Analysis of Antigen Receptor Repertoires Captured by High Throughput Sequencing

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

Sven-Léon Kuchenbecker

am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

Berlin 2018

Gutachter:
Prof. Dr. Knut Reinert, Freie Universität Berlin, Deutschland (1. Gutachter)
Prof. Dr. Peter Nick Robinson, The Jackson Laboratory, U.S.A.

Datum der Disputation: 29.05.2018

Abstract

In vertebrate species, the main mechanisms of defence against various types of pathogens are divided into the *innate* and the *adaptive* immune system. While the former relies on generic mechanisms, for example to detect the presence of bacterial cells, the latter features mechanisms that allow the individual to acquire defenses against specific, potentially novel features of pathogens and to maintain them throughout life. In a simplified sense, the adaptive immune system continuously generates new defenses against all kinds of structures randomly, carefully selecting them not to be reactive against the hosts own cells.

The underlying generative mechanism is a unique somatic recombination process modifying the genes encoding the proteins responsible for the recognition of such foreign structures, the so-called *antigen receptors*. With the advances of high throughput DNA sequencing, we have gained the ability to capture the repertoire of different antigen receptor genes that an individual has acquired by selectively sequencing the recombined loci from a cell sample. This enables us to examine and explore the development and behaviour of the adaptive immune system in a new way, with a variety of potential medical applications.

The main focus of this thesis is on two computational problems related to immune repertoire sequencing. Firstly, we developed a method to properly annotate the raw sequencing data that is generated in such experiments, taking into account various sources of biases and errors that either generally occur in the context of DNA sequencing or are specific for immune repertoire sequencing experiments. We will describe the algorithmic details of this method and then demonstrate its superiority in comparison with previously published methods on various datasets.

Secondly, we developed a machine learning based workflow to interpret this data in the sense that we attempted to classify such recombined genes functionally using a previously trained model. We implemented alternative models within this workflow, which we will first describe formally and then assess their performances on real data in the context of a binary functional feature in T cells, namely whether they have differentiated into cytotoxic or helper T cells.

Contents

Preface

In this thesis, I will describe my work in the context of immune repertoire sequencing, which ranged from the proper annotation of the early raw data to the interpretation and application of Rep-Seq data. The thesis is divided into three parts.

In the first part, comprising three chapters, I will give an introduction to genetical, biotechnological and immunological concepts that are relevant in the context of this thesis. Most importantly, it will be defined what immune repertoire sequencing is and what kind of data is generated in this type of experiment. Furthermore, I will introduce notations and algorithms used in the subsequent chapters.

The second part, *Clonotyping*, comprises two chapters about a method I developed to properly annotate the raw sequence data that is generated in repertoire sequencing experiments. This work was published in

Leon Kuchenbecker, Mikalai Nienen, Jochen Hecht, Avidan U. Neumann, Nina Babel, Knut Reinert, and Peter N. Robinson
IMSEQ — A fast and error aware approach to immunogenetic sequence analysis
Bioinformatics, 31(18):2963–2971, 2015.
doi: 10.1093/bioinformatics/btv309

and is described formally in the first chapter of the second part, followed by a chapter where the method is benchmarked and compared to other competing approaches. The benchmarks shown in this thesis have been extended in comparison to the evaluations shown in the aforementioned publication. They have furthermore been complemented by comparisons to a newer competing method that was published more recently.

In the third part of this thesis, *Applications*, I will cover the interpretation of the annotated repertoire sequencing data as previously obtained. The applications are divided into two general types, *clonotype identity* based applications and applications that aim to *functionally characterize* cells based on the clonotype information. This terminology will be defined in the corresponding chapters. The first described application is work conducted by our collaboration partners, which I supported with my research on clonotype annotation and quality control. It describes a potential clinical application for immune repertoire sequencing in the context of renal transplantation and was published in

Mikalai Dziubianau, Jochen Hecht, Leon Kuchenbecker, Arne Sattler, Ulrik Stervbo, Christian Rödelsperger, Peter Nickel, Avidan U. Neumann, Peter N. Robinson, Stefan Mundlos, Hans-Dieter Volk, Andreas Thiel, Petra Reinke, and Nina Babel.
TCR Repertoire Analysis by Next Generation Sequencing Allows Complex Differential Diagnosis of T Cell-Related Pathology
American Journal of Transplantation, 13(11):2842–2854, 2013.
doi: 10.1111/ajt.12431

The chapter continues with an overview of other immune repertoire sequencing applications from literature and concludes with a method aiming to classify cells from the sequenced sample on a functional level. For this purpose, I developed a workflow based on machine learning, more precisely Support Vector Machines. A number of alternative models are described and their classification performance comparatively evaluated. This work was not previously published.

In the last chapter of this thesis, I will then conclude with a short overview over current challenges in the context of immune repertoire sequencing and an outlook comprising latest developments that will potentially impact this field of research.

In accordance with the standard scientific protocol, throughout this thesis I will use the personal pronoun *we* to indicate the reader and the writer, or my scientific collaborators and myself.

Other Publications

I additionally contributed to the following articles, which appeared during my time as a graduate student, but are not part of this thesis:

Manuel Holtgrewe, Leon Kuchenbecker, and Knut Reinert.
Methods for the detection and assembly of novel sequence in high-throughput sequencing data
Bioinformatics, 31(12):1904–1912, 2015.
doi: 10.1093/bioinformatics/btv051

Guy Gorochov, Martin Larsen, Christophe Parizot, Hélène Brisson, Mikalai Nienen, Leon Kuchenbecker, Nina Babel, and Avidan U. Neumann
Comment on "Tracking donor-reactive T cells: Evidence for clonal deletion in tolerant kidney transplant patients"
Science Translational Medicine, 7(297):297le1–297le1, 2015.
doi: 10.1126/scitranslmed.aab1994

Hong Lei, Leon Kuchenbecker, Mathias Streitz, Birgit Sawitzki, Katrin Vogt, Sybille Landwehr-Kenzel, Jason Millward, Kerstin Juelke, Nina Babel, Avidan U. Neumann, Petra Reinke, and Hans-Dieter Volk
Human CD45RA$^-$ FoxP3^{hi} Memory-Type Regulatory T Cells Show Distinct TCR Repertoires With Conventional T Cells and Play an Important Role in Controlling Early Immune Activation
American Journal of Transplantation, 15(10):2625–2635, 2015.
doi: 10.1111/ajt.13315

Sascha Winter, Katharina Jahn, Stefanie Wehner, Leon Kuchenbecker, Manja Marz, Jens Stoye, and Sebastian Böcker
Finding approximate gene clusters with Gecko 3
Nucleic Acids Research, 44(20):gkw843, 2016.
doi: 10.1093/nar/gkw843

Acknowledgments

I would like to thank everyone who supported me during my PhD work, most importantly my supervisors Knut Reinert, Peter Nick Robinson and Nina Babel, who funded my research and patiently introduced me to the mysteries of immunobiology together with Mikalai Nienen. Furthermore, I would like to express my gratitude to my colleagues at the Charité and Freie Universität Berlin for making going to work an enjoyable experience, but also for fun after work beers, movie nights and outings. Thank you, Temesgen, for being the best officemate I could imagine!

My thanks also go the IMPRS coordinators Kirsten and Fabian for organizing great retreats, summer schools and other activities for our graduate school. Thank you René and Sabrina for proofreading my thesis and for your valuable feedback. Last but not least, I would like to thank my parents for their continuous support and Sabrina for always having my back over the last years!

