in the laboratory.

**Table 7: Proteinogenic amino acid groups.** The table lists the chemical groups used in ReadXplorer to further classify the 22 proteinogenic amino acids during SNP and DIP detection.

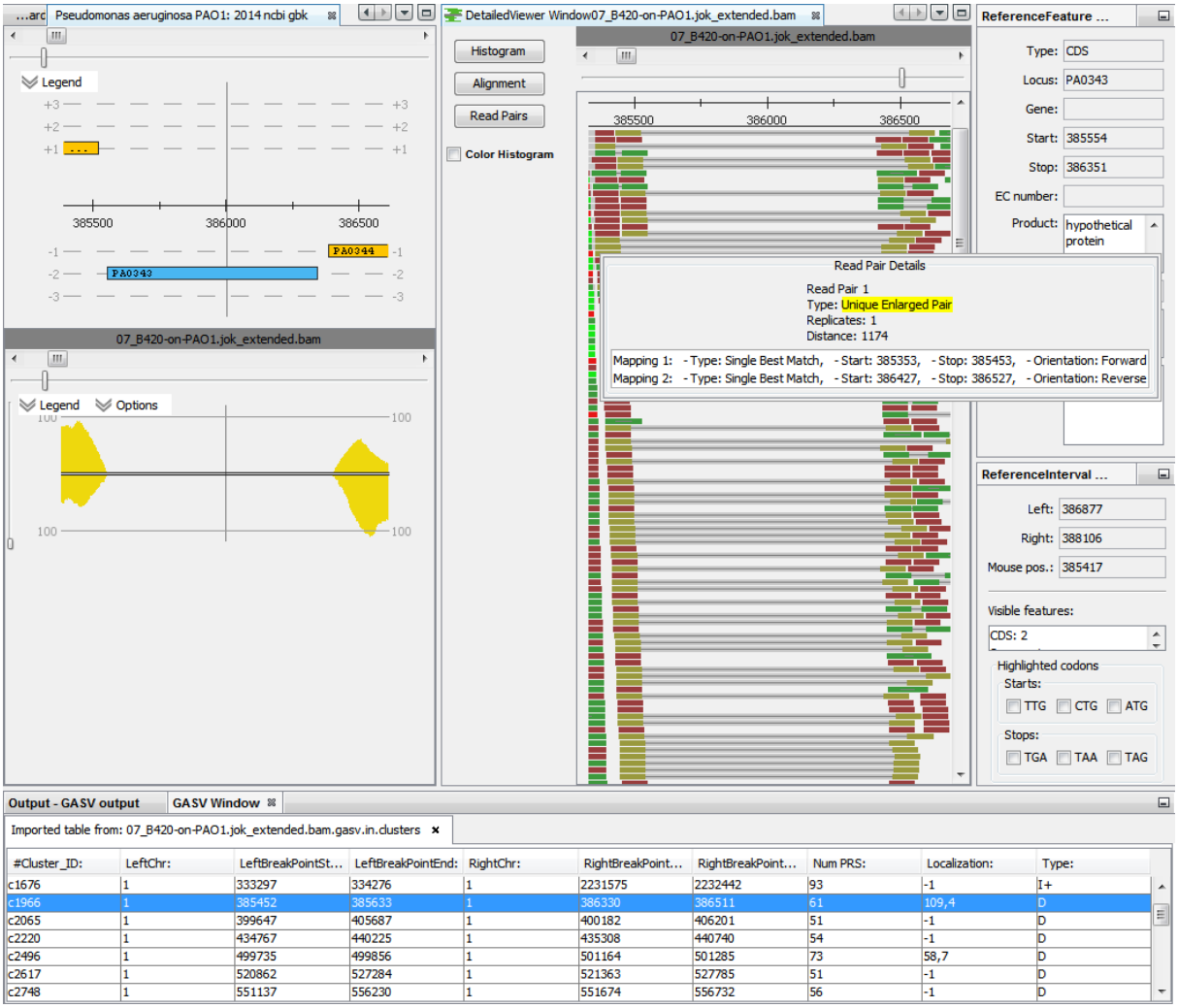
| Amino Acid Group | Amino Acids |
|---|---|
| Acidic | Aspartic acid, glutamic acid |
| Aliphatic | Alanine, valine, leucine, isoleucine, methionine |
| Amide group | Asparagine, glutamine |
| Aromatic | Phenylalanine, tyrosine, tryptophan |
| Basic | Arginine, lysine |
| Imidazole | Histidine |
| Imino group | Proline |
| Hydroxy | Serine, threonine |
| Pyrro group | Pyrrolysine |
| Small | Glycine |
| Seleno group | Selenocysteine |
| Thiol group | Cysteine |

Translation of codons to amino acids is defined by the genetic code selected by the user. It can be freely chosen from the list of NCBI genetic codes or an own user-defined genetic code can be added and selected within ReadXplorer (see Figure 69 in Appendix).
Besides the built-in SNP detection, ReadXplorer can also be used to visualize results of other statistical SNP detection tools (see Section 2.7.1) which use VCF output (Danecek *et al.*, 2011). ReadXplorer can import VCF files and display them in a table offering the same selection and filtering possibilities than for other tables. Direct comparison of SNP detection results from ReadXplorer and other tools is simplified by this feature.

### 5.3.2. Genome Rearrangements

SVs described in Section 2.7.2 are associated to many diseases including cancer (Iafrate *et al.*, 2004). As cancer research is a highly important research field in human health care, the first tools for SV detection based on read pair data were designed for human cancer research (Campbell *et al.*, 2008). Nonetheless, genome rearrangement detection is useful for the analysis of other eukaryotic and prokaryotic organisms as well.
In general, the tools for SV detection are command-line based. Integrating an established SV detection tool into ReadXplorer makes genome rearrangement detection accessible for researchers unfamiliar with command-line tools and enables immediate visual inspection of read pair configurations in genomic regions with predicted SVs (see Figure 48).
The tool chosen for integration into ReadXplorer is GASV (Sindi *et al.*, 2009) (see Section 2.7.2). Its development focused on the human genome, but this is also true for the other tools mentioned in Section 2.7.2. However, GASV has also been tested successfully on other genomes like yeast (Zeitouni *et al.*, 2010). The choice of GASV is advantageous, because it supports a broad range of SVs: insertions, deletions, inversion, translocations (see Figure 22) and can handle more divergent rearrangement events i.e. resulting from multiple

rearrangements at the same locus. Such cases hinder the reconstruction of the correct sequence of rearrangement operations for that locus.
A usability advantage of GASV is its implementation in Java. Thus, it can directly be incorporated in ReadXplorer without giving rise to additional installation requirements.
An analysis is started by configuring the GASV parameters in a wizard containing two pages: One for each GASV analysis step (`BamToGASV` and `GASVMain`). To guarantee flexibility for the user, all available options have been made adjustable through the wizard. Afterwards, GASV is started using an instance of the newly implemented `GASVCaller` class gathering all required data. GASV natively assumes that chromosome names are numbers. This is not the case for prokaryotes, though. To be able to use arbitrary chromosome names, ReadXplorer always utilizes the GASV option to set a chromosome naming file.
Two small changes of the GASV source code were necessary: Firstly, the status messages of GASV have been redirected to the console within ReadXplorer to show the progress of the analysis to the user. Secondly, a method assuring that only Single Perfect Match and Single Best Match mappings are allowed for the GASV analysis has been implemented. Testing different mapping classifications with different example data sets showed that it is inevitable to use only Single Perfect Match and Single Best Match mappings to receive reasonable results. Otherwise, repeat regions lead to many falsely predicted rearrangements obscuring correct predictions.




| #Cluster_ID: | LeftChr: | LeftBreakPointSt... | LeftBreakPointEnd: | RightChr: | RightBreakPoint... | RightBreakPoint... | Num PRS: | Localization: | Type: |
|---|---|---|---|---|---|---|---|---|---|
| c1676 | 1 | 333297 | 334276 | 1 | 2231575 | 2232442 | 93 | -1 | I+ |
| c1966 | 1 | 385452 | 385633 | 1 | 386330 | 386511 | 61 | 109,4 | D |
| c2065 | 1 | 399647 | 405687 | 1 | 400182 | 406201 | 51 | -1 | D |
| c2220 | 1 | 434767 | 440225 | 1 | 435308 | 440740 | 54 | -1 | D |
| c2496 | 1 | 499735 | 499856 | 1 | 501164 | 501285 | 73 | 58,7 | D |
| c2617 | 1 | 520862 | 527284 | 1 | 521363 | 527785 | 51 | -1 | D |
| c2748 | 1 | 551137 | 556230 | 1 | 551674 | 556732 | 56 | -1 | D |

**Figure 48: Visualization of genome rearrangement events using GASV.** ReadXplorer offers an effortless exploration of the data underlying the genome rearrangements detected by GASV (table at the bottom). The region of the deletion selected in the table is centered simultaneously in the reference, track and read pair viewer. The hypothetical protein PA0343 is deleted in *P. aeruginosa* strain B420. No reads map to the corresponding region of *P. aeruginosa* PAO1 and many read pairs are observed in B420 with an enlarged distance of about 1200bp instead of the expected 300bp.

GASV writes its results into a tab separated file in CSV format on the hard disk. In order to immediately visualize the results after finishing an analysis (see Figure 48), this file is autonomously parsed by ReadXplorer in the same manner as other CSV tables (see Section 5.1, Data Import). To increase usability and readability of the results, the left and right breakpoint borders have been split from a single column in the file to two columns. One lists the start and the other the end position of all breakpoints (see Figure 48).

### 5.3.3. Read Count & Normalization Calculations

This analysis offers an overview on the raw read counts and normalized read count values for each genomic feature. The normalization methods applied here are the TPM (B. Li *et al.*, 2010) and RPKM (Mortazavi *et al.*, 2008) methods. Details of the methods are clarified in Section 2.8.1.

Before normalizing read counts, the transcript boundaries and the inclusion model for the reads have to be determined. Thus, a decision has to be made if the exact boundaries given by the annotation are used or if an offset at the start, stop or both annotation ends is added. Introducing an offset can be crucial to downstream analyses, because ordinary gene annotations do not take 5'-UTRs and 3'-UTRs into account. Additionally, automatically annotated gene start and stop positions might be incorrect, leading to data loss. Therefore, offsets at the start and end of a reference feature can be defined by the user during configuration of the analysis in the corresponding wizard.

The implemented read assignment model for read mappings overlapping multiple genomic features (e.g. genes or CDS), is similar to the *union* model of HTSeq-count (Anders *et al.*, 2014). The difference is that instead of discarding read mappings marked as ambiguous in Figure 23, in the first case the reads are associated to gene_A and in the second case their proportional fraction is added to the read counts of each of the overlapped features.

Besides counting and normalizing the reads per reference feature, the analysis serves as filter: A minimum and maximum raw read count value can be set. Only reference features with read count values in the given range are returned in the result (see Figure 49). The TPM and RPKM values are well-suited for RNA-Seq data to identify genes with a certain expression level, whereas the read count column is applicable for both RNA-Seq and resequencing data to explore the read counts of reference features.

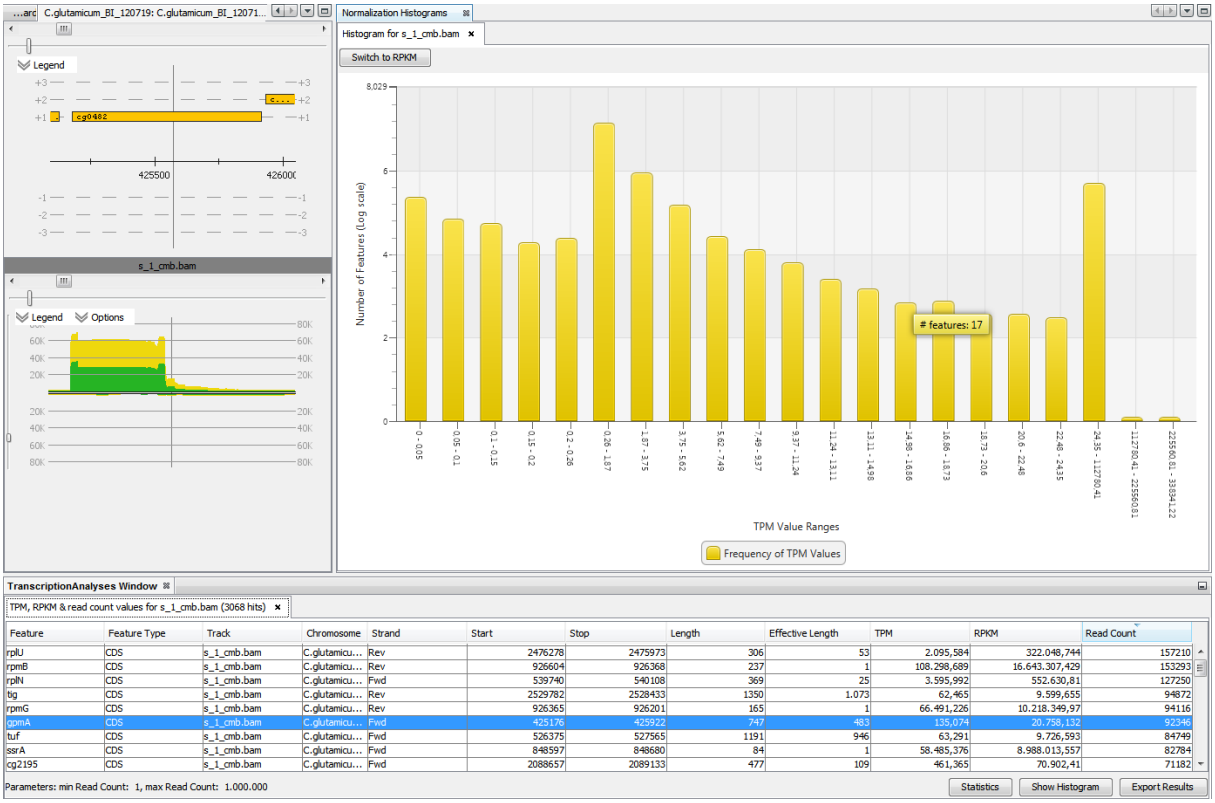



**Figure 49: TPM, RPKM and read count calculation result.** The result table (bottom) shows all important information regarding the genomic features and their read count (last column), TPM (third last column) and RPKM value (second last column). The genomic position and coverage are shown for the feature selected in the table in the reference and track viewer (left). The histogram comfortably visualizes the frequency of the TPM values observed within the data set at log scale. The histogram can be switched to show RPKM values using the dynamic "Switch to *" button at the top, where * is either RPKM or TPM. A figure of this plot can also be exported via the screenshot wizard (see Section 5.2.2, Data Export). The chosen example data set is an in-house RNA-seq data set from *Corynebacterium glutamicum*, kindly provided by Jörn Kalinowski.

In addition to the general export functionality, the TPM or RPKM-value distribution of the analyzed data set can be viewed in a log-scaled histogram (see Figure 49). This histogram is implemented based on the JavaFX library contained in the standard Java Development Kit since version 8 (JDK 8). It calculates and displays 21 bins of TPM or RPKM values. The first 5 bins are reserved for distinguishing values within the 0.2 quantile of the analyzed data. The boundaries of the next 13 and remaining 3 bins are reserved for distinguishing the 0.9 quantile and the largest TPM and RPKM values, respectively. All bin borders are dynamically calculated based on the mentioned quantiles observed in the result. Introduction of these three groupings was inspired by the thought to create a histogram which enables distinction of smaller values as well. A histogram scale geared to the largest normalization value would contain almost all values in the smallest bin. This becomes obvious when considering the largest RPKM value (16,643,308.429) from the example (see Figure 49). Such outliers are commonly observed in RNA-seq data.

The algorithm for the assignment of reads to genomic features and the TPM and RPKM normalization is implemented as follows: For each analyzed reference interval, the algorithm gathers the read counts for all genomic features in range. After all reads are counted, the normalization sum for TPM is calculated followed by the final calculation of TPM and RPKM values. Details are shown in Algorithm 3.

**Algorithm 3: TPM and RPKM calculation**

```
1:  features ← list of all genomic gene, CDS, mRNA, rRNA, tRNA and exon features
2:  m ← list of mappings from a single chromosome interval obtained by Algorithm 1
3:  li ← 0 (* Index of last assigned mapping *)
4:  for each feature f from features do
5:      if chromosome id of f = chromosome id of m do
6:          isFstFittingMapping ← true
7:          currentCount ← 0; readLengthSum ← 0
8:          for i ← li, ..., size of m do
9:              Calculate and use feature start and stop offset below according to parameters
10:             if stop of m(i) >= start of f and start of m(i) <= stop of f
11:                 if isFstFittingMapping = true do
12:                     li ← i, isFstFittingMapping ← false
13:                 end if
14:                 if m(i) is on the correct strand do
15:                     countMapping ← check according to union fraction model (see
16:                     above) if m(i) is allowed to be counted for f
17:                     if countMapping = true do
18:                         check if read count of preceding features also containing m(i)
19:                         has to be decreased and decrease if necessary
20:                         currentCount++; readLengthSum + length of m(i)
21:                     end if
22:                 end if
23:             else if start of m(i) > stop of f do
24:                 if isFstFittingMapping = true do
25:                     li ← i,break for
26:             end if
27:         end for
28:         increase read count of f by currentCount and store it
29:         increase read length sum of f by readLengthSum and store it
30:     end if
31: end for
32: for i = 0, ..., size of m do
33:     check if the count of m(i) has to be fractionated for its associated features according to
34:     the union fraction model
35: end for
36:
37: (* After gathering data for all intervals of all chromosomes *)
38: calculate normalization sum for TPM
39: for each feature f from features do
40:     sum read counts of nested features (e.g. for multiple exons of a gene) of f
41:     calculate effective length, TPM and RPKM value for stored read count of f
42: end for
```

### 5.3.4. Transcription Start Site Detection

The detection of novel transcription start sites (TSS) as well as the verification and correction of already annotated TSS are two of the key analysis features for RNA-Seq experiments. When using a suitable RNA-Seq protocol such as the selective analysis of primary transcripts via differential RNA-seq (Borries *et al.*, 2012) or a 5' prime RNA adapter ligation (Pfeifer-Sancar *et al.*, 2013), the mapped reads can be analyzed for TSSs.
One goal of this thesis was to develop and implement an automated method which detects TSSs within an appropriate RNA-seq data set. When the design of this analysis was started in 2011, no other computational methods were available to detect TSSs within prokaryotic RNA-seq data. Therefore, a novel approach has been developed and is presented here. This approach is based on analyzing the coverage properties of each neighbouring genomic position pair for TSSs. The analysis mainly relies on two parameters balancing each other. A TSS is only detected, if enough reads start at the second position of the pair and the increase of coverage in percentage from the first to the second position of the pair is high enough (see Figure 50).
Both parameters are user adjustable, but can also be automatically estimated for each track. In addition, a specialized treatment for low coverage regions has been implemented: A maximum coverage threshold for the low coverage regions can be set and the minimum number of read starts parameter can be lowered for these regions.
The automated parameter estimation only emits statistically significant positions as TSS, that satisfy both of the following two empirically chosen criteria:

1. The absolute number of read starts is in the upper 0.0025 quantile of all other read start counts for all positions in the track.
2. The percentage of coverage increase is in the upper 0.0025 quantile of all other coverage increase percentages in the track.

By combining both parameters, the analysis is capable of rejecting positions in areas of already high coverage, where the total number of read starts exceeds the threshold, but the coverage increase only accounts for a low increase in percentage. Additionally, positions in low coverage regions with a high coverage increase in percentage are rejected if not enough reads start at the examined position to exceed the minimum read start parameter.
This parameter choice is also suitable for 5' enriched RNA-Seq data sets, because the underlying distribution takes into account each $n$-1 pairs of neighbouring positions in a genome of size $n$.
Prokaryotes are known to harbor approximately one gene per kb of genome size (Rogozin *et al.*, 2002). With the explicated parameter choice, 2.5 genes per kb of genome size are allowed for each of the two parameters. Thus, there is enough room for both parameters to balance each other and in practice they have shown to be quite stringent (see evaluation in Table 8).
In some cases the coverage increases in steps at a TSS. To consider this case, an option is available to associate all predicted TSS within a small user-defined bp window to the statistically most significant TSS. Hereby, neither several predicted TSS appear for one gene, nor are they lost the analyst.
The bp window to associate TSSs with the next genomic feature can also be adjusted by the user. To identify at a glance if evidence for alternative TSSs exists e.g. for a gene, all identified TSS within this maximum feature distance are classified. The statistically most significant position is designated as "primary" TSS, while all other TSSs in this window on the same strand are designated as "secondary".

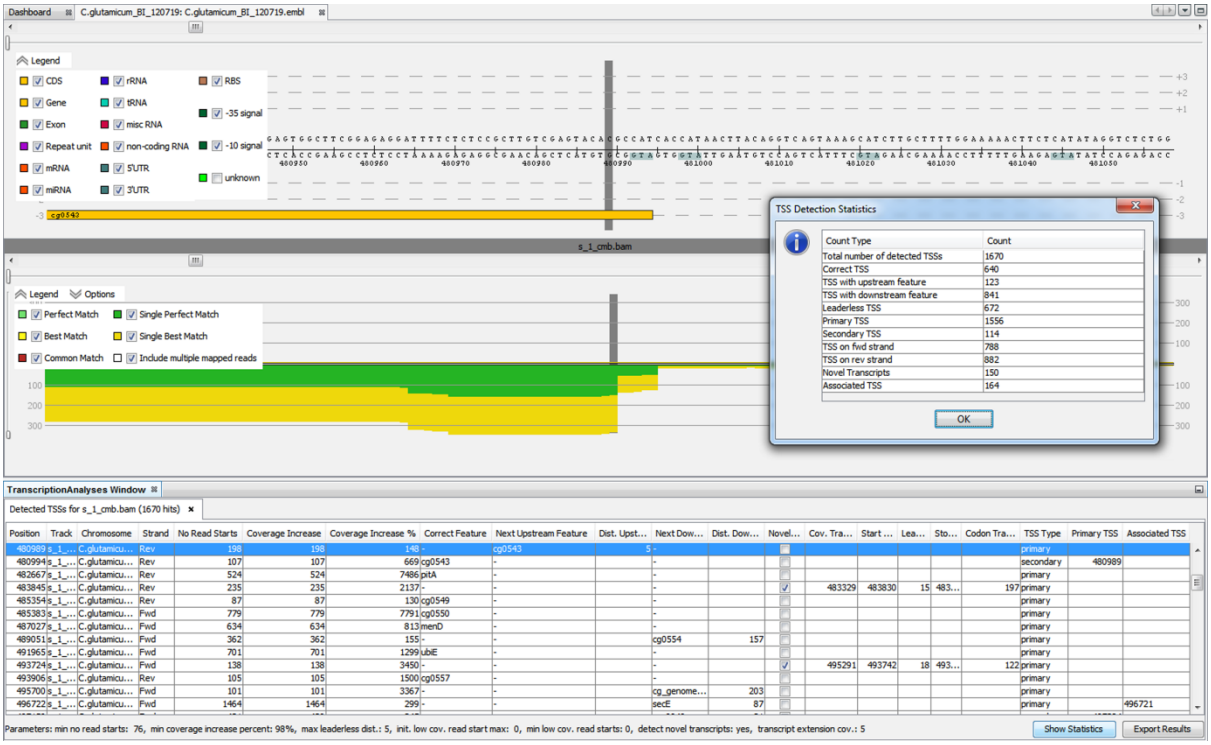



**Figure 50: TSS and novel transcript detection.** The highlighted TSS position in the table is centred in the Reference and Track viewers. The alternative secondary TSS (480,994) at the start of the gene has reads (transcripts) starting at exactly that position testifying for a correctly annotated gene having a leaderless transcript at the inspected position. The "Leaderless" column only appears in exported tables. Within ReadXplorer it can easily be deduced from the sortable "Correct Feature" and "Dist. Upstream Feature" columns. The selected primary TSS (480,989) harbours many more read starts. It reveals an interesting event suggesting an alternative transcript or a very long leader for the next gene. The chosen example data set is an in-house RNA-seq data set from *Corynebacterium glutamicum*, kindly provided by Jörn Kalinowski. The columns whose headers are not shown completely due to the extent of the table are (from left to right): Dist. Upstream Feature, Next Downstream Feature, Dist. Downstream Feature, Novel Transcript, Cov. Transcript Stop, Start Codon Pos, Leader Length, Stop Codon Pos and Codon CDS Length.

The majority of transcripts contains a leader, but transcripts can also be leaderless (Pfeifer-Sancar *et al.*, 2013). To distinguish both cases easily, leaderless transcripts are flagged in the result and can thus be retrieved instantly. The maximum distance of a TSS to the actual translation start site can be adjusted in the wizard by the user.

The TSS analysis in ReadXplorer also supports the detection of novel transcripts. Therefore, the nearest genomic feature starting in the right direction in a user-definable bp window around a TSS is listed. Internal TSSs are identified in the result by inspecting entries in the "Next Upstream Feature" column of the result table. In case a TSS is detected without a neighbouring genomic feature - an orphan TSS - it is marked and a novel transcript is suggested, starting at the TSS and ranging up to the position at which the coverage drops below a user-defined threshold (see Figure 50). To offer a deeper insight into the novel transcripts, the analysis further identifies and outputs the next start codon on the respective TSS strand, calculates the leader length, the next in-frame stop codon and the novel transcript length deduced from the assigned start and stop codons.

Furthermore, the method can detect *trans*-encoded as well as *cis*-antisense-transcripts and miRNAs. Additionally, highly conserved miRNA target sites can be detected by exploiting the fact that reads match to their origin as well as to their target site.

Because the detection algorithm does not need any annotations, the described method is well-suited as a starting point to identify novel transcripts in references without any genomic annotations. Afterwards they can be filtered manually and verified in the laboratory.

Additionally, ReadXplorer is capable of highlighting start and stop codons and their open reading frames in the reference (see Figure 43, o)). This facilitates instant comparison of RNA-Seq coverage with potential transcripts.

When the automatic parameter estimation is chosen, the read start and coverage distributions for each analyzed data set are retrieved from the database. During the first run of an analysis for a data set, the two distributions are initialized and calculated from the incoming data in parallel to the analysis. In this case, the initial parameter choice is very lenient. Thus, after analyzing the whole data sets, the read start and coverage distributions have also been computed. The resulting list of TSS is then filtered according to the strict parameter choice, obtained from the two distributions.

**Algorithm 4: TSS detection algorithm with automatic parameter estimation**

```
1:  minNoReadStarts← 0.0025 quantile cutoff taken from available read start distribution for data
2:  set or no. mappings / (0.1 * genome length) if not yet available
3:  minPercentIncrease ← 0.0025 quantile cutoff taken from available percent increase
4:  distribution for data set or 0 if not yet available
5:  coverage ← coverage arrays for current chromosome interval obtained by Algorithm 1
6:  readStarts ← arrays of read starts for current chromosome interval obtained by Algorithm 1
7:  add coverage and read starts from the last position of the previous chromosome interval to the
8:  current coverage and readStarts if this is not the first chromosome interval
9:  for each chromosome position p in current interval do
10:     (* Calculate TSS according to selected strand option. Only the default case is shown. *)
11:     if combine strands option is not set and feature strand is used do
12:         percentIncFwd ← coverage(forward)[p+1] / coverage(forward)[p] percentage
13:         percentIncRev ← coverage(reverse)[p] / coverage(reverse)[p+1] percentage
14:         readStartsFwd ← readStarts(forward)[p+1]
15:         readStartsRev ← readStarts(reverse)[p]
16:         if coverage and read start distributions have not been calculated yet do
17:             Add current read start and percent increase values to their respective distribution
18:         end if
19:         if readStartsFwd > min. no. read starts and percentIncFwd > min. percent increase
20:         do
21:             associatedFeatures ← compute associated features according to max. feature
22:             distance and strand option in linear time by keeping track of the index of the
23:             first feature in range. Features are natively sorted by genomic position.
24:             add current TSS to result including its position p+1, associatedFeatures,
25:             percentIncFwd and readStartsFwd
26:         end if
27:         (* Perform same calculation for reverse strand using percentIncRev and
28:         readStartsRev *)
29:     end if
30: end for
31:
32: if coverage and read count distributions have been calculated just now do
33:     Calculate new coverage percent increase and min. no. read start thresholds using the
34:     finished distributions
35:     for each TSS t in the result list do
36:         if read starts of t <= new min. no. read starts and percent increase of t <= new min.
37:         percent increase do
38:             remove TSS from result list
39:     end for
40: end if
```

The TSS detection algorithm (see Algorithm 4) runs on coverage and read start data and identifies all positions satisfying the parameters described above. Only for these positions the associated reference features are identified (correct, upstream or downstream features) to keep the computational overhead to a minimum. This feature identification step is designed to run efficiently in linear time for the whole analyzed data set.

As validation for the novel TSS analysis method, an RNA-Seq dataset from *Corynebacterium glutamicum* has been analyzed by this method using the automatic parameter estimation. It was then checked for a set of 223 experimentally validated *Corynebacterium glutamicum* TSS (see Table 8). The novel TSS detection approach was able to detect 78.87% of the TSS, which show at least the minimal required coverage increase. The remaining 21.13% do not satisfy the stringent automatic analysis parameters, either because of a low number of read starts or a stepwise coverage increase over multiple neighboring genome positions in this dataset. By relaxing the parameters manually, these TSS could also be identified.

**Table 8: Validation analysis of 223 known *C. glutamicum* TSS.** The table lists the detailed result of a comparison of 223 experimentally validated TSS with an RNA-Seq data set obtained from *C. glutamicum.* 194 of the 223 were manually identified in the analyzed data set. 29 TSS could not be observed, as there is no increase in coverage or read starts in the neighborhood of the expected genome position. 78.87% of the remaining 194 TSS were identified by our TSS detection method using the stringent automatic parameter estimation. For 13 TSS, an alternative start position has been detected in the analyzed data set in a distance between 13 and 45 bases up- or downstream of the expected TSS position. The 13 expected positions did not show any signs of a TSS in this data set. The lower part of the table apportions the undetected TSS according to their properties. Internal means that the start is observed within an expressed operon. Stepwise means that there is an observable amount of read starts/coverage increase around the analyzed position, but it is divided into multiple positions and thus, cannot be detected by our method. The undetected TSS can also be identified by manually choosing and relaxing the analysis parameters, depending on the user needs. The table and description are adapted from the supplement of (Hilker *et al.*, 2014).

| # TSS | Percentage | Description |
|---|---|---|
| 223 | 100.00% | All TSS |
| 29 | 13.00% | Absent TSS in analyzed data set |
| | | **Distribution of detectable TSS** |
| 194 | 100.00% | Detectable TSS in analyzed data set |
| 140 | 72.16% | Detected TSS |
| 13 | 6.70% | Alternative TSS in 13-45 bp distance |
| 153 | 78.87% | Total detected TSS |
| 41 | 21.13% | Undetected TSS |
| | | **Distribution of the undetected TSS** |
| 41 | 100.00% | Undetected TSS |
| 19 | 46.34% | Signal too weak |
| 9 | 21.95% | Signal too weak, internal |
| 5 | 12.20% | Stepwise increase, no single position with strong signal |
| 8 | 19.51% | Stepwise increase, no single position with strong signal, internal |

### 5.3.5. Differential Gene Expression Analysis

Transcript quantification by RNA-Seq offers a high resolution insight into expression levels. In comparison with other methods, like microarrays, it has been shown that RNA-Seq delivers far more accurate measurements (Wang *et al.*, 2009) for differential gene expression (DE) analysis. A brief description of several computational tools for DE analysis and their underlying concepts is given in Section 2.8.3. All mentioned tools are publicly available free of charge.

In ReadXplorer, two of these widely used tools for DE analysis have been integrated during the master's thesis of Kai Bernd Stadermann (Stadermann, 2013) under my supervision: **DESeq** (Anders and Huber, 2010) and **baySeq** (Hardcastle and Kelly, 2010). Since they both require R to be installed correctly, a third analysis has been implemented by the student: The **Express Test**.

The main goal for the additional approach was to implement an ultra-fast method running without external dependencies and without extensive statistical tests to complement the other two tools. This, of course, might come at the expense of accuracy, but it allows rapid insight into the data. The Express Test is limited to two conditions and calculates normalized gene expression values and a confidence value, but does not perform a statistical test for differential expression. If we have two conditions *A* and *B* with *n* and *m* replicates, we have the samples $\{a_1, a_2, \dots, a_n\}$ and $\{b_1, b_2, \dots, b_m\}$. $w \in \{n, m\}$ denotes the number of samples of one of the two conditions. The read count of the same $k$ regions $R_i$ with $i \in \{1, 2, \dots, k\}$ is analyzed in each of these samples. $R_{i,X}$ denotes all samples of one region belonging to a condition $X$, where $X \in \{A, B\}$. Mean and variance are then defined as:

$$mean(R_{i,X}) = \overline{R_{i,X}} = \frac{\sum_{v=1}^{w} R_{v,i,X}}{w}$$

$$variance(R_{i,X}) = \frac{\sum_{v=1}^{w} (R_{v,i,X} - \overline{R_{i,X}})^2}{w - 1}$$

Additionally to mean and variance, the ratios:

$$\frac{mean(R_{i,A})}{mean(R_{i,B})} \text{ and } \frac{mean(R_{i,B})}{mean(R_{i,A})}$$

are computed. If a mean value in the denominator is zero, it is replaced by one. In the last step, a confidence value $C_i$ is computed for each $R_i$ with $mean(R_{i,X}) > 0$ by the following equation:

$$C_i = -\log_{10}\left[\frac{1}{2}\left(\frac{variance(R_{i,A})}{mean(R_{i,A})} + \frac{variance(R_{i,B})}{mean(R_{i,B})}\right)\right]$$

For mean values of zero, $C_i$ is set to minus one. A high $C_i$ value can only be obtained by $R_i$ with a small variance. Hence, high ratio and confidence values indicate differential gene expression of the associated genomic region $R_i$. To account for inhomogeneous sequencing depth between the samples, the Express Test offers a normalization of the results. The normalization ratios can be computed based on all regions or based on a list of regions (e.g. housekeeping genes) adjustable by the user. The advantage of the Express Test is that it is completely implemented in Java. Therefore, it also works, if R is not available on the current machine.

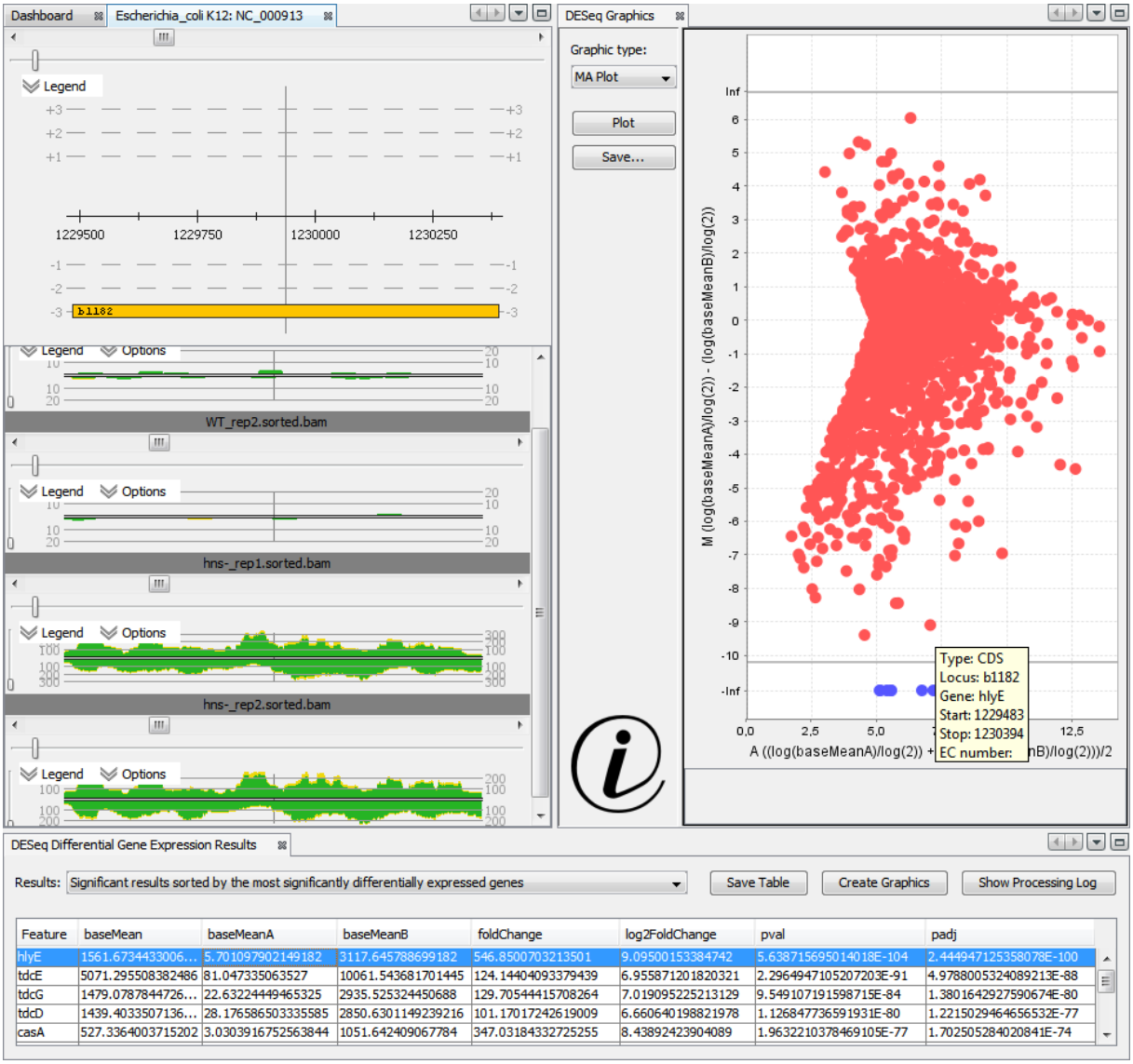



**Figure 51: DESeq differential gene expression result.** The table shows DESeq (Anders and Huber, 2010) results sorted by the most significantly differentially expressed genes (Feature column). A DESeq table contains the mean read counts of all tracks and the replicates of both conditions (baseMean/A/B) followed by the fold change (foldChange and log2FoldChange) and the original (pval) and adjusted p-value (padj). By clicking "Create Graphics" the plot window is opened. The MA plot enables visual exploration of the most significantly differentially expressed genes. In connection with the reference and track viewers (left side), the visual experience of the DE analysis is completed. They allow exploring the actual underlying data with a single mouse click. The *Escherichia coli* data set used to create this figure was published by Srinivasan *et al.* (2013).

All three DE tools can be configured conveniently by a wizard within ReadXplorer. This wizard queries the condition and for baySeq also group setup and the assignment of tracks to their respective condition or groups. An additional option available in the wizard is to export the raw count data into an XLS or CSV file for alternative downstream processing. The count data to be analyzed is gathered by ReadXplorer after completing the wizard. Algorithm 1 is used to read the data from the mapping files and specific analysis handlers have been implemented to carry out the analysis in R with the respective tool. To connect Java and R, the open source Java library rJava[26] is utilized. Note that rJava only supports one active R instance per Java virtual machine. This implies that currently only one DE analysis can be run simultaneously. Each DE analysis handler is capable of creating an R instance and transferring the read counts and parameter setup to R. Subsequently, the respective DE R-package analyzes the data set. After execution, the DE result is automatically re-imported into

[26] http://www.rforge.net/rJava

the Java environment of ReadXplorer (see Figure 51) and stored in an analysis result visualized in a result table. Further, the result panel offers buttons to show the log of the R-console for experts and to create and view analysis result plots. The set of available plots depends on the selected tool, but always contains an MA (log2 fold change (M) against normalized mean expression (A)) plot. The plots dependent on the analysis tool are static pre-rendered SVG images generated with R, while the additional MA plot is interactive. Thus, a selected point in the plot links back to the annotation it represents in the viewers (see Figure 51, tooltip in the plot). This feature allows effortless visual assessment of the original data on which the DE results are based.
For interactively plotting data, a new NBM has been created during the master's thesis of Kai Bernd Stadermann. It utilizes the open source JFreeChart[27] Java library to paint the plots.
The list of static plots for DESeq comprises:

- A gene dispersion (for details on the overdispersion problem see Section 2.8.3 and (Anders and Huber, 2010)) against normalized mean expression plot. This plot contains the empirical per gene dispersion values on the y-axis as black dots and the corresponding fitted dispersion values as a red line plotted against the mean expression strength of each gene on the x-axis. The plot is doubly logarithmic.
- An alternative log2 fold change (M) against normalized mean expression (A) plot generated by DESeq.
- A histogram of p-values. This plot visualizes the probability of genes not to be differentially expressed against its frequency in the experiment.

For baySeq, this list comprises:

- A priors plot. This plot shows the log prior probability means for a selected sample group.
- An alternative MA plot generated by baySeq.
- A plot of the posterior likelihoods of differential expression (see Section 2.8.3 and (Hardcastle and Kelly, 2010)) against log-ratio or log values. The log ratio is used when the value is not infinity and the log value is used where all data in the other sample group consists of zeros.

The convincing advantage of the integration of both command line R-packages is that the user does not need to know how to work with the console. A biologist can simply view the data in ReadXplorer, choose the DE tool of interest, design the DE experiment in the wizard and everything else is carried out automatically. All important results are directly accessible through ReadXplorer's visualizations (see Figure 51) and can be exported in standard, ready-for-publication formats.

### 5.3.6. Operon Detection

In prokaryotes, the annotation of operons (co-transcribed sets of genes resulting in a single polycistronic mRNA) is of high importance (Westover *et al.*, 2005). Operons can be identified in RNA-Seq data sets by analyzing the coverage (see Section 2.8.4). In terms of coverage, the most significant evidence in a track for two genes belonging to an operon is observed if these two neighbouring genes are connected by a reasonable amount of unique single reads overlapping both genes (see Figure 52). If the distance between both genes is too large to be spanned by single reads, also the minimal coverage in the interval between both genes can be taken into consideration.
In ReadXplorer, two neighbouring genes (protein-coding sequences or RNAs) are assigned to an operon if the number of observed reads spanning the two genes is higher than a user-

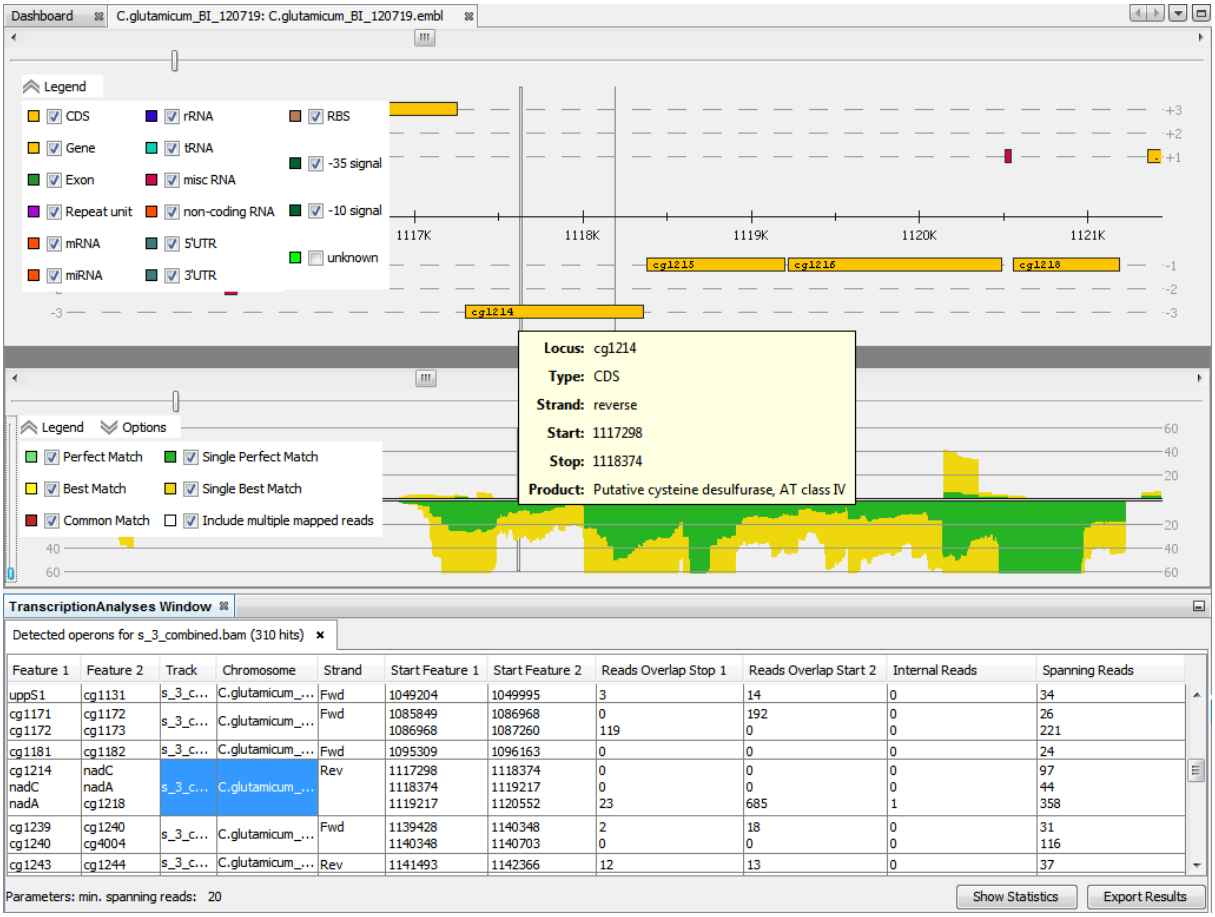
[27] http://www.jfree.org/jfreechart

defined threshold. This threshold varies from track to track amongst others depending on the background noise.

**Algorithm 5: Operon detection algorithm**

```
1:  features ← list of all features of all feature types included by the user for all chromosomes
2:  featurePairs ← map of the id of the first feature to a pair of neighbouring features with max.
3:  overlap of 20bp and max. distance of 1000bp of the same type on the same strand for all
4:  chromosomes and all feature types included by the user
5:  mappings ← list of mappings from a single chromosome interval obtained by Algorithm 1
6:  li ← 0 (* Index of last assigned mapping *)
7:
8:  for each feature f from features do
9:      if featurePairs contains id of f do
10:         isFstFittingMapping ← true
11:         fp ← featurePairs(id of f)
12:         strand ← determine analysis strand according to f and the parameters
13:         for i ← li, ..., size of mappings do
14:             if start of m > stop of second feature of fp do
15:                 break for (* Mappings are sorted by position *)
16:             else if strand of m does not satisfy strand do
17:                 continue with next mapping
18:             end if
19:             if start of m <= stop of first feature of fp and stop of m >= start of second feature
20:             of fp do
21:                 increase number of spanning reads for fp by 1
22:                 if isFstFittingMapping = true do
23:                     li ← i; isFstFittingMapping ← false
24:                 end if
25:             end if
26:         end for
27:     end if
28: end for
29:
30: (* After gathering data for all intervals of all chromosomes *)
31: lastFeatureId ← 0
32: operonAdjacencies ← empty list for feature pairs belonging to the same operon
33: for each entry fp in featurePairs do
34:     if no. spanning reads of fp >= min. no. spanning reads do
35:         if lastFeatureId ≠ first feature id of fp and lastFeatureId ≠ 0 do
36:             create operon in result list from current entries in operonAdjacencies
37:             clear operonAdjacencies
38:         end if
39:         add fp to operonAdjacencies
40:         lastFeatureId ← id of second feature from fp
41:     end if
42: end for
43: create operon in result list from current entries in operonAdjacencies
```

The operon detection algorithm (see Algorithm 5) is designed as follows: At first all neighbouring reference feature pairs are stored in a list. All genomic mappings are queried via Algorithm 1. Then, the mappings are assigned to their corresponding feature pairs. Mappings without features in their reach are not considered. The mappings are categorized in reads spanning both neighbouring features, reads overlapping only the end/start of one feature and reads located internally between both features without overlapping their borders. Only feature pairs whose number of spanning reads exceeds the given parameter are stored in the result. This process runs in linear time. Results of an operon detection are visualized in a table (see Figure 52).




| Feature 1 | Feature 2 | Track | Chromosome | Strand | Start Feature 1 | Start Feature 2 | Reads Overlap Stop 1 | Reads Overlap Start 2 | Internal Reads | Spanning Reads |
|---|---|---|---|---|---|---|---|---|---|---|
| uppS1 | cg1131 | s_3_c... | C.glutamicum_... | Fwd | 1049204 | 1049995 | 3 | 14 | 0 | 34 |
| cg1171<br>cg1172 | cg1172<br>cg1173 | s_3_c... | C.glutamicum_... | Fwd | 1085849<br>1086968 | 1086968<br>1087260 | 0<br>119 | 192<br>0 | 0<br>0 | 26<br>221 |
| cg1181 | cg1182 | s_3_c... | C.glutamicum_... | Fwd | 1095309 | 1096163 | 0 | 0 | 0 | 24 |
| cg1214<br>nadC<br>nadA | nadC<br>nadA<br>cg1218 | s_3_c... | C.glutamicum_... | Rev | 1117298<br>1118374<br>1119217 | 1118374<br>1119217<br>1120552 | 0<br>0<br>23 | 0<br>0<br>685 | 0<br>0<br>1 | 97<br>44<br>358 |
| cg1239<br>cg1240 | cg1240<br>cg4004 | s_3_c... | C.glutamicum_... | Fwd | 1139428<br>1140348 | 1140348<br>1140703 | 2<br>0 | 18<br>0 | 0<br>0 | 31<br>116 |
| cg1243 | cg1244 | s_3_c... | C.glutamicum_... | Rev | 1141493 | 1142366 | 12 | 13 | 0 | 37 |

**Figure 52: Operon detection result.** The table at the bottom shows an operon detection result. One operon is represented by one row entry. A single row represents two neighbouring features (left = Feature 1 column, right = Feature 2 column) being part of the same predicted operon. Hence, the selected predicted operon consists of four genes with continuous coverage (spanning reads column). By clicking the operon in the result table, it is centred in the reference and track viewers and the results can directly be inspected visually.

### 5.3.7. RNA Secondary Structure Prediction

The folding of RNA sequences, for example in untranslated regions, can reveal many structural and functional properties. To accommodate this, the ability to either fold a read alignment, or a sequence of interest chosen from the reference sequence has been included into ReadXplorer. For this purpose, the online service of **RNAfold** (Hofacker *et al.*, 1994; Hofacker, 2003) is queried. The graphical output is generated by an integrated version of **RNA Movies** (Evers and Giegerich, 1999) (see Figure 53).
The RNAMovies component also enables direct export of the visualization of the folded sequence.

A generic menu item to fold a sequence has been implemented. It is used in the `SequenceBar` to enable folding a sequence of choice in each viewer with a `SequenceBar` and in the `AlignmentViewer` to fold a single read alignment.

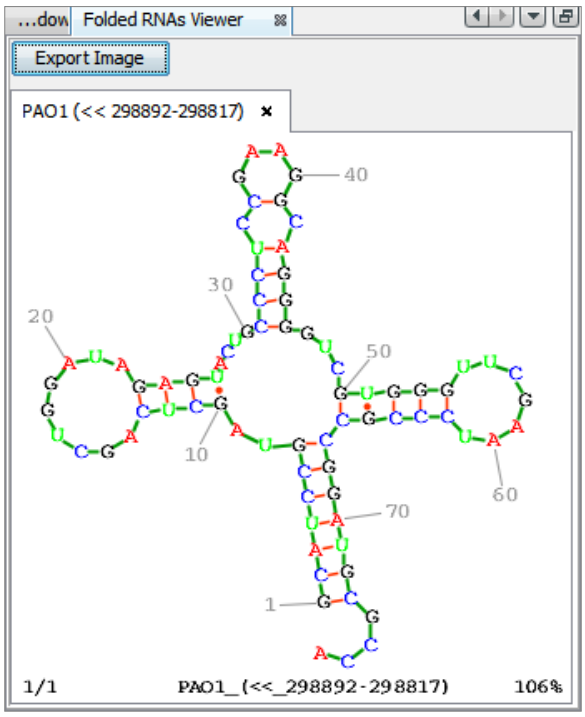


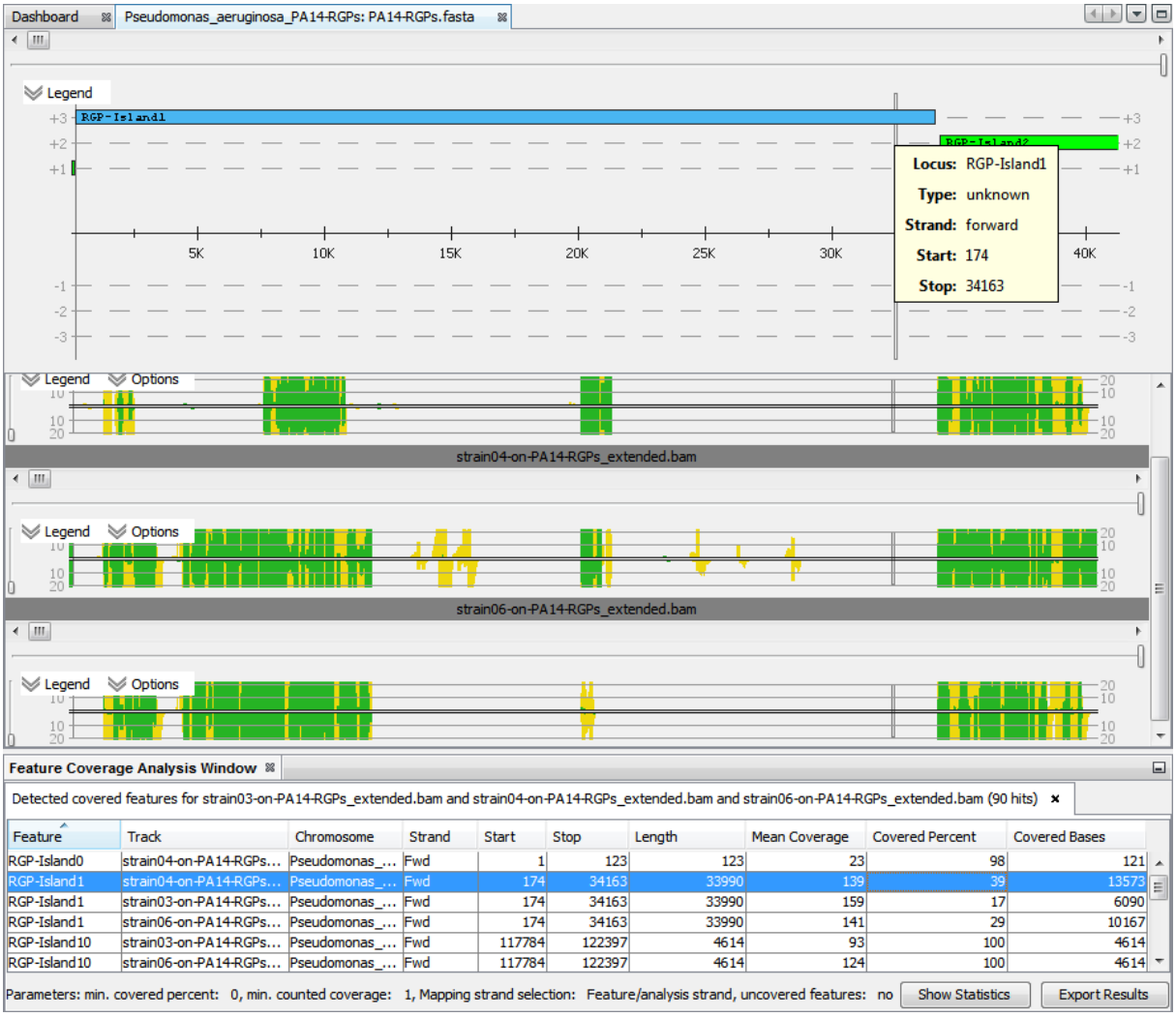

**Figure 53: Folded RNAs Viewer.** The embedded version of RNAMovies (Evers and Giegerich, 1999), showing the tRNA for arginine from *P. aeruginosa*. The tab header and the description below the RNA structure both reveal the name and the exact base positions of the folded sequence.

### 5.3.8. Coverage Analysis

Two coverage analyses are natively available in ReadXplorer and empower an experimenter to detect all reference features or reference intervals that show exceptional characteristics in terms of their coverage. They allow either listing of all features or intervals which are covered to a high extent by mapped reads or, on the contrary, which are not covered by a satisfactory amount of reads.

These are quite basic coverage analyses. But they are especially important, because in practice they can aid a variety of experiments by fast extraction of genomic regions of interest. They are useful for studies of both RNA-Seq and resequencing data sets. For RNA-Seq experiments, they facilitate the identification of genes or intervals which exhibit a certain minimum coverage or which are not expressed or covered at all. For resequencing experiments, the feature coverage analysis enables exploration of the set of common or divergent features between the reference and its tracks, but also among tracks mapped on the same reference (see Figure 54).

The general coverage analysis e.g. allows identification of regions which could not be sequenced, are not present in the mapped genome (see Figure 55) or show a significantly higher than the average coverage.




| Feature | Track | Chromosome | Strand | Start | Stop | Length | Mean Coverage | Covered Percent | Covered Bases |
|---|---|---|---|---|---|---|---|---|---|
| RGP-Island0 | strain04-on-PA14-RGPs... | Pseudomonas_... | Fwd | 1 | 123 | 123 | 23 | 98 | 121 |
| RGP-Island1 | strain04-on-PA14-RGPs... | Pseudomonas_... | Fwd | 174 | 34163 | 33990 | 139 | 39 | 13573 |
| RGP-Island1 | strain03-on-PA14-RGPs... | Pseudomonas_... | Fwd | 174 | 34163 | 33990 | 159 | 17 | 6090 |
| RGP-Island1 | strain06-on-PA14-RGPs... | Pseudomonas_... | Fwd | 174 | 34163 | 33990 | 141 | 29 | 10167 |
| RGP-Island10 | strain03-on-PA14-RGPs... | Pseudomonas_... | Fwd | 117784 | 122397 | 4614 | 93 | 100 | 4614 |
| RGP-Island10 | strain06-on-PA14-RGPs... | Pseudomonas_... | Fwd | 117784 | 122397 | 4614 | 124 | 100 | 4614 |

**Figure 54: Feature coverage analysis.** The coverage of the RGPs from *P. aeruginosa* strain PA14 is analyzed using the feature coverage analysis. Each RGP sequence is represented by a genomic feature of type "unknown" (green/blue) in the reference viewer. The feature coverage analysis was run on 3 data sets simultaneously. This universal ReadXplorer analysis option offers a great opportunity for a side by side comparison of as many data sets as desired. The result table gives in-depth information regarding the coverage of each RGP in the analyzed three *P. aeruginosa* strains E429 (strain03, top), 239A (strain04) and F429 (strain06). The table shows detailed information regarding the analyzed genomic feature (in the columns feature, chromosome, strand, start, stop, length) and its coverage (mean coverage, covered percent, covered bases). The linkage of the result table with the visualization enables direct comparison of the observed results with the actual coverage distribution, as shown here for the selected RGP-Island1 (blue in table and reference viewer).

The **feature coverage analysis** requires two parameters: The first parameter defines how many percent of a feature have to be covered with at least a coverage larger or equal to the second "minimum coverage" parameter value to report the feature in the result table. A checkbox enables switching the analysis to all features not satisfying the given parameters, thus, detecting the "uncovered" features of the reference genome.
The **coverage analysis** also relies on two parameters: A minimum coverage has to be given for each position. Additionally, the user has to choose between summing up the coverage of both strands or counting each strand separately. Then, the analysis lists all reference intervals fulfilling the minimum coverage parameter. As mentioned above, a check box is available for switching to the alternative mode computing all intervals which do not satisfy the minimum coverage parameter. Additionally, the coverage analysis allows exporting the DNA sequences of all detected (un)covered intervals in multiple FASTA format for downstream analyses (see Figure 55).

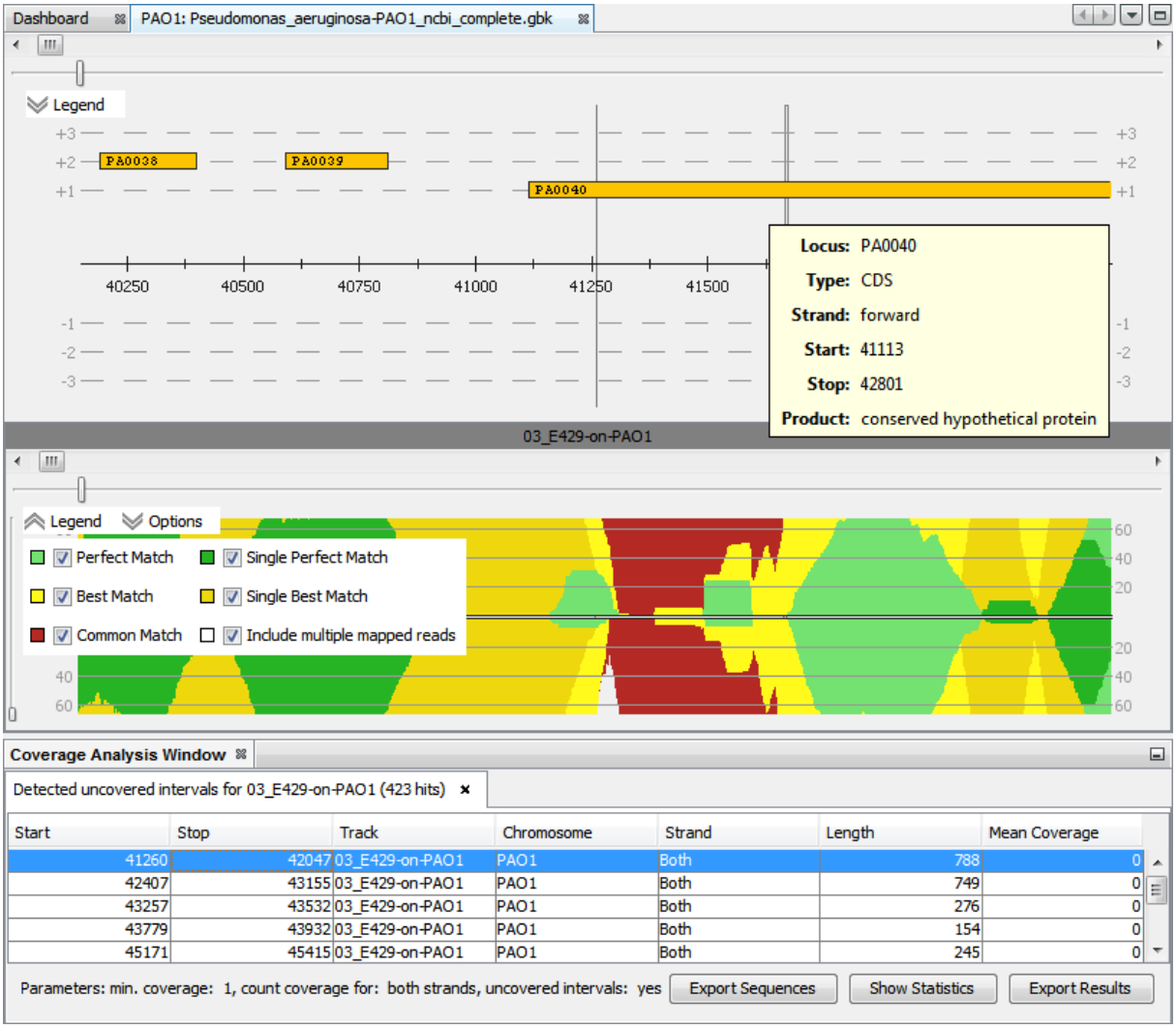



**Figure 55: Coverage analysis.** The coverage analysis was run for *P. aeruginosa* strain E429 to detect all regions of the genome which are not covered by at least one mapping of the Single Perfect or Single Best Match class. In the selected example, an interval of 788bp has been identified within PA0040. Due to all Perfect, Single and Common Match mappings in the detected interval, it is obvious that in this example we identified a repetitive region, occurring at least twice in the reference genome PAO1. The result table does not only list the interval boundaries and length, but also reveals its mean coverage. The "Export Sequences"-button enables convenient export of the underlying DNA sequences from the detected intervals.

The algorithm to identify covered or uncovered features first obtains all reference features. Next, the coverage of all selected data sets is queried via Algorithm 1 and all positions satisfying the given parameters are counted for each reference feature. After finishing the coverage queries, the map of features is iterated and only features satisfying the given analysis parameters are stored in the result list (for details see Algorithm 6).

**Algorithm 6: Feature coverage analysis algorithm**

```
1:  features ← list of all features of all feature types included by the user for all chromosomes
2:  coverage ← the list of coverage arrays of a single chromosome interval for each included
3:  mapping class obtained by Algorithm 1
4:  for each feature f from features do
5:      if chromosome id of f = chromosome id of m do
6:          noCoveredBases ← 0
7:          strand ← determine analysis strand according to f and the parameters
8:          for i ← start of f, ..., stop of f do
9:              cov ← coverage at i according to strand
10:             if cov >= min. coverage do
11:                 noCoveredBases++
12:                 increase coverage sum by cov
13:             end if
14:         end for
15:         store noCoveredBases for f
16:         compute and store mean coverage of f as coverage sum of f / noCoveredBases
17:     end if
18: end for
19:
20: (* After gathering data for all intervals of all chromosomes *)
21: for each feature f in features do
22:     covPercent ← noCoveredBases of f / length of f * 100
23:     if covPercent > min. covered percent and detect covered features or
24:       covPercent <= min. covered percent and detect uncovered features do
25:         add f and its coverage data (covPercent, noCoveredBases, mean coverage) to a result
26:         list
27:     end if
28: end for
```

The algorithm to identify covered or uncovered intervals of the reference (see Algorithm 7) queries the coverage of all selected data sets via Algorithm 1. Each position of the reference is then analyzed for the given analysis parameters and new intervals are started whenever the analysis parameters are satisfied for the first time. The current interval is extended until the first position not complying with the analysis parameters. Intervals spanning the borders of the chunks in which the coverage is queried are considered by additional temporary data structures not shown in Algorithm 7.

**Algorithm 7: Coverage analysis algorithm**

```
1:  coverage ← the array of the summed coverage of each included mapping class of a single
2:  chromosome interval obtained by Algorithm 1
3:  If this is not the first interval of the chromosome check if the previous adjacent interval can be
4:  extended by the first entry of the appropriate array from coverage and extend it if necessary
5:  isInInterval ← false
6:  summedCoverage ← 0
7:  for i ← 0, ..., length of coverage do
8:      if coverage[i] >= min. coverage and detect covered intervals or
9:        coverage[i] < min. coverage and detect uncovered intervals do
10:           isInInterval ← true
11:           summedCoverage + coverage[i]
12:     else if isInInterval = true do
13:           add current interval and its data (start, stop, mean coverage (as summedCoverage /
14:           length of interval)) to a result list
15:           isInInterval ← false
16:     end if
17: end for
18:
19: (* After gathering data for all intervals of all chromosomes *)
20: for each feature f in features do
21:     covPercent ← noCoveredBases of f / length of f * 100
22:     if covPercent > min. covered percent do
23:           add f and its coverage data (covPercent, noCoveredBases, mean coverage) to a result
24:           list
25:     end if
26: end for
```

## 5.4. ReadXplorer Evaluation

The features implemented in ReadXplorer are summarized and compared to other genome browsers reviewed during this work in Table 9. In total, ReadXplorer offers the most comprehensive set of features and it is the only tool offering an own in-depth read mapping classification. The effectiveness of ReadXplorer is illustrated in Table 10 by showing the computational performance and requirements for several common tasks. The import process is the bottleneck with the longest execution time, but this task only has to be performed once for each data set. Additionally, a data set which has been classified by ReadXplorer once can skip the time consuming steps of the import in all subsequent imports in other ReadXplorer databases. The automatic analysis functions only require a few seconds or minutes to run, scaling linearly with the number of reads to analyze. In general, the memory requirement of importing data and performing single analyses is less than 1GB. Note that the memory requirements can grow larger when multiple analyses are run in parallel.

**Table 9: Comparison of popular published read mapping visualization and analysis tools.** Here, the most important features of all compared tools are listed. Features present in a tool are depicted by a "✓"and highlighted in green. Available features requiring a more detailed explanation are also highlighted in green. Features which are present with constraints are marked by "(✓)" and highlighted in yellow. Features which are planned to be implemented are shown as "× (planned)" and highlighted in orange. All features not supported by a tool are marked by an "×" and highlighted in red. The last two columns show the RAM usage for two common visualization tasks tested on the same region of the same data set. The table and descriptions are an updated version of Table S3 from (Hilker *et al.*, 2014).

| **Feature** | **Savant 2.0.4** | **IGB 8.0.1** | **Artemis 16.0** | **IGV 2.3.26 + Rockhopper 1.3.0** | **GenomeView 2450** | **ReadXplorer 2.1** |
|---|---|---|---|---|---|---|
| Reference formats | BED, GFF3, GTF, Fasta | Genbank, GFF2/3, Fasta, BED | Genbank, EMBL, Fasta | BED, GFF2/3, GTF, Fasta | BED, EMBL, Genbank, GFF3, GTF, Fasta, PTT, TBL | Genbank, EMBL, GFF2/3, GTF, Fasta |
| Mapping formats | BAM | SAM, BAM | BAM, only one at a time or combination of multiple files | SAM, BAM | BAM, MAF | SAM, BAM, JOK |
| 6 frame view | × | × | ✓ | (✓) one strand at a time | ✓ | ✓ |
| Mapping classification | × | × | × | × | × | ✓ |
| Filter mappings | Mainly pair SAM flags & mapping quality | Some SAM flags, mapping & base quality | All SAM flags & mapping quality | × | × | Read class, mapping & base quality |
| Color mappings | Mapping & base quality | Base quality & some SAM flags | (✓) Base quality on lowest zoom level | Some SAM flags & SAM tags | Mapping quality | Read (pair) class, mapping & base quality |
| Coverage graph | ✓ | ✓ | (✓) no position specific details | ✓ | ✓ | ✓ |
| Read pair visualization | ✓ | × | ✓ | ✓ | ✓ | ✓ |
| Alignment view | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Histogram viewer | × | × | × | ✓ | ✓ | ✓ |
| Methylation viewer | × | × | × | ✓ | × | × (planned) |
| Thumbnail viewer | × | × | × | × | × | ✓ |
| G/C plot | × | × | ✓ | × | × | × (planned) |
| Track comparison | × | × | × | × | (✓) Only for multiple genome alignment | ✓ |
| Track combination | × | × | ✓ | ✓ | × | ✓ |
| Scaling | ✓ | ✓ | ✓ | ✓ | × | ✓ |
| Strand specific visualizations | (✓) only for coverage plot | ✓ | ✓ | (✓) only for alignments | ✓ | ✓ |
| Genetic code selection | × | × | × | ✓ | × | ✓ + Addition of own codes |

| | | | | | | |
|---|---|---|---|---|---|---|
| SNP calling | ✓ | × | × | × | × | ✓ |
| VCF-support | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| RNA-Seq analyses | Isoforms, abundancies, fragment length | × | Read count and RPKM calculation for selected features | Normalization , transcript assembly, RPKM, read count, operons | × | TSS, novel transcript, TPM, RPKM, read count, operon, (feature) coverage |
| Differential gene expression | edgeR | × | × | Similar to DESeq | × | DESeq, baySeq, Express Test |
| Genome rearrangements | × | × | × | × | × | GASV |
| Continuous data support | TDF, BigWIG | BigWIG, BedGraph | × | TDF, BigWIG, WIG | TDF, BigWIG, WIG, pileup | × (planned) |
| Indexing | ✓ | × | ✓ except Fasta | ✓ except BAM | ✓ | ✓ |
| Bookmarks | ✓ | ✓ | ✓ | ✓ | × | × (planned) |
| Reformat data | Indexing and zipping BED, GenePred, GFF, GTF, VCF, WIG, BedGraph, BAM | × | Indexing BAM | Count, Sort, Index (except BAM), toTDF | Indexing Fasta, BED, BAM | Tab-separated (csv), Indexing BAM, SAM to BAM, JOK to BAM |
| Project based | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Predefined genome list | ✓ | ✓ | × | ✓ | ✓ | × (planned) |
| Restriction sites | × | ✓ | × | × | × | × |
| Screenshots | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Edit annotations | × | × | ✓ | × | ✓ | × (planned) |
| Pattern search | × | × | ✓ | ✓ | ✓ | ✓ |
| Database searches | × | NCBI | NCBI, Pfam, Rfam | × | NCBI | Enzyme DBs (e.g. ExPASy) |
| Export data | Fasta | BED, bedgraph | Fasta, PIR database, EMBL, Genbank, GFF, Sequin | BED, txt, TDM | EMBL, GFF3 | Xls, CSV, Fasta |
| Genome visualization (10kb) | 220 MB | 230 MB | 230 MB | 250 MB | 250 MB | 125 MB |
| Mapping visualization (10kb) | 450 MB | 330 MB | 380 MB | 460 MB | 390 MB | 220 MB |

**Table 10: Analysis and import benchmark of ReadXplorer.** This table shows the time and memory requirements for common tasks within the ReadXplorer software on a Windows 7 x64 Laptop with an i7-2630QM 2GHz processor and 8GB RAM. Additionally, the genome size and the number of mapped reads of the investigated data set are listed for each task. Reference imports include the genomic features and sequence. The "Overall Memory" column refers to the maximum observed memory usage during the task. The memory usage does not only include the memory footprint of the task, but the overall used memory for displaying the reference with genomic features, the data sets used in the task and the task itself. Further note that the observed memory footprint might be higher than necessary, since ReadXplorer was run with more than 1GB of available memory. This means, that the Java garbage collector does not always clean up the memory and possibly unneeded objects are still in memory. The table is taken from the supplement of (Hilker *et al.*, 2014). "s" abbreviates "second".

| Task | Genome Size | Mapped reads | Time | Overall Memory |
|---|---|---|---|---|
| Import/index *Pseudomonas aeruginosa* | ~ 6.26 mb | - | 8 s | 150 MB |
| Import/index *Caenorhabditis elegans* | ~ 99 mb | - | 14 s | 200 MB |
| Import/index *Homo sapiens* | ~ 30 gb | - | 8:05 min | 500 MB |
| Import *P. aeruginosa* read pair reads | ~ 6.26 mb | 22 356 000 | 45:39 min | 900 MB |
| Import *P. aeruginosa* read pair reads | ~ 6.26 mb | 16 000 000 | 29:03 min | 800 MB |
| Import *P. aeruginosa* read pair reads | ~ 6.26 mb | 8 780 000 | 15:52 min | 800 MB |
| Import *P. aeruginosa* reads | ~ 6.26 mb | 22 356 000 | 45:10 min | 900 MB |
| Import *P. aeruginosa* reads | ~ 6.26 mb | 16 000 000 | 29:07 min | 800 MB |
| Import *P. aeruginosa* reads | ~ 6.26 mb | 8 780 000 | 16:18 min | 800 MB |
| Import *Chlamydomonas reinhardtii* reads | ~ 120 mb | 82,000,000 | 151 min | 1000 MB |
| DESeq analysis | ~ 3 mb | 19,400,000 | 1:29 min | 510 MB |
| DESeq analysis | ~ 120 mb | 82,000,000 | 8:39 min | 800 MB |
| baySeq analysis | ~ 3 mb | 19,400,000 | 6:03 min | 390 MB |
| baySeq analysis | ~ 120 mb | 82,000,000 | 62 min | 800 MB |
| Express test | ~ 3 mb | 10,770,000 | 45 s | 400 MB |
| Express test | ~ 120 mb | 82,000,000 | 5:23 min | 700 MB |
| TSS detection | ~ 3 mb | 7,839,000 | 19 s | 510 MB |
| TSS detection | ~ 3 mb | 17,069,000 | 1:10 min | 440 MB |
| TSS detection | ~ 120 mb | 82,000,000 | 6:27 min | 480 MB |
| Read count and normalization calculation | ~ 3 mb | 7,839,000 | 25 s | 520 MB |
| Read count and normalization calculation | ~ 120 mb | 82,000,000 | 2:32 min | 860 MB |
| Operon detection | ~ 3 mb | 17,069,000 | 36 s | 430 MB |
| Operon detection | ~ 120 mb | 82,000,000 | 5:19 min | 590 MB |
| Coverage analysis | ~ 6.26 mb | 12,115,000 | 37 s | 420 MB |
| Coverage analysis | ~ 120 mb | 82,000,000 | 3:19 min | 480 MB |
| Feature Coverage analysis | ~ 6.26 mb | 12,115,000 | 41 s | 480 MB |
| Feature Coverage analysis | ~ 120 mb | 82,000,000 | 3:20 min | 560 MB |
| SNP detection (55,000 SNPs found) | ~ 6.26 mb | 12,115,000 | 7:03 min | 480 MB |
| SNP detection (57.000 SNPs found) | ~ 120 mb | 82,000,000 | 8:35 min | 710 MB |

# Chapter 6

# *Pseudomonas aeruginosa* Pan Genome Analysis

This chapter starts with an introduction to *P. aeruginosa*, the state of the art of comparative genomics approaches regarding this pathogen and the 20 *P. aeruginosa* strains subject to this dissertation. After this introduction, the here established workflow for comparative genomics analyses is defined and the individual steps are explicated. The next section describes the comparative genomics results obtained with the software EDGAR (see Section 2.5.3). It is followed by another comparative genomics study based on the SNPs and DIPs identified against a common reference strain with the newly implemented analysis methods of the software ReadXplorer (see Chapter 5). The last section deals with the comparative analysis of genomic islands and regions of genomic plasticity also performed with analysis methods implemented during this work into ReadXplorer (see Section 5.3.8). This chapter is mainly based on the *P. aeruginosa* pan genome analysis publication by Hilker *et al.* (2014).