Part I.

Introduction

1. Genetics and Biotechnology

In this chapter, we will briefly introduce the biological and biotechnological background of the methods discussed in the subsequent chapters. This comprises primarily basic genetic principles and, building on that, the two biotechnological methods that are most important in the context of this thesis: DNA amplification based on PCR and DNA Sequencing. Since an in-depth coverage of these subjects is out of the scope of this thesis, the interested reader is referred to standard literature such as "Lewin's Genes" by Krebs et al. [1].

1.1. Genetic Principles

The hereditary information of life forms as we know them is encoded in their *genome*, a set of *deoxyribonucleic acid* (DNA) and molecules contained in every nucleated cell of an organism. In this section, we will outline the terminology and basics of DNA, protein biosynthesis and genetic variation as it will be discussed in the context of this thesis.

1.1.1. DNA

A DNA molecule is a large polymer chain composed of monomers referred to as *nucleotides*, which are composed of three subunits: a sugar (deoxyribose), a phosphate group and a nitrogenous base. The latter can be either *adenine (A)*, *cytosine (C)*, *guanine (G)* or *thymine (T)*. DNA mostly occurs as two paired chains, referred to as *strands*, bound through hydrogen bonds at the bases: each adenine is paired with a thymine and each cytosine is paired with a guanine, and vice versa. The paired molecule forms a coiled helix structure with the sugar-phosphate backbone pointing outwards and the *base pairs (bp)* pointing inwards. A DNA strand is *directional*, the two ends of the strand are referred to as the *3' end* and the *5' end*, corresponding to the carbon atom labelling within the sugar. The two strands are oriented in opposite directions. In combination with the previously described complementary system of bases, one strand is referred to as being *reverse complementary* to the other. Due to the complementary nature of the strands, the information contained in the order of bases in the strand is encoded redundantly, a feature that is exploited both in DNA replication as well as in DNA repair mechanisms.

In most complex organisms, the genome comprises multiple double stranded DNA molecules, referred to as *chromosomes*. The set of chromosomes itself again occurs multiple times, a phenomenon known as *ploidy*. Mammals, including humans, are *diploid*, i.e. carry two sets of chromosomes - one maternal and one paternal set.

While more and more functional mechanisms of DNA molecules are discovered, a core feature of DNA molecules is their role in the biosynthesis of proteins.

1.1.2. Protein Biosynthesis

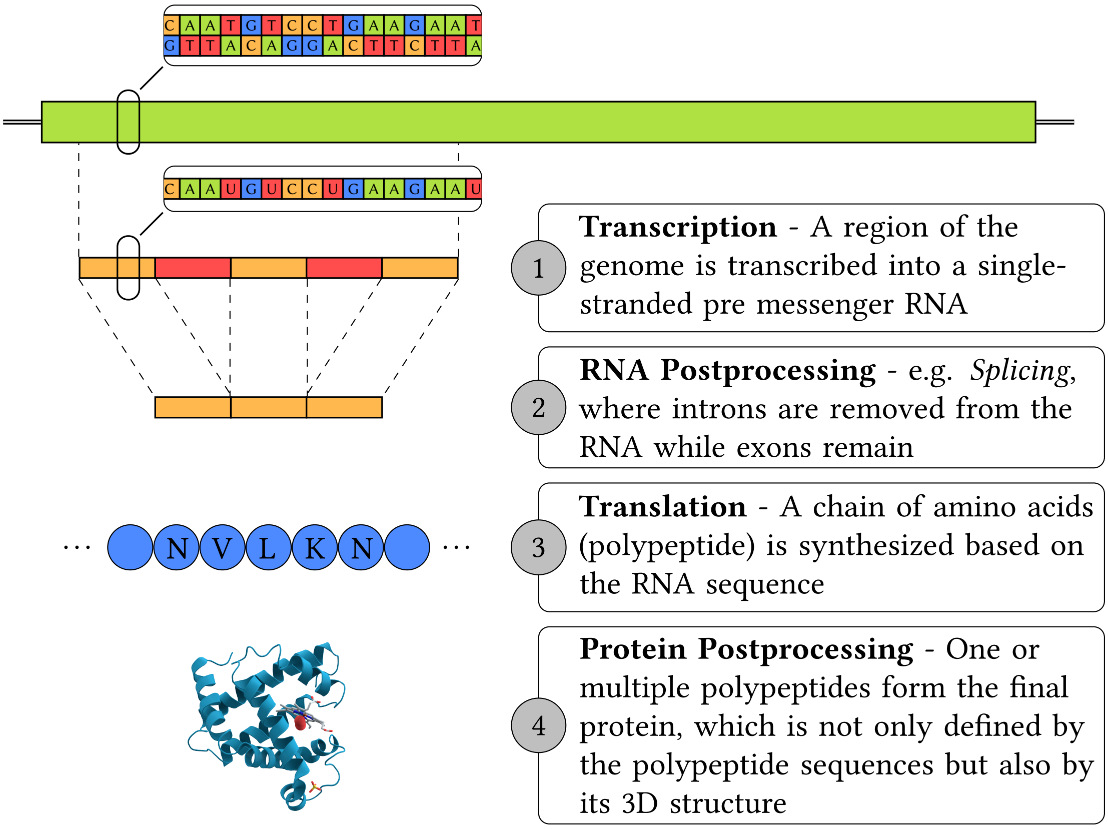
Proteins are macromolecules comprising one or multiple amino acid chains, also referred to as *polypeptides*. Cells can synthesize these molecules based on information contained in the base pair sequence of the DNA. In eukaryotes, the synthesis process comprises four main steps: transcription, RNA splicing, translation and protein post-processing.

Initially, a region of the genome encoding for a protein chain, a *gene*, is *transcribed*, i.e. copied into a single-stranded *ribonucleic acid* (RNA) molecule. RNA molecules differ from DNA molecules in two main aspects: the nucleotides contain ribose instead of deoxyribose and the base *uracil (U)* is synonymously used instead of thymine. While the genomic sequence of the DNA remains unmodified in this process, the RNA molecule acts as a temporary intermediate product in the biosynthesis process and is subject to modifications. Most importantly, in eukaryotes certain regions of the gene, so-called *introns*, are excised from the RNA molecule, while the rest of the gene sequence, known as *exons*, remain in the RNA - a process known as *splicing*. The RNA molecule is then *translated* into a polypeptide chain, according to the *genetic code* which maps triplets of nucleotides (*codons*) to single amino acids, e.g. "CAU" to histidine. Lastly, the polypeptide is post-processed into its final spacial conformation, i.e. *folded*, mostly in conjunction with other, independently synthesized polypeptides in order to form the final, functional protein. A schematic overview of the entire process is shown in Figure 1.1.

1.1.3. Genetic Variation

Variation in the genetic code occurs at various levels: across species, across individuals of the same species as well as across cells within the same individual. In the context of this thesis it is important to make a distinction in the latter case.