## 6.1. Comparative Genomics of *Pseudomonas aeruginosa*

*P. aeruginosa* is a gram-negative, metabolically very versatile gammaproteobacterium which can be found around the globe in soil as well as in water habitats. It prefers moist surfaces and has been reported to colonize plants, mammals, insects and worms. In animals and humans, it is found within the skin microbiome and thrives within wounds and in immunocompromised hosts. The habitat temperatures can range from 4°C - 42°C. (Ramos, 2004-2010a, 2004-2010b)

As an opportunistic pathogen, *P. aeruginosa* causes inflammation and sepsis within infected hosts. *P. aeruginosa* infections can severely sicken and even kill the host. Therefore, this ubiquitous bacterium plays an important role for human health care. It is involved in a wide range of infections mainly affecting the pulmonary or urinary tract, open wounds and burns. Typical diseases involving *P. aeruginosa* are the Chronic obstructive pulmonary disease (COPD) (Murphy, 2008), ventilator associated pneumonia (VAP) (Crouch Brewer *et al.*, 1996) and urinary catheter infections (Mittal *et al.*, 2009).

The population structure of *P. aeruginosa* has been shown to be epidemic (Pirnay *et al.*, 2009; Selezska *et al.*, 2012). Most of the several hundred clonal complexes from environmental and disease habitats identified by genotyping in the recent years are rare (Wiehlmann *et al.*, 2007; Cramer *et al.*, 2012). 40% of today's *P. aeruginosa* population is made up by the 15 most frequent clones. Representatives of the two major clones C (Römling *et al.*, 2005) and PA14 (Rahme *et al.*, 1995) have been identified to populate numerous habitats around the globe. A few examples are: chronic and acute human infections, man-made environments, fresh and

salt water, solitary national reserves, animals and plants. However, the next frequent clones preponderate for geographic areas and/or habitats. Numerous clones have no representative as yet among the subset of human infections and conversely, clones that had caused outbreaks of nosocomial infection still lack an environmental isolate in the strain collection. These data suggest that the *P. aeruginosa* population consists of global and local generalists on one hand and niche specialists on the other. There even may exist an interclonal gradient of pathogenicity ranging from innocuous to highly virulent clones.

The first completely sequenced *P. aeruginosa* strain was PAO1 (see Figure 56) (Stover *et al.*, 2000). It serves as major reference strain until today. In the meantime, several other strains like the aforementioned highly virulent PA14 (Lee *et al.*, 2006) followed and have been deposited into online databases (Winsor *et al.*, 2011). The genome size of *P. aeruginosa* is between 5.5 and 7.5mb[28] and consists of a single chromosome and a variable number of different plasmids. All strains are characterized by a high GC content of 65 - 67% and mostly contain approximately 5500 - 6500 genes. Besides the general core genome, the dispensable genome consists of an additional clone specific core genome (Wiehlmann *et al.*, 2007). Wiehlmann *et al.* also state that the *P. aeruginosa* genome is assembled non-randomly: "Individual clones prefer a specific repertoire of accessory segments. Moreover, some parts of the core genome tolerate only a subset of the possible combinations of sequence variants, whereas other segments are freely recombining." In *P. aeruginosa* many important genes for pathogenicity are located in the dispensable genome (Klockgether *et al.*, 2011). Hot spots of genomic island and RGP integration in the *P. aeruginosa* genome are tRNA genes. Notable differences to core genome regions are the anomalous mono- to tetradecanucleotide usage and GC content of RGP sequences (Klockgether *et al.*, 2011; Kung *et al.*, 2010). For *P. aeruginosa*, the GC content of RGPs is mostly lower than the high GC content of the core genome (Kung *et al.*, 2010). Further, RGPs can be identified by mobility factors in their flanking regions. Frequently, the content of RGPs is even a mosaic of regions from different mobile elements (Klockgether *et al.*, 2004). The above mentioned studies act on the assumption that *P. aeruginosa* has an open pan genome, because all newly analyzed strains added several genes to the pan genome while the core genome slowly decreased.

The relevance of the dispensable genome for medicine becomes apparent when the acquisition of prevalence increasing secondary pathways (Aguilar-Barajas *et al.*, 2010; Campos-García, 2010), new virulence factors (He *et al.*, 2004), or antibiotic resistances (Mesaros *et al.*, 2007) is considered. Another important source of adaptability are plasmids. In medical science very often antibiotic resistance plasmids play an important role, especially since multidrug resistant bacteria have become a severe issue (Shahid *et al.*, 2003).

We have described the study in our publication (Hilker *et al.*, 2014) as follows:
A drawback of the so far sequenced *P. aeruginosa* strains is that except PA14 they all belong to uncommon clonal complexes in the bacterial population. In this study, the genomes of 15 *P. aeruginosa* strains that are representative for the 15 most frequent clonal complexes according to Wiehlmann *et al.* (2007) in the *P. aeruginosa* population have been analyzed. Based on the hypothesis that clonal complexes may occur in the environment which cannot cause disease in humans, five more strains from soil, plants and aquatic habitats have been added to this panel for genome sequencing, each of which representing a common clonal complex of which so far no isolate from a human niche has been detected. To address this issue of whether there is an association between clonal frame and virulence, the chosen strain panel has been analyzed computationally and by infection models in the laboratory.

[28] http://www.ncbi.nlm.nih.gov/genome/genomes/187

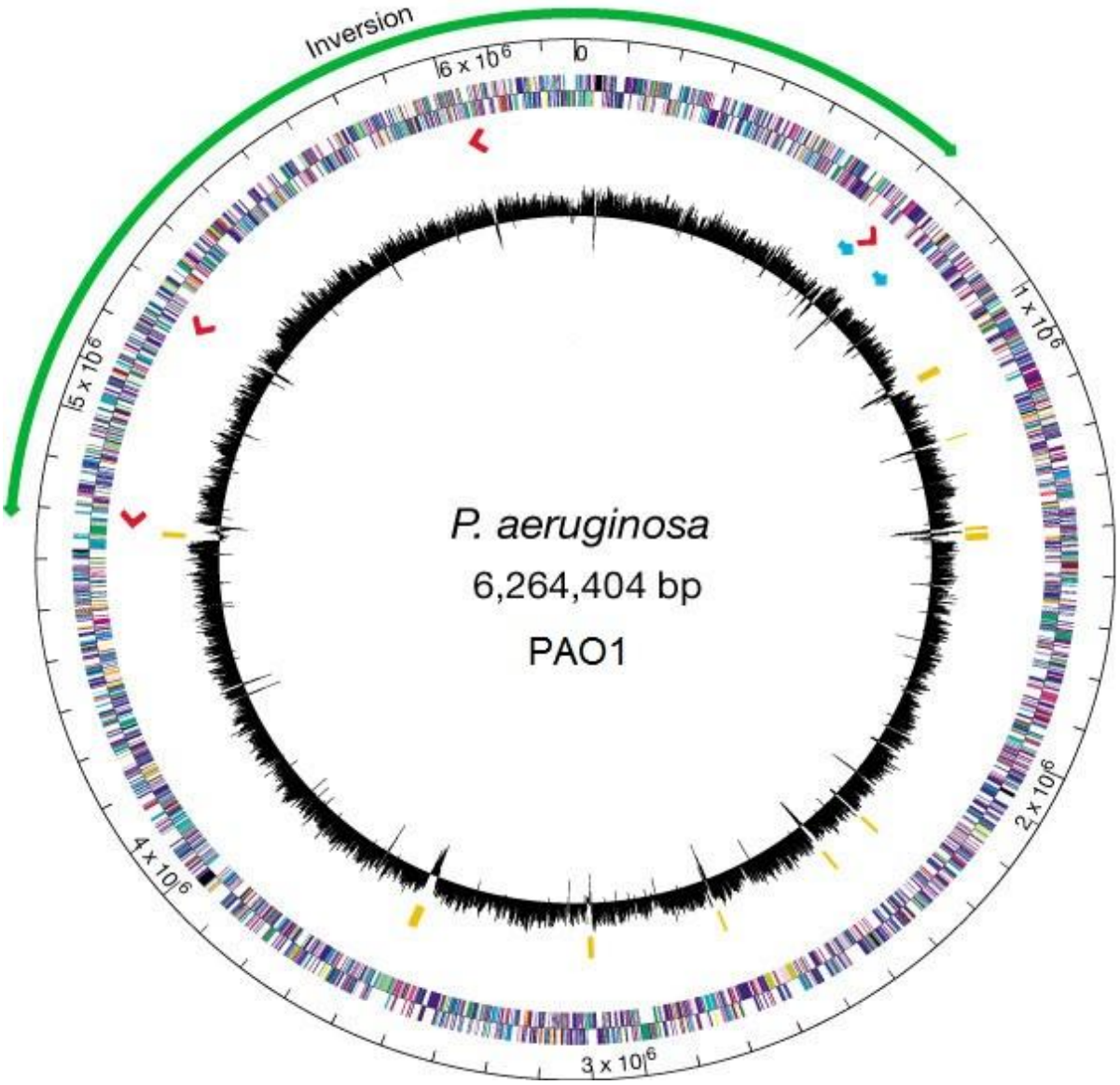



**Figure 56: Circular representation of the *P. aeruginosa* genome.** The outermost circle indicates the chromosomal location in base pairs (each tick is 100kb). The distribution of genes is depicted by coloured boxes according to functional category and direction of transcription (outer band is the plus strand; inner band is the minus strand). Red arrows, the locations and direction of transcription of ribosomal RNA genes; green arrow, the inverted region that resulted from a homologous recombination event between *rrnA* and *rrnB*; blue arrows, location of two regions containing probable bacteriophages. The black plot in the centre is percentage G+C content plotted as the average for non-overlapping 1-kb windows spanning one strand for the entire *P. aeruginosa* genome. Yellow bars, regions of ≥ 3.0 kb with G+C content of two standard deviations (< 58.8%) below the mean (66.6%). A linear map of the genes, with the colour code for functional categories, is available at http://www.pseudomonas.com/functionClasses.jsp. Figure and description are adapted from (Stover *et al.*, 2000).

Three different infection models have been used to determine the virulence of the separate strains in the laboratory. The computational analysis refers to NGS analyses of the pan genome of the chosen strain panel. The infection model experiments are an important part of our publication (Hilker *et al.*, 2014), but not subject of this dissertation. However, results from this analysis are correlated to the computational analysis results generated within the framework of this dissertation in the following subsections. The selection of the 20 strains enables analyzing the sequence diversity and gene content among the most common clonal complexes and allows estimating the gene pool of the pan genome of *P. aeruginosa*. For 19 of the 20 genomes it was planned to deliver draft genome sequences, while for the pathogenic and prevalent clone C the complete and finished genome sequence should be decoded. With a complete high quality genome sequence in hand, a powerful foundation for the analysis of the pathogenicity and prevalence of clone C is made available (Fischer *et al.*, in preparation).

Further, the complete genome sequence of relevant strains is needed as reference to investigate the pathogenic properties and potential in future analyses comparing *P. aeruginosa* strains.

## 6.2. Pan Genome Analysis Workflow Establishment

The 15 most common clonal complexes in the *P. aeruginosa* population were represented by isolates from the environment (2×), acute eye infection (1×), community-acquired pneumonia (1×), intubated patients (3×) and chronic airway infections of individuals with cystic fibrosis (CF) (5×) or chronic obstructive pulmonary disease (COPD) (3×). Five further environmental isolates were recovered from plants (2×), soil (1×), fresh (1×) and salt water (1×) respectively. The detailed list of the strains is shown in Table 11. The strains were deposited at the German Collection of Microorganisms and Cell Cultures (DSMZ) and are available under accession numbers DSM29238-29241, DSM29272-29281 and DSM29304-29311.

**Table 11: *P. aeruginosa* strain information.** The first 15 strains represent the 15 most common clonal complexes in the *P. aeruginosa* population (sorted according to decreasing frequency). The lower five strains were from the most common clonal lineages without any clinical representatives found so far. Table and description taken from (Hilker *et al.*, 2014). Clinical strains have been highlighted in red, the five most common environmental strains in green and the other two river isolates in light green. This coloration is used in all subsequent strain tables.

| Clone | Isolate | Source |
|---|---|---|
| C40A | NN2 | CF lungs, Germany |
| D421 | RN3 | CF lungs, Germany |
| F469 | 60P57PA | COPD airways, USA |
| 0C2E | RP1 | CF lungs, Germany |
| E429 | 15108/-1 | Intubated patient, Spain |
| 239A | 13121/-1 | Intubated patient, France |
| 2C22 | 57P31PA | COPD airways, USA |
| F429 | A5803 | Pneumonia, Germany |
| B420 | 120SD3 | River, Germany |
| EA0A | 39177 | Keratitis, UK |
| 0812 | 27103 | Intubated patient, France |
| 2C1A | 18P17PA | COPD airways, USA |
| 1BAE | KK1 | CF lungs, Germany |
| 3C2A | TR1 | CF lungs, Germany |
| EC2A | PT22 | River, Germany |
| EC21 | 100 | Pacific Ocean, Japan |
| 843A | BP35 | Pepper plant, India |
| 0822 | E501 | Tomato plant, Italy |
| 149A | 120SB2 | River, Germany |
| 478A | M41A.1 | Soil, Colombia |

Sequencing of the 20 genomes was carried out by GATC Biotech[29] on an Illumina Genome Analyzer II (see Section 2.1.3). Beforehand, tagged paired-end libraries were prepared by my collaborators at the MH Hanover following the manufacturer's instructions. To be able to finish the clone C genome sequence, they prepared an additional 3kb mate pair library for the Illumina Genome Analyzer II and a 454 sequencing library for a Roche Genome Sequencer FLX (see Section 2.1.2), both sequenced by GATC. The resulting genome sequence reads,

[29] GATC Biotech AG, Constance, Germany

except the two additional libraries for clone C, were deposited in the European Nucleotide Archive (ENA) hosted by EMBL/EBI, accession no. PRJEB4961.
All computational processing steps are described in detail in their corresponding subsection and a global overview of the analysis workflow is given in Figure 57.
At first, the sequencing data sets were analyzed for their quality (see Section 6.2.1). From here on, two different processing paths were taken: 1. *de novo* assembly (see Sections 6.2.2) and annotation (see Section 0) and 2. read mapping on the PAO1 genome (see Section 6.2.4), both followed by pan genome analyses described in the following Sections 6.3 - 6.5.

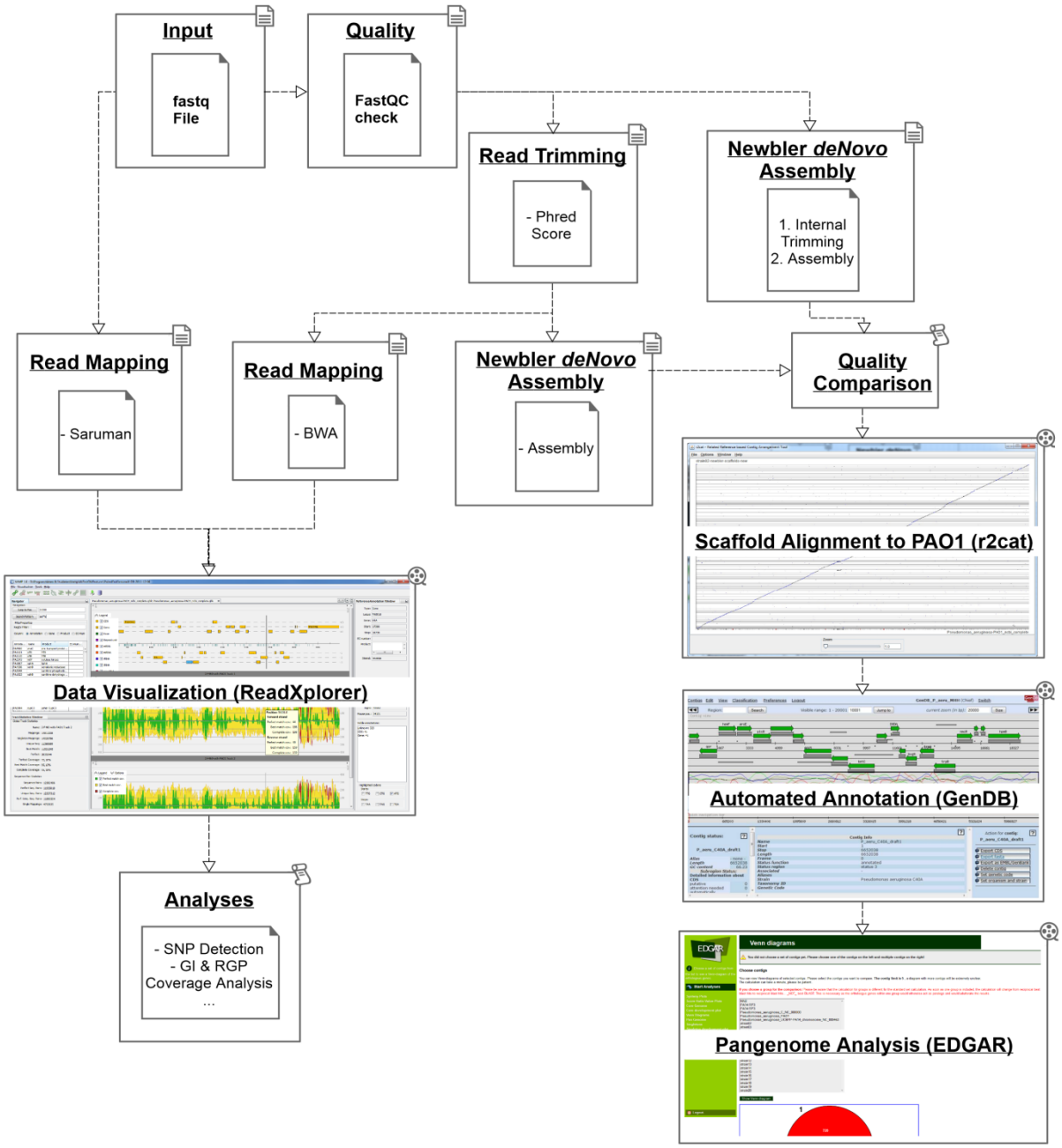



**Figure 57: *P. aeruginosa* pan genome analysis workflow.** The figure visualizes the steps and software used to analyze the pan genome of the 20 *P. aeruginosa* strains of this study. It starts with the input of raw sequencing reads in fastq format and then follows two routes: 1. *de novo* assembly and subsequent pan genome analysis of the assembled and annotated genome sequences (right path), 2. Mapping of the sequencing reads onto the genome sequence of the reference strain PAO1 followed by subsequent analyses (left path).

### 6.2.1. Read Quality Assessment

A variety of tools is available for different tasks related to read quality assessment (see Section 2.2). In this work, the tools were chosen on the basis of the required tasks. Thus, the most appropriate and easy to install tools for each task were used.

All data sets presented in this work were at first statistically and visually analyzed with FastQC (Andrews, 2010). This analysis revealed that 10 of the 20 data sets had a strong GC-bias. Here, regions with a GC-content above ~75% were strongly underrepresented (see Figure 70 in Appendix for an example). During closer inspection of the affected data sets, it became clear that these 10 genomes had to be sequenced a second time to include also high-GC regions of the genomes. The sequencing procedure was the same as mentioned in the introduction to this section and yielded data sets of much better quality including high-GC regions. This incident is a textbook example proving the importance of read quality assessment.

The read quality check further revealed that sequencing adapters and primers (see Appendix, Section 10.3 for the identified sequences) were still present in some of the data sets and had to be removed prior to further processing and that the base quality of most reads dropped below a PHRED score (see Section 2.1) of 20 towards the read ends. The latter is a common issue with the Illumina sequencing technology (see Section 2.1.3).

The adapter and primer sequences were removed with the FASTX-toolkit (Gordon, 2009) (see Section 2.2 and Appendix, Section 10.4 for program calls). Adapter and primer trimming was performed for reads containing a contiguous alignment of at least 6 bases to the adapter/primer and all resulting reads shorter than 25 bases were discarded.

Since the requirements of assembly and read mapping tools were different, two trimming strategies were applied next.

For the assembly path (see Figure 57), reads were trimmed by base quality using a trimming workflow provided by the Conveyor workflow engine (Linke *et al.*, 2011). All read ends were trimmed from the position on where a stretch of at least 3 bases had a PHRED quality value below 20 (see Appendix, Section 10.4 for program calls).

For the read mapping path (see Figure 57), the reads were trimmed from 101 to 80bp with the FASTX-toolkit (Gordon, 2009) (see Section 2.2 and the Appendix, Section 10.4 for program calls) to yield high quality data sets. The more detailed trimming approach used for the assembly path cannot be used in conjunction with the read mapper SARUMAN (Blom *et al.*, 2011) (see Section 2.6.4). SARUMAN requires all reads to have the same length which is not given by this approach. Nonetheless, SARUMAN had already been chosen for read mapping, because it does not use a heuristic approach as most other widely used mapping tools. Instead, it performs an exact and complete mapping of the reads in competitive time, which is especially beneficial to identify reads mapping to multiple regions of the reference genome.

After the quality assessment three data sets were available for each *P. aeruginosa* strain: The original untrimmed data set, a data set trimmed by PHRED quality score 20 and a data set with reads trimmed to 80bp length. Processing and analysis were continued in parallel with these data sets where possible to evaluate the impact of filtering and trimming (see Figure 57).

### 6.2.2. Genome Assemblies

Assemblers considered for this work were Newbler (Margulies *et al.*, 2005), Velvet (Zerbino and Birney, 2008) and ABySS (Simpson *et al.*, 2009). ABySS had to be excluded very early in the comparison, because it was not possible to correctly install it on our system due to irresolvable program dependencies. When comparing the results of test assemblies of Newbler and Velvet generated with the C40A and F469 paired-end data sets (for Velvet results see Table S5 from the Appendix, Table 17), it became evident that the Newbler results by far outperformed the Velvet assemblies. This finding has been confirmed lately. Newbler has been shown to generate reliable results in one of the latest assembly comparison studies (Bradnam *et al.*, 2013).

The final *P. aeruginosa de novo* draft assemblies were generated with Newbler version 2.8. To produce the best possible assembly and reveal the impact of quality trimming, three different assembly strategies were tested:

1. Assembling the raw and untrimmed reads.
2. Assembling the raw and untrimmed reads utilizing the internal adapter and quality trimming of Newbler.
3. Assembling the reads trimmed by PHRED quality score 20 (see Section 6.2.1).

As expected, the result with the untrimmed reads was the worst among the three assemblies. Interestingly, the internal trimming of Newbler resulted in significantly better results than using the external base quality trimming approach (for details of the comparison see Table S6 from the Appendix, Table 17). On this account, the genome assemblies generated with the internal trimming by Newbler (see Table 13) were used for all downstream processing and analyses. Besides the trimming flag, the minimum contig length to output was set to 150 bases (see Appendix, Section 10.4 for the applied Newbler command). The quality of all assemblies is relatively similar, except for strain EC2A. This data set posed unknown problems to the assembly and contains more than twice as many contigs than the other assemblies. However, the Velvet assembly of this strain was still worse in comparison. To test the impact of the updates in the recently released Newbler version 3.0, new assemblies have been generated for EC2A using this version. The best of the new assemblies (see Table 13) was still not as good as for the other strains, but clearly superior to the previous assembly. Hence, all downstream analyses involving the draft genome sequence of this strain have been recalculated using the new draft genome sequence. On this account, the resulting numbers slightly vary in comparison to the publication (Hilker *et al.*, 2014).

In general, the genome size is between 6.3 and 7.05mb with an average of 6.7mb. This is a slightly larger size than the 6.2-6.8mb with an average of 6.5mb of the other finished genome sequences available in the Pseudomonas Genome Database (Winsor *et al.*, 2011). Except for strain EC2A, the N50 value of contigs > 500bp ranges from 31.1 to 157.3kb with an average of 57.8kb, the number of scaffolds ranges from 53 to 127 with an average of 89 and the N50 value of the scaffolds ranges from 105.1 to 291.4kb with an average of 174.3kb.

Clone C has been treated separately, as for this strain three sequencing data sets were available: A 300bp Illumina paired-end data set, a 3kb Illumina mate pair data set and a 454 single end data set. In this case, the 454 data set was first assembled with Newbler using the same parameters. Afterwards, the resulting clone C assembly was combined with the two other data sets using SSPACE (Boetzer *et al.*, 2010). SSPACE was chosen because it also extends contigs by yet unmapped reads where possible and is one of the most reliable programs for this task (Hunt *et al.*, 2014). Application of SSPACE led to a significantly improved assembly result (see Table 12): Before the combination, the assembly still had 109 contigs. Afterwards, only 29 scaffolds with 74 gaps remained.

**Table 12: Improvement of the clone C assembly by data set combination.** The first line lists the characteristics of the Newbler assembly only including the 454 data set. The numbers given here refer to contigs. The second line lists the characteristics of the SSPACE (Boetzer *et al.*, 2010) assembly combining the 454 assembly with the two other data sets. The numbers given here refer to scaffolds.

| Assembly | # contigs / scaffolds | Sum (bp) | Max size | Min size | Average size | N50 |
|---|---|---|---|---|---|---|
| Newbler with only 454 (contigs) | 109 | 6 872 252 | 495 312 | 150 | 63 048 | 198 672 |
| SSPACE combination (scaffolds) | 29 | 6 889 368 | 2 950 570 | 118 | 237 564 | 2 515 587 |

The resulting scaffolds of all assemblies were ordered according to the reference strain PAO1 (Stover *et al.*, 2000) using the software r2cat (Husemann and Stoye, 2010) (see Figure 57 and Section 2.3.3). All scaffolds, which could not be mapped on PAO1, were appended at the end of the scaffolds fasta file.

To prepare manual finishing of the clone C assembly, the size of the 74 gaps was estimated according to the gaps seen in the alignment of neighboring scaffolds to the PAO1 reference sequence: 18 had a length of 1 - 2bp, 29 were smaller than 220bp, 7 were 220 - 1000bp long and 20 were larger than 1000bp. All these gaps have been successfully closed by my cooperation partners at the MH Hanover (Fischer *et al.*, in preparation). The smallest gaps could already be closed *in silico*, while ~40 gaps had to be closed manually in the laboratory.

**Table 13: Genome assembly characteristics of the sequenced *P. aeruginosa* strains except C40A.** Clone C has been left out since the assembly strategy was different for this strain. For assembly results of clone C see Table 12.

| Strain | # Reads | # Assembled Reads | Genome size (bp) | Average Alignment Depth | # Contigs Total | # Contigs > 500bp | N50 Contigs > 500bp | # Scaffolds | # Scaffold Contigs | N50 Scaffolds | [%] GC Content |
|---|---|---|---|---|---|---|---|---|---|---|---|
| D421 | 21 555 950 | 18 424 853 | 6 902 893 | 249 | 836 | 292 | 64 490 | 79 | 247 | 147 505 | 65.37 |
| F469 | 50 622 084 | 35 457 943 | 6 910 555 | 442 | 478 | 423 | 32 053 | 102 | 401 | 156 565 | 64.12 |
| OC2E | 23 613 860 | 21 081 757 | 6 914 674 | 285 | 303 | 162 | 96 592 | 75 | 142 | 193 120 | 65.83 |
| E429 | 15 001 126 | 13 722 786 | 7 039 190 | 182 | 441 | 226 | 89 852 | 124 | 190 | 105 063 | 65.46 |
| 239A | 21 737 820 | 20 106 731 | 6 915 596 | 272 | 506 | 340 | 46 617 | 127 | 299 | 122 854 | 65.32 |
| 2C22 | 58 743 200 | 48 970 069 | 6 519 939 | 644 | 426 | 352 | 34 499 | 71 | 336 | 195 426 | 65.37 |
| F429 | 19 073 646 | 16 583 740 | 6 801 202 | 228 | 359 | 210 | 79 653 | 105 | 175 | 113 680 | 65.91 |
| B420 | 11 840 794 | 9 557 706 | 6 368 472 | 140 | 286 | 170 | 83 786 | 97 | 154 | 152 318 | 66.01 |
| EA0A | 17 618 710 | 15 267 462 | 6 629 320 | 215 | 359 | 189 | 91 761 | 88 | 153 | 197 743 | 65.92 |
| 0812 | 16 187 298 | 14 576 315 | 6 652 994 | 205 | 265 | 139 | 157 306 | 68 | 119 | 227 478 | 65.92 |
| 2C1A | 16 922 710 | 15 598 163 | 6 425 255 | 227 | 402 | 323 | 38 208 | 69 | 310 | 188 951 | 65.24 |
| 1BAE | 30 392 666 | 24 634 302 | 6 798 656 | 288 | 678 | 404 | 31 453 | 77 | 388 | 195 431 | 65.01 |
| 3C2A | 35 419 090 | 28 737 477 | 6 774 714 | 338 | 649 | 402 | 34 502 | 95 | 387 | 148 219 | 65.05 |
| EC2A | 28 832 664 | 2 8229 896 | 7 044 851 | 270 | 1 203 | 741 | 17 514 | 585 | 585 | 17 922 | 65.73 |
| EC21 | 46 799 478 | 37 632 112 | 6 652 626 | 451 | 619 | 390 | 34 187 | 90 | 374 | 172 679 | 65.23 |
| 843A | 46 633 194 | 37 454 053 | 6 351 657 | 469 | 561 | 336 | 31 129 | 76 | 324 | 187 394 | 65.67 |
| 0822 | 40 049 592 | 32 784 497 | 7 037 971 | 375 | 471 | 397 | 32 219 | 93 | 374 | 144 373 | 64.69 |
| 149A | 35 606 102 | 29 020 165 | 7 007 315 | 333 | 698 | 493 | 31 303 | 97 | 431 | 195 945 | 64.62 |
| 478A | 40 849 968 | 32 582 423 | 6 370 109 | 406 | 378 | 337 | 31 678 | 53 | 334 | 291 401 | 65.45 |

### 6.2.3. Annotation

When the clone C genome sequence had been finished, it was ready for genome annotation. However, the other 19 draft genomes had to be prepared for the annotation first. The ordered scaffolds were concatenated by a stop linker on all six reading frames (`CTAGCTAGCTAG`) with a script written in-house by Dr. Jochen Blom (see Appendix, Section 10.4 for the script call). The advantage of the stop linker over other linker sequences is that it prevents wrong prediction of genes across contig borders.
The resulting fasta files only contain a single draft genome sequence and were automatically annotated with the GenDB annotation platform version 2.2 (Meyer *et al.*, 2003) using the standard pipeline based on gene prediction with Prodigal (Hyatt *et al.*, 2010) (see Appendix, Section 10.4 for the used GenDB commands). The corresponding GenDB project is "GenDB_Paepan" and is still hosted on the GenDB server of Bielefeld University[30] with restricted access. To use the annotated draft genomes for subsequent comparative genomics analyses, they were all exported as GenBank formatted files from GenDB. The number of annotated genes per strain ranges from 5,892 - 6,904 with a mean of 6,358 genes (see Table 14). This observed mean is a bit higher than that of the genomes already published in the NCBI database. The available *P. aeruginosa* genomes contain 6,178 genes on average[31].

**Table 14: Genome annotation results of all 20 analyzed *P. aeruginosa* strains.** The table lists the number of genes predicted for each genome by the automatic annotation platform GenDB (Meyer *et al.*, 2003).

| Clone | # predicted genes |
|---|---|
| C40A | 6 583 |
| D421 | 6 412 |
| F469 | 6 566 |
| 0C2E | 6 529 |
| E429 | 6 585 |
| 239A | 6 575 |
| 2C22 | 6 135 |
| F429 | 6 397 |
| B420 | 5 892 |
| EA0A | 6 213 |
| 0812 | 6 157 |
| 2C1A | 5 959 |
| 1BAE | 6 381 |
| 3C2A | 6 386 |
| EC2A | 6 904 |
| EC21 | 6 285 |
| 843A | 5 931 |
| 0822 | 6 715 |
| 149A | 6 536 |
| 478A | 6 009 |

[30] https://gendb.cebitec.uni-bielefeld.de
[31] http://www.ncbi.nlm.nih.gov/genome/genomes/187

### 6.2.4. Read Mapping and Analysis

The paired-end reads trimmed to 80bp were used to generate read alignments matching the reference genome PAO1 (NC_002516.2) for each strain. As read mapping tool, the exact and complete read mapper SARUMAN version 1.0.7 (Blom *et al.*, 2011) was used with a maximum of 8% mismatches per read and standard Levenshtein distance (see Appendix, Section 10.4 for the applied command). Then the ReadXplorer (Hilker *et al.*, 2014) database "*P.aeruginosa*-on-PAO1.h2.db" was created and the annotated PAO1 genome (in GenBank format) was imported into that database. The JOK mapping files obtained from SARUMAN were imported into this database as read pair data sets with ReadXplorer (fragment size set to 300bp and an allowed deviation of 80%). Especially for the JOK format, a Java converter module (see Section 5.2.2, Data Import) was written and integrated into the official ReadXplorer version to support conversion of JOK files into BAM formatted alignment files (H. Li *et al.*, 2009). This module was used to write the extended BAM files containing the ReadXplorer read mapping classification (see Section 5.2.2, Read Mapping Classification and Read Pair Classification) for all sequenced *P. aeruginosa* strains. The resulting number of read mappings and the amount of mappings in each ReadXplorer read mapping classification are shown in Table 15.

Next, SNPs and DIPs were computed with the SNP and DIP detection functionality of ReadXplorer (see Section 5.3.1). Each examined reference position had to be covered by at least 30 reads of which at least 90% had to show the same polymorphism. The read mapping classes included in the SNP detection were the four classes Single Perfect, Single Best, Perfect and Best Match (see Section 5.2.2, Read Mapping Classification). All Common Match mappings were discarded as non-reliable. The results of the SNP and DIP detection are explicated in Section 6.4.

To analyze the presence or absence of known genomic islands and RGPs, the mapping tool BWA (Li and Durbin, 2009) (see Section 2.6.4) was chosen in conjunction with the paired-end reads trimmed to PHRED quality score 20 (see Section 6.2.1). Apart from the default parameters, BWA was run with a maximum of 2 allowed gap extensions and a maximum of 5 mappings per read (see the Appendix, Section 10.4 for program calls) to map the reads on the mobile element sequences.

**Table 15: Read Mapping Data Overview.** The table presents the number of read mappings in total and in each read mapping class available within ReadXplorer for all 20 *P. aeruginosa* strains. The percentage columns list the coverage of the reference genome PAO1 in the different mapping classes. The last two columns show the number of Read Pairs and Single Mappings as classified by ReadXplorer. For further details on the Read Pair statistics of these data sets see Table S4 from the Appendix, Table 17.

| Strain | Mappings | Single Perfect Match | Perfect Match | Single Best Match | Best Match | Common Match | Total Covered Bases | Single Perfect Match Coverage | Perfect Match Coverage | Single Best Match Coverage | Best Match Coverage | Common Match Coverage | Read Pairs | Single Mappings |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C40A | 37 880 131 | 13 288 989 | 1 076 968 | 22 345 382 | 771 622 | 397 170 | 95.56% | 81.48% | 4.86% | 94.47% | 13.12% | 1.45% | 16 555 645 | 4 604 427 |
| D421 | 15 997 937 | 4 205 150 | 226 676 | 11 066 542 | 341 346 | 158 223 | 96,52% | 74,27% | 3,42% | 95,56% | 6,67% | 1,70% | 7 389 251 | 864 978 |
| F469 | 24 484 716 | 11 410 186 | 751 258 | 11 669 287 | 305 299 | 348 686 | 95.41% | 81.79% | 1.25% | 95.30% | 1.70% | 2.59% | 11 601 176 | 50 628 |
| OC2E | 19 167 378 | 7 561 794 | 260 235 | 10 861 332 | 321 632 | 162 385 | 97,25% | 87,58% | 3,24% | 96,21% | 5,23% | 1,65% | 9 041 735 | 813 136 |
| E429 | 11 999 891 | 3 165 472 | 183 257 | 8 252 827 | 267 025 | 131 310 | 96.43% | 74.03% | 3.15% | 95.43% | 6.26% | 1.64% | 5 605 980 | 654 222 |
| 239A | 18 021 718 | 6 938 646 | 320 167 | 10 120 875 | 447 158 | 194 872 | 97.12% | 87.21% | 3.47% | 96.10% | 5.39% | 1.72% | 8 442 394 | 826 386 |
| 2C22 | 37 047 772 | 23 166 865 | 1 069 960 | 11 901 112 | 354 117 | 555 718 | 96.73% | 90.02% | 1.29% | 95.67% | 1.66% | 2.80% | 18 138 627 | 501 274 |
| F429 | 15 022 231 | 4 118 238 | 221 584 | 10 245 349 | 303 882 | 133 178 | 95.98% | 73.78% | 3.34% | 95.00% | 7.03% | 1.50% | 6 921 597 | 842 898 |
| B420 | 8 778 724 | 1 257 243 | 113 268 | 7 122 971 | 215 189 | 70 053 | 93.29% | 45.51% | 2.34% | 92.28% | 7.61% | 1.34% | 3 809 418 | 776 912 |
| EA0A | 14 537 456 | 5 757 284 | 214 354 | 8 165 008 | 257 899 | 142 911 | 96.92% | 87.25% | 3.46% | 95.92% | 5.86% | 1.68% | 6 704 230 | 755 849 |
| 0812 | 13 578 108 | 5 522 655 | 208 698 | 7 487 193 | 233 140 | 126 422 | 98.50% | 90.02% | 3.27% | 97.60% | 4.85% | 1.71% | 6 375 267 | 567 331 |
| 2C1A | 15 624 607 | 6 256 280 | 244 625 | 8 710 813 | 261 375 | 151 514 | 96.99% | 86.89% | 3.28% | 96.04% | 5.46% | 1.59% | 7 354 611 | 747 160 |
| 1BAE | 16 241 253 | 9 803 978 | 488 081 | 5 505 429 | 189 575 | 254 190 | 96.55% | 89.57% | 1.34% | 95.49% | 1.70% | 2.68% | 7 960 935 | 257 938 |
| 3C2A | 18 606 086 | 11 176 196 | 614 743 | 6 288 450 | 227 058 | 299 639 | 95.43% | 88.49% | 1.36% | 94.36% | 1.70% | 2.69% | 9 108 553 | 321 376 |
| EC2A | 22 355 197 | 13 094 241 | 787 813 | 7 840 372 | 286 614 | 346 157 | 96.83% | 89.88% | 1.49% | 95.68% | 1.86% | 2.80% | 10 899 564 | 469 670 |
| EC21 | 26 126 649 | 11 965 799 | 821 278 | 12 559 793 | 338 482 | 441 297 | 97.88% | 84.31% | 1.44% | 96.82% | 1.90% | 2.94% | 12 763 476 | 493 119 |
| 843A | 26 973 987 | 15 858 613 | 780 418 | 9 573 824 | 348 775 | 412 357 | 96.85% | 89.89% | 1.31% | 95.60% | 1.69% | 2.77% | 13 215 376 | 427 099 |
| 0822 | 20 581 839 | 12 262 556 | 639 214 | 7 077 732 | 261 754 | 340 583 | 96,87% | 89,90% | 1,32% | 95,84% | 1,69% | 2,74% | 10 083 475 | 342 352 |
| 149A | 17 776 695 | 10 493 885 | 543 798 | 6 205 930 | 240 169 | 292 913 | 96,59% | 89,17% | 1,42% | 95,49% | 1,78% | 2,79% | 8 714 117 | 299 540 |
| 478A | 22 863 267 | 13 694 605 | 692 174 | 7 896 835 | 252 024 | 327 629 | 96,30% | 89,63% | 1,28% | 95,19% | 1,66% | 2,70% | 11 209 930 | 349 526 |

# 6.3. Comparative Genomics using EDGAR

The GenBank files of the draft genomes and the reference strain PAO1 were used to create an EDGAR version 1.2 (Blom *et al.*, 2009) (for a description of the EDGAR functionality see Section 2.5.3) project for the pan genome and core genome analysis. This project is named "EDGAR_Paeru_MHH" and available with restricted access on the EDGAR server[32].
The score ratio value (SRV) (Lerat *et al.*, 2003) used as master cut-off for this project is 30%. This means that only reciprocal BLASTP hits for coding sequences with an SRV higher than or equal to 30% are marked as being present in two compared genomes. All further calculations are based on this parameter choice[33].
The core and pan genome of all 20 sequenced strains and the additional reference strain PAO1 have been calculated iteratively. The prevalent clone C was chosen as reference to start with. Then all other genomes were added to the calculations one after another. The complete core genome consists of 4,787 genes and the pan genome contains 13,277 genes (see Figure 58). The actual lists of core and pan genome genes can be found in the supplementary data files listed in the Appendix, Table 17 as Table S1 and S2. The theoretical development of the core and pan genomes as predicted by EDGAR is shown in Figure 59 and Figure 60, respectively.