Generally, the mechanisms of cell proliferation, which include the replication of the cells DNA, aim for a perfect copy, i.e. all cells of the same individual carry a copy of the same genome. One exception are *gametes*, i.e. ova and sperm cells, that contain a single set of chromosomes, which was generated by pairwise *recombination* of both chromosome sets, creating a new chromosome set. Upon sexual reproduction, the chromosome sets of two gametes are joined and form the unique diploid genome of a new individual. This recombination process is *germline* specific, i.e. it does not occur in cells that do not develop into ova or sperm cells. Variation that occurs outside the



Transcription - A region of the genome is transcribed into a single-stranded pre messenger RNA

RNA Postprocessing - e.g. *Splicing*, where introns are removed from the RNA while exons remain

Translation - A chain of amino acids (polypeptide) is synthesized based on the RNA sequence

Protein Postprocessing - One or multiple polypeptides form the final protein, which is not only defined by the polypeptide sequences but also by its 3D structure

Figure 1.1.: A schematic overview of protein biosynthesis. (1) A region of the genome, typically a gene, is transcribed into RNA, which is then (2) spliced, i.e. its introns are excised. (3) The RNA sequence is then translated into an amino acid chain. (4) Multiple amino acid chains are combined and folded in order to form the final protein.

germline is referred to as *somatic* variation. It generally comprises accumulated, in most cases non-deleterious, mutations which develop due to environmental factors or errors during DNA replication. However, in the context of cells that are part of the immune system, another targeted recombination process has evolved that acts on the somatic and not on the germline level, leading to genetic variation that is inherited somatically, i.e. upon cell proliferation, but not to offspring. This mechanism will be discussed in more detail in Chapter 2.

1.2. Biotechnological Methods

Since the first major steps in understanding DNA were undertaken by Watson and Crick [2], a huge number of biochemical mechanisms related to DNA and its function have been discovered. Some of these mechanisms have been adapted to be applied in a controllable fashion under laboratory conditions. Two of these biotechnological methods stand out as they form the foundation for many other, more complex methods and thus have contributed substantially to genetic research: the artificial replication of DNA and DNA sequencing. As both of them play a major role in the methods described in this thesis and in the interpretation of the generated data, we will briefly describe them in the following sections.

1.2.1. Polymerase Chain Reaction

Polymerase chain reaction (PCR) [3] describes a group of protocols designed to massively amplify, i.e. replicate DNA molecules. They are based on the usage of *DNA polymerases*, proteins which occur naturally and are responsible for DNA replication. DNA polymerases can synthesize, more precisely fill gaps in a DNA strand in 5' to 3' direction, given a partially single stranded DNA molecule.

A typical PCR mix contains the original, *template DNA* molecules, the DNA polymerases, a sufficient amount of single nucleotides (dNTPs) as building blocks for the amplification process and *PCR primers*. PCR primers are short stretches of DNA, which have to be reverse complementary to a target region within the strands of the template DNA and can be produced artificially. All components reside in a buffer medium and the reaction is controlled through cyclic changes of the temperature.

The reaction comprises three main stages, which are controlled by applying different temperatures: At first, the double stranded DNA is *denatured* into single stranded DNA. In the second, *annealing* phase the primers bind to the reverse complementary sites in the templates. The temperature and primer concentration are chosen such that pairing occurs predominantly between primers and templates and not between the original template strand pairs. In the final phase, the complementary strand is *extended* by the DNA polymerases, starting from the bound primer. The temperature for the extension step is chosen according to the optimal range of the used polymerase. After the extension step we again have double stranded DNA molecules in the reaction mix and the process is repeated until the desired amplification is achieved. The process is illustrated in Figure 1.2. Since the number of DNA fragments is approximately doubled in every cycle, the PCR amplification performance is *exponential* in the number of cycles.

PCR has become an indispensable tool in most experimental methods involving DNA. It can be used to amplify short DNA fragments as well as longer fragments with sizes up to several kilobases (kb). It does, however, also introduce certain biases and errors, which can pose a difficulty in many applications. Most importantly, the polymerases introduce *sequence errors*, i.e. false base incorporations, at a certain rate. Due to the branching nature of the PCR process, these errors propagate further in subsequent PCR cycles, thus erroneous molecules can make up a significant portion of the final product. The issue can to some degree be addressed by the choice of the polymerase. *Taq Polymerase*, the first polymerase to be used for PCR, has an estimated error rate in the order of $\sim 10^{-5}$. There exist other PCR polymerases which feature *proof reading* activity and reach error rates in the order of $\sim 10^{-6}$ [4]. However,

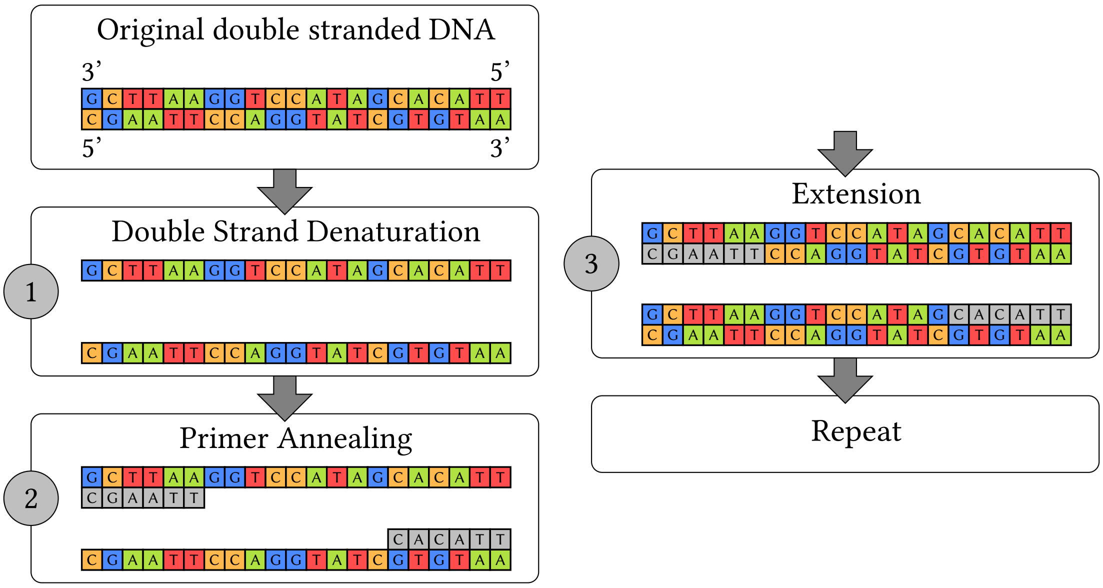


Figure 1.2.: The basic PCR steps. The originally double stranded DNA molecules are (1) denatured at a high temperature into single stranded DNA. (2) The forward and reverse primers then anneal at the target sites at a low temperature and (3) the DNA polymerase extends the second strand. The entire process is repeated multiple times.

polymerases with higher fidelities are often more difficult to handle in the reaction, making Taq polymerases a standard tool up until today.

Another type of error that is introduced by PCR particularly affects quantitative methods which take into account the ratio of different fragments. Since the amplification performance is not perfect, i.e. only a certain fraction of fragments is amplified in every step, the relative abundances of unique fragments after the amplification do not necessarily reflect the original abundances of templates. Additionally, biochemical properties of the templates such as the base sequence itself, the fragment length or the primer binding affinity can influence the amplification performance systematically. One approach to counteract such biases is the use of unique molecular identifiers, which will be described in the context of immune repertoire sequencing in Section 2.4.3.

1.2.2. DNA Sequencing

The term *DNA Sequencing* describes methods and technologies that aim to detect the base pair sequence of DNA molecules. Given a mixture of DNA molecules, the goal is to obtain a list of sequences, referred to as *reads*, over the alphabet $\Sigma_{\text{DNA}} = \{A, C, G, T\}$ which reflect the order of bases in the DNA molecules present in the sample.

Most sequencing methods are based on the extension of reverse complementary strands using DNA polymerases ("sequencing by synthesis"), similar to the PCR described in the previous section. Starting off with a high number of copies of a single

DNA molecule to be sequenced, early approaches [5] are based on mixing a certain fraction of terminating dNTPs for one single base, e.g. adenine, into the reaction. These modified dNTPs prevent the incorporation of further, subsequent nucleotides. Every incorporation of an adenine into a newly synthesized strand will then have a certain chance to lead to the termination of the elongation process at that position. Thus, the independent synthesis reactions lead to molecules of different lengths - each ending in the same base. The positions of that particular base can then be recovered based on the lengths of the fragments that were produced.

Second generation sequencing approaches, commonly referred to as *next generation sequencing* (NGS) or *high throughput sequencing* (HTS), which are still the most used sequencing technologies until today, are a further improvement of this approach and yield a significantly higher throughput at lower costs. While in the original approach, one DNA fragment was used to report one position in the sequence, NGS methods are able to detect subsequent base incorporations in various molecules in a highly parallel fashion. This is made possible by two main features: (1) The spatial immobilization of DNA molecules, which allows for the association of multiple base incorporation events to the same molecule and (2) an alternative solution to the permanent termination of strand elongation. Multiple such solutions have been developed and commercialized, the most prominent being the recently discontinued *454 Pyrosequencing* and the *Illumina sequencing* [6] platforms. The latter has been established as the de facto standard today due to its low sequence error rates.

The first step in the Illumina sequencing procedure is the *library preparation*. This primarily involves the attachment of *sequencing adapters* at each end of each single stranded DNA molecule, a specific 5' adapter and a specific 3' adapter. The actual sequencing reaction takes place on a glass surface, the *flow cell*, which was manufactured such that two types of short DNA oligonucleotides complementary to the adapters are uniformly distributed and fixated on the glass surface. The sequencing library is then added into the flow cell, where the single stranded DNA templates bind to the oligonucleotides on the flow cell surface. A DNA polymerase complements the missing strand, using the surface oligonucleotides as a primer, and the original template strand is denatured and washed away. Single stranded DNA molecules with the sequence of interest are now immobilized on the flow cell surface.

Next follows a step referred to as *bridge amplification*. The single stranded molecules are amplified in a PCR reaction, where previously unused oligonucleotides fixated on the flow cell serve as primers. These bind to the unbound adapters of the DNA strands, which then form a connection between two surface fixated oligonucleotides, hence the name bridge amplification. After the denaturation step the newly synthesized DNA strand is therefore already fixated to the flow cell surface. Since both template and product are spatially immobilized, the bridge amplification causes the formation of *clusters* of copies on the flow cell surface. After cleaving off one of the two types of complementary strands from the flow cell surface and washing them away, the clusters now consist of identical DNA molecules and the actual sequencing reaction can begin.

The sequence detection is performed by synthesis using primers that bind to the free ends of the DNA molecules. The four terminating dNTPs are added with four different fluorescent dyes. After one dNTP was incorporated at every DNA molecule, the dyes are excited by a light source and the signal is detected by a camera. The simultaneous emissions from an entire cluster amplify the signal sufficiently to be reliably detected, thus one cluster produces one read. The terminator is then washed away from the 5' ends of the currently synthesized strands and the procedure is repeated.

Like PCR, sequencing methods introduce a certain amount of errors, i.e. report false, missing or additional bases in the read sequences. On Illumina systems, the error rate increases with the read length and is additionally biased by the base sequence itself [7]. Over time, errors such as false, missing or multiple dNTP incorporations accumulate within each cluster. However, unlike PCR, most sequencers provide a *quality score* with each base they report, thus delivering a measure of confidence in the correctness of that particular base call which can be used in subsequent applications to quantify, filter or correct errors. On Illumina systems that score is based on the homogeneity of the signal emitted after every cycle.

Another quality related issue specific for Illumina systems is the proper detection of cluster boundaries. The design aims at a uniform spatial distribution of clusters across the flow cell surface, however, to a certain extent clusters can form in close proximity. The cluster detection happens in the first few cycles (4-5) [8], i.e. if clusters have emitted distinct signals in early cycles they can be separated by the Illumina software. If the DNA molecules are, however, identical at the 5' end, clusters in close proximity may be treated as one, resulting in erroneous reads. This is problematic in PCR amplicon sequencing applications, including the immune repertoire sequencing approach which we will describe in Section 2.4. These methods generally achieve lower sequencing qualities than applications with more diverse sequence content do.

2. Immunology and Immunogenetics

In this chapter, we will briefly introduce the immunobiological background of the topics and methods handled in the subsequent chapters of this thesis. The concepts described here apply to mechanisms known from *vertebrate species* unless stated otherwise. For more background on this subject, we refer the interested readers to "Janeway's Immunobiology" by Murphy and Weaver [9].

2.1. General Concepts

The term *immunology* describes the study of the body's protection mechanisms against pathogens - primarily viruses, bacteria, fungi and parasites. Generally, defense mechanisms can be divided in three types or stages: *anatomical barriers*, *innate immunity* and *adaptive immunity*. While anatomical barriers such as skin or mucosal tissue aim to prevent the pathogenic intrusion, the latter two mechanisms become active once a pathogen has successfully violated those boundaries. The types of countermeasures that are applied by those systems primarily depend on on the type of infection, e.g. intracellular pathogens such as viruses or small bacteria can be defeated by killing the affected (own) cells, while extracellular pathogens like large bacteria are isolated and digested.

2.2. Innate Immune System

In comparison with adaptive immunity, the components of the innate immune system are evolutionary older. They comprise specialized cells and mechanisms that act against infection in a *generalized* way, i.e. without being specific for one particular pathogen but by recognition of so-called *inflammatory inducers*. These are markers for the presence of a pathogen, e.g. extracellular ATP as a marker for bacterial activity. Once cells that are capable of detecting inflammatory inducers pick up such a marker, they signal other cells which then act situation dependent, e.g. by triggering the synthesis of antiviral proteins or by ingesting and killing microbes.

The innate immune system is a *fast* responder that is able to act within minutes after an infection. It is, however, limited due to its fixed response triggers. The high mutation rate in many pathogenic species constantly leads to the evolution of new, adapted pathogens that are able to remain unseen by the innate immune system.

2.3. Adaptive Immune System

The key mechanism that has evolved to supplement the innate immune system is the *adaptive immune system*. While innate immune mechanisms can also be found in plants, fungi and many other organisms, the adaptive immune system can only be found in *vertebrate species*. Instead of being an entirely independent component, it both integrates into the innate mechanisms, e.g. by making pathogens visible to the innate system through signalling, but also adds its own effector mechanisms. Cells of the adaptive system are able to recognize pathogenic molecules, primarily proteins and polysaccharides, referred to as *antigens*. In contrast to innate mechanisms, they do not rely on a limited number of specific features of those molecules, but are able to adapt to new, previously unseen substances. The adaptive mechanisms respond substantially slower than the innate mechanisms, with response times in the range of hours to days. However, the adaptive immune system is able to *memorize* pathogenic encounters, leading to a more efficient response at future infections known as *immunity*.