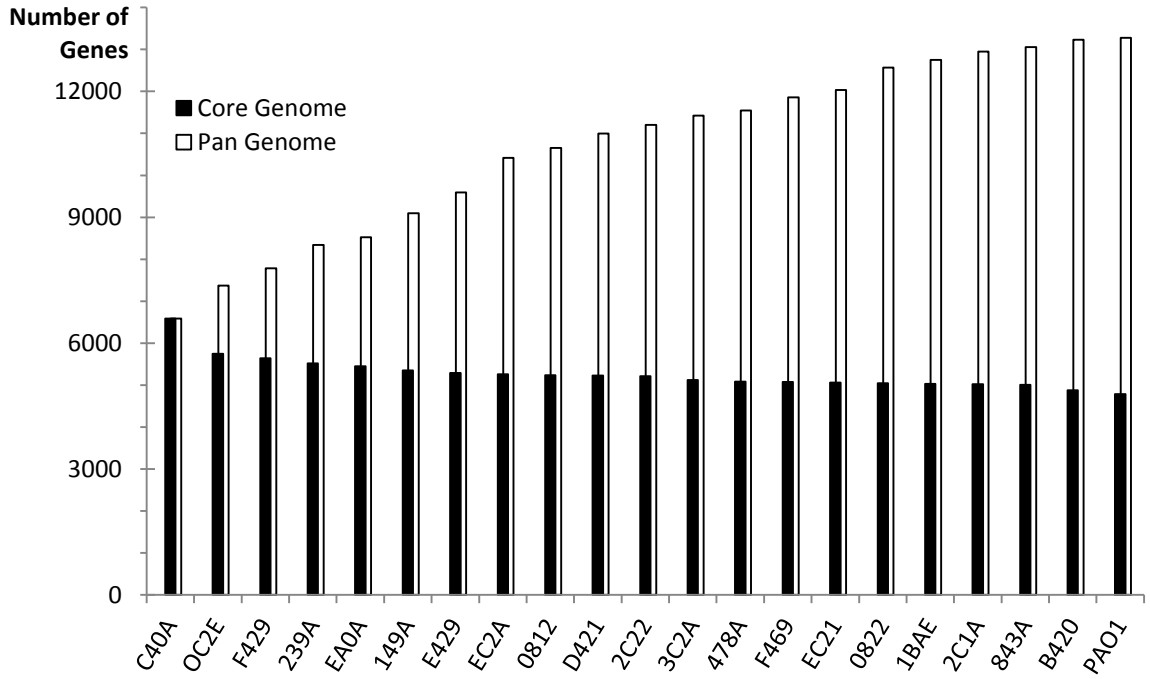



**Figure 58: The pan genome of the 20 sequenced strains representing the most common clonal complexes in the *P. aeruginosa* population.** *P. aeruginosa* PAO1 was included as the reference strain. The coding genetic repertoire of the core genome and of the pan genome was constructed as follows: starting from the most frequent clone C (hexadecimal code C40A), core and pan genome were stepwise constructed by the addition of genes not present in the predecessor (pan genome) or by the subtraction of genes absent in the successor (core genome). The numbers were obtained by the core and pan genome functions of EDGAR. Figure and description are updated versions from (Hilker *et al.*, 2014). The lists of core and pan genome genes can be found in the supplementary data files listed in the Appendix, Table 17 as Table S1 and S2.

[32] https://www.edgar.computational.bio
[33] Note that the numbers here are slightly different than in the publication (Hilker *et al.*, 2014). This is due to the fact that the assembly with the by far most contigs, the assembly of strain EC2A (see Table 13), could be improved by using the recently released version 3.0 of the Newbler assembler. This version was not yet available when the publication was submitted.

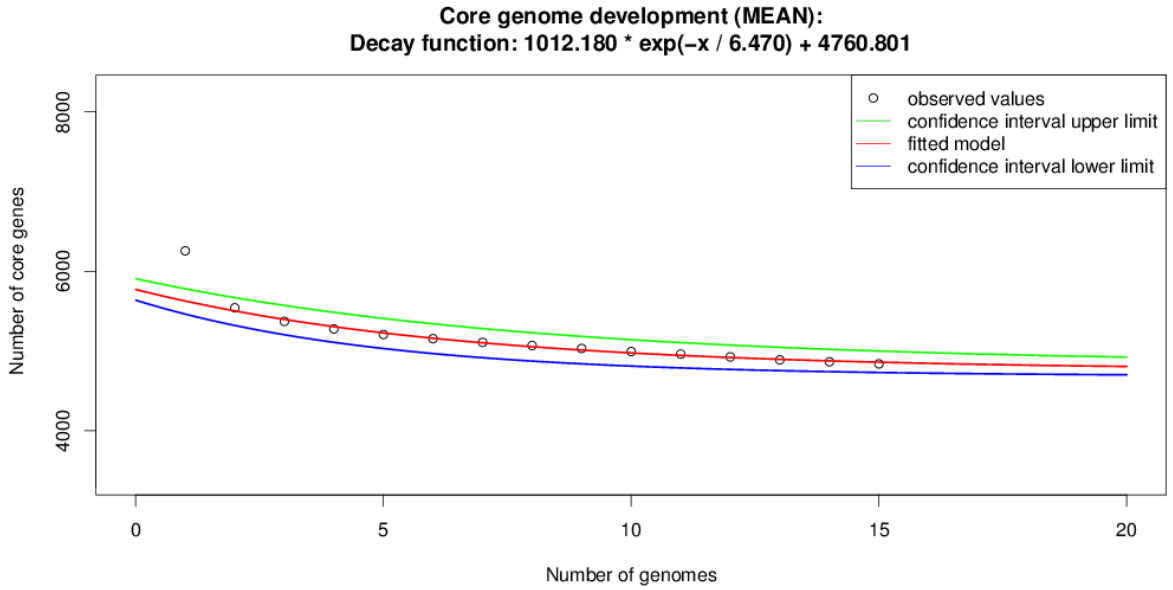



**Figure 59: *P. aeruginosa* core genome development plot.** The maximum number of 15 genomes was chosen to calculate the prediction for the core genome development with increasing genome numbers using a nonlinear least squares model fit as implemented in EDGAR (Blom *et al.*, 2009). To generate a conservative estimate, the 15 most divergent strains were chosen as input for the calculation. The plot suggests a number higher than 4000 genes for the complete *P. aeruginosa* core genome.

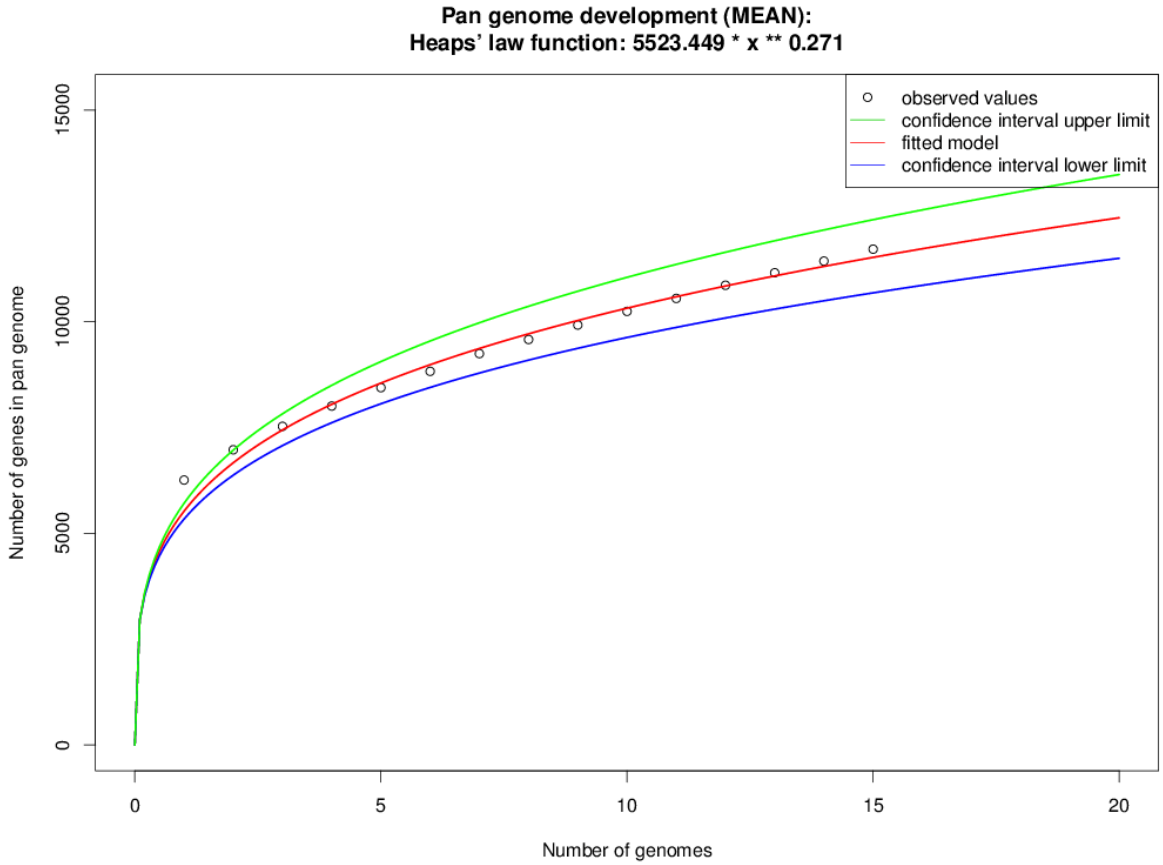



**Figure 60: *P. aeruginosa* pan genome development plot.** The maximum number of 15 genomes was chosen to calculate the prediction for the pan genome development with increasing genome numbers using a nonlinear least squares model fit as implemented in EDGAR (Blom *et al.*, 2009). To generate a conservative estimate, the 15 most divergent strains were chosen as input for the calculation. The plot suggests an open pan genome for *P. aeruginosa*.

Further, the quantity of unique genes, occurring only in one of the genomes, was evaluated with EDGAR. The result is visualized in the histogram in Figure 61 and the actual list of

unique genes per strain is available in the supplementary data file listed in the Appendix, Table 17 as Table S2.
The strains contain between 27 (PAO1) and 406 (0822) unique genes with an average of 126 genes. The major portion of these genes is organized in operons or DNA blocks of up to 312 CDS. Most genes were assigned to the categories of nucleic acid metabolism, mobile genetic elements or hypotheticals. The disease isolates additionally carried a few paralogues of housekeeping enzymes and/or elements of mobility or secretion, the most spectacular example being an additional set of pilus biogenesis genes in the intubated patient isolate E429 (see Table S2). The environmental isolates harboured larger sets of strain-specific genes than the clinical isolates, and these genes conferred numerous extra potential for bioenergetics, metabolism, transport and immunity such as a CRISPR-Cas system (Bondy-Denomy *et al.*, 2013). Recurrent features were operons for the biogenesis of heme proteins and for the transport and metabolism of amino acids and sulphur compounds.

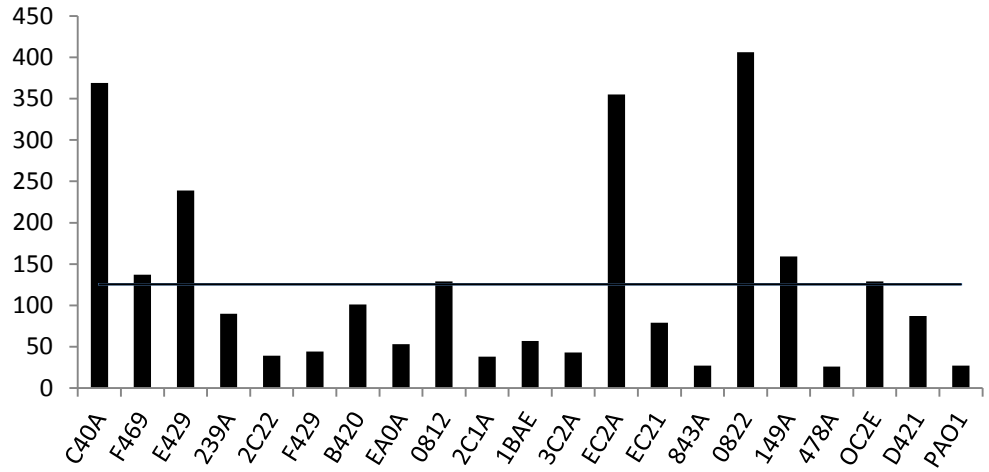



**Figure 61: Number of unique genes per strain.** All genes unique per strain were extracted with EDGAR. On average, 126 genes were unique per strain. This average is indicated by the line. The complete list of genes can be found in the supplementary data file listed in the Appendix, Table 17 as Table S2.

To measure and weigh the evolutionary distance among all the strains, a phylogenetic tree was calculated based on the core genome (see Figure 62). The tree shows that there is one outlier strain among the selected strain panel - B420. This is an innocuous and the most common strain among isolates from the inanimate environment (Selezska *et al.*, 2012). In this case, the analyzed representative was isolated from a river. All environmental strains are spread across the whole tree and do not cluster together as could be suspected. Only the two environmental strains EC2A and 843A cluster together as closest relatives.

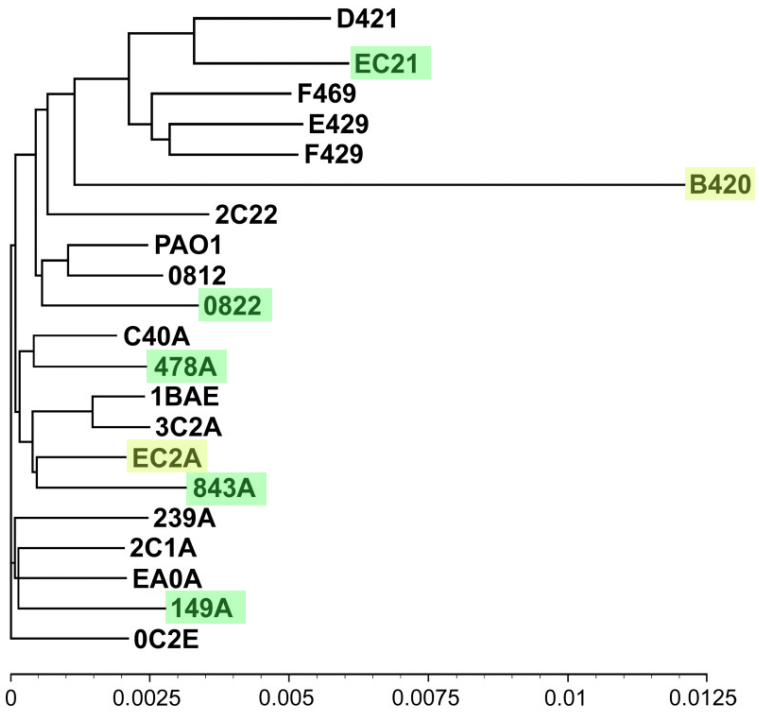



**Figure 62: Phylogenetic tree of the 20 sequenced *P. aeruginosa* strains including the PAO1 reference strain.** The tree has been generated with EDGAR (Blom *et al.*, 2009) based on the amino acid sequences of the 4787 genes from the core genome of all listed strains. The distances in the tree are based on the amino acid substitution model established by (Kimura, 1983) and the tree layout is calculated by an implementation of the neighbor-joining algorithm (Saitou and Nei, 1987). The scale indicates their sequence diversity. The 5 strains without any clinical representatives to date are highlighted in green. The two other river isolates are marked in light green. The figure and the description are adapted from (Hilker *et al.*, 2014).

The gene content of *P. aeruginosa* has been described as combinatorial (Lee *et al.*, 2006). This primarily reflects the combinatorial composition of the accessory genome with genomic islands (GIs) and insertions in RGPs (Mathee *et al.*, 2008; Klockgether *et al.*, 2011).
To review this argument, four exemplary genome sets of four strains each are shown in Figure 63. The sets have been chosen to cover the different strain groups contained in the sequenced strain panel: Environmental strains (C), pathogenic strains (D), close relatives (B) and finally the most abundant pathogenic and apathogenic strains (A). All four gene set comparisons confirm the previous findings, i.e. all possible fields of singles, dyads, triples and quad are occupied by genes. The core genome has a similar size in all four comparisons (5217-5446 genes). In Figure 63, (C), strain 0822 and 139A have the largest phylogenetic distance and also hold the largest number of unique genes (926 for 0822 and 705 for 139A) among all displayed comparisons. The outlier B420 contains an unexceptional number of unique genes with 327 genes. Even among the four closest relatives shown in Figure 63, (B) the number of unique genes is quite high and varies strongly between the four genomes (239-646 genes).

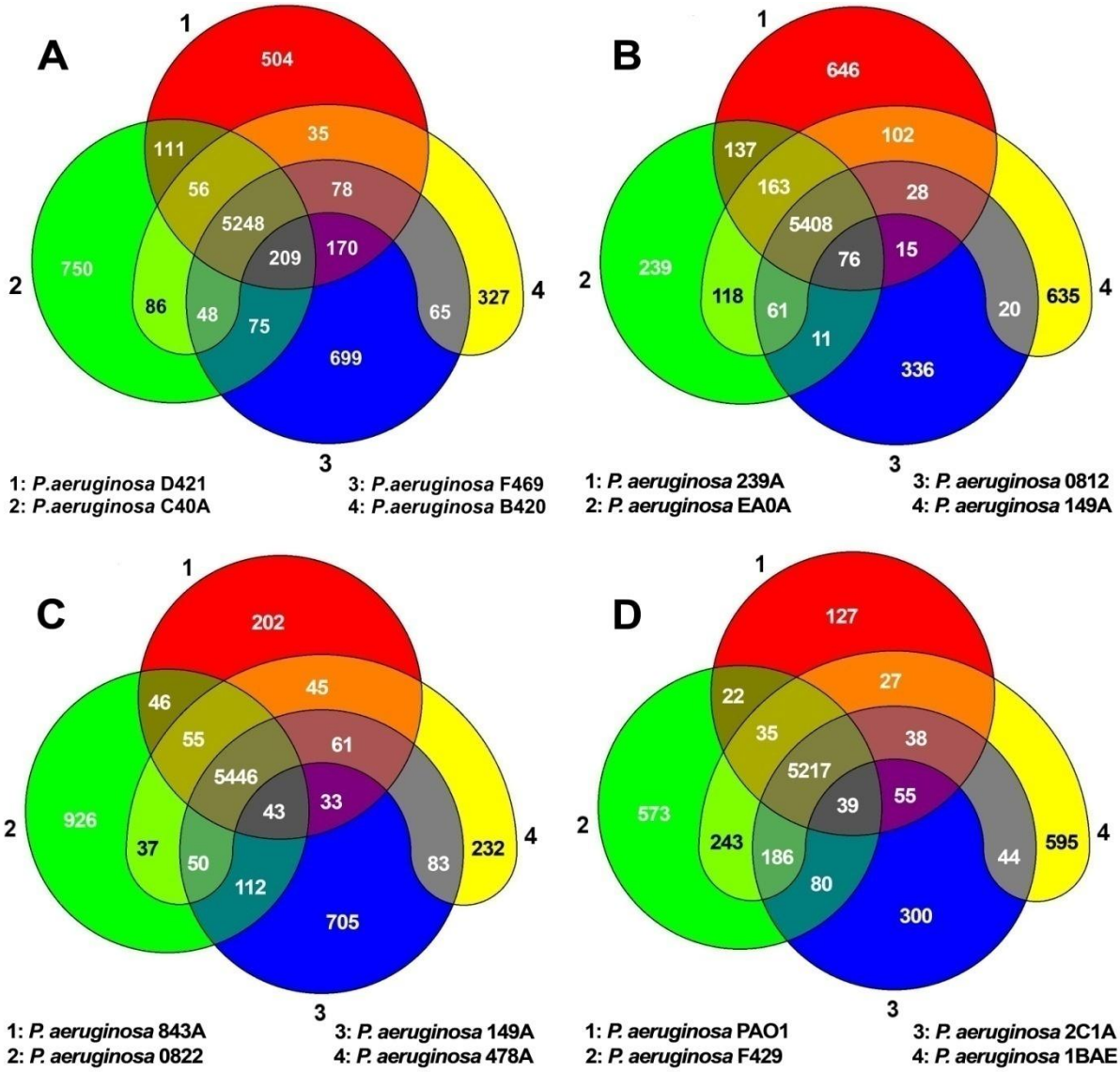



**Figure 63: Genome gene set comparisons of selected *P. aeruginosa* strains.** Venn diagrams of the number of strain-specific and shared genes each in a panel of four *P. aeruginosa* strains. (A) Shows the gene sets of the apathogenic B420 strain, the highly virulent F469 strain and the two strains belonging to the major clones C (C40A) and PA14 (D421). (B) Shows the gene sets of a set of four closest relatives from the phylogenetic tree. (C) Shows the gene sets of four environmental strains without any known infection history. (D) Shows the gene sets of four pathogenic strains. Figure A and its description are adapted from (Hilker *et al.*, 2014).

# 6.4. Comparative SNP and DIP Analysis

Mapping of the 20 genome sequences onto the *P. aeruginosa* PAO1 reference genome (see Section 6.2.4) identified sequence variation between PAO1 and the respective strain at 23,907-114,260 positions of the PAO1 genome, which corresponds to a sequence diversity at the single nucleotide level of 0.38-1.82% (see Table 16, and supplementary tables S3A and S3B). A portion of 17.2-19.97% of single nucleotide polymorphisms (SNPs) caused an amino acid substitution in CDS. Based on the criterion of single nucleotide substitution (SNP) diversity, the clonal complexes were grouped into two clusters and one outlier (B420) (see Figure 62). The larger cluster including the most abundant clone C (C40A) differs at 0.38-0.48% positions from the PAO1 genome, whereas the smaller cluster including the second most abundant clone PA14 (D421) exhibits a sequence diversity of 0.85-0.89% (see Table 16).

The two clusters do not segregate clinical from environmental isolates, as both contain members of each group of strains.

The DIP frequency within the genomes is much lower than the SNP frequency at a level of 0.0041-0.019%. Only 13.97-20.39% of these DIPs account for events within a gene. In absolute numbers the count of DIPs affecting CDS ranges from 40-216.

Another rare case of polymorphism is a stop codon mutation within a gene. In our examined strain panel, the count within each genome ranges from 7 stop mutations in strain 478A to 43 stop mutations in strain B420 accounting for 0.03-0.07% of all polymorphisms in the respective strain.

When examining the non-synonymous polymorphisms within genes, which make up between 17.2-19.97% of all polymorphisms of the respective strain, a tendency towards base exchanges resulting in chemically different amino acids becomes apparent (for the chemical amino acid classification within ReadXplorer see Section 5.3.1). Only 36.53-39.13% of them are associated to the same chemical amino acid group as the original amino acid, leaving the larger portion of 60.87-63.47% to code for a chemically different amino acid.

Most oligonucleotide insertions or deletions were in frame and caused the incorporation or loss of a single codon (see supplementary tables S3A and S3B). Single nucleotide frameshift mutations predominantly affected conserved hypotheticals. Deleterious frameshift mutations in functionally characterized genes were only detected in the 13 clinical isolates of our panel but in none of the seven environmental isolates. Recurrent loss-of-function hits were observed in gene clusters encoding the biosynthesis or regulation of flagella, pili, quinolones, the O-antigen of lipopolysaccharides, effectors of type III secretion, siderophores and their receptors and the biosynthesis of the antimetabolite L-2-amino-4-methoxy-trans-3-butenoic acid (Lee *et al.*, 2010) (Tables S3A and S3B). This spectrum of mutations is consistent with a conversion of bacterial phenotype that often occurs during human infections, i.e. loss of motility, LPS deficiency and modulation of virulence, signalling and iron homeostasis (Döring *et al.*, 2011).

**Table 16: Number of SNPs and DIPs of each of the 20 sequenced strains to the reference PAO1.** The three SNP frequency clusters are indicated by white, orange and aquamarine background color in the total number of SNPs column.

| Strain | Total number of SNPs | Substitutions | Insertions | Deletions | Intergenic SNPs | Synonymous SNPs | Chemically neutral SNPs | Chemically different SNPs | Insertions in Gene | Deletions in Gene | Stop Mutations |
|---|---|---|---|---|---|---|---|---|---|---|---|
| C40A | 24 856 | 24 504 | 147 | 205 | 3 983 | 16 292 | 1 711 | 2 803 | 25 | 42 | 10 |
| D421 | 55 854 | 55 171 | 331 | 352 | 8 066 | 37 995 | 3 715 | 5 957 | 48 | 73 | 20 |
| F469 | 53 456 | 52 869 | 263 | 324 | 7 807 | 36 252 | 3 600 | 5 715 | 30 | 52 | 20 |
| OC2E | 27 657 | 27 246 | 183 | 228 | 4 081 | 18 076 | 2 036 | 3 387 | 37 | 40 | 9 |
| E429 | 55 122 | 54 469 | 303 | 350 | 7 923 | 37 417 | 3 732 | 5 939 | 50 | 61 | 20 |
| 239A | 27 879 | 27 521 | 185 | 173 | 3 883 | 18 492 | 2 024 | 3 407 | 43 | 30 | 14 |
| 2C22 | 27 513 | 27 196 | 148 | 186 | 4290 | 17878 | 1964 | 3338 | 26 | 34 | 13 |
| F429 | 55 281 | 54 602 | 331 | 348 | 7 930 | 37 559 | 3 745 | 5 938 | 43 | 66 | 25 |
| B420 | 114 260 | 113 067 | 518 | 675 | 13 876 | 80 516 | 7 690 | 11 962 | 83 | 133 | 43 |
| EA0A | 27 373 | 26 991 | 179 | 203 | 4 078 | 17 837 | 2 001 | 3 380 | 35 | 42 | 12 |
| 0812 | 23 907 | 23 578 | 167 | 162 | 3 408 | 15 668 | 1 740 | 3 034 | 30 | 27 | 8 |
| 2C1A | 28 332 | 27 932 | 174 | 226 | 4 079 | 18 685 | 2 061 | 3 441 | 31 | 35 | 9 |
| 1BAE | 26 547 | 26 285 | 100 | 162 | 3 854 | 17 475 | 1 933 | 3 245 | 15 | 25 | 13 |
| 3C2A | 26 637 | 26 369 | 107 | 161 | 3 863 | 17 553 | 1 938 | 3 241 | 16 | 26 | 19 |
| EC2A | 27 043 | 26 760 | 101 | 182 | 4 002 | 17 779 | 1 935 | 3 285 | 17 | 25 | 11 |
| EC21 | 52 974 | 52 506 | 210 | 258 | 7 595 | 36 155 | 3 570 | 5 578 | 30 | 46 | 16 |
| 843A | 29 870 | 29 555 | 123 | 192 | 4 306 | 19 900 | 2 088 | 3 517 | 23 | 36 | 10 |
| 0822 | 27 581 | 27 306 | 110 | 165 | 4 006 | 18 204 | 1 944 | 3 377 | 19 | 31 | 11 |
| 149A | 28 453 | 28 153 | 130 | 170 | 4 208 | 18 845 | 2 033 | 3 318 | 18 | 31 | 13 |
| 478A | 27 260 | 26 980 | 118 | 162 | 3 881 | 18 258 | 1 893 | 3 177 | 18 | 33 | 7 |

## 6.5. Comparative Analysis of Shared Genomic Islands and Regions of Genomic Plasticity

Lateral gene transfer and recombination shape the dynamics of bacterial genomes. An extensive list of known mobile genomic elements from *P. aeruginosa* (GIs as well as RGPs) can be found in the publication of Klockgether *et al.* (2011). The combinatorial repertoire of GIs (see Figure 64) and RGPs (see Figure 65) of the 20 strains indicates a continuous horizontal gene flow between the clonal complexes of *P. aeruginosa*. Numerous but not all strains, for example, shared complete copies of the LESprophage 1 and LESGI-4 (Winstanley *et al.*, 2009) and harboured similar but not identical members of the PAGI-2 and pKLC102 island families (Klockgether *et al.*, 2007). The RGPs identified in PA14 are all completely preserved in the PA14 representative D421. A few other islands are not present in any of the strains: PAGI-3, PAGI-11, except for a complete copy in strain EA0A, LESGI-5, LES-prophage 4, except for a minor preserved part in the strains OC2E and D421, a few RGPs like C3719-RGP8, C3719-RGP16, C3719-RGP28, C3719-RGP52 and PACS2-RGP62 and most of the RGPs found in PA7 - only very few strains contain a few of the PA7 RGPs with high identity. In fact, PA7-RGP73 is the only PA7-RGP present with high identity in several of the 20 sequenced strains (in 11).

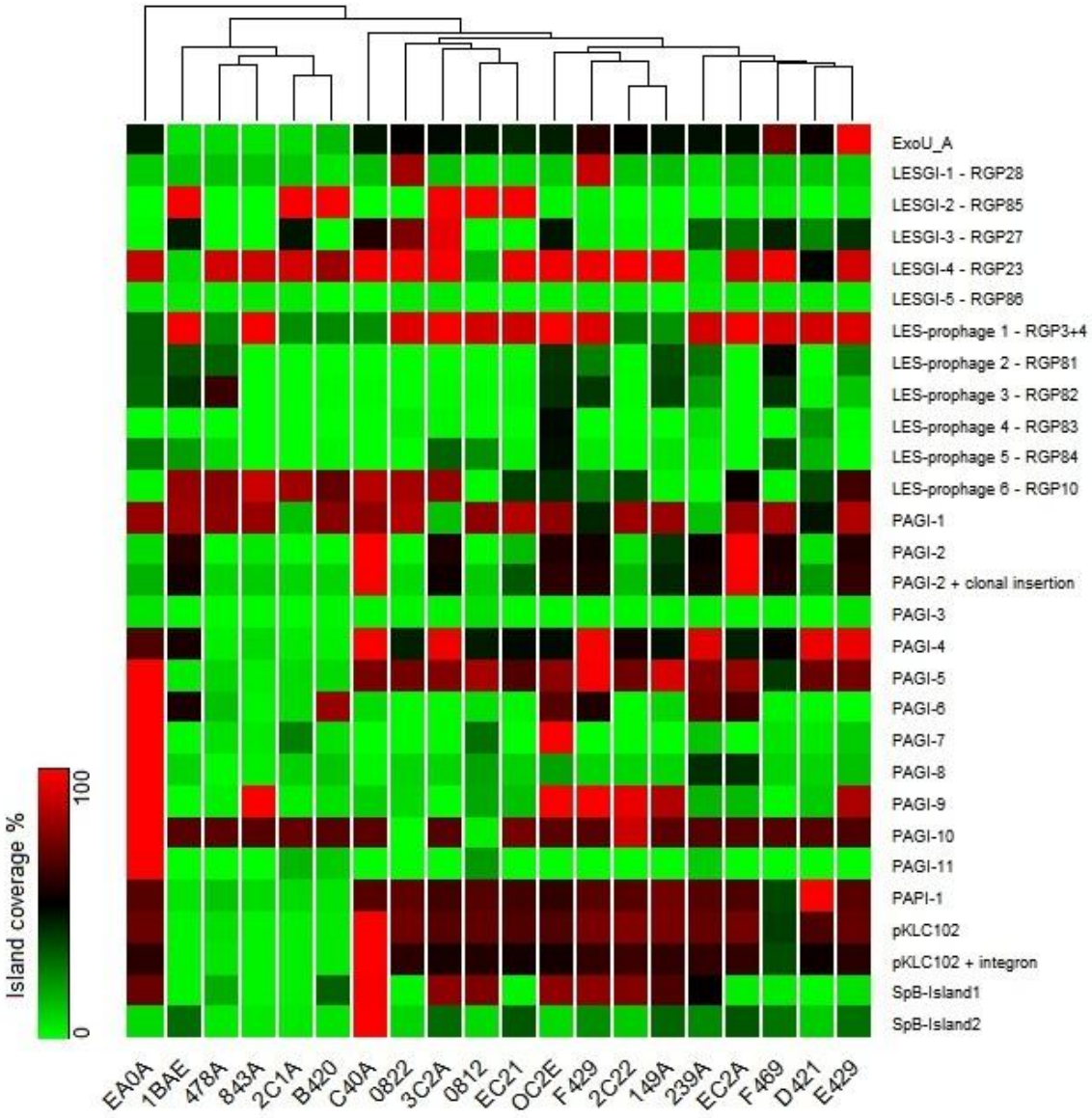



**Figure 64: Genomic island coverage heatmap.** The variable presence and conservation of each of the 29 known genomic islands among all 20 analyzed *P. aeruginosa* strains is visualized by this heatmap. None of the genomic islands is contained in all strains as complete copy. The degree of their presence has been determined by read mapping on the respective island sequence and automatic coverage analysis with ReadXplorer (Hilker *et al.*, 2014). Absent islands are depicted in green, while highly conserved islands are shown in red. The dark green and red colors in between depict islands only partly conserved in the respective strain.

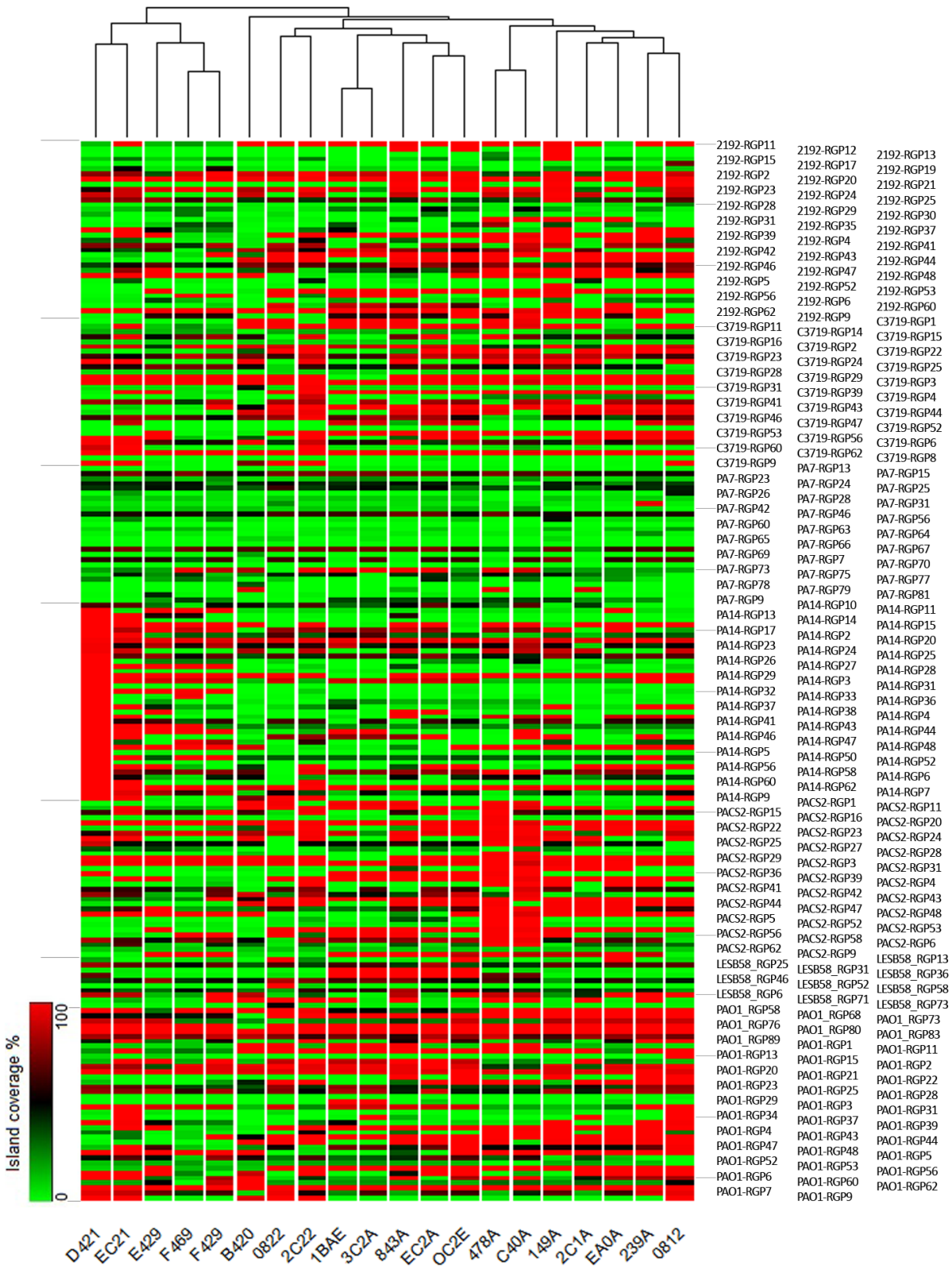



**Figure 65: RGP coverage heatmap.** The variable presence and conservation of each of the 161 known RGPs among all 20 analyzed *P. aeruginosa* strains is visualized by this heatmap. The grey lines on the left side indicate the RGP strain borders, while the grey lines on the right side connect each fifth RGP legend entry of the first column to the heatmap for guidance. Only 8 of the genomic islands are contained in all strains as complete copy. The applied procedure and color code is the same as for Figure 64.