2.3.1. T and B Lymphocytes

Two types of lymphocytes form the key components of the adaptive immune system: T and B cells. They express a membrane protein, the *antigen receptor*, which in both cases comprises two kinds of polypeptide chains: the receptor of the predominant type of T cells contains an α (TRA) and a β (TRB) chain and the B cell receptor, in its basic form, contains one identical pair of *heavy* (IGH) and one identical pair of *light* (IGL) chains. The antigen receptor is responsible for the recognition of antigens and the transmission of an activation signal to the cell upon interaction with a matching ligand. Instead of capturing an antigen as a whole, antigen receptors recognize small substructures of antigens, so-called *epitopes*. In a simplified sense, every antigen receptor is specific for only one antigen epitope and each B or T cell expresses only a single antigen receptor. However, different cells express different antigen receptors, a mechanism made possible by the somatic recombination of the gene encoding the receptor. The recombination occurs early after the differentiation of a stem cell into the T or B cell lineage and will be described in detail in Section 2.3.3.

While a cell is, in most cases, initially unique with respect to its antigen receptor, as soon as it encounters a matching antigen it starts to proliferate and transmits its recombined receptor gene to the daughter cells. Through the proliferation of active cells the response to antigens is boosted for the time of the infection. Furthermore, after the pathogen is successfully defeated, the subpopulation of antigen specific cells reduces again, however, not down to a single cell but rather to a population of *memory* cells. That mechanism ensures that future encounters with the same antigen will be responded to more rapidly and more efficiently.

While T and B cells are evolutionary closely related [10], there are some major differences regarding their function and their way of recognizing antigens. The *T cell receptor* (TCR) is always membrane bound, whereas B cell receptors can be secreted into the extracellular space and are then referred to as *immunoglobulin*s (IGs) or *antibodies*. Since it has been established as the standard terminology, we will refer to B cell receptors as immunoglobulins throughout this thesis, unless we want to explicitly distinguish the membrane bound from the secreted state.

Another fundamental difference between the TCRs and IGs is the way they bind antigen epitopes: while immunoglobulins can recognize epitopes that are still bound in larger, complex molecules, T cell receptor epitopes have to be preprocessed, i.e. excised from the original molecules and loaded onto a presenting membrane protein for recognition. The details are briefly described in the following section.

2.3.2. T Cell Antigen Recognition

In order for a T cell receptor to recognize an antigen epitope, the epitope has to be mounted onto an *MHC molecule*, a group of proteins encoded in the eponymous region of the genome referred to as the *major histocompatibility complex*. In humans, the term *human leucocyte antigen* (HLA) is used equivalently. The ligand of a TCR is therefore not a single antigen epitope, but the complex of the MHC molecule and the epitope, known as the *peptide MHC complex* (pMHC).

There are two main types of MHC molecules, *class I* and *class II*. The MHC is polygenic and highly polymorphic, i.e. each MHC molecule class is encoded by multiple genes and for each gene there is a high number of variants within a species' population. For example, there are three genes encoding the MHC-I molecule, thus a cell expressing MHC-I expresses up to six different variants of the molecule. While MHC-I is expressed by all nucleated cells, MHC-II is expressed only by certain immune cells specialized in the absorption, digestion and presentation of antigens. Corresponding to the two subtypes of MHC molecules, there exist two subtypes of T cells: those that express the TCR co-receptor CD4 and those that express the co-receptor CD8. CD8 T cells bind MHC-I molecules and are known as *cytotoxic T cells* based on their primary function to kill cells that show signs of infection, e.g. present virus epitopes on their MHC-I molecules. CD4 T cells on the other hand bind MHC-II molecules and are known as *helper T cells*. Their primary function is to mediate an immune response by secreting signalling molecules called *cytokines*, which guide other immune cells to the site of infection.

A schematic view of the presentation of peptides and the interaction of T cells with the pMHC complex is shown in Figure 2.1. The target antigen, in this case a protein, is digested inside the presenting cell and the peptides are loaded onto MHC molecules. The pMHC complex is then embedded into the cell membrane, presenting the peptide

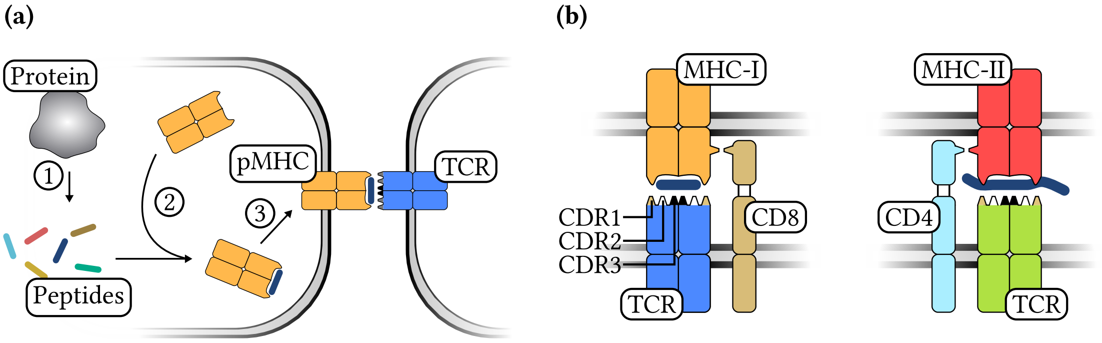


Figure 2.1.: (a) A schematic view of how antigen epitopes are presented to the TCR. (1) The presenting cell digests larger proteins to smaller peptides. (2) The peptides are loaded into an empty MHC molecule right after its biosynthesis, which is then (3) embedded into the membrane presenting the peptide to the extracellular space. **(b)** A detailed view of the TCR-pMHC interaction. The CDRs of each TCR chain point towards the pMHC complex. The CDR3 interacts primarily with the peptide, while the germline encoded CDR1 and CDR2 interact primarily with the MHC. MHC-I molecules enclose both ends of the peptide, present peptides of a fixed length and bind T cells expressing the CD8 co-receptor. MHC-II molecules can bind T cells expressing the CD4 co-receptor and can load longer peptides with overhangs outside the binding cleft.

to the extracellular space. Depending on the class of the MHC molecule, either CD4 or a CD8 T cell that matches the pMHC can bind to the complex. Within each T cell receptor chain, the so-called *complementary determining regions* (CDRs) face the pMHC. The CDRs show a higher degree of variability across different TCRs, while the other regions of the protein are relatively conserved. Both in IGs and TCRs the CDRs are what is primarily in contact with the ligand, where in T cells the CDR3 interacts with the center of the peptide and the CDR1 and CDR2 interact with the MHC and the peptide termini [11, 12].

While MHC-I molecules are known only to present protein peptides, the MHC-II pathway can also process polysaccharides to low molecular weight carbohydrates and present them outside the cell [13].

2.3.3. V(D)J Recombination

In the early 1980s, Tonegawa [14] and his team successfully deciphered the genetic mechanism behind the diversity observed in the antigen receptors of different cells. They discovered that the high number of antigen receptors was not (solely) based on a high number of encoding genes in the germline genome, but that instead a *somatic modification* of the gene loci takes place upon the differentiation of T and B cells. In 1990, Tonegawa was awarded with the Nobel Prize in Physiology or Medicine for his contribution to this discovery.

The genes of T and B cell receptor chain genes contain a variable region, referred

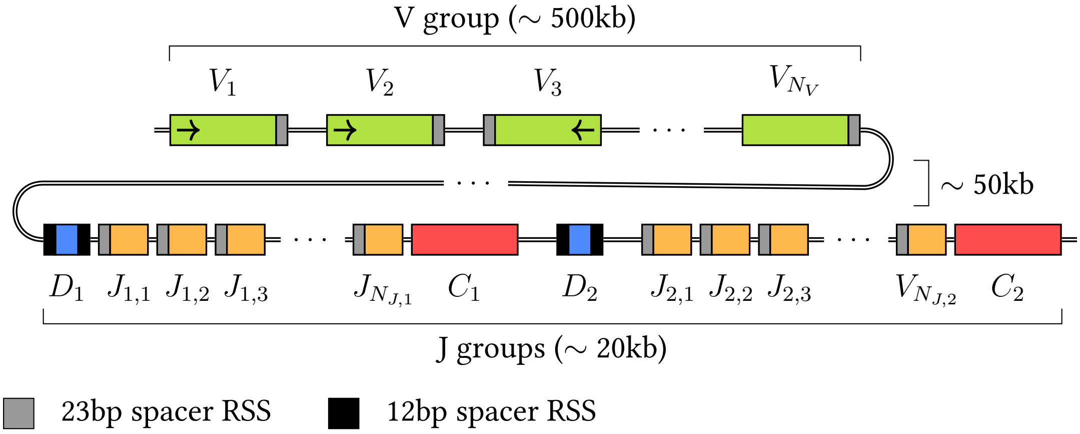


Figure 2.2.: A schematic view of the human T cell receptor β chain germline locus before recombination. The locus comprises $N_V = 65$ V gene segments and $N_J = 15$ J gene segments, the latter being separated into two groups, each lead by a single D gene segment. There are recombination signal sequences (*recombination signal sequences*s (RSSs)) with either 12bp or 23bp spacers flanking each segment at the potential junction site(s). Gene segments occur in both orientations.

to as the *V region*. In cells other than T or B cells, that region is not present in a functional configuration. Instead, the loci comprise sets of building blocks or templates for the V region, so-called *gene segments*. There are three types of gene segments: *variable (V)*, *diversity (D)* and *joining (J)*. [1] A functional antigen receptor gene V region always comprises exactly one V and one J segment in exactly this order. For the B cell heavy chains as well as for T cell β chains, one (or in rare cases two) D gene segments occur between the V and J segment. In the recombined locus, the V region is followed by an intron and subsequently the *constant (C) region*, which does not undergo any modifications. The diversity of antigen receptor genes arises from the fact that multiple variants of V, D and J segments are encoded in the germline genome, thus part of the variability is accounted for by the combinatorial possibilities of selecting one of each segment type. This variability is further enhanced by the fact that during the recombination of the locus additional modifications of the nucleotide sequence occur at the junction sites at which two gene segments are joined.

The overall architecture of an antigen receptor gene locus in the unmodified, germline genome is shown in Figure 2.2 which shows the human T cell receptor β chain locus. It comprises $\sim$ 75 V gene segments spanning a range of $\sim$ 500kb, $\sim$ 15 J gene segments, 2 D and 2 C gene segments which are organized in two groups and span $\sim$ 20 kb together [15]. The other antigen receptor gene loci are organized in a slightly different way, but the general mechanism of the recombination process described in the following section applies to all loci, with the only difference being that the D segment is not incorporated into the gene loci that encode the smaller chains (T

[1] Note the potentially confusing choice of nomenclature regarding the character *V*: a V *segment* is one of three building blocks for a V *region*. See Figure 2.5.

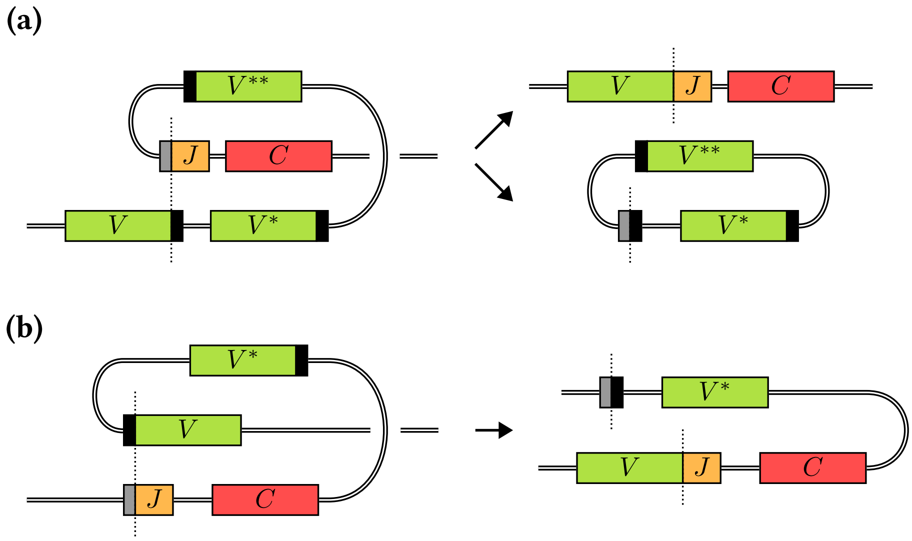


Figure 2.3.: During the V(D)J recombination, the gene segments are brought together by cut and join operations performed on the gene locus. **(a)** If the gene segments are in the same orientation, they are brought together through the removal of the intermediate DNA, leading to the formation of excision circles. **(b)** If the target segments are not in the same orientation, a inversion is performed instead, with no DNA being removed. Adapted, based on Lefranc and Lefranc [15].

cell receptor $alpha$ chain, immunoglobulin light chain).

Every gene segment is flanked by *recombination signal sequences* (RSS) at their termini that can form a junction to another gene segment. They are recognized by the proteins involved in the recombination process, which then bring the two RSSs and therefore the two gene segments together to initiate the formation of a junction. The RSSs begin and end with well conserved motifs which are separated either by a 12bp or a 23bp spacer sequence. Junctions can only be formed between a site flanked by a 12bp spacer RSS and a second site flanked by a 23bp spacer RSS. The two different RSS types ensure that only V-D and D-J junctions are formed in those genes that contain a D segment, that only V-J junctions are formed in the other genes and that the gene segments are joined in the proper orientation.

The alteration into a locus that comprises single V, (D) and J segments consecutively is performed by cut and join operations. In most cases, the gene segments occur in the same orientation on the genome [15], in which case two segments are joined by excision of the intermediate DNA as shown in Figure 2.3(a). The excised DNA is joined as well and therefore forms circles, known as *excision circles*. Since the circular DNA is not replicated upon cell division, the presence of such molecules can be used as a marker for T cells that have recently emigrated from the thymus [9]. In a minority of cases, the segments are in opposite orientation on the chromosome, in which case two such segments are joined by inversion as shown in Figure 2.3(b). In this case, no genetic material is removed from the genome, i.e. no excision circles are being generated.