## 6.6. Conclusion

The conducted *P. aeruginosa* pan genome analyses (see Sections 6.3 - 6.5) provide comprehensive results elucidating the genomic content of the analyzed strain panel and the phylogenetic relationships of the strains. The earlier ascertained combinatorial composition of the accessory genome (Lee *et al.*, 2006) is also supported by all of these analyses. Further, the EDGAR and SNP results complement one another with regard to the segregation of the strains into two clusters and one outlier. These clusters do not segregate clinical from environmental strains. When considering the composition of GIs and RGPs, the picture is more diverse and indicates a continuous horizontal gene flow. This observation is in concordance with the observations from (Mathee *et al.*, 2008; Klockgether *et al.*, 2011).

# Chapter 7

# Discussion and Outlook

This chapter discusses the work described in the previous two chapters and pictures the future perspective in both subject areas treated in this dissertation.

## 7.1. A Novel Platform for Read Mapping Visualization and Analysis

The implementation of the comprehensive open source Java desktop client application ReadXplorer (Hilker *et al.*, 2014) was a substantial contribution to conduct a state-of-the-art pan genome analysis of 20 prevalent *Pseudomonas aeruginosa* strains possible.
Certainly, the value of ReadXplorer (see Chapter 5) reaches much further. It is a substantial tool to bridge the problematic gap between visualization and automatic analysis functions for read mapping data (see Chapter 4). ReadXplorer enables visual and meaningful conjunction of reference genome sequences and annotations, NGS read mapping data and additional related tabular data.
By employing the Netbeans Rich Client Platform as foundation for ReadXplorer, the extensibility of the software for other programmers is guaranteed. It is based on the classical three-tier architecture (see Figure 38) and makes extensive use of common programming design patterns to ease the entrance for new developers.
Furthermore, the modular software composition and plug-in framework enable simple integration of additional highly specialized modules by other developers. Therefore, selected ReadXplorer features have already been implemented by students within the framework of a two weeks course without any prior knowledge of ReadXplorer.
The efficient data handling and support of various reference formats (see Section 5.2.2) enables fast and responsive handling of all kinds of genomes, ranging from viruses to prokaryotes to eukaryotic genomes.

The most important data for which ReadXplorer has especially been developed, the read mapping data, is not simply displayed, but further classified by quality and quantity measures based on the read mappings from the original mapping file fed into ReadXplorer (see Section 5.2.2, Read Mapping Classification and Read Pair Classification). This classification approach does not only cover detailed decoding of the mappings, but also of read pair data. An algorithm considering all mapped reads and classifying them into the most probable pairing configurations has been developed and implemented in ReadXplorer. The distinctiveness of this classification is constituted by keeping all read mappings available to

the researcher in contrast to ultimately discarding unwanted mappings, e.g. multiple mapped reads. This read mapping classification is available throughout the whole software in all analysis functions and corresponding colour-coded visualizations, offering a deeper insight into the characteristics of the read mapping data than the other presented tools with a comparable scope (see Chapter 3).

Besides the classification, ReadXplorer comprehensively features mapping and base quality handling. A mapping quality threshold can be set in each data viewer and during configuration of each analysis function. A base quality threshold can be set during configuration of the SNP and DIP detection and the alignment viewer takes care of the base quality visualization by shading each base of an alignment according to its base quality.
Another unique aspect of ReadXplorer is the availability of several views illuminating different aspects of the underlying data. Notable specialized viewers are the Double Track Viewer for direct comparison of two tracks, the Thumbnail Viewer for genomic feature comparisons at a glance, the Read Pair Viewer for color-coded visualization of read pair configurations and the Multiple Track Viewer for combined data analysis. Additionally, normalization of coverage has been introduced for all viewers displaying coverage.
ReadXplorer makes the genetic code user-selectable. Then, it makes global use of this property in visualizations as well as analysis functions. The Reference Viewer is capable of interactively visualizing selected start and stop codons for the currently selected genetic code. They can highlight the sequence from the start to the next in-frame stop codon and allow copying, translating and extracting the corresponding CDS. This feature e.g. facilitates instant comparison of RNA-Seq (Wang *et al.*, 2009) coverage with potential transcripts obtained from the implemented automatic analysis methods.

An automatic analysis framework has been developed to scan read mapping tracks for certain characteristics. By using this single analysis framework for all implemented analysis functions, the recognition value for researchers testing a different than their usual analysis for the first time in ReadXplorer is fueled.
The automatic analysis functions offered by ReadXplorer (see Sections 2.7 and 2.8 for the theory and Section 5.3 for the implementations) include differential gene expression analysis, detection of TSS, novel transcripts, operons as well as calculating TPM, RPKM and feature read count values especially for RNA-Seq (Wang *et al.*, 2009) data sets. Further, it contains automatic detection of SNPs and DIPs, genome rearrangements, RNA secondary structure prediction and two unique automatic analysis methods: An analysis of the coverage of reference features and an analysis of the coverage of the whole reference sequence.
In addition to the implemented analyses, ReadXplorer also features import of tabular data. SNP and DIP tables obtained from other tools can be reviewed effortlessly by using the VCF import. The tabular data import is also available for any other kind of tabular data and enables working with the data along with the other visualizations and ReadXplorer internal analysis results. Results generated with ReadXplorer itself are treated even more specific for the best possible processing of the data.
Another unique feature of ReadXplorer is the ability to combine multiple mapping data sets and treat them as one not only in the visualization, but also for each analysis.
Also comparative analysis of multiple tracks from the same reference is provided by ReadXplorer through comparative filters which can be applied to analysis result tables (see Section 5.2.2, Viewer Logic).

By providing all the different analysis functions in conjunction with the visualizations from one source, there is no need to install and handle several different tools. The manageability of complicated multi-dimensional NGS experiments is greatly simplified while all program

features are provided with solid usability and performance. Thus, the learning curve for users is shortened and tedious conversion of different output from different tools is no longer required.
Subsequent use of ReadXplorer results is easy. The results of all analysis functions can be exported in two popular tabular formats and high quality screenshots can be obtained directly from within the software.

As a matter of course, ReadXplorer has not been developed solely for the *P. aeruginosa* pan genome project. In fact, it supplies a large range of application areas dealing with genome browsing and read mapping data. ReadXplorer is available for everyone at the homepage www.readxplorer.org and thus already in use at external sites (e.g. see (Soares-Castro and Santos, 2015; Tong *et al.*, 2015)). The most important experiments conducted by myself in cooperations or by colleagues from our universities and their main results generated by ReadXplorer are explicated in the following.

The effective handling of medium sized eukaryotes has been demonstrated during an in-house collaboration for the SNP and DIP analysis of three closely related *Chlamydomonas reinhardtii* (Merchant *et al.*, 2007) strains in a resequencing experiment (Schierenbeck *et al.*, 2015).
The TSS and operon detection, as well as the differential gene expression analysis tools have effectively been applied to experiments on different *Corynebacterium glutamicum* strains (e.g. (Mentz *et al.*, 2013; Pfeifer-Sancar *et al.*, 2013)). The integrated differential gene expression toolkits delivered similar results as corresponding microarray experiments (data not shown). The differences in both assays might be induced by the higher resolution and flexibility of RNA-Seq in comparison to microarrays.
Recently, I have used ReadXplorer successfully to analyze *Haloferax volcanii* dRNA-seq data for TSS (Maier *et al.*, 2015). Currently, we are preparing a more comprehensive TSS analysis of *Haloferax volcanii* dRNA-seq data using ReadXplorer.
In the past year 44 *Listeria* strains have been sequenced in-house to analyze their pathogenicity potential from a genomics point of view. Within this framework, I have performed comparative pan genome studies utilizing the here developed pan genome analysis workflow. The SNP and DIP analyses of all these strains have been carried out with ReadXplorer, too (Hilker *et al.*, in preparation).
Furthermore, I have recently applied ReadXplorer's differential gene expression capabilities to three *Listeria* experiments each comparing two different environmental conditions (Schultze *et al.*, in preparation; Hilker *et al.*, in preparation).

# Outlook

Forthcoming features complementing the abilities of ReadXplorer in terms of user-friendly visualization and automatic analysis are explicated in the following paragraphs.
The SNP and DIP, TSS and operon detections could all be enhanced by a statistical detection model. For the SNP and DIP detection, a variety of methods including machine learning approaches are available to choose from (see Section 2.7.1). For the TSS detection three statistical models have been developed recently (Amman *et al.*, 2014; Jorjani and Zavolan, 2014; McClure *et al.*, 2013). One of these methods could be used to enhance the here presented empirical model to provide even more reliable analysis results. Similar to Rockhopper (McClure *et al.*, 2013), the TSS detection algorithm could be developed further into a full transcript recovery algorithm, predicting both transcript boundaries. The decision

for the most accurate approach should be made on the basis of an evaluation of the tools against experimentally validated TSS from different organisms. Further, an algorithm for the identification of promoter regions could be incorporated to predict transcripts in conjunction with their promoters. 17 tools for eukaryotic promoter identification have been reviewed in (Abeel *et al.*, 2009), 4 of which score best in their evaluation. One of them or a combination of these tools could be integrated into ReadXplorer. For prokaryotes 7 freely available tools are mentioned in the publication of Jong *et al.* (2012). They propose and use MEME (Bailey *et al.*, 2009) in their PePPER web-service. For integration or implementation of the most suitable algorithm, the available open-source tools can be evaluated regarding specificity, sensitivity and convenience of integration.

For the operon detection one can adopt the methods presented by Bischler *et al.* (2014) for the RNAseg tool or integrate parts of Rockhopper (McClure *et al.*, 2013) for the operon detection. Further, comparisons to existing operon databases can be established as proposed by Fortino *et al.* (2014). They check possible operons against the DOOR database (Mao *et al.*, 2009).
The SNP and DIP detection can be further developed in the direction of comparative analysis of large sets of tracks. Besides the implemented occurrence filter (see Section 5.3.1), plots visualizing the SNPs and DIPs of multiple tracks comparatively would be highly beneficial. A starting point has been made during the master's thesis of Margarita Steinhauer (Steinhauer, 2014) which I have supervised. In this context, the generation and visualization of a phylogenetic tree based on the SNPs within selected tracks is highly desirable.

ReadXplorer offers several automatic analysis functions. For all these analyses complementary plots intuitively visualizing the results can be added to the software. E.g. a SNP distribution plot, showing the distribution of SNPs and DIPs along the whole genome in a histogram could be implemented. Similar plots can be realized for the other automatic analysis methods, i.e. showing TPM and RPKM values and read counts of genes, their distribution, operons, covered genomic features and covered intervals. For TSS, besides their distribution also their expression strength can be visualized in the histogram by showing either the number of read starts or the coverage increase in percent on the y-axis.
A great enhancement, which will however take a considerable amount of time to design and implement, would be to offer annotation editing. This could be based on analysis results calculated from the tracks belonging to the reference or the user could have independent access to this feature to adapt and add genomic features as required. Updating of genomic annotations on the basis of analysis results can be implemented in an automatic fashion. In this case, the researcher defines certain criteria as foundation for the update. E.g. the results of a TSS detection from a high-quality RNA-seq data set can be used to automatically identify and correct wrongly annotated gene starts.
In connection to annotation editing, also the export of the updated annotations in a common format, such as GFF3, GenBank or EMBL should be implemented to enable their direct use in downstream applications or for an updated online genome database submission.

Establishing links to other cross related databases, such as NCBI[34], KEGG[35], Pfam[36], Rfam[37], would be highly beneficial to shorten the path for attaining a holistic cognition regarding particular data elements of interest (e.g. genes or motifs).

The recent and growing field of epigenetics (Carey, 2013) demands suitable visualization and automatic analysis tools. Epigenetic methylation assays produce the same sequence data already visualized in ReadXplorer (Bock *et al.*, 2010), but an additional mode is needed to extract and visualize methylation information from read mapping files. Only IGV is currently able to visualize methylation data. Still, the data sets have to be scanned manually in IGV. Automatic analyses listing genomic areas with certain methylation characteristics and allowing differential methylation analysis on multiple data sets, similar to differential gene expression analysis, could largely simplify the analysis of such data.
The support of external continuous data such as bigWig[38] and BedGraph[39] could further enhance and complete the scope of ReadXplorer. This feature would provide another effortless possibility to combine results from ReadXplorer and other tools while simultaneously providing the corresponding graphical visualization of all corresponding data.
A feature simplifying the locating and import of reference genomes would be their supply via online genome databases. Then, the researcher can simply choose from the list of available genomes, like in the other presented genome viewers (see Table 9).
The addition of a bookmark system would be an adjuvant usability enhancement allowing each user to mark and share genomic positions and data sets of special interest for quick access.

## 7.2. Insights into the *Pseudomonas aeruginosa* pan genome

The workflow developed for the pan genome analysis of the ubiquitous pathogenic bacterium *Pseudomonas aeruginosa* (see Section 6.2) has been successfully applied to all 20 *P. aeruginosa* genomes.
The goal of completely finishing the clone C (C40A) genome sequence was achieved by the here presented assembly approach combining multiple data sets (see Section 6.2.2) and the finishing by (Fischer *et al.*, in preparation). These efforts provide a valuable reference genome to the community, since clone C is the most abundant among all *P. aeruginosa* strains.
The automatic genome annotation with GenDB (Meyer *et al.*, 2003) provided reliable results which could subsequently be used to elucidate the phylogenetic relationship, the core, pan and accessory genome of the sequenced strains in an EDGAR (Blom *et al.*, 2009) evaluation. This whole genome evaluation revealed that the major clonal complexes of the *P. aeruginosa* population segregate into outliers and two clusters with the ubiquitous clones C and PA14 as the most prominent representatives. According to the results, the *P. aeruginosa* pan genome

[34] www.ncbi.nlm.nih.gov
[35] www.genome.jp/kegg
[36] http://pfam.xfam.org
[37] http://rfam.xfam.org
[38] http://genome.ucsc.edu/goldenpath/help/bigWig.html
[39] http://genome.ucsc.edu/goldenpath/help/bedgraph.html

consists of a conserved core genome of at least 4,000 genes (see also Figure 59), an accessory genome of common GIs and RGPs of about a further 10,000 genes and rare genes that are only present in few strains or clonal complexes. In this and other genome sequencing projects, dozens to hundreds of genes previously unknown in *P. aeruginosa* have regularly been observed whenever a strain of a yet uncharacterized clonal lineage was subjected to genome sequencing. Because more than 300 clonal complexes have been identified for *P. aeruginosa* to date, we can estimate a pool of at least 30,000 'private' genes that are rare or very rare in the *P. aeruginosa* population (see also Figure 60). Our empirical data fit perfectly with Koonin and Wolf's (2012) concept that a prokaryotic pan genome is made up of a small, highly conserved core, a much larger 'shell' of genes with limited conservation and a vast 'cloud' of rare poorly conserved genes.

In conjunction with mouse infection experiments conducted by my colleagues within the framework of our pan genome study (Hilker *et al.*, 2014), it was possible to rank the involved pathogenicity factors by relevance: Type III secretion, adhesins and Cif seem to be major stand-alone factors for virulence in mice, whereas the contribution of other elements such as siderophores or type II secretion is combinatorial depending on the asset of pathogenicity factors in the core and accessory genome of the individual strain.
The individual *P. aeruginosa* genome consists of a conserved core, a variable composition of common gene islands and a small set of rare genes that are chiefly hypotheticals of unknown function. Although there are associations between clonal frame and the composition of the accessory genome (Wiehlmann *et al.*, 2007), the clonal complexes freely exchange their genes by recombination and transfer of gene islands.
The detection of the presence or absence of the genomic islands and RGPs benefitted from the coverage analysis (see Section 5.3.8) implemented in ReadXplorer (Hilker *et al.*, 2014). Mapping of the reads on the accessory element followed by the automatic ReadXplorer analysis conveniently provided the exact base pair count and percentage of presence and conservation of the accessory element.
Results of complementing analyses all allowed the deduction that the *P. aeruginosa* core genome freely recombines. The evenly distribution of environmental isolates throughout the whole phylogenetic tree (see Figure 62) and the closer inspection of four subsets of four genomes each (see Figure 63) both support this hypothesis. In contrast, a delimitation of pathogenic and innocuous strains should be observable if these strains would evolve directed in their specific pathogenic or innocuous genome cluster - which is not the case here. Finally, the free exchange of DNA between the *P. aeruginosa* strains was confirmed by the combination of the SNP detection results (see Section 6.4 and Table 16) computed by ReadXplorer (see Section 5.3.1) for each sequenced strain with a further investigation of the frequency distribution of syntenic SNPs executed by my colleagues (Hilker *et al.*, 2014).

The benefit of the read mapping classification approach used in ReadXplorer was directly evident. Alone for the most divergent strain B420 1,300 SNPs, of which more than 900 are intragenic, would have not been identified, if only uniquely mapped reads had been considered. The choice of Single Perfect and Single Best Match mappings, discarding all Common Match mappings, enabled these additional findings. The SNP detection results are also in concordance with the phylogenetic tree. The number of SNPs within the strains clearly separates the same two groups and the outlier B420 from each other.
Additionally, the varying number of unique genes per strain (see Figure 61) indicates that *P. aeruginosa* frequently exchanges DNA with other organisms.
The key question of molecular epidemiology of whether a bacterial species has a clonal, panmictic or epidemic population structure has been typically investigated in the past by multilocus sequence typing (Maiden, 2006) and/or analysis of polyphasic datasets of

phenotypic polymorphisms (Pirnay *et al.*, 2009). The outcome was often dependent on the set of parameters selected for analysis.
The pan genome analysis workflow undertaken in this dissertation and complemented by the additional methods as described in (Hilker *et al.*, 2014), allows an unbiased and complete analysis with a definitive outcome. Therefore, this analysis workflow can be highly recommended for future pan genome and population structure studies.
All types of SNP and DIP events occur at approximately the same frequencies within the analyzed strain panel. This suggests that they are all exposed to the same steady evolutionary forces advancing the mutation rate at a fixed pace disregarding the location and habitat of the bacterium. This observation is consentient with the observations from (Drake *et al.*, 1998).

The here presented study shows that single to few bp insertions and deletions are rarely observed in the analyzed strain panel. However, this does not allow the conclusion that insertions and deletions in general occur more seldom than substitutions. More complex insertions and deletions are still hard to identify by read mapping, as the reads are differing more and more from the reference until they exceed the maximal allowed error threshold for a valid alignment. As an example, a ReadXplorer coverage analysis of the closest relative of the reference PAO1 (see Figure 62), strain 0812, reveals 43 intervals between 1-100bp covering ~1.7kb and further 16 intervals between 100-500bp covering ~3.5kb without any mapped reads. In total 94 uncovered intervals covering ~94kb of PAO1 can be identified for 0812 (see Table S7 from the Appendix, Table 17). These intervals either indicate deletions in 0812 or insertions in PAO1 or these regions posed a problem to sequencing. Even if some intervals originated from sequencing problems, the remaining ones are evidence that deletions and insertions are common rearrangement operations. Additional laboratory experiments will be necessary to further evaluate this issue.
Especially within genes, the number of DIPs is only a small fraction of all DIPs. A likely explanation is the destructive effect of a frame shift in the DNA sequence on the resulting protein. Due to the high gene density in bacteria, most SNPs arise in intragenic regions (~85%). However, only ~18% of these SNPs are causing amino acid exchanges - showing a clear preference of evolution to synonymous SNPs in the examined genomes.

To investigate whether the oppositional phenotypes of the most virulent strain F469 (Cramer *et al.*, 2012) and the innocuous strain B420 are reflected in their genetic material and to identify the causes of virulence, both genomes were subject to a detailed comparison in (Hilker *et al.*, 2014). This comparison was enabled by the groundwork established during this dissertation.
EDGAR revealed that both genomes share 92.3% of their genes. This large overlap suggests that just a few features of their genetic repertoire may account for their differential pathogenicity. By this comparison, my collaborators could identify a couple of unique gene variants involved in the course of human diseases. In the future, these genes can be analyzed experimentally for their role in the high pathogenicity of F469. For strain B420, they discovered that it lacks several genes and gene clusters which are known to play an important role in the course of *P. aeruginosa* infections. Additionally, the high number of SNPs and DIPs identified here for B420 in comparison to PAO1 (see Section 6.4) may constitute another reason for the innocuousness of B420. My collaborators also closely investigated the DIP positions supplied by ReadXplorer for strain B420 (see Table S3A from the Appendix, Table 17). A notable finding is that frequently combined compensatory frameshift mutations were observed among the DIPs.

# Outlook

Continuative analyses of *P. aeruginosa* on the basis of the here presented results are explicated in the following.
First of all, the data basis could be improved in a similar fashion as for the strain C40A, utilizing a current state-of-the-art sequencing technology. With SMRT sequencing (see Section 2.1.6) in combination with the already available Illumina data sets (see Section 6.2) a promising attempt can be made to completely finish the here presented draft genome sequences (Koren *et al.*, 2013). Using the long SMRT sequencing reads, the processing step of aligning the assembly scaffolds to a reference (see Section 6.2.2) can be omitted if the long reads allow scaffolding of all contigs. Denying the reference alignment step is a great chance to correctly reconstruct the corresponding genome and readily detect genome rearrangements without any bias introduced by the reference alignment. Additionally, this effort would result in even more precise genome annotation and pan genome analysis.

In connection to an improved data basis, also the number of pan genome, core genome and unique genes can be honed by broadening the strain panel and applying the here proposed workflow to the enlarged set of genomes. Such a project is feasible, as sequencing costs have again dropped considerably in comparison to the beginning of 2011 (see Section 2.1).
Sequencing more *P. aeruginosa* strains will also make it easier to classify a fresh isolate from a patient. The knowledge gained from all the fundamental bioinformatics analyses and laboratory experiments can be used to aid personalized medicine. Eventually, rapid tests can be developed on this basis for supporting doctors in their diagnosis and pharmaceutical choice. To achieve this large-scale goal, further in-depth analyses of possible pathogenicity factors (as discussed earlier in this Section and in (Hilker *et al.*, 2014)) including laboratory experiments as verification should be conducted.

To reveal new medically and metabolically interesting genes, genes of yet unknown function can be analyzed in detail in the laboratory. Promising candidates for such a laborious task can be chosen from the core genome and from highly pathogenic strains like F469. Highly preserved hypothetical genes from the core genome are at least likely to play an important role in *P. aeruginosa*. Researching predicted genes of yet unknown function with a possible relation to pathogenicity is vital as well. Hereby useful points of attack to cure *P. aeruginosa* infections could be revealed.

The *P. aeruginosa* strains can be analyzed for genome rearrangements. An overview image could summarize and visualize the genome remodelling. On the one hand, this can be achieved with a Mauve analysis (Darling *et al.*, 2004) of the assembled genome sequences; on the other hand, ReadXplorer could be used with the read mapping data and the integrated version of GASV (Sindi *et al.*, 2009). The result of such an analysis is expected to endorse the phylogenetic relationships received from the here presented work.

# Chapter 8

# Conclusion

The localization of this work between bioinformatical software engineering and applied bioinformatics is an important aspect of its successful completion. Both main issues of this dissertation, the software development of the short read mapping visualization and analysis tool ReadXplorer and the pan genome analysis of the pathogenic bacterium *P. aeruginosa*, could benefit from this combination. Fundamental research questions that emerged during the pan genome analysis lead to the development of several useful features and improved algorithms for ReadXplorer, like the feature coverage and the coverage analyses. Otherwise, ReadXplorer facilitated the efficient solution of several biological research questions from the pan genome analysis, like the determination of the amount of SNPs and DIPs and the existence of known RGPs in the analyzed strain panel.

Several different studies using ReadXplorer show that many of its features can now be applied to other research projects as well. Thus, ReadXplorer is a valuable contribution to the field of read mapping analysis and visualization. Its amount of integrated analysis functions and the multiplicity of visualizations allow the application of the software to a wide range of research questions - as long as they involve the analysis of read mapping data or a genome sequence. The modular framework of the software enables easy integration of new analyses and visualizations on demand.

The pan genome analysis of *P. aeruginosa* enabled comprehensive insights into the genomes of its major clonal complexes and the most prominent clones. Not least, the pathogenicity of the strains could be evaluated and ranked and pathogenicity factors could be identified. Further, the workflow enabled estimation of the amount of core and pan genome genes as well as singleton genes expected for *P. aeruginosa*. Naturally, the unbiased and complete pan genome analysis workflow applied to *P. aeruginosa* can be recommended for future pan genome and population structure studies, thus forming a permanent scientific value.

Hence, the integration of both research fields, bioinformatical software engineering and applied bioinformatics, constitutes a high virtue in this context.

# Chapter 9

# Bibliography

Abeel,T., Van Parys,T., Saeys,Y., Galagan,J., and Van de Peer,Y. (2012) GenomeView: a next-generation genome browser. *Nucleic Acids Res.*, **40**, e12, doi: 10.1093/nar/gkr995.

Abeel,T., Peer,Y.V. de, and Saeys,Y. (2009) Toward a gold standard for promoter prediction evaluation. *Bioinformatics*, **25**, i313–i320, doi: 10.1093/bioinformatics/btp191.

Adey,A., Morrison,H.G., Asan, Xun,X., Kitzman,J.O., Turner,E.H., Stackhouse,B., MacKenzie,A.P., Caruccio,N.C., Zhang,X., *et al.* (2010) Rapid, low-input, low-bias construction of shotgun fragment libraries by high-density in vitro transposition. *Genome Biol.*, **11**, R119, doi: 10.1186/gb-2010-11-12-r119.

Aguilar-Barajas,E., Ramírez-Díaz,M.I., Riveros-Rosas,H., and Cervantes,C. (2010) Heavy Metal Resistance in Pseudomonads. In, Ramos,J.L. and Filloux,A. (eds), *Pseudomonas*. Springer Netherlands, pp. 255–282.

Alkan,C., Kidd,J.M., Marques-Bonet,T., Aksay,G., Antonacci,F., Hormozdiari,F., Kitzman,J.O., Baker,C., Malig,M., Mutlu,O., *et al.* (2009) Personalized copy number and segmental duplication maps using next-generation sequencing. *Nat. Genet.*, **41**, 1061–1067, doi: 10.1038/ng.437.

Almeida,L.G.P., Paixão,R., Souza,R.C., Costa,G.C. da, Barrientos,F.J.A., Santos,M.T. dos, Almeida,D.F. de, and Vasconcelos,A.T.R. (2004) A System for Automated Bacterial (genome) Integrated Annotation--SABIA. *Bioinforma. Oxf. Engl.*, **20**, 2832–2833, doi: 10.1093/bioinformatics/bth273.

Altschul,S.F., Gish,W., Miller,W., Myers,E.W., and Lipman,D.J. (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410, doi: 10.1016/S0022-2836(05)80360-2.

Altshuler,D., Pollara,V.J., Cowles,C.R., Van Etten,W.J., Baldwin,J., Linton,L., and Lander,E.S. (2000) An SNP map of the human genome generated by reduced representation shotgun sequencing. *Nature*, **407**, 513–516, doi: 10.1038/35035083.

Amman,F., Wolfinger,M.T., Lorenz,R., Hofacker,I.L., Stadler,P.F., and Findeiß,S. (2014) TSSAR: TSS annotation regime for dRNA-seq data. *BMC Bioinformatics*, **15**, 89, doi: 10.1186/1471-2105-15-89.

Anders,S. and Huber,W. (2010) Differential expression analysis for sequence count data. *Genome Biol.*, **11**, R106, doi: 10.1186/gb-2010-11-10-r106.

Anders,S., Pyl,P.T., and Huber,W. (2014) HTSeq - A Python framework to work with high-throughput sequencing data. *bioRxiv*, doi: 10.1101/002824.

Andrews,S. (2010) FastQC A Quality Control tool for High Throughput Sequence Data. http://www.bioinformatics.babraham.ac.uk/projects/fastqc/ accessed on 30.01.2014.

Angiuoli,S.V., Gussman,A., Klimke,W., Cochrane,G., Field,D., Garrity,G., Kodira,C.D., Kyrpides,N., Madupu,R., Markowitz,V., *et al.* (2008) Toward an online repository of Standard Operating Procedures (SOPs) for (meta)genomic annotation. *Omics J. Integr. Biol.*, **12**, 137–141, doi: 10.1089/omi.2008.0017.

Aparicio,O., Geisberg,J.V., and Struhl,K. (2004) Chromatin immunoprecipitation for determining the association of proteins with specific genomic sequences in vivo. *Curr. Protoc. Cell Biol. Editor. Board Juan Bonifacino Al*, **Chapter 17**, Unit 17.7, doi: 10.1002/0471143030.cb1707s23.

Assefa,S., Keane,T.M., Otto,T.D., Newbold,C., and Berriman,M. (2009) ABACAS: algorithm-based automatic contiguation of assembled sequences. *Bioinformatics*, **25**, 1968–1969, doi: 10.1093/bioinformatics/btp347.

Aury,J.-M., Cruaud,C., Barbe,V., Rogier,O., Mangenot,S., Samson,G., Poulain,J., Anthouard,V., Scarpelli,C., Artiguenave,F., *et al.* (2008) High quality draft sequences for prokaryotic genomes using a mix of new sequencing technologies. *BMC Genomics*, **9**, 603, doi: 10.1186/1471-2164-9-603.

Aziz,R.K., Bartels,D., Best,A.A., DeJongh,M., Disz,T., Edwards,R.A., Formsma,K., Gerdes,S., Glass,E.M., Kubal,M., *et al.* (2008) The RAST Server: rapid annotations using subsystems technology. *BMC Genomics*, **9**, 75, doi: 10.1186/1471-2164-9-75.

Bachhawat,A.K. (2006) Comparative genomics. *Resonance*, **11**, 22–40, doi: 10.1007/BF02855776.

Bailey,T.L., Boden,M., Buske,F.A., Frith,M., Grant,C.E., Clementi,L., Ren,J., Li,W.W., and Noble,W.S. (2009) MEME SUITE: tools for motif discovery and searching. *Nucleic Acids Res.*, **37**, W202–208, doi: 10.1093/nar/gkp335.

Baker,M. (2012) *De novo* genome assembly: what every biologist should know. *Nat. Methods*, **9**, 333–337, doi: 10.1038/nmeth.1935.

Bankevich,A., Nurk,S., Antipov,D., Gurevich,A.A., Dvorkin,M., Kulikov,A.S., Lesin,V.M., Nikolenko,S.I., Pham,S., Prjibelski,A.D., *et al.* (2012) SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing. *J. Comput. Biol.*, **19**, 455–477, doi: 10.1089/cmb.2012.0021.

Barnes,M.R. and Gray,I.C. eds. (2003) Bioinformatics for Geneticists 1 edition. Wiley, Chichester, West Sussex, England ; Hoboken, N.J.

Bartels,D., Kespohl,S., Albaum,S., Drüke,T., Goesmann,A., Herold,J., Kaiser,O., Pühler,A., Pfeiffer,F., Raddatz,G., *et al.* (2005) BACCardI--a tool for the validation of genomic assemblies, assisting genome finishing and intergenome comparison. *Bioinforma. Oxf. Engl.*, **21**, 853–859, doi: 10.1093/bioinformatics/bti091.

Bennett,S. (2004) Solexa Ltd. *Pharmacogenomics*, **5**, 433–438, doi: 10.1517/14622416.5.4.433.

Benson,D.A., Clark,K., Karsch-Mizrachi,I., Lipman,D.J., Ostell,J., and Sayers,E.W. (2014) GenBank. *Nucleic Acids Res.*, **42**, D32–37, doi: 10.1093/nar/gkt1030.

Bentley,D.R., Balasubramanian,S., Swerdlow,H.P., Smith,G.P., Milton,J., Brown,C.G., Hall,K.P., Evers,D.J., Barnes,C.L., Bignell,H.R., *et al.* (2008) Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, **456**, 53–59, doi: 10.1038/nature07517.

Van Berkum,N.L., Lieberman-Aiden,E., Williams,L., Imakaev,M., Gnirke,A., Mirny,L.A., Dekker,J., and Lander,E.S. (2010) Hi-C: a method to study the three-dimensional architecture of genomes. *J. Vis. Exp. JoVE*, doi: 10.3791/1869.

Bertone,P., Stolc,V., Royce,T.E., Rozowsky,J.S., Urban,A.E., Zhu,X., Rinn,J.L., Tongprasit,W., Samanta,M., Weissman,S., *et al.* (2004) Global identification of human transcribed sequences with genome tiling arrays. *Science*, **306**, 2242–2246, doi: 10.1126/science.1103388.

Bischler,T., Kopf,M., and Voß,B. (2014) Transcript mapping based on dRNA-seq data. *BMC Bioinformatics*, **15**, 122, doi: 10.1186/1471-2105-15-122.

Blom,J., Albaum,S.P., Doppmeier,D., Pühler,A., Vorhölter,F.-J., Zakrzewski,M., and Goesmann,A. (2009) EDGAR: A software framework for the comparative analysis of prokaryotic genomes. *BMC Bioinformatics*, **10**, 154, doi: 10.1186/1471-2105-10-154.

Blom,J., Jakobi,T., Doppmeier,D., Jaenicke,S., Kalinowski,J., Stoye,J., and Goesmann,A. (2011) Exact and complete short-read alignment to microbial genomes using Graphics Processing Unit programming. *Bioinformatics*, **27**, 1351–1358, doi: 10.1093/bioinformatics/btr151.

Bock,C., Tomazou,E.M., Brinkman,A.B., Müller,F., Simmer,F., Gu,H., Jäger,N., Gnirke,A., Stunnenberg,H.G., and Meissner,A. (2010) Quantitative comparison of genome-wide DNA methylation mapping technologies. *Nat. Biotechnol.*, **28**, 1106–1114, doi: 10.1038/nbt.1681.

Boetzer,M., Henkel,C.V., Jansen,H.J., Butler,D., and Pirovano,W. (2010) Scaffolding pre-assembled contigs using SSPACE. *Bioinformatics*, btq683, doi: 10.1093/bioinformatics/btq683.

Boisvert,S., Laviolette,F., and Corbeil,J. (2010) Ray: Simultaneous Assembly of Reads from a Mix of High-Throughput Sequencing Technologies. *J. Comput. Biol.*, **17**, 1519–1533, doi: 10.1089/cmb.2009.0238.

Bolotin,A., Quinquis,B., Renault,P., Sorokin,A., Ehrlich,S.D., Kulakauskas,S., Lapidus,A., Goltsman,E., Mazur,M., Pusch,G.D., *et al.* (2004) Complete sequence and comparative genome analysis of the dairy bacterium Streptococcus thermophilus. *Nat. Biotechnol.*, **22**, 1554–1558, doi: 10.1038/nbt1034.

Bondy-Denomy,J., Pawluk,A., Maxwell,K.L., and Davidson,A.R. (2013) Bacteriophage genes that inactivate the CRISPR/Cas bacterial immune system. *Nature*, **493**, 429–432, doi: 10.1038/nature11723.

Borries,A., Vogel,J., and Sharma,C.M. (2012) Differential RNA Sequencing (dRNA-Seq): Deep-Sequencing-Based Analysis of Primary Transcriptomes. In, Harbers, tthias and Kahl,G. (eds), *Tag-Based Next Generation Sequencing*. Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, pp. 109–121.

Bowers,J., Mitchell,J., Beer,E., Buzby,P.R., Causey,M., Efcavitch,J.W., Jarosz,M., Krzymanska-Olejnik,E., Kung,L., Lipson,D., *et al.* (2009) Virtual terminator nucleotides for next-generation DNA sequencing. *Nat. Methods*, **6**, 593–595, doi: 10.1038/nmeth.1354.

Bradnam,K.R., Fass,J.N., Alexandrov,A., Baranay,P., Bechner,M., Birol,I., Boisvert,S., Chapman,J.A., Chapuis,G., Chikhi,R., *et al.* (2013) Assemblathon 2: evaluating *de novo* methods of genome assembly in three vertebrate species. *GigaScience*, **2**, 10, doi: 10.1186/2047-217X-2-10.

Branscomb,E. and Predki,P. (2002) On the High Value of Low Standards. *J. Bacteriol.*, **184**, 6406–6409, doi: 10.1128/JB.184.23.6406-6409.2002.

Breitbart,R.E., Andreadis,A., and Nadal-Ginard,B. (1987) Alternative splicing: a ubiquitous mechanism for the generation of multiple protein isoforms from single genes. *Annu. Rev. Biochem.*, **56**, 467–495, doi: 10.1146/annurev.bi.56.070187.002343.

Brenner,S., Johnson,M., Bridgham,J., Golda,G., Lloyd,D.H., Johnson,D., Luo,S., McCurdy,S., Foy,M., Ewan,M., *et al.* (2000) Gene expression analysis by massively parallel signature sequencing (MPSS) on microbead arrays. *Nat. Biotechnol.*, **18**, 630–634, doi: 10.1038/76469.

De Bruijn,N.G. (1946) A Combinatorial Problem. *K. Ned. Akad. V Wet.*, **49**, 758–764.

Bryson,K., Loux,V., Bossy,R., Nicolas,P., Chaillou,S., van de Guchte,M., Penaud,S., Maguin,E., Hoebeke,M., Bessières,P., *et al.* (2006) AGMIAL: implementing an annotation strategy for prokaryote genomes as a distributed system. *Nucleic Acids Res.*, **34**, 3533–3545, doi: 10.1093/nar/gkl471.

Brzuszkiewicz,E., Brüggemann,H., Liesegang,H., Emmerth,M., Ölschläger,T., Nagy,G., Albermann,K., Wagner,C., Buchrieser,C., Emődy,L., *et al.* (2006) How to become a uropathogen: Comparative genomic analysis of extraintestinal pathogenic *Escherichia coli* strains. *Proc. Natl. Acad. Sci.*, **103**, 12879–12884, doi: 10.1073/pnas.0603038103.

Burrows,M. and Wheeler,D.J. (1994) A block-sorting lossless data compression algorithm. *DEC Syst. Res. Cent. Tech. Rep.*, **124**, 18.

Campbell,P.J., Stephens,P.J., Pleasance,E.D., O'Meara,S., Li,H., Santarius,T., Stebbings,L.A., Leroy,C., Edkins,S., Hardy,C., *et al.* (2008) Identification of somatically acquired rearrangements in cancer using genome-wide massively parallel paired-end sequencing. *Nat. Genet.*, **40**, 722–729, doi: 10.1038/ng.128.

Campos-García,J. (2010) Metabolism of Acyclic Terpenes by Pseudomonas. In, Ramos,J.L. and Filloux,A. (eds), *Pseudomonas*. Springer Netherlands, pp. 235–253.

Carey,N. (2013) The Epigenetics Revolution: How Modern Biology Is Rewriting Our Understanding of Genetics, Disease, and Inheritance Reprint edition. Columbia University Press.

Carneiro,M.O., Russ,C., Ross,M.G., Gabriel,S.B., Nusbaum,C., and DePristo,M.A. (2012) Pacific biosciences sequencing technology for genotyping and variation discovery in human data. *BMC Genomics*, **13**, 375, doi: 10.1186/1471-2164-13-375.

Carver,T., Böhme,U., Otto,T.D., Parkhill,J., and Berriman,M. (2010) BamView: viewing mapped read alignment data in the context of the reference sequence. *Bioinforma. Oxf. Engl.*, **26**, 676–677, doi: 10.1093/bioinformatics/btq010.

Carver,T., Harris,S.R., Berriman,M., Parkhill,J., and McQuillan,J.A. (2012) Artemis: an integrated platform for visualization and analysis of high-throughput sequence-based experimental data. *Bioinforma. Oxf. Engl.*, **28**, 464–469, doi: 10.1093/bioinformatics/btr703.

Carver,T., Harris,S.R., Otto,T.D., Berriman,M., Parkhill,J., and McQuillan,J.A. (2013) BamView: visualizing and interpretation of next-generation sequencing read alignments. *Brief. Bioinform.*, **14**, 203–212, doi: 10.1093/bib/bbr073.

Centers for Disease Control and Prevention (2013) Antibiotic resistance threats in the United States. http://www.cdc.gov/drugresistance/threat-report-2013/ accessed on 03.04.2014.

Chapman,J.A., Ho,I., Sunkara,S., Luo,S., Schroth,G.P., and Rokhsar,D.S. (2011) Meraculous: *De novo* Genome Assembly with Short Paired-End Reads. *PLoS ONE*, **6**, e23501, doi: 10.1371/journal.pone.0023501.

Chaudhuri,R.R., Loman,N.J., Snyder,L.A.S., Bailey,C.M., Stekel,D.J., and Pallen,M.J. (2008) xBASE2: a comprehensive resource for comparative bacterial genomics. *Nucleic Acids Res.*, **36**, D543–546, doi: 10.1093/nar/gkm928.

Chen,K., Wallis,J.W., McLellan,M.D., Larson,D.E., Kalicki,J.M., Pohl,C.S., McGrath,S.D., Wendl,M.C., Zhang,Q., Locke,D.P., *et al.* (2009) BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. *Nat. Methods*, **6**, 677–681, doi: 10.1038/nmeth.1363.

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

Cox,M.P., Peterson,D.A., and Biggs,P.J. (2010) SolexaQA: At-a-glance quality assessment of Illumina second-generation sequencing data. *BMC Bioinformatics*, **11**, 485, doi: 10.1186/1471-2105-11-485.

Crabtree,J., Angiuoli,S.V., Wortman,J.R., and White,O.R. (2007) Sybil: methods and software for multiple genome comparison and visualization. *Methods Mol. Biol. Clifton NJ*, **408**, 93–108, doi: 10.1007/978-1-59745-547-3_6.

Cramer,N., Wiehlmann,L., Ciofu,O., Tamm,S., Høiby,N., and Tümmler,B. (2012) Molecular Epidemiology of Chronic *Pseudomonas aeruginosa* Airway Infections in Cystic Fibrosis. *PLoS ONE*, **7**, e50731, doi: 10.1371/journal.pone.0050731.

Craven,M., Page,D., Shavlik,J., Bockhorst,J., and Glasner,J. (2000) A probabilistic learning approach to whole-genome operon prediction. *Proc. Int. Conf. Intell. Syst. Mol. Biol. ISMB Int. Conf. Intell. Syst. Mol. Biol.*, **8**, 116–127.

Crouch Brewer,S., Wunderink,R.G., Jones,C.B., and Leeper,K.V. (1996) Ventilator-associated pneumonia due to *Pseudomonas aeruginosa*. *Chest*, **109**, 1019–1029.

Cruveiller,S., Le Saux,J., Vallenet,D., Lajus,A., Bocs,S., and Médigue,C. (2005) MICheck: a web tool for fast checking of syntactic annotations of bacterial genomes. *Nucleic Acids Res.*, **33**, W471–479, doi: 10.1093/nar/gki498.

Dam,P., Olman,V., Harris,K., Su,Z., and Xu,Y. (2007) Operon prediction using both genome-specific and general genomic information. *Nucleic Acids Res.*, **35**, 288–298, doi: 10.1093/nar/gkl1018.

Danecek,P., Auton,A., Abecasis,G., Albers,C.A., Banks,E., DePristo,M.A., Handsaker,R.E., Lunter,G., Marth,G.T., Sherry,S.T., *et al.* (2011) The variant call format and VCFtools. *Bioinformatics*, **27**, 2156–2158, doi: 10.1093/bioinformatics/btr330.

Darling,A.C.E., Mau,B., Blattner,F.R., and Perna,N.T. (2004) Mauve: multiple alignment of conserved genomic sequence with rearrangements. *Genome Res.*, **14**, 1394–1403, doi: 10.1101/gr.2289704.

Dayarian,A., Michael,T.P., and Sengupta,A.M. (2010) SOPRA: Scaffolding algorithm for paired reads via statistical optimization. *BMC Bioinformatics*, **11**, 345, doi: 10.1186/1471-2105-11-345.

Delcher,A.L., Bratke,K.A., Powers,E.C., and Salzberg,S.L. (2007) Identifying bacterial genes and endosymbiont DNA with Glimmer. *Bioinforma. Oxf. Engl.*, **23**, 673–679, doi: 10.1093/bioinformatics/btm009.

DePristo,M.A., Banks,E., Poplin,R., Garimella,K.V., Maguire,J.R., Hartl,C., Philippakis,A.A., del Angel,G., Rivas,M.A., Hanna,M., *et al.* (2011) A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat. Genet.*, **43**, 491–498, doi: 10.1038/ng.806.

Van De Wiel,M.A., Leday,G.G.R., Pardo,L., Rue,H., Van Der Vaart,A.W., and Van Wieringen,W.N. (2013) Bayesian analysis of RNA sequencing data by estimating multiple shrinkage priors. *Biostat. Oxf. Engl.*, **14**, 113–128, doi: 10.1093/biostatistics/kxs031.

Dobin,A., Davis,C.A., Schlesinger,F., Drenkow,J., Zaleski,C., Jha,S., Batut,P., Chaisson,M., and Gingeras,T.R. (2013) STAR: ultrafast universal RNA-seq aligner. *Bioinforma. Oxf. Engl.*, **29**, 15–21, doi: 10.1093/bioinformatics/bts635.

Doppmeier,D. (2009) VAMP - Visualization and Analysis of MapPed sequences.

Döring,G., Parameswaran,I.G., and Murphy,T.F. (2011) Differential adaptation of microbial pathogens to airways of patients with cystic fibrosis and chronic obstructive pulmonary disease. *FEMS Microbiol. Rev.*, **35**, 124–146, doi: 10.1111/j.1574-6976.2010.00237.x.

Drake,J.W., Charlesworth,B., Charlesworth,D., and Crow,J.F. (1998) Rates of Spontaneous Mutation. *Genetics*, **148**, 1667–1686.

Druley,T.E., Vallania,F.L., Wegner,D.J., Varley,K.E., Knowles,O.L., Bonds,J.A., Robison,S.W., Doniger,S.W., Hamvas,A., Cole,F.S., *et al.* (2009) Quantification of rare allelic variants from pooled genomic DNA. *Nat. Methods*, **6**, 263–265, doi: 10.1038/nmeth.1307.

Dugar,G., Herbig,A., Förstner,K.U., Heidrich,N., Reinhardt,R., Nieselt,K., and Sharma,C.M. (2013) High-resolution transcriptome maps reveal strain-specific regulatory features of multiple Campylobacter jejuni isolates. *PLoS Genet.*, **9**, e1003495, doi: 10.1371/journal.pgen.1003495.

Earl,D., Bradnam,K., John,J.S., Darling,A., Lin,D., Fass,J., Yu,H.O.K., Buffalo,V., Zerbino,D.R., Diekhans,M., *et al.* (2011) Assemblathon 1: A competitive assessment of *de novo* short read assembly methods. *Genome Res.*, **21**, 2224–2241, doi: 10.1101/gr.126599.111.

Eid,J., Fehr,A., Gray,J., Luong,K., Lyle,J., Otto,G., Peluso,P., Rank,D., Baybayan,P., Bettman,B., *et al.* (2009) Real-Time DNA Sequencing from Single Polymerase Molecules. *Science*, **323**, 133–138, doi: 10.1126/science.1162986.

English,A.C., Richards,S., Han,Y., Wang,M., Vee,V., Qu,J., Qin,X., Muzny,D.M., Reid,J.G., Worley,K.C., *et al.* (2012) Mind the Gap: Upgrading Genomes with Pacific Biosciences RS Long-Read Sequencing Technology. *PLoS ONE*, **7**, e47768, doi: 10.1371/journal.pone.0047768.

Eppinger,M., Baar,C., Raddatz,G., Huson,D.H., and Schuster,S.C. (2004) Comparative analysis of four Campylobacterales. *Nat. Rev. Microbiol.*, **2**, 872–885, doi: 10.1038/nrmicro1024.

Eppstein,D., Galil,Z., Giancarlo,R., and Italiano,G.F. (1990) Sparse Dynamic Programming. In, *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '90. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 513–522.

Evers,D. and Giegerich,R. (1999) RNA movies: visualizing RNA secondary structure spaces. *Bioinformatics*, **15**, 32–37, doi: 10.1093/bioinformatics/15.1.32.

Ewing,B. and Green,P. (1998) Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities. *Genome Res.*, **8**, 186–194.

Eyre-Walker,A. and Keightley,P.D. (2007) The distribution of fitness effects of new mutations. *Nat. Rev. Genet.*, **8**, 610–618, doi: 10.1038/nrg2146.

Del Fabbro,C., Scalabrin,S., Morgante,M., and Giorgi,F.M. (2013) An Extensive Evaluation of Read Trimming Effects on Illumina NGS Data Analysis. *PLoS ONE*, **8**, e85024, doi: 10.1371/journal.pone.0085024.

Falgueras,J., Lara,A.J., Fernández-Pozo,N., Cantón,F.R., Pérez-Trabado,G., and Claros,M.G. (2010) SeqTrim: a high-throughput pipeline for pre-processing any type of sequence read. *BMC Bioinformatics*, **11**, 38, doi: 10.1186/1471-2105-11-38.

Farrar,M. (2007) Striped Smith–Waterman speeds database searches six times over other SIMD implementations. *Bioinformatics*, **23**, 156–161, doi: 10.1093/bioinformatics/btl582.

Ferragina,P. and Manzini,G. (2005) Indexing Compressed Text. *J ACM*, **52**, 552–581, doi: 10.1145/1082036.1082039.

Ferragina,P. and Manzini,G. (2000) Opportunistic data structures with applications. In, *41st Annual Symposium on Foundations of Computer Science, 2000. Proceedings.*, pp. 390–398.

Fischer,S., Cramer,N., Klockgehter,J., Losada,P.M., Dethlefsen,S., Davenport,C.F., Dorda,M., Goesmann,A., Hilker,R., Mielke,S., *et al.* (in preparation) Genome diversity of the major *Pseudomonas aeruginosa* clones C and PA14.

Fitch,W.M. (1970) Distinguishing homologous from analogous proteins. *Syst. Zool.*, **19**, 99–113.

Fitch,W.M. (2000) Homology a personal view on some of the problems. *Trends Genet. TIG*, **16**, 227–231.

Fiume,M., Smith,E.J.M., Brook,A., Strbenac,D., Turner,B., Mezlini,A.M., Robinson,M.D., Wodak,S.J., and Brudno,M. (2012) Savant Genome Browser 2: visualization and analysis for population-scale genomics. *Nucleic Acids Res.*, **40**, W615–621, doi: 10.1093/nar/gks427.

Fiume,M., Williams,V., Brook,A., and Brudno,M. (2010) Savant: genome browser for high-throughput sequencing data. *Bioinformatics*, **26**, 1938–1944, doi: 10.1093/bioinformatics/btq332.

Flusberg,B.A., Webster,D.R., Lee,J.H., Travers,K.J., Olivares,E.C., Clark,T.A., Korlach,J., and Turner,S.W. (2010) Direct detection of DNA methylation during single-molecule, real-time sequencing. *Nat. Methods*, **7**, 461–465, doi: 10.1038/nmeth.1459.

Foquet,M., Samiee,K.T., Kong,X., Chauduri,B.P., Lundquist,P.M., Turner,S.W., Freudenthal,J., and Roitman,D.B. (2008) Improved fabrication of zero-mode waveguides for single-molecule detection. *J. Appl. Phys.*, **103**, 034301–034301–9, doi: 10.1063/1.2831366.

Ford,C.B., Lin,P.L., Chase,M.R., Shah,R.R., Iartchouk,O., Galagan,J., Mohaideen,N., Ioerger,T.R., Sacchettini,J.C., Lipsitch,M., *et al.* (2011) Use of whole genome sequencing to estimate the mutation rate of Mycobacterium tuberculosis during latent infection. *Nat. Genet.*, **43**, 482–486, doi: 10.1038/ng.811.

Fortino,V., Smolander,O.-P., Auvinen,P., Tagliaferri,R., and Greco,D. (2014) Transcriptome dynamics-based operon prediction in prokaryotes. *BMC Bioinformatics*, **15**, 145, doi: 10.1186/1471-2105-15-145.

Fox,J. and Kling,J. (2010) Chinese institute makes bold sequencing play. *Nat. Biotechnol.*, **28**, 189–191, doi: 10.1038/nbt0310-189c.

Fraser,C.M., Eisen,J.A., Nelson,K.E., Paulsen,I.T., and Salzberg,S.L. (2002) The value of complete microbial genome sequencing (you get what you pay for). *J. Bacteriol.*, **184**, 6403–6405; discusion 6405.

Frazer,K.A., Pachter,L., Poliakov,A., Rubin,E.M., and Dubchak,I. (2004) VISTA: computational tools for comparative genomics. *Nucleic Acids Res.*, **32**, W273–279, doi: 10.1093/nar/gkh458.

Fullwood,M.J., Wei,C.-L., Liu,E.T., and Ruan,Y. (2009) Next-generation DNA sequencing of paired-end tags (PET) for transcriptome and genome analyses. *Genome Res.*, **19**, 521–532, doi: 10.1101/gr.074906.107.

Gao,S., Sung,W.-K., and Nagarajan,N. (2011) Opera: reconstructing optimal genomic scaffolds with high-throughput paired-end sequences. *J. Comput. Biol. J. Comput. Mol. Cell Biol.*, **18**, 1681–1691, doi: 10.1089/cmb.2011.0170.

Gentleman,R.C., Carey,V.J., Bates,D.M., Bolstad,B., Dettling,M., Dudoit,S., Ellis,B., Gautier,L., Ge,Y., Gentry,J., *et al.* (2004) Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol.*, **5**, R80, doi: 10.1186/gb-2004-5-10-r80.

Gilles,A., Meglécz,E., Pech,N., Ferreira,S., Malausa,T., and Martin,J.-F. (2011) Accuracy and quality assessment of 454 GS-FLX Titanium pyrosequencing. *BMC Genomics*, **12**, 245, doi: 10.1186/1471-2164-12-245.

Giuliani,M.M., Adu-Bobie,J., Comanducci,M., Aricò,B., Savino,S., Santini,L., Brunelli,B., Bambini,S., Biolchi,A., Capecchi,B., *et al.* (2006) A universal vaccine for serogroup B meningococcus. *Proc. Natl. Acad. Sci. U. S. A.*, **103**, 10834–10839, doi: 10.1073/pnas.0603940103.

Glenn,T.C. (2011) Field guide to next-generation DNA sequencers. *Mol. Ecol. Resour.*, **11**, 759–769, doi: 10.1111/j.1755-0998.2011.03024.x.

Gnerre,S., MacCallum,I., Przybylski,D., Ribeiro,F.J., Burton,J.N., Walker,B.J., Sharpe,T., Hall,G., Shea,T.P., Sykes,S., *et al.* (2011) High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc. Natl. Acad. Sci.*, **108**, 1513–1518, doi: 10.1073/pnas.1017351108.

Goeddel,D.V., Kleid,D.G., Bolivar,F., Heyneker,H.L., Yansura,D.G., Crea,R., Hirose,T., Kraszewski,A., Itakura,K., and Riggs,A.D. (1979) Expression in *Escherichia coli* of chemically synthesized genes for human insulin. *Proc. Natl. Acad. Sci. U. S. A.*, **76**, 106–110.

Good,I.J. (1946) Normal Recurring Decimals. *J. Lond. Math. Soc.*, **s1-21**, 167–169, doi: 10.1112/jlms/s1-21.3.167.

Gordon,A. (2009) FASTX Toolkit. http://hannonlab.cshl.edu/fastx_toolkit/ accessed on 22.03.2014.

Gordon,D., Abajian,C., and Green,P. (1998) Consed: A Graphical Tool for Sequence Finishing. *Genome Res.*, **8**, 195–202, doi: 10.1101/gr.8.3.195.

Gordon,D. and Green,P. (2013) Consed: a graphical editor for next-generation sequencing. *Bioinformatics*, **29**, 2936–2937, doi: 10.1093/bioinformatics/btt515.

Grabherr,M.G., Haas,B.J., Yassour,M., Levin,J.Z., Thompson,D.A., Amit,I., Adiconis,X., Fan,L., Raychowdhury,R., Zeng,Q., *et al.* (2011) Trinity: reconstructing a full-length transcriptome without a genome from RNA-Seq data. *Nat. Biotechnol.*, **29**, 644–652, doi: 10.1038/nbt.1883.

Grim,C.J., Kotewicz,M.L., Power,K.A., Gopinath,G., Franco,A.A., Jarvis,K.G., Yan,Q.Q., Jackson,S.A., Sathyamoorthy,V., Hu,L., *et al.* (2013) Pan-genome analysis of the emerging foodborne pathogen Cronobacter spp. suggests a species-level bidirectional divergence driven by niche adaptation. *BMC Genomics*, **14**, 366, doi: 10.1186/1471-2164-14-366.

De Groot,A.S. and Rappuoli,R. (2004) Genome-derived vaccines. *Expert Rev. Vaccines*, **3**, 59–76, doi: 10.1586/14760584.3.1.59.

Güell,M., van Noort,V., Yus,E., Chen,W.-H., Leigh-Bell,J., Michalodimitrakis,K., Yamada,T., Arumugam,M., Doerks,T., Kühner,S., *et al.* (2009) Transcriptome complexity in a genome-reduced bacterium. *Science*, **326**, 1268–1271, doi: 10.1126/science.1176951.

Hall,B.G. (2004) Predicting the evolution of antibiotic resistance genes. *Nat. Rev. Microbiol.*, **2**, 430–435, doi: 10.1038/nrmicro888.

Handelsman,J. (2004) Metagenomics: application of genomics to uncultured microorganisms. *Microbiol. Mol. Biol. Rev. MMBR*, **68**, 669–685, doi: 10.1128/MMBR.68.4.669-685.2004.

Hardcastle,T.J. and Kelly,K.A. (2010) baySeq: Empirical Bayesian methods for identifying differential expression in sequence count data. *BMC Bioinformatics*, **11**, 422, doi: 10.1186/1471-2105-11-422.

Hayes,M., Pyon,Y.S., and Li,J. (2012) A model-based clustering method for genomic structural variant prediction and genotyping using paired-end sequencing data. *PloS One*, **7**, e52881, doi: 10.1371/journal.pone.0052881.

He,J., Baldini,R.L., Déziel,E., Saucier,M., Zhang,Q., Liberati,N.T., Lee,D., Urbach,J., Goodman,H.M., and Rahme,L.G. (2004) The broad host range pathogen *Pseudomonas aeruginosa* strain PA14 carries two pathogenicity islands harboring plant and animal virulence genes. *Proc. Natl. Acad. Sci. U. S. A.*, **101**, 2530–2535.

He,Y., Xiang,Z., and Mobley,H.L.T. (2010) Vaxign: the first web-based vaccine design program for reverse vaccinology and applications for vaccine development. *J. Biomed. Biotechnol.*, **2010**, 297505, doi: 10.1155/2010/297505.

Van Hijum,S.A.F.T., Zomer,A.L., Kuipers,O.P., and Kok,J. (2005) Projector 2: contig mapping for efficient gap-closure of prokaryotic genome sequence assemblies. *Nucleic Acids Res.*, **33**, W560–566, doi: 10.1093/nar/gki356.

Hilker,R., Munder,A., Klockgether,J., Moran Losada,P., Chouvarine,P., Cramer,N., Davenport,C.F., Dethlefsen,S., Fischer,S., Peng,H., *et al.* (2014) Interclonal gradient of virulence in the *Pseudomonas aeruginosa* pangenome from disease and environment. *Environ. Microbiol.*, n/a–n/a, doi: 10.1111/1462-2920.12606.

Hilker,R., Stadermann,K.B., Doppmeier,D., Kalinowski,J., Stoye,J., Straube,J., Winnebald,J., and Goesmann,A. (2014) ReadXplorer - visualization and analysis of mapped sequences. *Bioinformatics*, **30**, 2247–2254, doi: 10.1093/bioinformatics/btu205.

Hiller,N.L., Janto,B., Hogg,J.S., Boissy,R., Yu,S., Powell,E., Keefe,R., Ehrlich,N.E., Shen,K., Hayes,J., *et al.* (2007) Comparative genomic analyses of seventeen Streptococcus pneumoniae strains: insights into the pneumococcal supragenome. *J. Bacteriol.*, **189**, 8186–8195, doi: 10.1128/JB.00690-07.

Hoberman,R., Dias,J., Ge,B., Harmsen,E., Mayhew,M., Verlaan,D.J., Kwan,T., Dewar,K., Blanchette,M., and Pastinen,T. (2009) A probabilistic approach for SNP discovery in high-throughput human resequencing data. *Genome Res.*, **19**, 1542–1552, doi: 10.1101/gr.092072.109.

Hofacker,I.L. (2003) Vienna RNA secondary structure server. *Nucleic Acids Res.*, **31**, 3429–3431, doi: 10.1093/nar/gkg599.

Hofacker,I.L., Fontana,W., Stadler,P.F., Bonhoeffer,L.S., Tacker,M., and Schuster,P. (1994) Fast folding and comparison of RNA secondary structures. *Monatshefte Für Chem. Chem. Mon.*, **125**, 167–188, doi: 10.1007/BF00818163.

Homer,N. and Nelson,S.F. (2010) Improved variant discovery through local re-alignment of short-read next-generation sequencing data using SRMA. *Genome Biol.*, **11**, R99, doi: 10.1186/gb-2010-11-10-r99.

Huang,Y.-F., Chen,S.-C., Chiang,Y.-S., Chen,T.-H., and Chiu,K.-P. (2012) Palindromic sequence impedes sequencing-by-ligation mechanism. *BMC Syst. Biol.*, **6**, S10, doi: 10.1186/1752-0509-6-S2-S10.

Huber,W., Toedling,J., and Steinmetz,L.M. (2006) Transcript mapping with high-density oligonucleotide tiling arrays. *Bioinforma. Oxf. Engl.*, **22**, 1963–1970, doi: 10.1093/bioinformatics/btl289.

Hunt,M., Newbold,C., Berriman,M., and Otto,T.D. (2014) A comprehensive evaluation of assembly scaffolding tools. *Genome Biol.*, **15**, R42, doi: 10.1186/gb-2014-15-3-r42.

Hurst,L.D., Pál,C., and Lercher,M.J. (2004) The evolutionary dynamics of eukaryotic gene order. *Nat. Rev. Genet.*, **5**, 299–310, doi: 10.1038/nrg1319.

Husemann,P. and Stoye,J. (2010) r2cat: synteny plots and comparative assembly. *Bioinformatics*, **26**, 570–571, doi: 10.1093/bioinformatics/btp690.

Hyatt,D., Chen,G.-L., Locascio,P.F., Land,M.L., Larimer,F.W., and Hauser,L.J. (2010) Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, **11**, 119, doi: 10.1186/1471-2105-11-119.

Iafrate,A.J., Feuk,L., Rivera,M.N., Listewnik,M.L., Donahoe,P.K., Qi,Y., Scherer,S.W., and Lee,C. (2004) Detection of large-scale variation in the human genome. *Nat. Genet.*, **36**, 949–951, doi: 10.1038/ng1416.

Jacob,F., Perrin,D., Sánchez,C., and Monod,J. (1960) L'opéron : groupe de gènes à expression coordonnée par un opérateur [C. R. Acad. Sci. Paris 250 (1960) 1727–1729]. *C. R. Biol.*, **328**, 514–520, doi: 10.1016/j.crvi.2005.04.005.

Jenkinson,A.M., Albrecht,M., Birney,E., Blankenburg,H., Down,T., Finn,R.D., Hermjakob,H., Hubbard,T.J., Jimenez,R.C., Jones,P., *et al.* (2008) Integrating biological data – the Distributed Annotation System. *BMC Bioinformatics*, **9**, S3, doi: 10.1186/1471-2105-9-S8-S3.

Johnson,D.S., Mortazavi,A., Myers,R.M., and Wold,B. (2007) Genome-wide mapping of in vivo protein-DNA interactions. *Science*, **316**, 1497–1502, doi: 10.1126/science.1141319.

Jokinen,P. and Ukkonen,E. (1991) Two algorithms for approximate string matching in static texts. In, Tarlecki,A. (ed), *Mathematical Foundations of Computer Science 1991*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 240–248.

Jong,A. de, Pietersma,H., Cordes,M., Kuipers,O.P., and Kok,J. (2012) PePPER: a webserver for prediction of prokaryote promoter elements and regulons. *BMC Genomics*, **13**, 299, doi: 10.1186/1471-2164-13-299.

Jorjani,H. and Zavolan,M. (2014) TSSer: an automated method to identify transcription start sites in prokaryotic genomes from differential RNA sequencing data. *Bioinforma. Oxf. Engl.*, **30**, 971–974, doi: 10.1093/bioinformatics/btt752.

Joshi,N. (2011) Sickle - Windowed Adaptive Trimming for fastq files using quality. *GitHub*, https://github.com/ucdavis-bioinformatics/sickle accessed on 22.03.2014.

Jothi,R., Cuddapah,S., Barski,A., Cui,K., and Zhao,K. (2008) Genome-wide identification of in vivo protein-DNA binding sites from ChIP-Seq data. *Nucleic Acids Res.*, **36**, 5221–5231, doi: 10.1093/nar/gkn488.

Karolchik,D., Barber,G.P., Casper,J., Clawson,H., Cline,M.S., Diekhans,M., Dreszer,T.R., Fujita,P.A., Guruvadoo,L., Haeussler,M., *et al.* (2014) The UCSC Genome Browser database: 2014 update. *Nucleic Acids Res.*, **42**, D764–D770, doi: 10.1093/nar/gkt1168.

Kasianowicz,J.J., Brandin,E., Branton,D., and Deamer,D.W. (1996) Characterization of individual polynucleotide molecules using a membrane channel. *Proc. Natl. Acad. Sci. U. S. A.*, **93**, 13770–13773.

Kent,W.J., Zweig,A.S., Barber,G., Hinrichs,A.S., and Karolchik,D. (2010) BigWig and BigBed: enabling browsing of large distributed datasets. *Bioinformatics*, **26**, 2204–2207, doi: 10.1093/bioinformatics/btq351.

Kieleczawa,J., Dunn,J.J., and Studier,F.W. (1992) DNA sequencing by primer walking with strings of contiguous hexamers. *Science*, **258**, 1787–1791.

Kim,D., Pertea,G., Trapnell,C., Pimentel,H., Kelley,R., and Salzberg,S.L. (2013) TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biol.*, **14**, R36, doi: 10.1186/gb-2013-14-4-r36.

Kimura,M. (1983) The Neutral Theory of Molecular Evolution Cambridge University Press, Cambridge.

Klockgether,J., Cramer,N., Wiehlmann,L., Davenport,C.F., and Tümmler,B. (2011) *Pseudomonas aeruginosa* Genomic Structure and Diversity. *Front. Microbiol.*, **2**, 150, doi: 10.3389/fmicb.2011.00150.

Klockgether,J., Reva,O., Larbig,K., and Tümmler,B. (2004) Sequence analysis of the mobile genome island pKLC102 of *Pseudomonas aeruginosa* C. *J. Bacteriol.*, **186**, 518–534.

Klockgether,J., Würdemann,D., Reva,O., Wiehlmann,L., and Tümmler,B. (2007) Diversity of the Abundant pKLC102/PAGI-2 Family of Genomic Islands in *Pseudomonas aeruginosa*. *J. Bacteriol.*, **189**, 2443–2459, doi: 10.1128/JB.01688-06.

Knippers,R. (2006) Molekulare Genetik 9. edition, complete revised edition, 614 color figures, 68 tables. Thieme, Stuttgart inter alia.

Koboldt,D.C., Chen,K., Wylie,T., Larson,D.E., McLellan,M.D., Mardis,E.R., Weinstock,G.M., Wilson,R.K., and Ding,L. (2009) VarScan: variant detection in massively parallel sequencing of individual and pooled samples. *Bioinforma. Oxf. Engl.*, **25**, 2283–2285, doi: 10.1093/bioinformatics/btp373.

Kodzius,R., Kojima,M., Nishiyori,H., Nakamura,M., Fukuda,S., Tagami,M., Sasaki,D., Imamura,K., Kai,C., Harbers,M., *et al.* (2006) CAGE: cap analysis of gene expression. *Nat. Methods*, **3**, 211–222, doi: 10.1038/nmeth0306-211.

Kong,Y. (2011) Btrim: a fast, lightweight adapter and quality trimming program for next-generation sequencing technologies. *Genomics*, **98**, 152–153, doi: 10.1016/j.ygeno.2011.05.009.

Koonin,E.V. and Wolf,Y.I. (2012) Evolution of microbes and viruses: a paradigm shift in evolutionary biology? *Front. Cell. Infect. Microbiol.*, **2**, 119, doi: 10.3389/fcimb.2012.00119.

Koren,S., Harhay,G.P., Smith,T.P., Bono,J.L., Harhay,D.M., Mcvey,S.D., Radune,D., Bergman,N.H., and Phillippy,A.M. (2013) Reducing assembly complexity of microbial genomes with single-molecule sequencing. *Genome Biol.*, **14**, R101, doi: 10.1186/gb-2013-14-9-r101.

Koren,S., Treangen,T.J., and Pop,M. (2011) Bambus 2: scaffolding metagenomes. *Bioinforma. Oxf. Engl.*, **27**, 2964–2971, doi: 10.1093/bioinformatics/btr520.

Koressaar,T. and Remm,M. (2007) Enhancements and modifications of primer design program Primer3. *Bioinforma. Oxf. Engl.*, **23**, 1289–1291, doi: 10.1093/bioinformatics/btm091.

Kosugi,S., Natsume,S., Yoshida,K., MacLean,D., Cano,L., Kamoun,S., and Terauchi,R. (2013) Coval: Improving Alignment Quality and Variant Calling Accuracy for Next-Generation Sequencing Data. *PLoS ONE*, **8**, e75402, doi: 10.1371/journal.pone.0075402.

Krawitz,P., Rödelsperger,C., Jäger,M., Jostins,L., Bauer,S., and Robinson,P.N. (2010) Microindel detection in short-read sequence data. *Bioinforma. Oxf. Engl.*, **26**, 722–729, doi: 10.1093/bioinformatics/btq027.

Kung,V.L., Ozer,E.A., and Hauser,A.R. (2010) The Accessory Genome of *Pseudomonas aeruginosa*. *Microbiol. Mol. Biol. Rev.*, **74**, 621–641, doi: 10.1128/MMBR.00027-10.

Kunin,V., Copeland,A., Lapidus,A., Mavromatis,K., and Hugenholtz,P. (2008) A Bioinformatician's Guide to Metagenomics. *Microbiol. Mol. Biol. Rev.*, **72**, 557–578, doi: 10.1128/MMBR.00009-08.

Kurtz,S., Phillippy,A., Delcher,A.L., Smoot,M., Shumway,M., Antonescu,C., and Salzberg,S.L. (2004) Versatile and open software for comparing large genomes. *Genome Biol.*, **5**, R12, doi: 10.1186/gb-2004-5-2-r12.

Laing,C., Buchanan,C., Taboada,E.N., Zhang,Y., Kropinski,A., Villegas,A., Thomas,J.E., and Gannon,V.P. (2010) Pan-genome sequence analysis using Panseq: an online tool for the rapid analysis of core and accessory genomic regions. *BMC Bioinformatics*, **11**, 461, doi: 10.1186/1471-2105-11-461.

Lander,E.S., Linton,L.M., Birren,B., Nusbaum,C., Zody,M.C., Baldwin,J., Devon,K., Dewar,K., Doyle,M., FitzHugh,W., *et al.* (2001) Initial sequencing and analysis of the human genome. *Nature*, **409**, 860–921, doi: 10.1038/35057062.

Langmead,B. and Salzberg,S.L. (2012) Fast gapped-read alignment with Bowtie 2. *Nat. Methods*, **9**, 357–359, doi: 10.1038/nmeth.1923.

Langmead,B., Trapnell,C., Pop,M., and Salzberg,S.L. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol.*, **10**, R25, doi: 10.1186/gb-2009-10-3-r25.

Lee,D.G., Urbach,J.M., Wu,G., Liberati,N.T., Feinbaum,R.L., Miyata,S., Diggins,L.T., He,J., Saucier,M., Déziel,E., *et al.* (2006) Genomic analysis reveals that *Pseudomonas aeruginosa* virulence is combinatorial. *Genome Biol.*, **7**, R90, doi: 10.1186/gb-2006-7-10-r90.

Lee,S., Hormozdiari,F., Alkan,C., and Brudno,M. (2009) MoDIL: detecting small indels from clone-end sequencing with mixtures of distributions. *Nat. Methods*, **6**, 473–474, doi: 10.1038/nmeth.f.256.

Lee,S.J., Lee,D.-Y., Kim,T.Y., Kim,B.H., Lee,J., and Lee,S.Y. (2005) Metabolic engineering of *Escherichia coli* for enhanced production of succinic acid, based on genome comparison and *in silico* gene knockout simulation. *Appl. Environ. Microbiol.*, **71**, 7880–7887, doi: 10.1128/AEM.71.12.7880-7887.2005.

Lee,S.Y., Lee,D.-Y., and Kim,T.Y. (2005) Systems biotechnology for strain improvement. *Trends Biotechnol.*, **23**, 349–358, doi: 10.1016/j.tibtech.2005.05.003.

Lee,X., Fox,Á., Sufrin,J., Henry,H., Majcherczyk,P., Haas,D., and Reimmann,C. (2010) Identification of the Biosynthetic Gene Cluster for the *Pseudomonas aeruginosa* Antimetabolite l-2-Amino-4-Methoxy-trans-3-Butenoic Acid. *J. Bacteriol.*, **192**, 4251–4255, doi: 10.1128/JB.00492-10.

Lerat,E., Daubin,V., and Moran,N.A. (2003) From Gene Trees to Organismal Phylogeny in Prokaryotes:The Case of the γ-Proteobacteria. *PLoS Biol*, **1**, e19, doi: 10.1371/journal.pbio.0000019.

Levene,M.J., Korlach,J., Turner,S.W., Foquet,M., Craighead,H.G., and Webb,W.W. (2003) Zero-Mode Waveguides for Single-Molecule Analysis at High Concentrations. *Science*, **299**, 682–686, doi: 10.1126/science.1079700.

Liang,W., Zhao,Y., Chen,C., Cui,X., Yu,J., Xiao,J., and Kan,B. (2012) Pan-Genomic Analysis Provides Insights into the Genomic Variation and Evolution of Salmonella Paratyphi A. *PLoS ONE*, **7**, e45346, doi: 10.1371/journal.pone.0045346.

Li,B., Ruotti,V., Stewart,R.M., Thomson,J.A., and Dewey,C.N. (2010) RNA-Seq gene expression estimation with read mapping uncertainty. *Bioinformatics*, **26**, 493–500, doi: 10.1093/bioinformatics/btp692.

Li,H. (2011) A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinforma. Oxf. Engl.*, **27**, 2987–2993, doi: 10.1093/bioinformatics/btr509.

Li,H. and Durbin,R. (2009) Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics*, **25**, 1754–1760, doi: 10.1093/bioinformatics/btp324.

Li,H., Handsaker,B., Wysoker,A., Fennell,T., Ruan,J., Homer,N., Marth,G., Abecasis,G., and Durbin,R. (2009) The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, **25**, 2078–2079, doi: 10.1093/bioinformatics/btp352.

Li,H. and Homer,N. (2010) A survey of sequence alignment algorithms for next-generation sequencing. *Brief. Bioinform.*, **11**, 473–483, doi: 10.1093/bib/bbq015.

Li,H., Ruan,J., and Durbin,R. (2008) Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res.*, **18**, 1851–1858, doi: 10.1101/gr.078212.108.

Linke,B., Giegerich,R., and Goesmann,A. (2011) Conveyor: a workflow engine for bioinformatic analyses. *Bioinforma. Oxf. Engl.*, **27**, 903–911, doi: 10.1093/bioinformatics/btr040.

Li,R., Li,Y., Fang,X., Yang,H., Wang,J., Kristiansen,K., and Wang,J. (2009) SNP detection for massively parallel whole-genome resequencing. *Genome Res.*, **19**, 1124–1132, doi: 10.1101/gr.088013.108.

Li,R., Li,Y., Kristiansen,K., and Wang,J. (2008) SOAP: short oligonucleotide alignment program. *Bioinformatics*, **24**, 713–714, doi: 10.1093/bioinformatics/btn025.

Li,R., Zhu,H., Ruan,J., Qian,W., Fang,X., Shi,Z., Li,Y., Li,S., Shan,G., Kristiansen,K., *et al.* (2010) *De novo* assembly of human genomes with massively parallel short read sequencing. *Genome Res.*, **20**, 265–272, doi: 10.1101/gr.097261.109.

Li,S., Dong,X., and Su,Z. (2013) Directional RNA-seq reveals highly complex condition-dependent transcriptomes in E. coli K12 through accurate full-length transcripts assembling. *BMC Genomics*, **14**, 520, doi: 10.1186/1471-2164-14-520.

Liu,L., Cheng,G., Wang,C., Pan,X., Cong,Y., Pan,Q., Wang,J., Zheng,F., Hu,F., and Tang,J. (2009) Identification and Experimental Verification of Protective Antigens Against Streptococcus suis Serotype 2 Based on Genome Sequence Analysis. *Curr. Microbiol.*, **58**, 11–17, doi: 10.1007/s00284-008-9258-x.

Liu,L., Li,Y., Li,S., Hu,N., He,Y., Pong,R., Lin,D., Lu,L., and Law,M. (2012) Comparison of Next-Generation Sequencing Systems. *J. Biomed. Biotechnol.*, **2012**, doi: 10.1155/2012/251364.

Li,Z., Chen,Y., Mu,D., Yuan,J., Shi,Y., Zhang,H., Gan,J., Li,N., Hu,X., Liu,B., *et al.* (2012) Comparison of the two major classes of assembly algorithms: overlap–layout–consensus and de-bruijn-graph. *Brief. Funct. Genomics*, **11**, 25–37, doi: 10.1093/bfgp/elr035.

Lohse,M., Bolger,A.M., Nagel,A., Fernie,A.R., Lunn,J.E., Stitt,M., and Usadel,B. (2012) RobiNA: a user-friendly, integrated software solution for RNA-Seq-based transcriptomics. *Nucleic Acids Res.*, **40**, W622–W627, doi: 10.1093/nar/gks540.

Love,M.I., Huber,W., and Anders,S. (2014) Moderated estimation of fold change and dispersion for RNA-Seq data with DESeq2. *bioRxiv*, doi: 10.1101/002832.