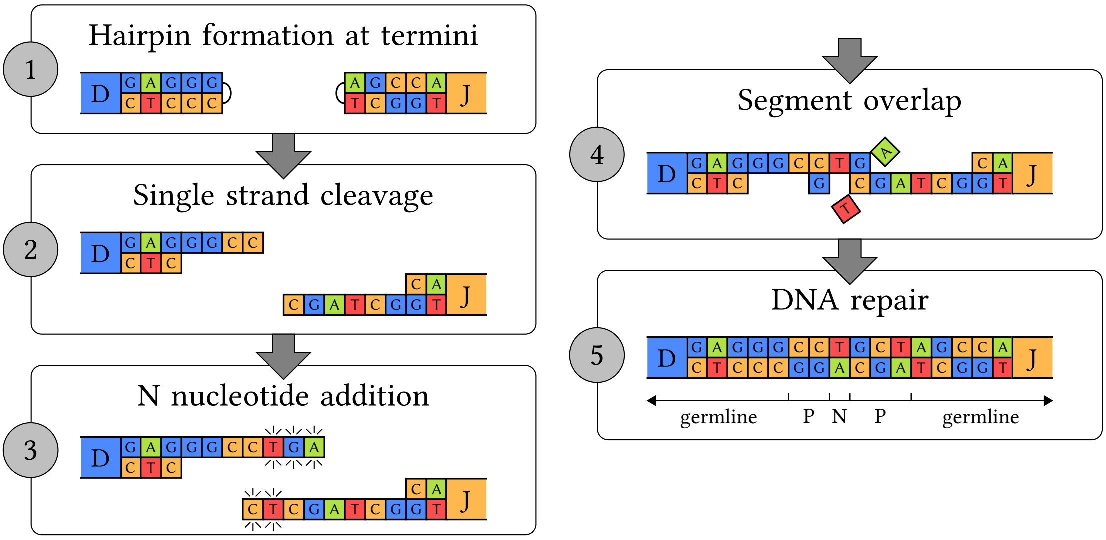


Figure 2.4.: The junction formation using the example of the DJ junction. **(1)** Hairpin formation at the blunt ends of the gene segments. **(2)** Single strand cleavage potentially causing overhang. **(3)** Addition of random nucleotides at the overhang. **(4)** Pairing of the strands. **(5)** The junction is completed by means of DNA repair mechanisms.

The diversity of genes for the same type of antigen receptor is however much larger than the possible number of combinations of gene segments. As previously mentioned, it is additionally boosted when the junctions between gene segments are formed. Instead of just being joined, the ends of the gene segments are modified on the single nucleotide level when the junction is formed. The process of junction formation is illustrated in Figure 2.4. Initially, after the double stranded DNA was cut open precisely at the site between the RSS and the coding gene segment, a hairpin is formed at the blunt end of the double strand. Next, the hairpin is opened by induction of a single-strand cut whose position is variable, resulting in a *palindromic overhang*, since the now consecutive nucleotides were formerly complementary. The nucleotides forming the palindromic overhang are referred to as *P nucleotides*. Subsequently, both endonucleases and exonucleases act on the single-stranded overhang, inducing *random* removals and additions of single nucleotides. If additional nucleotides remain after this process, they are referred to as *N nucleotides*. Lastly, the modified overhangs of the two segments are overlapped and the junction is completed by means of DNA repair of missing and mismatching nucleotides.

Due to the random nature of the recombination, the efficiency of the process itself is relatively low. In fact, the majority of the recombination events leads to non-functional genes, mostly because the random nucleotide insertions and deletions induce frame shifts or stop codons [9]. Furthermore, many of the germline encoded gene segments are in fact pseudo gene segments, meaning they are non-functional, which additionally decreases recombination efficiency.

The recombination of the longer heavy and β chain loci occurs before the recom-

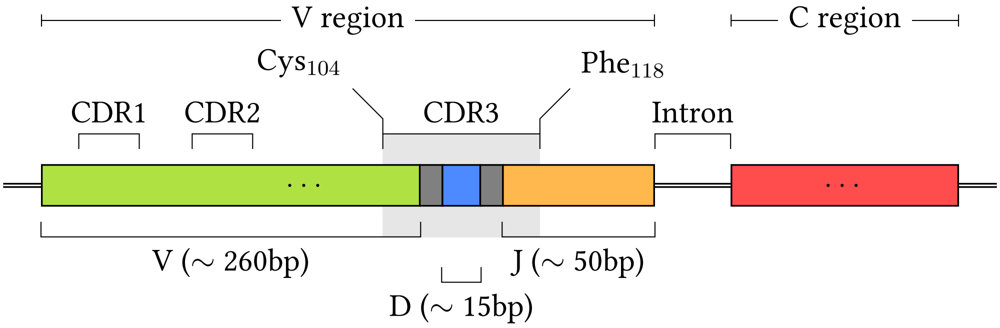


Figure 2.5.: The schema of the mature, recombined human TRB locus. The V, D and J gene segments are joined with additional P and N nucleotides (gray) at the junction sites. The CDR1 and CDR2 are enclosed in the entirely germline encoded V gene segment, while the CDR3 (∼ 40bp) comprises the junction sites and thus random, not germline encoded nucleotides. The C region remains separated from the V region by an intron.

bination of the shorter light and α chain loci. An evaluation mechanism validates whether the recombination was successful, i.e. whether e.g. a β chain is expressed. If the recombinations fails, it may be repeated on the same locus if sufficient material is still available in the genome or on the second chromosome. Only once the first chain is expressed is the recombination of the second chain locus initiated.

An overview of a recombined human TRB gene locus is shown in Figure 2.5. The CDR1 (∼ 15bp) and CDR2 (∼ 18bp) are encoded in the germline V gene segments and their positions defined according to the IMGT numbering scheme [16]. The CDR3 on the other hand comprises the segment junctions and therefore nucleotide sequences that are not germline encoded. It is defined as the region between the conserved Cys$_{104}$ encoding triplet in the V gene segment and the conserved P$_{118}$ encoding triplet in the J gene segment, again, following the IMGT numbering scheme.

2.3.4. Additional Recombination in B Cells

In T cells, the TCR locus is not further modified following V(D)J recombination during thymic development. In mature B cells however, there are two mechanisms that continue to alter the antigen receptor gene loci.

The first mechanism is known as *class switching*. There are five main classes of immunoglobulins, IgM, IgD, IgG, IgE and IgA. The class is determined by alternative C regions: while the switch between IgM and IgD is controlled by alternatively splicing the corresponding C region to the V region, the switch to the other IG classes requires another somatic recombination of the gene loci, which happens long after the V(D)J recombination when the B cell was stimulated by a matching antigen.

The second mechanism of late somatic recombination are *somatic hypermutations*

(SHMs), which describes point mutations that are induced in the V region upon cell division. The mutations occur at a rate between 10^{-5} and 10^{-3} [17], accumulating with each generation. This mechanism adds another layer of adaption, which can further optimize the antigen affinity by selection. While this thesis focusses primarily on T cells, SHMs will to some degree become relevant in Chapter 4 where we will discuss a method to annotate recombined V regions, since they are hard to distinguish from technical artifacts.

2.3.5. Allelic Exclusion

Generally, the concept of "one cell - one receptor" is crucial to the idea of an effective and efficient adaptive immune system. If a cell was specific for multiple antigens simultaneously, activation by one antigen and subsequent proliferation of that cell would boost the immune response against other antigens as well, with potentially unwanted side effects. Such a situation could arise if both alleles of one antigen receptor chain were recombined into functional genes. To prevent such a situation, various mechanisms ensure that the V(D)J recombination is limited to one chromosome and locus at a time and immediately suppressed if a recombination is successful. This process is known as *allelic exclusion* and is successfully achieved in the large majority of cells [18]. There are, however, exceptions, such as the T cell receptor α chain alleles which are known to be recombined simultaneously and the recombination machinery is deactivated only relatively late during early cell development, thus a fraction of T cells break with the one receptor rule and express two receptors that differ in their α chain [19].