Lowe,T.M. and Eddy,S.R. (1997) tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.*, **25**, 955–964.

Lu,B., Zeng,Z., and Shi,T. (2013) Comparative study of *de novo* assembly and genome-guided assembly strategies for transcriptome reconstruction based on RNA-Seq. *Sci. China Life Sci.*, **56**, 143–155, doi: 10.1007/s11427-013-4442-z.

Luo,R., Liu,B., Xie,Y., Li,Z., Huang,W., Yuan,J., He,G., Chen,Y., Pan,Q., Liu,Y., *et al.* (2012) SOAPdenovo2: an empirically improved memory-efficient short-read *de novo* assembler. *GigaScience*, **1**, 18, doi: 10.1186/2047-217X-1-18.

Lyons,E., Pedersen,B., Kane,J., Alam,M., Ming,R., Tang,H., Wang,X., Bowers,J., Paterson,A., Lisch,D., *et al.* (2008) Finding and comparing syntenic regions among Arabidopsis and the outgroups papaya, poplar, and grape: CoGe with rosids. *Plant Physiol.*, **148**, 1772–1781, doi: 10.1104/pp.108.124867.

Maddison,D.R., Swofford,D.L., and Maddison,W.P. (1997) Nexus: An Extensible File Format for Systematic Information. *Syst. Biol.*, **46**, 590–621, doi: 10.1093/sysbio/46.4.590.

Magoc,T., Pabinger,S., Canzar,S., Liu,X., Su,Q., Puiu,D., Tallon,L.J., and Salzberg,S.L. (2013) GAGE-B: an evaluation of genome assemblers for bacterial organisms. *Bioinformatics*, **29**, 1718–1725, doi: 10.1093/bioinformatics/btt273.

Maiden,M.C.J. (2006) Multilocus sequence typing of bacteria. *Annu. Rev. Microbiol.*, **60**, 561–588, doi: 10.1146/annurev.micro.59.030804.121325.

Maier,L.-K., Benz,J., Fischer,S., Alstetter,M., Jaschinski,K., Hilker,R., Becker,A., Allers,T., Soppa,J., and Marchfelder,A. (2015) Deletion of the Sm1 encoding motif in the lsm gene results in distinct changes in the transcriptome and enhanced swarming activity of Haloferax cells. *Biochimie*, doi: 10.1016/j.biochi.2015.02.023.

Malhis,N. and Jones,S.J.M. (2010) High quality SNP calling using Illumina data at shallow coverage. *Bioinforma. Oxf. Engl.*, **26**, 1029–1035, doi: 10.1093/bioinformatics/btq092.

Mao,F., Dam,P., Chou,J., Olman,V., and Xu,Y. (2009) DOOR: a database for prokaryotic operons. *Nucleic Acids Res.*, **37**, D459–463, doi: 10.1093/nar/gkn757.

Mardis,E., McPherson,J., Martienssen,R., Wilson,R.K., and McCombie,W.R. (2002) What is Finished, and Why Does it Matter. *Genome Res.*, **12**, 669–671, doi: 10.1101/gr.032102.

Margulies,M., Egholm,M., Altman,W.E., Attiya,S., Bader,J.S., Bemben,L.A., Berka,J., Braverman,M.S., Chen,Y.-J., Chen,Z., *et al.* (2005) Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, **437**, 376–380, doi: 10.1038/nature03959.

Martin,E.R., Kinnamon,D.D., Schmidt,M.A., Powell,E.H., Zuchner,S., and Morris,R.W. (2010) SeqEM: an adaptive genotype-calling approach for next-generation sequencing studies. *Bioinforma. Oxf. Engl.*, **26**, 2803–2810, doi: 10.1093/bioinformatics/btq526.

Martin,M. (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal*, **17**, pp. 10–12, doi: 10.14806/ej.17.1.200.

Mathee,K., Narasimhan,G., Valdes,C., Qiu,X., Matewish,J.M., Koehrsen,M., Rokas,A., Yandava,C.N., Engels,R., Zeng,E., *et al.* (2008) Dynamics of *Pseudomonas aeruginosa* genome evolution. *Proc. Natl. Acad. Sci. U. S. A.*, **105**, 3100–3105, doi: 10.1073/pnas.0711982105.

Maxam,A.M. and Gilbert,W. (1977) A new method for sequencing DNA. *Proc. Natl. Acad. Sci.*, **74**, 560–564.

McCarthy,D.J., Chen,Y., and Smyth,G.K. (2012) Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. *Nucleic Acids Res.*, **40**, 4288–4297, doi: 10.1093/nar/gks042.

McClure,R., Balasubramanian,D., Sun,Y., Bobrovskyy,M., Sumby,P., Genco,C.A., Vanderpool,C.K., and Tjaden,B. (2013) Computational analysis of bacterial RNA-Seq data. *Nucleic Acids Res.*, **41**, e140, doi: 10.1093/nar/gkt444.

McCullagh,P. and Nelder,J.A. (1989) Generalized Linear Models. In, *Generalized Linear Models*, CRC Monographs on Statistics & Applied Probability. Chapman & Hall, London, United Kingdom.

McGeoch,D.J. and Davison,A.J. (1986) DNA sequence of the herpes simplex virus type 1 gene encoding glycoprotein gH, and identification of homologues in the genomes of varicella-zoster virus and Epstein-Barr virus. *Nucleic Acids Res.*, **14**, 4281–4292, doi: 10.1093/nar/14.10.4281.

McKenna,A., Hanna,M., Banks,E., Sivachenko,A., Cibulskis,K., Kernytsky,A., Garimella,K., Altshuler,D., Gabriel,S., Daly,M., *et al.* (2010) The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res.*, **20**, 1297–1303, doi: 10.1101/gr.107524.110.

Medini,D., Donati,C., Tettelin,H., Masignani,V., and Rappuoli,R. (2005) The microbial pan-genome. *Curr. Opin. Genet. Dev.*, **15**, 589–594, doi: 10.1016/j.gde.2005.09.006.

Mendoza-Vargas,A., Olvera,L., Olvera,M., Grande,R., Vega-Alvarado,L., Taboada,B., Jimenez-Jacinto,V., Salgado,H., Juárez,K., Contreras-Moreira,B., *et al.* (2009) Genome-Wide Identification of Transcription Start Sites, Promoters and Transcription Factor Binding Sites in E. coli. *PLoS ONE*, **4**, e7526, doi: 10.1371/journal.pone.0007526.

Mentz,A., Neshat,A., Pfeifer-Sancar,K., Pühler,A., Rückert,C., and Kalinowski,J. (2013) Comprehensive discovery and characterization of small RNAs in Corynebacterium glutamicum ATCC 13032. *BMC Genomics*, **14**, 714, doi: 10.1186/1471-2164-14-714.

Merchant,S.S., Prochnik,S.E., Vallon,O., Harris,E.H., Karpowicz,S.J., Witman,G.B., Terry,A., Salamov,A., Fritz-Laylin,L.K., Maréchal-Drouard,L., *et al.* (2007) The Chlamydomonas genome reveals the evolution of key animal and plant functions. *Science*, **318**, 245–250, doi: 10.1126/science.1143609.

Mesaros,N., Nordmann,P., Plésiat,P., Roussel-Delvallez,M., Van Eldere,J., Glupczynski,Y., Van Laethem,Y., Jacobs,F., Lebecque,P., Malfroot,A., *et al.* (2007) *Pseudomonas aeruginosa*: resistance and therapeutic options at the turn of the new millennium. *Clin. Microbiol. Infect. Off. Publ. Eur. Soc. Clin. Microbiol. Infect. Dis.*, **13**, 560–578, doi: 10.1111/j.1469-0691.2007.01681.x.

Meyer,F., Goesmann,A., McHardy,A.C., Bartels,D., Bekel,T., Clausen,J., Kalinowski,J., Linke,B., Rupp,O., Giegerich,R., *et al.* (2003) GenDB—an open source genome annotation system for prokaryote genomes. *Nucleic Acids Res.*, **31**, 2187–2195, doi: 10.1093/nar/gkg312.

Miele,V., Penel,S., and Duret,L. (2011) Ultra-fast sequence clustering from similarity networks with SiLiX. *BMC Bioinformatics*, **12**, 116, doi: 10.1186/1471-2105-12-116.

Miller,J.R., Delcher,A.L., Koren,S., Venter,E., Walenz,B.P., Brownley,A., Johnson,J., Li,K., Mobarry,C., and Sutton,G. (2008) Aggressive assembly of pyrosequencing reads with mates. *Bioinformatics*, **24**, 2818–2824, doi: 10.1093/bioinformatics/btn548.

Minoche,A.E., Dohm,J.C., and Himmelbauer,H. (2011) Evaluation of genomic high-throughput sequencing data generated on Illumina HiSeq and Genome Analyzer systems. *Genome Biol.*, **12**, R112, doi: 10.1186/gb-2011-12-11-r112.

Mittal,R., Aggarwal,S., Sharma,S., Chhibber,S., and Harjai,K. (2009) Urinary tract infections caused by *Pseudomonas aeruginosa*: a minireview. *J. Infect. Public Health*, **2**, 101–111, doi: 10.1016/j.jiph.2009.08.003.

Moore,G.E. (1998) Cramming More Components Onto Integrated Circuits. *Proc. IEEE*, **86**, 82–85, doi: 10.1109/JPROC.1998.658762.

Morrissy,A.S., Morin,R.D., Delaney,A., Zeng,T., McDonald,H., Jones,S., Zhao,Y., Hirst,M., and Marra,M.A. (2009) Next-generation tag sequencing for cancer gene expression profiling. *Genome Res.*, **19**, 1825–1835, doi: 10.1101/gr.094482.109.

Mortazavi,A., Williams,B.A., McCue,K., Schaeffer,L., and Wold,B. (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat. Methods*, **5**, 621–628, doi: 10.1038/nmeth.1226.

Murphy,T.F. (2008) The Many Faces of *Pseudomonas aeruginosa* in Chronic Obstructive Pulmonary Disease. *Clin. Infect. Dis.*, **47**, 1534–1536, doi: 10.1086/593187.

Myers,E.W. (1986) AnO(ND) difference algorithm and its variations. *Algorithmica*, **1**, 251–266, doi: 10.1007/BF01840446.

Myers,E.W., Sutton,G.G., Delcher,A.L., Dew,I.M., Fasulo,D.P., Flanigan,M.J., Kravitz,S.A., Mobarry,C.M., Reinert,K.H.J., Remington,K.A., *et al.* (2000) A Whole-Genome Assembly of Drosophila. *Science*, **287**, 2196–2204, doi: 10.1126/science.287.5461.2196.

Nagalakshmi,U., Wang,Z., Waern,K., Shou,C., Raha,D., Gerstein,M., and Snyder,M. (2008) The transcriptional landscape of the yeast genome defined by RNA sequencing. *Science*, **320**, 1344–1349, doi: 10.1126/science.1158441.

Nagarajan,N., Cook,C., Bonaventura,M.D., Ge,H., Richards,A., Bishop-Lilly,K.A., DeSalle,R., Read,T.D., and Pop,M. (2010) Finishing genomes with limited resources: lessons from an ensemble of microbial genomes. *BMC Genomics*, **11**, 242, doi: 10.1186/1471-2164-11-242.

Needleman,S.B. and Wunsch,C.D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**, 443–453, doi: 10.1016/0022-2836(70)90057-4.

Nicol,J.W., Helt,G.A., Blanchard,S.G., Raja,A., and Loraine,A.E. (2009) The Integrated Genome Browser: free software for distribution and exploration of genome-scale datasets. *Bioinformatics*, **25**, 2730–2731, doi: 10.1093/bioinformatics/btp472.

Nielsen,H., Engelbrecht,J., Brunak,S., and von Heijne,G. (1997) Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Protein Eng.*, **10**, 1–6.

Nielsen,R., Paul,J.S., Albrechtsen,A., and Song,Y.S. (2011) Genotype and SNP calling from next-generation sequencing data. *Nat. Rev. Genet.*, **12**, 443–451, doi: 10.1038/nrg2986.

Niemann,S., Köser,C.U., Gagneux,S., Plinke,C., Homolka,S., Bignell,H., Carter,R.J., Cheetham,R.K., Cox,A., Gormley,N.A., *et al.* (2009) Genomic Diversity among Drug Sensitive and Multidrug Resistant Isolates of Mycobacterium tuberculosis with Identical DNA Fingerprints. *PLoS ONE*, **4**, e7407, doi: 10.1371/journal.pone.0007407.

Nyrén,P. (1987) Enzymatic method for continuous monitoring of DNA polymerase activity. *Anal. Biochem.*, **167**, 235–238.

O'Geen,H., Echipare,L., and Farnham,P.J. (2011) Using ChIP-seq technology to generate high-resolution profiles of histone modifications. *Methods Mol. Biol. Clifton NJ*, **791**, 265–286, doi: 10.1007/978-1-61779-316-5_20.

Ohnishi,J., Mitsuhashi,S., Hayashi,M., Ando,S., Yokoi,H., Ochiai,K., and Ikeda,M. (2002) A novel methodology employing Corynebacterium glutamicum genome information to generate a new L-lysine-producing mutant. *Appl. Microbiol. Biotechnol.*, **58**, 217–223.

Overbeek,R., Begley,T., Butler,R.M., Choudhuri,J.V., Chuang,H.-Y., Cohoon,M., de Crécy-Lagard,V., Diaz,N., Disz,T., Edwards,R., *et al.* (2005) The subsystems approach to genome annotation and its use in the project to annotate 1000 genomes. *Nucleic Acids Res.*, **33**, 5691–5702, doi: 10.1093/nar/gki866.

Pandey,V., Nutter,R.C., and Prediger,E. (2008) Applied Biosystems SOLiD™ System: Ligation-Based Sequencing. In, Janitz,M. (ed), *Next Generation Genome Sequencing*. Wiley-VCH Verlag GmbH & Co. KGaA, pp. 29–42.

Pandya,G.A., Holmes,M.H., Petersen,J.M., Pradhan,S., Karamycheva,S.A., Wolcott,M.J., Molins,C., Jones,M., Schriefer,M.E., Fleischmann,R.D., *et al.* (2009) Whole genome single nucleotide polymorphism based phylogeny of Francisella tularensis and its application to the development of a strain typing assay. *BMC Microbiol.*, **9**, 213, doi: 10.1186/1471-2180-9-213.

Passalacqua,K.D., Varadarajan,A., Ondov,B.D., Okou,D.T., Zwick,M.E., and Bergman,N.H. (2009) Structure and Complexity of a Bacterial Transcriptome. *J. Bacteriol.*, **191**, 3203–3211, doi: 10.1128/JB.00122-09.

Pevzner,P.A., Tang,H., and Tesler,G. (2004) *De novo* Repeat Classification and Fragment Assembly. *Genome Res.*, **14**, 1786–1796, doi: 10.1101/gr.2395204.

Pfeifer-Sancar,K., Mentz,A., Rückert,C., and Kalinowski,J. (2013) Comprehensive analysis of the Corynebacterium glutamicum transcriptome using an improved RNAseq technique. *BMC Genomics*, **14**, 888, doi: 10.1186/1471-2164-14-888.

Phillippy,A.M., Schatz,M.C., and Pop,M. (2008) Genome assembly forensics: finding the elusive mis-assembly. *Genome Biol.*, **9**, R55, doi: 10.1186/gb-2008-9-3-r55.

Pico,A.R., Kelder,T., van Iersel,M.P., Hanspers,K., Conklin,B.R., and Evelo,C. (2008) WikiPathways: pathway editing for the people. *PLoS Biol.*, **6**, e184, doi: 10.1371/journal.pbio.0060184.

Pirnay,J.-P., Bilocq,F., Pot,B., Cornelis,P., Zizi,M., Van Eldere,J., Deschaght,P., Vaneechoutte,M., Jennes,S., Pitt,T., *et al.* (2009) *Pseudomonas aeruginosa* Population Structure Revisited. *PLoS ONE*, **4**, e7740, doi: 10.1371/journal.pone.0007740.

Porter,D., Yao,J., and Polyak,K. (2006) SAGE and related approaches for cancer target identification. *Drug Discov. Today*, **11**, 110–118, doi: 10.1016/S1359-6446(05)03694-9.

Price,M.N., Arkin,A.P., and Alm,E.J. (2006) The life-cycle of operons. *PLoS Genet.*, **2**, e96, doi: 10.1371/journal.pgen.0020096.

Rahme,L.G., Stevens,E.J., Wolfort,S.F., Shao,J., Tompkins,R.G., and Ausubel,F.M. (1995) Common virulence factors for bacterial pathogenicity in plants and animals. *Science*, **268**, 1899–1902.

Ramos,J.-L. (2004a) Pseudomonas. Plenum Press., New York, NY, USA.

Ramos,J.-L. (2004b) Pseudomonas. Springer, Heidelberg, Germany.

Rasmussen,K.R., Stoye,J., and Myers,E.W. (2006) Efficient q-gram filters for finding all epsilon-matches over a given length. *J. Comput. Biol. J. Comput. Mol. Cell Biol.*, **13**, 296–308, doi: 10.1089/cmb.2006.13.296.

Reumers,J., De Rijk,P., Zhao,H., Liekens,A., Smeets,D., Cleary,J., Van Loo,P., Van Den Bossche,M., Catthoor,K., Sabbe,B., *et al.* (2012) Optimized filtering reduces the error rate in detecting genomic variants by short-read sequencing. *Nat. Biotechnol.*, **30**, 61–68, doi: 10.1038/nbt.2053.

Richter,D.C., Schuster,S.C., and Huson,D.H. (2007) OSLay: optimal syntenic layout of unfinished assemblies. *Bioinformatics*, **23**, 1573–1579, doi: 10.1093/bioinformatics/btm153.

Roach,J.C., Boysen,C., Wang,K., and Hood,L. (1995) Pairwise end sequencing: a unified approach to genomic mapping and sequencing. *Genomics*, **26**, 345–353.

Robinson,J.T., Thorvaldsdóttir,H., Winckler,W., Guttman,M., Lander,E.S., Getz,G., and Mesirov,J.P. (2011) Integrative genomics viewer. *Nat. Biotechnol.*, **29**, 24–26, doi: 10.1038/nbt.1754.

Robinson,M.D., McCarthy,D.J., and Smyth,G.K. (2010) edgeR: A Bioconductor Package for Differential Expression Analysis of Digital Gene Expression Data. *Bioinformatics*, **26**, 139–140, doi: 10.1093/bioinformatics/btp616.

Robinson,M.D. and Smyth,G.K. (2007) Moderated statistical tests for assessing differences in tag abundance. *Bioinforma. Oxf. Engl.*, **23**, 2881–2887, doi: 10.1093/bioinformatics/btm453.

Roger,A.J. and Simpson,A.G.B. (2009) Evolution: Revisiting the Root of the Eukaryote Tree. *Curr. Biol.*, **19**, R165–R167, doi: 10.1016/j.cub.2008.12.032.

Rogozin,I.B., Makarova,K.S., Natale,D.A., Spiridonov,A.N., Tatusov,R.L., Wolf,Y.I., Yin,J., and Koonin,E.V. (2002) Congruent evolution of different classes of non-coding DNA in prokaryotic genomes. *Nucleic Acids Res.*, **30**, 4264–4271.

Römling,U., Kader,A., Sriramulu,D.D., Simm,R., and Kronvall,G. (2005) Worldwide distribution of *Pseudomonas aeruginosa* clone C strains in the aquatic environment and cystic fibrosis patients. *Environ. Microbiol.*, **7**, 1029–1038, doi: 10.1111/j.1462-2920.2005.00780.x.

Ronaghi,M., Karamohamed,S., Pettersson,B., Uhlén,M., and Nyrén,P. (1996) Real-Time DNA Sequencing Using Detection of Pyrophosphate Release. *Anal. Biochem.*, **242**, 84–89, doi: 10.1006/abio.1996.0432.

Rothberg,J.M., Hinz,W., Rearick,T.M., Schultz,J., Mileski,W., Davey,M., Leamon,J.H., Johnson,K., Milgrew,M.J., Edwards,M., *et al.* (2011) An integrated semiconductor device enabling non-optical genome sequencing. *Nature*, **475**, 348–352, doi: 10.1038/nature10242.

Rückert,C., Pühler,A., and Kalinowski,J. (2003) Genome-wide analysis of the L-methionine biosynthetic pathway in Corynebacterium glutamicum by targeted gene deletion and homologous complementation. *J. Biotechnol.*, **104**, 213–228.

Rutherford,K., Parkhill,J., Crook,J., Horsnell,T., Rice,P., Rajandream,M.A., and Barrell,B. (2000) Artemis: sequence visualization and annotation. *Bioinforma. Oxf. Engl.*, **16**, 944–945.

Sahlin,K. (2013) BESST - scaffolder for genomic assemblies. *GitHub*, https://github.com/ksahlin/BESST accessed on 24.03.2014.

Saitou,N. and Nei,M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, **4**, 406–425.

Sallet,E., Gouzy,J., and Schiex,T. (2014) EuGene-PP: a next-generation automated annotation pipeline for prokaryotic genomes. *Bioinforma. Oxf. Engl.*, **30**, 2659–2661, doi: 10.1093/bioinformatics/btu366.

Salzberg,S.L., Phillippy,A.M., Zimin,A., Puiu,D., Magoc,T., Koren,S., Treangen,T.J., Schatz,M.C., Delcher,A.L., Roberts,M., *et al.* (2012) GAGE: A critical evaluation of genome assemblies and assembly algorithms. *Genome Res.*, **22**, 557–567, doi: 10.1101/gr.131383.111.

Sambrook,J. and Russell,D.W. (2006) Fragmentation of DNA by Nebulization. *Cold Spring Harb. Protoc.*, **2006**, pdb.prot4539, doi: 10.1101/pdb.prot4539.

Sanger,F. and Coulson,A.R. (1975) A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *J. Mol. Biol.*, **94**, 441–448, doi: 10.1016/0022-2836(75)90213-2.

Sanger,F., Donelson,J.E., Coulson,A.R., Kössel,H., and Fischer,D. (1973) Use of DNA Polymerase I Primed by a Synthetic Oligonucleotide to Determine a Nucleotide Sequence in Phage f1 DNA. *Proc. Natl. Acad. Sci.*, **70**, 1209–1213.

Schatz,M.C., Phillippy,A.M., Sommer,D.D., Delcher,A.L., Puiu,D., Narzisi,G., Salzberg,S.L., and Pop,M. (2013) Hawkeye and AMOS: visualizing and assessing the quality of genome assemblies. *Brief. Bioinform.*, **14**, 213–224, doi: 10.1093/bib/bbr074.

Schierenbeck,L., Ries,D., Rogge,K., Grewe,S., Weisshaar,B., and Kruse,O. (2015) Fast forward genetics to identify mutations causing a high light tolerant phenotype in *Chlamydomonas reinhardtii* by whole-genome-sequencing. *BMC Genomics*, **16**, 57, doi: 10.1186/s12864-015-1232-y.

Schmieder,R. and Edwards,R. (2011) Quality control and preprocessing of metagenomic datasets. *Bioinformatics*, btr026, doi: 10.1093/bioinformatics/btr026.

Schmieder,R., Lim,Y.W., Rohwer,F., and Edwards,R. (2010) TagCleaner: Identification and removal of tag sequences from genomic and metagenomic datasets. *BMC Bioinformatics*, **11**, 341, doi: 10.1186/1471-2105-11-341.

Schultze,T., Hilker,R., Mannala,G.K., Weigel,M., Farmani,N., Goesmann,A., Chakraborty,T., and Hain,T. (in preparation) Intracellular mRNA transcriptome analysis of Listeria monocytogenes by RNA-seq led to discovery of the involvement of a DNA methyltransferase in virulence.

Schulz,M.H., Zerbino,D.R., Vingron,M., and Birney,E. (2012) Oases: robust *de novo* RNA-seq assembly across the dynamic range of expression levels. *Bioinformatics*, **28**, 1086–1092, doi: 10.1093/bioinformatics/bts094.

Schwientek,P., Szczepanowski,R., Rückert,C., Stoye,J., and Pühler,A. (2011) Sequencing of high G+C microbial genomes using the ultrafast pyrosequencing technology. *J. Biotechnol.*, **155**, 68–77, doi: 10.1016/j.jbiotec.2011.04.010.

Seemann,T. (2014) Prokka: rapid prokaryotic genome annotation. *Bioinforma. Oxf. Engl.*, **30**, 2068–2069, doi: 10.1093/bioinformatics/btu153.

Selezska,K., Kazmierczak,M., Müsken,M., Garbe,J., Schobert,M., Häussler,S., Wiehlmann,L., Rohde,C., and Sikorski,J. (2012) *Pseudomonas aeruginosa* population structure revisited under environmental focus: impact of water quality and phage pressure. *Environ. Microbiol.*, **14**, 1952–1967, doi: 10.1111/j.1462-2920.2012.02719.x.

Sette,A. and Rappuoli,R. (2010) Reverse vaccinology: developing vaccines in the era of genomics. *Immunity*, **33**, 530–541, doi: 10.1016/j.immuni.2010.09.017.

Shahid,M., Malik,A., and Sheeba (2003) Multidrug-resistant *Pseudomonas aeruginosa* strains harbouring R-plasmids and AmpC β-lactamases isolated from hospitalised burn patients in a tertiary care hospital of North India. *FEMS Microbiol. Lett.*, **228**, 181–186, doi: 10.1016/S0378-1097(03)00756-0.

Sharma,C.M., Hoffmann,S., Darfeuille,F., Reignier,J., Findeiss,S., Sittka,A., Chabas,S., Reiche,K., Hackermüller,J., Reinhardt,R., *et al.* (2010) The primary transcriptome of the major human pathogen Helicobacter pylori. *Nature*, **464**, 250–255, doi: 10.1038/nature08756.

Sharon,D., Tilgner,H., Grubert,F., and Snyder,M. (2013) A single-molecule long-read survey of the human transcriptome. *Nat. Biotechnol.*, **31**, 1009–1014, doi: 10.1038/nbt.2705.

Shen,Y., Wan,Z., Coarfa,C., Drabek,R., Chen,L., Ostrowski,E.A., Liu,Y., Weinstock,G.M., Wheeler,D.A., Gibbs,R.A., *et al.* (2010) A SNP discovery method to assess variant allele probability from next-generation resequencing data. *Genome Res.*, **20**, 273–280, doi: 10.1101/gr.096388.109.

Shiraki,T., Kondo,S., Katayama,S., Waki,K., Kasukawa,T., Kawaji,H., Kodzius,R., Watahiki,A., Nakamura,M., Arakawa,T., *et al.* (2003) Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage. *Proc. Natl. Acad. Sci. U. S. A.*, **100**, 15776–15781, doi: 10.1073/pnas.2136655100.

Shizuya,H., Birren,B., Kim,U.J., Mancino,V., Slepak,T., Tachiiri,Y., and Simon,M. (1992) Cloning and stable maintenance of 300-kilobase-pair fragments of human DNA in *Escherichia coli* using an F-factor-based vector. *Proc. Natl. Acad. Sci. U. S. A.*, **89**, 8794–8797.

Simon,C. and Daniel,R. (2011) Metagenomic Analyses: Past and Future Trends. *Appl. Environ. Microbiol.*, **77**, 1153–1161, doi: 10.1128/AEM.02345-10.

Simpson,J.T. and Durbin,R. (2012) Efficient *de novo* assembly of large genomes using compressed data structures. *Genome Res.*, **22**, 549–556, doi: 10.1101/gr.126953.111.

Simpson,J.T., Wong,K., Jackman,S.D., Schein,J.E., Jones,S.J.M., and Birol,I. (2009) ABySS: A parallel assembler for short read sequence data. *Genome Res.*, **19**, 1117–1123, doi: 10.1101/gr.089532.108.

Sindi,S., Helman,E., Bashir,A., and Raphael,B.J. (2009) A geometric approach for classification and comparison of structural variants. *Bioinforma. Oxf. Engl.*, **25**, i222–230, doi: 10.1093/bioinformatics/btp208.

Skovgaard,O., Bak,M., L?bner-Olesen,A., and Tommerup,N. (2011) Genome-wide detection of chromosomal rearrangements, indels, and mutations in circular chromosomes by short read sequencing. *Genome Res.*, **21**, 1388–1393, doi: 10.1101/gr.117416.110.

Slater,G.S. and Birney,E. (2005) Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics*, **6**, 31, doi: 10.1186/1471-2105-6-31.

Smeds,L. and Künstner,A. (2011) ConDeTri - A Content Dependent Read Trimmer for Illumina Data. *PLoS ONE*, **6**, e26314, doi: 10.1371/journal.pone.0026314.

Soares-Castro,P. and Santos,P.M. (2015) Deciphering the genome repertoire of Pseudomonas sp. M1 toward β-myrcene biotransformation. *Genome Biol. Evol.*, **7**, 1–17, doi: 10.1093/gbe/evu254.

Srinivasan,R., Chandraprakash,D., Krishnamurthi,R., Singh,P., Scolari,V.F., Krishna,S., and Seshasayee,A.S.N. (2013) Genomic analysis reveals epistatic silencing of 'expensive' genes in *Escherichia coli* K-12. *Mol. Biosyst.*, **9**, 2021–2033, doi: 10.1039/c3mb70035f.

Stadermann,K.B. (2013) Integration of differential expression analysis capabilities into a Viewer for Next Generation Sequencing Data utilizing baySeq, DESeq and an own approach.

Steinhauer,M. (2014) Comparative analysis of SNP data from genome re-sequencing.

Stein,L. (2001) Genome annotation: from sequence to biology. *Nat. Rev. Genet.*, **2**, 493–503, doi: 10.1038/35080529.

Stewart,A.C., Osborne,B., and Read,T.D. (2009) DIYA: a bacterial annotation pipeline for any genomics lab. *Bioinforma. Oxf. Engl.*, **25**, 962–963, doi: 10.1093/bioinformatics/btp097.

Stoesser,G., Baker,W., van den Broek,A., Camon,E., Garcia-Pastor,M., Kanz,C., Kulikova,T., Leinonen,R., Lin,Q., Lombard,V., *et al.* (2002) The EMBL Nucleotide Sequence Database. *Nucleic Acids Res.*, **30**, 21–26.

Stover,C.K., Pham,X.Q., Erwin,A.L., Mizoguchi,S.D., Warrener,P., Hickey,M.J., Brinkman,F.S., Hufnagle,W.O., Kowalik,D.J., Lagrou,M., *et al.* (2000) Complete genome sequence of *Pseudomonas aeruginosa* PAO1, an opportunistic pathogen. *Nature*, **406**, 959–964, doi: 10.1038/35023079.

Talbot,G.H., Bradley,J., Edwards,J.E.,Jr, Gilbert,D., Scheld,M., Bartlett,J.G., and Antimicrobial Availability Task Force of the Infectious Diseases Society of America (2006) Bad bugs need drugs: an update on the development pipeline from the Antimicrobial Availability Task Force of the Infectious Diseases Society of America. *Clin. Infect. Dis. Off. Publ. Infect. Dis. Soc. Am.*, **42**, 657–668, doi: 10.1086/499819.

Tamas,I., Klasson,L., Canbäck,B., Näslund,A.K., Eriksson,A.-S., Wernegreen,J.J., Sandström,J.P., Moran,N.A., and Andersson,S.G.E. (2002) 50 Million Years of Genomic Stasis in Endosymbiotic Bacteria. *Science*, **296**, 2376–2379, doi: 10.1126/science.1071278.

Tettelin,H. (2009) The bacterial pan-genome and reverse vaccinology. *Genome Dyn.*, **6**, 35–47, doi: 10.1159/000235761.

Tettelin,H., Masignani,V., Cieslewicz,M.J., Donati,C., Medini,D., Ward,N.L., Angiuoli,S.V., Crabtree,J., Jones,A.L., Durkin,A.S., *et al.* (2005) Genome analysis of multiple pathogenic isolates of Streptococcus agalactiae: Implications for the microbial 'pan-genome'. *Proc. Natl. Acad. Sci. U. S. A.*, **102**, 13950–13955, doi: 10.1073/pnas.0506758102.

Tettelin,H., Radune,D., Kasif,S., Khouri,H., and Salzberg,S.L. (1999) Optimized multiplex PCR: efficiently closing a whole-genome shotgun sequencing project. *Genomics*, **62**, 500–507, doi: 10.1006/geno.1999.6048.

Tettelin,H., Riley,D., Cattuto,C., and Medini,D. (2008) Comparative genomics: the bacterial pan-genome. *Curr. Opin. Microbiol.*, **11**, 472–477, doi: 10.1016/j.mib.2008.09.006.

Thomas,R.K., Nickerson,E., Simons,J.F., Jänne,P.A., Tengs,T., Yuza,Y., Garraway,L.A., LaFramboise,T., Lee,J.C., Shah,K., *et al.* (2006) Sensitive mutation detection in heterogeneous cancer specimens by massively parallel picoliter reactor sequencing. *Nat. Med.*, **12**, 852–855, doi: 10.1038/nm1437.

Thorvaldsdóttir,H., Robinson,J.T., and Mesirov,J.P. (2013) Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Brief. Bioinform.*, **14**, 178–192, doi: 10.1093/bib/bbs017.

Tong,Y., Charusanti,P., Zhang,L., Weber,T., and Lee,S.Y. (2015) CRISPR-Cas9 Based Engineering of Actinomycetal Genomes. *ACS Synth. Biol.*, doi: 10.1021/acssynbio.5b00038.

Tran,T.T., Zhou,F., Marshburn,S., Stead,M., Kushner,S.R., and Xu,Y. (2009) *De novo* computational prediction of non-coding RNA genes in prokaryotic genomes. *Bioinforma. Oxf. Engl.*, **25**, 2897–2905, doi: 10.1093/bioinformatics/btp537.

Trapnell,C., Hendrickson,D.G., Sauvageau,M., Goff,L., Rinn,J.L., and Pachter,L. (2013) Differential analysis of gene regulation at transcript resolution with RNA-seq. *Nat. Biotechnol.*, **31**, 46–53, doi: 10.1038/nbt.2450.

Trapnell,C., Pachter,L., and Salzberg,S.L. (2009) TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics*, **25**, 1105–1111, doi: 10.1093/bioinformatics/btp120.

Trapnell,C., Roberts,A., Goff,L., Pertea,G., Kim,D., Kelley,D.R., Pimentel,H., Salzberg,S.L., Rinn,J.L., and Pachter,L. (2012) Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nat. Protoc.*, **7**, 562–578, doi: 10.1038/nprot.2012.016.

Trapnell,C., Williams,B.A., Pertea,G., Mortazavi,A., Kwan,G., Baren,M.J. van, Salzberg,S.L., Wold,B.J., and Pachter,L. (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat. Biotechnol.*, **28**, 511–515, doi: 10.1038/nbt.1621.

Travers,K.J., Chin,C.-S., Rank,D.R., Eid,J.S., and Turner,S.W. (2010) A flexible and efficient template format for circular consensus sequencing and SNP detection. *Nucleic Acids Res.*, **38**, e159–e159, doi: 10.1093/nar/gkq543.

Trifonov,V., Pasqualucci,L., Tiacci,E., Falini,B., and Rabadan,R. (2013) SAVI: a statistical algorithm for variant frequency identification. *BMC Syst. Biol.*, **7**, S2, doi: 10.1186/1752-0509-7-S2-S2.

Uchiyama,I. (2003) MBGD: microbial genome database for comparative analysis. *Nucleic Acids Res.*, **31**, 58–62.

UniProt Consortium (2011) Ongoing and future developments at the Universal Protein Resource. *Nucleic Acids Res.*, **39**, D214–219, doi: 10.1093/nar/gkq1020.

Ureta-Vidal,A., Ettwiller,L., and Birney,E. (2003) Comparative genomics: genome-wide analysis in metazoan eukaryotes. *Nat. Rev. Genet.*, **4**, 251–262, doi: 10.1038/nrg1043.

Vallenet,D., Belda,E., Calteau,A., Cruveiller,S., Engelen,S., Lajus,A., Le Fèvre,F., Longin,C., Mornico,D., Roche,D., *et al.* (2013) MicroScope--an integrated microbial resource for the curation and comparative analysis of genomic and metabolic data. *Nucleic Acids Res.*, **41**, D636–647, doi: 10.1093/nar/gks1194.

Vallenet,D., Labarre,L., Rouy,Z., Barbe,V., Bocs,S., Cruveiller,S., Lajus,A., Pascal,G., Scarpelli,C., and Médigue,C. (2006) MaGe: a microbial genome annotation system supported by synteny results. *Nucleic Acids Res.*, **34**, 53–65, doi: 10.1093/nar/gkj406.

Veesenmeyer,J.L., Hauser,A.R., Lisboa,T., and Rello,J. (2009) *Pseudomonas aeruginosa* Virulence and Therapy: Evolving Translational Strategies. *Crit. Care Med.*, **37**, 1777–1786, doi: 10.1097/CCM.0b013e31819ff137.

Velculescu,V.E., Zhang,L., Vogelstein,B., and Kinzler,K.W. (1995) Serial analysis of gene expression. *Science*, **270**, 484–487.

Vezzi,F., Fabbro,C.D., Tomescu,A.I., and Policriti,A. (2012) rNA: a fast and accurate short reads numerical aligner. *Bioinformatics*, **28**, 123–124, doi: 10.1093/bioinformatics/btr617.

Wald,A. (1943) Tests of Statistical Hypotheses Concerning Several Parameters When the Number of Observations is Large. *Trans. Am. Math. Soc.*, **54**, 426, doi: 10.2307/1990256.

Wang,Z., Gerstein,M., and Snyder,M. (2009) RNA-Seq: a revolutionary tool for transcriptomics. *Nat. Rev. Genet.*, **10**, 57–63, doi: 10.1038/nrg2484.

Watson,J.D. and Crick,F.H.C. (1953) Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid. *Nature*, **171**, 737–738, doi: 10.1038/171737a0.

Wayne,L.G., Brenner,D.J., Colwell,R.R., Grimont,P. a. D., Kandler,O., Krichevsky,M.I., Moore,L.H., Moore,W.E.C., Murray,R.G.E., Stackebrandt,E., *et al.* (1987) Report of the Ad Hoc Committee on Reconciliation of Approaches to Bacterial Systematics. *Int. J. Syst. Bacteriol.*, **37**, 463–464, doi: 10.1099/00207713-37-4-463.

Van de Werken,H.J.G., de Vree,P.J.P., Splinter,E., Holwerda,S.J.B., Klous,P., de Wit,E., and de Laat,W. (2012) 4C technology: protocols and data analysis. *Methods Enzymol.*, **513**, 89–112, doi: 10.1016/B978-0-12-391938-0.00004-5.

Westover,B.P., Buhler,J.D., Sonnenburg,J.L., and Gordon,J.I. (2005) Operon prediction without a training set. *Bioinformatics*, **21**, 880–888, doi: 10.1093/bioinformatics/bti123.

Wetterstrand,K.A. (2015) DNA Sequencing Costs: Data from the NHGRI Genome Sequencing Program (GSP).

Whitaker,L. (1914) On the Poisson Law of Small Numbers. *Biometrika*, **10**, 36–71, doi: 10.1093/biomet/10.1.36.

Wiehlmann,L., Wagner,G., Cramer,N., Siebert,B., Gudowius,P., Morales,G., Köhler,T., Delden,C. van, Weinel,C., Slickers,P., *et al.* (2007) Population structure of *Pseudomonas aeruginosa*. *Proc. Natl. Acad. Sci.*, **104**, 8101–8106, doi: 10.1073/pnas.0609213104.

Winsor,G.L., Lam,D.K.W., Fleming,L., Lo,R., Whiteside,M.D., Yu,N.Y., Hancock,R.E.W., and Brinkman,F.S.L. (2011) Pseudomonas Genome Database: improved comparative analysis and population genomics capability for Pseudomonas genomes. *Nucleic Acids Res.*, **39**, D596–600, doi: 10.1093/nar/gkq869.

Winstanley,C., Langille,M.G.I., Fothergill,J.L., Kukavica-Ibrulj,I., Paradis-Bleau,C., Sanschagrin,F., Thomson,N.R., Winsor,G.L., Quail,M.A., Lennard,N., *et al.* (2009) Newly introduced genomic prophage islands are critical determinants of in vivo competitiveness in the Liverpool Epidemic Strain of *Pseudomonas aeruginosa*. *Genome Res.*, **19**, 12–23, doi: 10.1101/gr.086082.108.

Wittler,R. (2013) Unraveling overlapping deletions by agglomerative clustering. *BMC Genomics*, **14**, S12, doi: 10.1186/1471-2164-14-S1-S12.

Wu,H., Wang,C., and Wu,Z. (2013) A new shrinkage estimator for dispersion improves differential expression detection in RNA-seq data. *Biostat. Oxf. Engl.*, **14**, 232–243, doi: 10.1093/biostatistics/kxs033.

Wu,T.D. and Nacu,S. (2010) Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics*, **26**, 873–881, doi: 10.1093/bioinformatics/btq057.

Xiang,Z., Todd,T., Ku,K.P., Kovacic,B.L., Larson,C.B., Chen,F., Hodges,A.P., Tian,Y., Olenzek,E.A., Zhao,B., *et al.* (2008) VIOLIN: vaccine investigation and online information network. *Nucleic Acids Res.*, **36**, D923–928, doi: 10.1093/nar/gkm1039.

Xie,Y., Wu,G., Tang,J., Luo,R., Patterson,J., Liu,S., Huang,W., He,G., Gu,S., Li,S., *et al.* (2014) SOAPdenovo-Trans: *de novo* transcriptome assembly with short RNA-Seq reads. *Bioinformatics*, btu077, doi: 10.1093/bioinformatics/btu077.

Yandell,M. and Ence,D. (2012) A beginner's guide to eukaryotic genome annotation. *Nat. Rev. Genet.*, **13**, 329–342, doi: 10.1038/nrg3174.

Yukawa,H., Omumasaba,C.A., Nonaka,H., Kós,P., Okai,N., Suzuki,N., Suda,M., Tsuge,Y., Watanabe,J., Ikeda,Y., *et al.* (2007) Comparative analysis of the Corynebacterium glutamicum group and complete genome sequence of strain R. *Microbiol. Read. Engl.*, **153**, 1042–1058, doi: 10.1099/mic.0.2006/003657-0.

Zeitouni,B., Boeva,V., Janoueix-Lerosey,I., Loeillet,S., Legoix-né,P., Nicolas,A., Delattre,O., and Barillot,E. (2010) SVDetect: a tool to identify genomic structural variations from paired-end and mate-pair sequencing data. *Bioinforma. Oxf. Engl.*, **26**, 1895–1896, doi: 10.1093/bioinformatics/btq293.

Zerbino,D.R. and Birney,E. (2008) Velvet: Algorithms for *de novo* short read assembly using de Bruijn graphs. *Genome Res.*, **18**, 821–829, doi: 10.1101/gr.074492.107.

Zhou,S., Herschleb,J., and Schwartz,D.C. (2007) Chapter 9 A Single Molecule System for Whole Genome Analysis. In, Keith R. Mitchelson (ed), *Perspectives in Bioanalysis*, New High Throughput Technologies for DNA Sequencing and Genomics. Elsevier, pp. 265–300.

Zimin,A.V., Marçais,G., Puiu,D., Roberts,M., Salzberg,S.L., and Yorke,J.A. (2013) The MaSuRCA genome assembler. *Bioinformatics*, **29**, 2669–2677, doi: 10.1093/bioinformatics/btt476.

Zuckerkandl,E. and Pauling,L. (1965) Molecules as documents of evolutionary history. *J. Theor. Biol.*, **8**, 357–366.

# Chapter 10

# Appendix

## 10.1. Algorithms

**Algorithm 8: Read Pair Classification Algorithm Outer Part**

Note that this algorithm is designed for mapping data sets sorted by read name

```
1:  bamFileReader ← a reader for BAM files sorted by read name
2:  bamFileWriter ← a writer for BAM files
3:  lastReadName ← empty String
4:  classification ← classification data and mappings of current read pair
5:  while bamFileReader has next mapping do
6:      if r is mapped to given reference do
7:          if lastReadName ≠ read name of r do
8:              call Algorithm 9 to classify the current read pair
9:              use the bamFileWriter to store all classified mappings of the pair
10:             clear classification
11:         end if
12:         if r is first read of the pair do
13:             add mismatch count for first read to classification
14:             update smallest no. of mismatches for first read in classification if this is the
15:             smallest
16:         end if (* Do the same for r being the second of the pair or for unpaired reads *)
17:         lastReadName ← read name of r
18:     end if
19: end while
```

**Algorithm 9: Read Pair Classification Algorithm Inner Part**

Note that the designation "(type x)" refers to the read pair types listed in Table 6.

```
1:  classification ← classification data and mappings of current read pair
2:  if classification has exactly 1 mapping for both reads of the pair do
3:      d ← distance of both mappings
4:      if orientation of both mappings is concordant with expected orientation do
5:          if d is in the valid range do
6:              store both mappings as perfect pair (type 7)
7:          else if d is too small do
8:              store both mappings as distance too small pair (type 9)
9:          else if d is too large do
10:             store both mappings as distance too large pair (type 8)
11:         end if
12:     else do
13:         if d is in the valid range do
14:             store both mappings as wrongly oriented pair (type 10)
15:         else if d is too small do
16:             store both mappings as wrongly oriented distance too small pair (type 12)
17:         else if d is too large do
18:             store both mappings as wrongly oriented distance too large pair (type 11)
19:         end if
20:     end if
21: else if classification has more mappings for current read pair do
22:     potentialLists ← create hierarchical set for potential pair lists
23:     omitList ← empty list for mappings already assigned to a mapping pair
24:     for each mapping m1 of read 1 do
25:     for each mapping m2 of read 2 do
26:         if omitList does not contain m1 and m2 do
27:             if orientation of m1 and m2 is concordant with expected orientation do
28:                 if d is in the valid range do
29:                     if m1 and m2 are Perfect or Best Match mappings do
30:                         store m1 and m2 as perfect read pair
31:                         add m1 and m2 to omitList
32:                     else do
33:                         add m1 and m2 to potential pair list of potentialLists
34:                     end if
35:                 else if d is too small do
36:                     if d is currently the largest too small distance and m1 and m2 are
37:                     Perfect or Best Match mappings do
38:                         add m1 and m2 to potential small pair list of potentialLists
39:                     else if d is currently the largest potential too small distance do
40:                         add m1 and m2 to potential small pair level two list of
41:                         potentialLists
42:                     end if
43:                 end if (* No pairs are created when d is too large here *)
44:             else do
45:                 if d is in the valid range do
46:                     if m1 and m2 are Perfect or Best Match mappings do
```

```
                                add m1 and m2 to potential wrongly oriented pair list of
                                potentialLists
                            else do
                                add m1 and m2 to potential wrongly oriented pair level two list
                                of potentialLists

                            end if
                        else if d is too small do
                            if d is currently the largest wrongly oriented too small distance and
                            m1 and m2 are Perfect or Best Match mappings do
                                add m1 and m2 to potential wrongly oriented small pair list of
                                potentialLists
                            else if d is currently the largest potential wrongly oriented too small
                            distance do
                                add m1 and m2 to potential wrongly oriented small pair level
                                two list of potentialLists
                            end if
                        end if
                    end if
                end if
            end for
            end for
            for each list l from potentialLists in the following order: small pair list, wrongly
            oriented pair list, wrongly oriented small pair list, pair list, small pair level two list,
            wrongly oriented small pair level two list, wrongly oriented pair list do
                store all potential pairs from l if both mappings of a pair are not in the omitList
                (type 0-5)
                add all mappings from l to the omitList
            end for
            store remaining mapping not in omitList as single mapping (type 6)
        else do (* Only one read of the pair is mapped *)
            store all mappings of the mapped read as single mapping (type 6)
        end if
```

## 10.2. Programming Example

Below is an example class illustrating that only a few steps are necessary for implementing a custom viewer, the `DotViewer`, for ReadXplorer:

```
package your.package;

import de.cebitec.readxplorer.data back end.IntervalRequest;
import de.cebitec.readxplorer.data back end.ThreadListener;
import de.cebitec.readxplorer.data back end.connector.TrackConnector;
import de.cebitec.readxplorer.data back end.dataobjects.Mapping;
import de.cebitec.readxplorer.data back end.dataobjects.MappingResult;
import de.cebitec.readxplorer.data back end.dataobjects.PersistentReference;
import de.cebitec.readxplorer.ui.datavisualisation.BoundsInfo;
import de.cebitec.readxplorer.ui.datavisualisation.BoundsInfoManager;
import de.cebitec.readxplorer.ui.datavisualisation.abstractviewer.AbstractViewer;
import de.cebitec.readxplorer.ui.datavisualisation.basepanel.BasePanel;
import java.awt.Color;
import java.awt.Graphics;
import java.util.ArrayList;
import java.util.List;


/**
 * An exemplary viewer painting a black dot at genomic positions where at least one
 * read starts.
 * @author Rolf Hilker <rolf.hilker at mikrobio.med.uni-giessen.de>
 */
public class DotViewer extends AbstractViewer implements ThreadListener {

    private final TrackConnector trackConnector;
    private MappingResult mappingResult;
    private List<Integer> pointList;

    public DotViewer( BoundsInfoManager boundsManager, BasePanel basePanel,
                      PersistentReference reference, TrackConnector tc ) {
        super( boundsManager, basePanel, reference );
        this.trackConnector = tc;
        mappingResult = new MappingResult( new ArrayList<>(), null );
        pointList = new ArrayList<>();
    }

    @Override
    protected int getMaximalHeight() { 230; } //a standard height

    @Override
    public void boundsChangedHook() {
        BoundsInfo bounds = getBoundsInfo();
        IntervalRequest request = new IntervalRequest(
                bounds.getLogLeft(), //start position of the mapping request
                bounds.getLogRight(), //stop position of the mapping request
                getReference().getActiveChromId(), //id of the chromosome to query
                                                   //data from
                this, //the sending object receiving the result of the request
                false, //true, if diffs and gaps shall be included in the result,
                       //false otherwise
                getReadClassParams() ); //Contains ReadXplorer's read mapping
                                        //classification parameters and if only uniquely
                                        //mapped reads shall be used or all reads.
        trackConnector.addMappingRequest( request );
    }

    @Override
    public void changeToolTipText( int logPos ) {
        setToolTipText( String.valueOf( logPos ) );
    }
```

```
    @Override
    public void notifySkipped() { /* not supported here */ }

    @Override
    public void receiveData( Object data ) {
        if(data.getClass().equals( mappingResult.getClass() ) ) {
            pointList.clear();
            mappingResult = ((MappingResult) data);
            for( Mapping m : mappingResult.getMappings() ) {
                int readStart = m.isFwdStrand() ? m.getStart() : m.getStop();
                int readStartPixel =
                    (int) getPhysBoundariesForLogPos( readStart ).getPhyMiddle();
                //getPhyMiddle() returns the middle pixel of the position area
                pointList.add( readStartPixel );
            } //Note that anything else can be done with the mappings here instead
        }
    }

    @Override
    public void paintComponent( Graphics graphics ) {
        super.paintComponent( graphics );
        graphics.setColor( Color.BLACK );
        for( Integer xCoord : pointList ) { //draw a point where a mapping starts
            graphics.drawLine( xCoord, 10, xCoord, 10 );
        }
    }
}
```

## 10.3. Primers and Adapters

In the following, Illumina **adapter** and **primer** sequences which had to be removed from several data sets prior to any other processing are listed:

- Oligonucleotide sequences for the Multiplexing Sample Prep Oligo Only Kit
  - Multiplexing Index Read Sequencing Primer

    `5'-GATCGGAAGAGCACACGTCTGAACTCCAGTCAC-3'`

- Oligonucleotide sequences for Genomic DNA
  - Adapters

    `5'-GATCGGAAGAGCTCGTATGCCGTCTTCTGCTTG-3'`
  - PCR Primers

    `5'-CAAGCAGAAGACGGCATACGAGCTCTTCCGATCT-3'`



## 10.4. Program Calls

This is the **FastQC** command to evaluate the read quality of all data sets in the current directory:

```
fastqc *.fastq.gz
```

This command was applied for the **Conveyor** read trimming workflow:

```
singleendqualityfilter -l 25 -q 20 -i <dataset>.fastq -o <dataset>-trimmed20.fastq
-l = minimum read length to keep the read
-q = PHRED quality value cutoff
-i = input file
-o = output file
```

This command was applied to trim the reads from 101 to 80 bp using the **FASTX-toolkit**:

```
fastx_trimmer -l 80 -o <dataset>-trimmed80.fastq <dataset>.fastq
-l = length to trim all reads to
-o = output file
```

This command was applied to filter adapter sequences from read data sets using the **FASTX-toolkit**:

```
fastx_clipper -a <adapter> -l 25 -n -M 6 -o <dataset>-filtered.fastq
<dataset>.fastq
-l = minimum read length to keep the read
-n = keep reads containing N's
-M = minimum contiguous alignment required to identify an adapter or primer
-o = output file
```

These **Newbler** commands were used to assemble a paired end data set:

1. Create assembly folder:

   ```
   newAssembly <assemblyFolderName>
   ```

2. Add all read data sets to the folder:

   ```
   addRun <assemblyFolderName> <dataFiles>
   ```

3. Start the assembly:

   ```
   runProject -cpu 0 -minlen 20 -trim -tr -a 150 -vs
   suspectedPrimersAdapters.fasta <assemblyfolder>

   -cpu 0 = use all CPUs
   -minlen = minimum read length to include reads in computation
   -trim = flag to switch on internal quality and primer trimming
   -tr =flag to output the trimmed read data sets
   -a = minimum length of contigs
   -vs = adapter/primer database in fasta format for filtering
   ```

This command was used to **concatenate** all sequences of a multiple fasta file when concatenating all scaffolds of a draft genome:

```
concat.pl -f input_file.fasta -n output_file -S
-f = multiple fasta file to concatenate
-n = output file name omitting file ending
-S = scaffold-mode: Insert linkers at start and end of multi-N-Regions
```

This command was used to import and annotate genomes with **GenDB**:

```
run_gendb_pipeline -p GenDB_Paepan -f <genome>.fasta -G NEG -a -D B
-p = project to import the data into
-f = fasta file to import
-G = gram type (POS or NEG, NEG for P. aeruginosa)
-a = create automatic annotator of the pipeline
-D = domain (bacteria in this case (B))
```

This command was used to map reads to the PAO1 reference sequence with **SARUMAN**:

```
saruman-1.0.7 -q -e 0.08 -r <reads>.fastq -g Pseudomonas_aeruginosa-PAO1_ncbi.fna -
l 80 -c 2000000 -s 19500 -m 0 -i 1 -G 10000000 > <reads>-with-PAO1.jok
-q = input is in fastq format
-e = maximal allowed number of mismatches in percent
-r = input reads file
-g = reference sequence fasta file
-l = read length
-c = number of reads to gather from input file in one chunk
-s = number of reads to align in one chunk
-m = cost for a match
-i = cost for a mismatch
-G = size of the genome chunks gathered in one pass
```

These **BWA** commands were used to map reads to the genomic islands and RGPs of *P. aeruginosa*:

```
bwa index <GIorRGP>.fasta
bwa aln -e 2 -t 4 <GIorRGP>.fasta <DataSet>-trimmed20-<read1|read2>.fastq >
<DataSet>-trimmed20-<read1|read2>.sai
-e = Maximum number of gap extensions
-t = Number of threads for calculations
bwa sampe -n 5 <GIorRGP>.fasta <DataSet>-trimmed20-read1.sai <DataSet>-trimmed20-
read2.sai <DataSet>-trimmed20-read1.fastq <DataSet>-trimmed20-<read2.fastq >
<DataSet>-trimmed20-on-<GIorRGP>.sam
-n = Maximum number of alignments per read
```

This **BWA** command using the newer maximal exact matches algorithm can be used to map data appropriately for ReadXplorer:

```
bwa mem -t 16 -a <Reference>.fasta <DataSet> [<MatesDataSet>]
-t = Number of threads for calculations
-a = Output all found alignments. They are flagged as secondary
```

This **Bowtie 2** command can be used to map data appropriately for ReadXplorer:

```
bowtie2-build <Reference>.fasta <ReferenceIndexBaseNameToUse>
bowtie2 --very-sensitive -k 5 -N 1 -p 32 -x <ReferenceIndexBaseNameToUse> -1
<DataSet-Read1>.fastq -2 <DataSet-Read2>.fastq -S <Output>.sam
--very-sensitive = The most sensible preset option
-N = either 1 or 0. 1 allows for 1 mismatch in the seed, but is slower. Choose 0 if
the calculation lasts too long
-k = maximum number of alignments to output per read
-S = output file
-p = Number of threads for calculations
```

## 10.5. Additional Figures

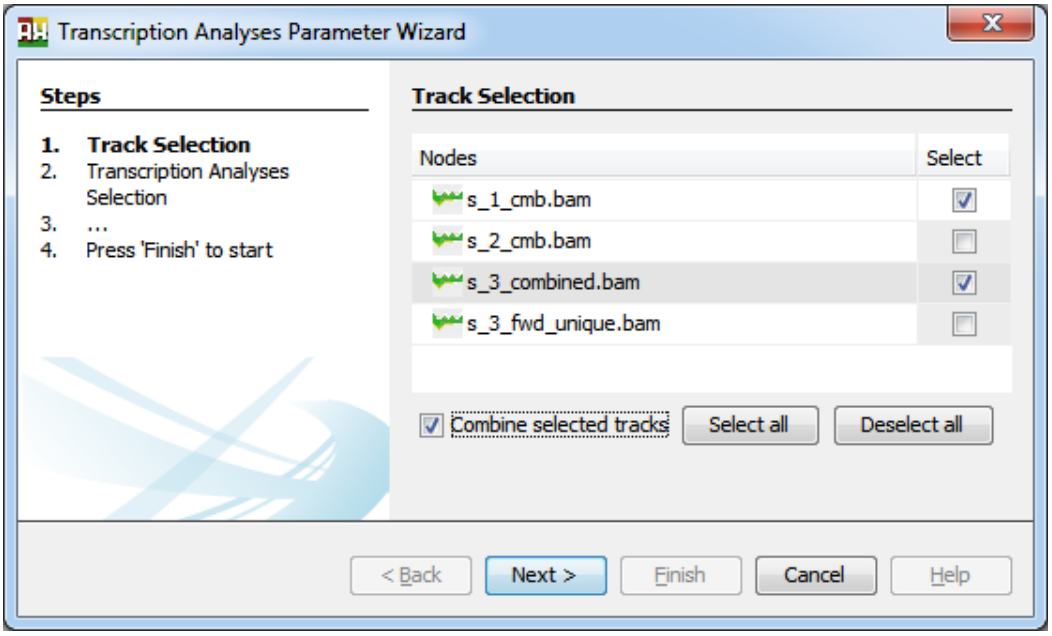



**Figure 66: ReadXplorer Track Selection.** The explorer component listing the tracks is used for both opening tracks and selecting tracks for any of the incorporated analysis methods. This example shows the track selection for the transcription analyses wizard. To combine multiple tracks into one data set for the Track Viewer or an analysis simply the "Combine selected tracks"-button has to be checked. Selecting all tracks or deleting the current selection is simplified by the "Select all" and "Deselect all" buttons.

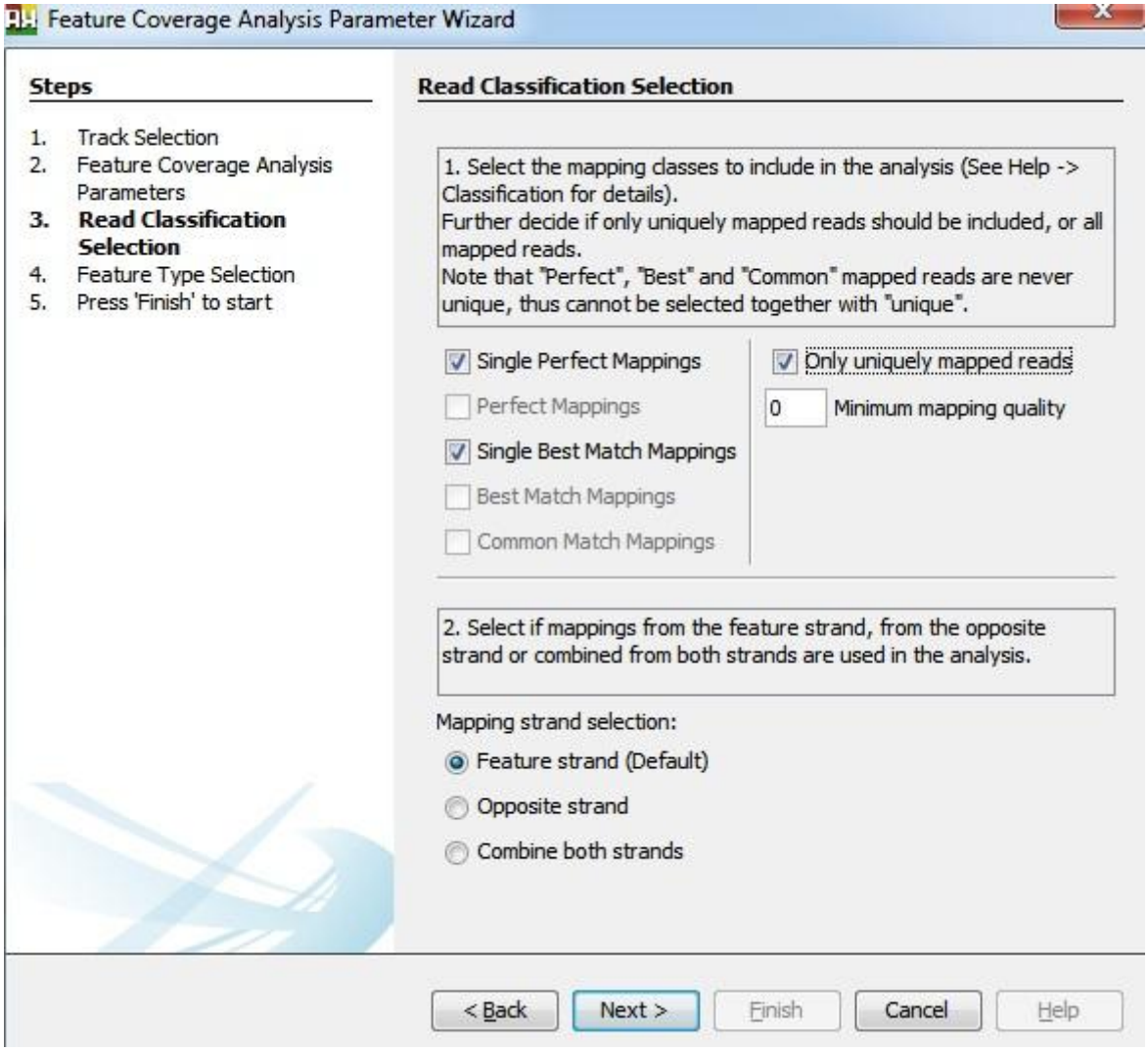



**Figure 67: Read mapping classification and analysis strand configuration.** This wizard page is re-used in each wizard and enables the selection of the read mapping classes to include in the analysis. Also the minimum mapping quality of a mapping can be set if the analysis requires such information. In the lower part of the panel, the strand configuration of the analyzed tracks can be adjusted. For a track originating from a stranded sequencing library the default value is normally the desired choice. But for unstranded libraries it is useful to combine the data from both strands during an analysis.

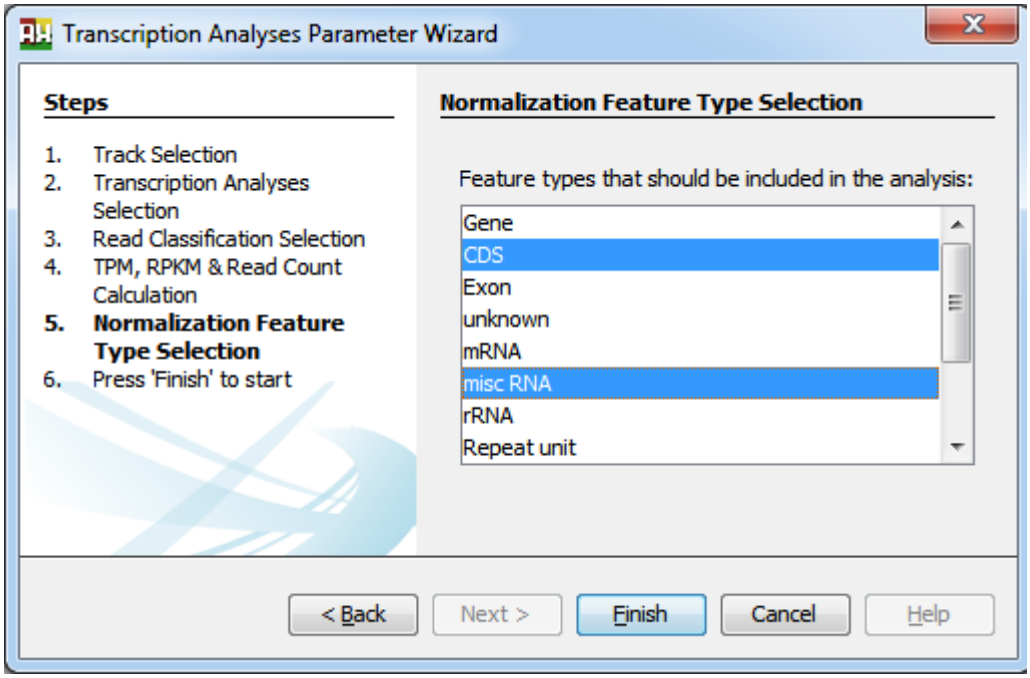



**Figure 68: Genomic feature type selection wizard page.** This wizard page is incorporated in each analysis wizard requiring reference features. It enables selection of a single or multiple feature types, depending on the user needs.

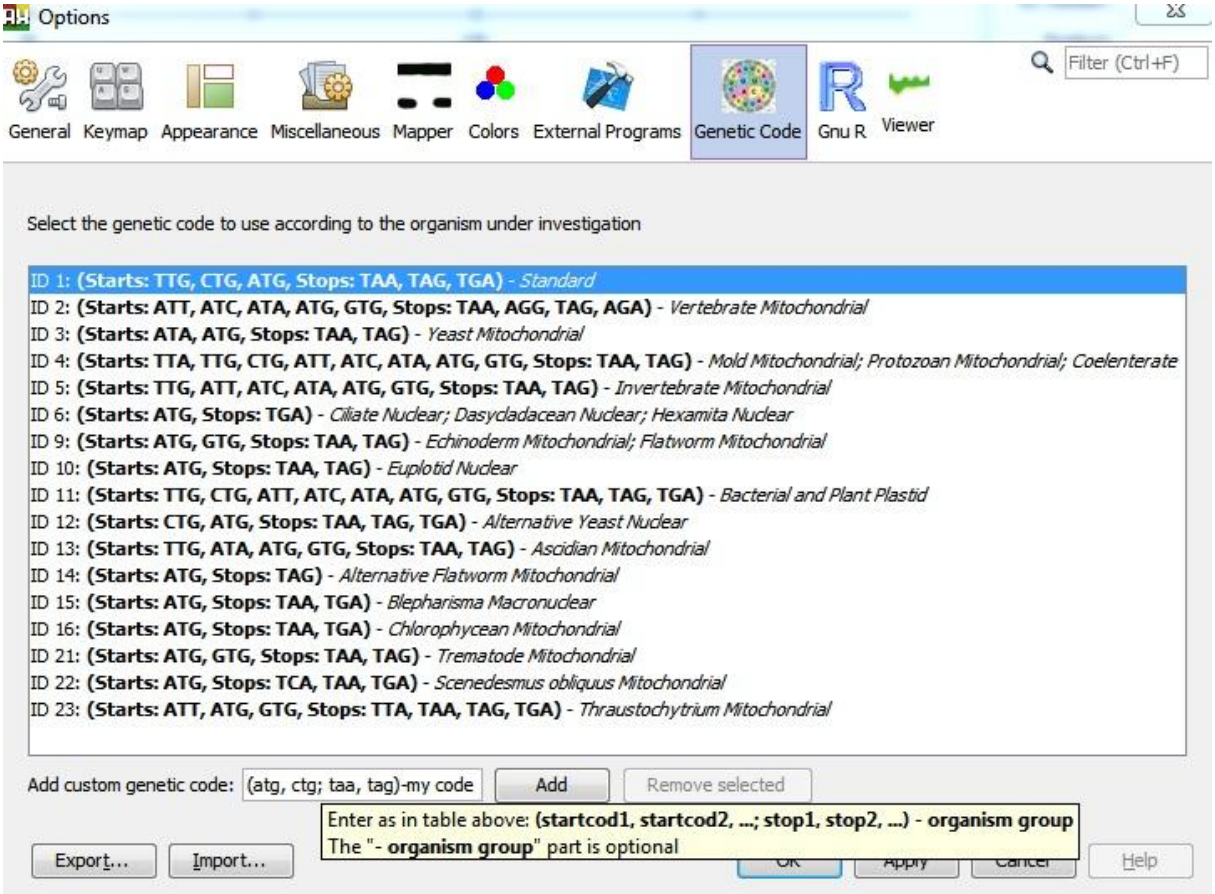



**Figure 69: Genetic code selection.** This options panel enables selection of one of the NCBI standard genetic codes for all ReadXplorer features involving translation of DNA sequences. Additionally, own codes consisting of lists of start and stop codons can be added and chosen.

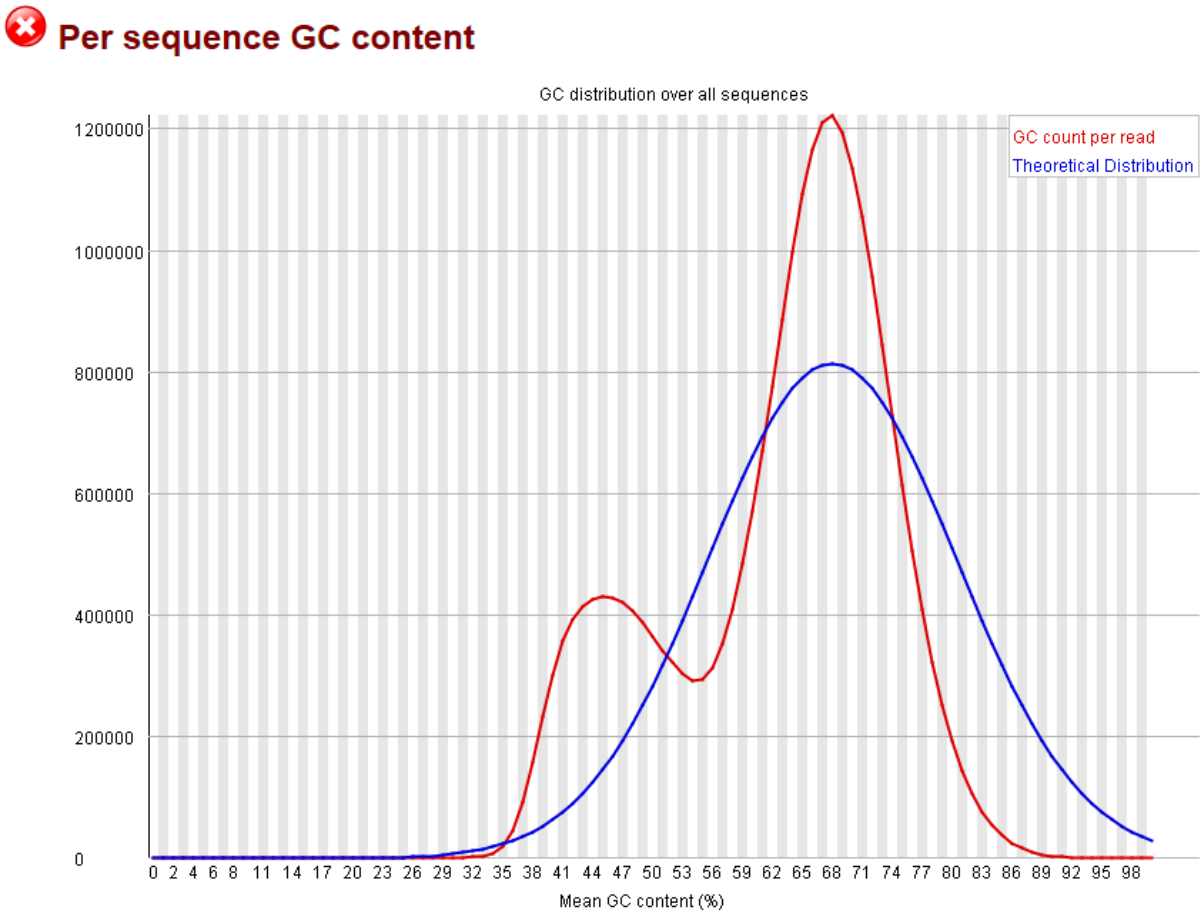



**Figure 70: FastQC GC content analysis of strain F469.** The actual GC content distribution strongly deviates from the theoretical distribution. Thus, the data set contains more low GC reads than expected (between 35 and 50%) and, despite the fact that the *P. aeruginosa* genome contains regions with a higher GC content, it lacks reads with GC content above 75%.

## 10.6. Supplementary Data Description

**Table 17: Description of additional files.** These files are stored on the included supplemental data CD.

| ID | File name | Description |
|---|---|---|
| Table S1 | *P.aeruginosa*-CoreGenome-EDGAR.xls | List of genes contained in the core genome of all 20 sequenced *P. aeruginosa* strains and the reference strain PAO1 generated with EDGAR. |
| Table S2 | emi12606-SuppInfo_Table_S1.xls | The first sheet lists all pan genome genes of all 20 sequenced *P. aeruginosa* strains and the reference strain PAO1. The second sheet lists and visualizes all genes unique per strain. The data has been generated using EDGAR. This file is also a supplementary file from (Hilker *et al.*, 2014). |
| Table S3A | emi12606-SuppInfo_TableS2A.xls | List of SNPs and DIPs of the first 8 *P. aeruginosa* genomes computed by ReadXplorer. This file is also a supplementary file from (Hilker *et al.*, 2014) |
| Table S3B | emi12606-SuppInfo_TableS2B.xls | List of SNPs and DIPs of the remaining 12 *P. aeruginosa* genomes computed by ReadXplorer. The last sheet contains the SNP and DIP detection overview statistics. This file is also a supplementary file from (Hilker *et al.*, 2014) |
| Table S4 | ReadXplorerMappingStatistics.xlsx | Complete Read Mapping Data Overview, complementing Table 15 with all Read Pair track statistics available in ReadXplorer. |
| Table S5 | P._aeruginosa-Velvet_Assemblies.xlsx | Overview of the *P. aeruginosa* Velvet assemblies, having a considerably lower quality than the Newbler assemblies. |
| Table S6 | P._aeruginosa-TrimmingStrategyEvaluation.xlsx | Gives examples elucidating the differences of the 3 tested assembly strategies. |
| Table S7 | CoverageAnalysis-09-0812-uncovered-1-both.xls | ReadXplorer analysis of uncovered intervals in strain 0812 mapped on PAO1. |
| - | ReadXplorer folder | ReadXplorer 2.1 source code |

# Acknowledgements

First of all, I have to express my gratitude to my supervisors Prof. Dr. Alexander Goesmann and Prof. Dr. Jens Stoye for providing me the opportunity to write this dissertation. I appreciate their permanent interest and fruitful discussions. My special thanks go to Alexander Goesmann for always ensuring my funding.
In this regard, I would like to acknowledge the German Federal Ministry of Education and Research for funding my work (grants 0315827C "*P. aeruginosa* pan genome" and 8000 701-3 "DZIF Bioinformatics Platform").

Next, I want to thank all my colleagues from the BRF, Computational Biology and Genome Informatics workgroups during my time in Bielefeld and our new Bioinformatics and Systems Biology workgroup in Gießen. I appreciate all help, hints, useful discussions, the always enjoyable work atmosphere and our countless 'yummy' barbecues. In particular, I thank Dr. Jochen Blom for proof-reading and valuable discussions, Jörn Winnebald for his great groundwork on ReadXplorer and helpful advices, Dr. Burkhard Linke for providing hints and our computer infrastructure in Gießen and Lukas Jelonek, Oliver Rupp, Oliver Schwengers and again Jochen Blom for the convenient time we spent in our office together.
From the master's theses I have supervised, I would like to thank Kai Bernd Stadermann for his highly beneficial implementation of the differential gene expression and Evgeny Anisiforov for his efficient and useful implementations. Jennifer Heß I thank for her contribution to ReadXplorer's SNP detection during her bachelor's thesis and Jasmin Straube for all features she implemented as a student assistant. My thanks also go to Daniel Doppmeier for his well-structured code which made it easy to get involved.
I wish to thank Burkhard Tümmler, Jens Klockgether, Sarah Dethlefsen and Sebastian Fischer from the MH Hanover for our productive collaboration in the *P. aeruginosa* pan genome project and Jörn Kalinowski and his workgroup "Microbial Genomics and Biotechnology" from Bielefeld for their excellent feedback and pleasant cooperation.

I would like to thank all my supporting friends both from schooldays and my years of study for making life richer.
I deeply thank my mother for all her love and trust and the countless days she took good and loving care of my sons to enable me to finish writing. I thank both of my sons for all pleasant and amusing distractions always cheering me up.
Last but not least I am eternally grateful to my wife Henni for her love and inestimable support during all the ups and downs in the past four years of working on this dissertation and her ever loving care for our children. Without your backing, this would not have been possible!

# Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Dissertation selbstständig verfasst und gelieferte Datensätze und Grafiken selbstständig erstellt habe. Ich habe die benutzen Quellen vollständig angegeben und die Stellen der Arbeit, die anderen Publikationen dem Wortlaut oder Sinn nach entnommen sind, in jedem einzelnen Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht.
Außerdem erkläre ich, dass diese Dissertation noch keiner anderen Fakultät oder Universität zur Prüfung vorgelegen hat und noch nicht veröffentlicht worden ist.

Gießen, 15. April 2015

_____
(Rolf Hilker)