2.3.6. Antigen Receptor Affinity Assessment

Once the cell expresses an antigen receptor on the surface, it undergoes a process known as *positive* and *negative selection*. Basically, the antigen receptor has to show a minimum degree of activation, i.e. binding affinity when exposed to a variety of antigens (positive selection). Additionally, the affinity to self-antigens must no be too high, i.e. recombinations that are likely to contribute to auto-immune responses rather than to respond to foreign antigens are negatively selected. Only cells that pass both selection stages emigrate from the thymus or bone marrow into the periphery.

Since T cells do not bind antigens directly, but require epitopes to be presented by MHC molecules, they are exposed to pMHC ligands during their thymic development. For them the binding affinity assessment furthermore includes the distinction between MHC-I and MHC-II specificity. T cells start off as double positives - depending on which ligand the newly generated antigen receptor binds best, the cell will loose either the CD4 or the CD8 TCR cofactor and thus become either a cytotoxic cell or a cytokine-secreting helper cell respectively.

2.4. Measuring TCR / IG Repertoires

Repertoire Sequencing (Rep-Seq) describes methods that aim to capture the repertoire of B or T cells quantitatively based on their antigen specificities using high throughput sequencing. We refer to the somatically recombined configurations of the antigen receptor genes that define the cells antigen specificity as the cell's *clonotype*. Thus, the goal of Rep-Seq can be defined as determining the clonotypes and their abundances given a sample of cells. Due to the separate loci of the TRA and TRB (or IGH and IGL) genes, it is technically difficult to detect both genes in a combined fashion. It is therefore established practice to capture only one of the two genes. Since the TRB and IGH genes feature the larger variability in comparison to their counterparts, these genes are typically selected for sequencing. Throughout this thesis, we will use the term Rep-Seq to refer to single chain immune repertoire sequencing and the term clonotype to describe the antigen specificity as defined based on a single chain. We will, however, provide an outlook on the currently emerging technologies regarding the simultaneous capture of both antigen receptor chain genes in Section 8.3.1.

The basic principles of single chain Rep-Seq protocols and differences in common approaches are described in the following sections.

2.4.1. Cell Characterization

Depending on the experimental goal, it is often desired to isolate cells of different subpopulations according to functional features of interest. This is typically achieved through flow cytometric methods such as *fluorescence-activated cell sorting (FACS)*, which is a well established tool to quantify and separate cells based on markers on their surface, usually membrane proteins [20] (*immunophenotyping*). Typical markers are those that uniquely correspond to T cell subtypes such as CD4 (helper T cells), CD8 (cytotoxic T cells), CD45RO (memory T cells), or activation markers like Interferon gamma (IFNγ), which can be used to select *antigen specific* T cells after exposing them to a particular antigen in culture.

2.4.2. Gene Enrichment

Once the target population of cells is defined and prepared, the initial step before sequencing is a *targeted enrichment* of the gene of interest, e.g. TRB. The enrichment is achieved through a PCR using primers that specifically bind at sites close to the perimeters of the region of interest, as one typically wants to recover as much of the V region as possible. Approaches based both on genomic DNA and on RNA have been proposed.

DNA based Rep-Seq enrichment [21, 22] requires a *multiplex PCR*, i.e. multiple forward and reverse primers. This is due to the fact that even for the germline encoded part of the V region, namely the V and the J gene segment, there are multiple variants that may occur in the recombined gene. The sets of forward and reverse primers is typically not as large as the set of V and J gene segments, since some of them share common conserved sequences that can be used as a primer binding site. Multiplex PCRs come with additional sources of errors compared to simplex PCRs. Apart from general issues that arise from the reaction being more difficult to control, e.g. due to increased hybridization between primers, there are also two main issues that specifically affect Rep-Seq. Firstly, while every PCR with mixed templates will be skewed to some degree after the amplification [23], this effect is further enhanced in a multiplex setting, due to some primers performing better than others. In a quantitative application such as Rep-Seq this can drastically reduce the data quality, since the measured proportions do not reflect the biological truth. Secondly, primers may be cross reactive, e.g. the primer designed for one J gene segment might bind to a gene that incorporated another J gene segment. The product will then be a hybrid of J gene segments, since the primer itself is incorporated during the amplification and further cycles will further amplify such products as they are. If the primer binding site is considered in the subsequent analysis steps, this can lead to false gene segment annotations.

RNA based Rep-Seq [24] can be used to overcome the limitations of the multiplex PCR: after splicing, the well conserved C region is directly adjacent to the J gene segment, therefore allowing a single reverse primer. Moreover, instead of using a multiplex forward primer set to target the V gene segment, one can utilize the *template switching* [25] protocol. The template switching protocol makes use of specific reverse transcriptases which add a few non-template C-nucleotides to the DNA product after reaching the 5' end of the template RNA. A previously added 5' adapter with a poly-G tail can then bind to the product currently being synthesized, leading to a continued transcription against the adapter. The adapter sequence is subsequently used as a forward primer binding site in the PCR amplification.

The RNA-based protocol reduces the amplification biases induced by the heterogeneous amplification performances within the primer sets, however, adds complexity over the DNA based approach. Furthermore, using RNA as the original material again introduces a certain degree of quantitative skewing due to the potentially different expression levels of the antigen receptor gene across the cell sample. Depending on the C region primer site, part of the read length might be lost due to the remaining C region part of the amplicon. In the case of IG repertoire sequencing this might however be intended, if the B cell isotype is to be determined.

2.4.3. Unique Molecular Identifiers

An experimental effort to overcome or at least contain two main issues that arise during the gene enrichment stage, i.e. the induction of PCR errors and quantitative skewing, is the usage of short stretches of random nucleotides, which are introduced into the target fragments as early in the amplification process as possible [26, 27]. They are often referred to as *unique molecular identifier*s (UMIs), and have been used in various quantitative NGS applications [28, 29, 30, 31]. The random nucleotide sequence is used as a barcode for a particular template, i.e. every subsequent copy will carry the same barcode. Fragments with PCR errors can then be identified based on the observation that they carry the same barcode as a more abundant, highly similar sequence. A quantitative correction can be performed by counting unique barcodes instead of sequences.

In Rep-Seq protocols, UMIs can be introduced as part of the template switch fragments [24] or the target primers [32].

2.4.4. Sequencing

After the target fragments have been enriched, sequencing adapters are appended. Since the ends of the fragments are known from the previously used primers, the sequencing adapters can be incorporated in another PCR step with primers carrying an adapter tail instead of using a ligation step, leading to a fixed orientation of the sequencing reads. The region of the highest interest is the CDR3, thus when performing *single end* sequencing the reads are generated from the J/C end of the fragments, since the V segment is substantially longer and the CDR3 region would not be reached with common read lengths. If both ends of the amplified fragments are sequenced in a *paired end* configuration, the second read will in most setups contain only V gene segment sequence. Note that under error-free conditions, due to the protocol being fragmentation free and based on well defined primer locations, two genetically identical T or B cells should generate the exact same read (pair)s. The most common read configurations are illustrated in Figure 2.6. We will refer to the reverse read that covers the target fragment from the J/C end as the *V(D)J read* and to the optional forward read that covers the additional V gene segment information as the *V read*.

For T cell Rep-Seq, it is sufficient to cover as much of the V and J gene segments as necessary for a unique identification against the germline reference. In IG Rep-Seq, if somatic hypermutations are of interest, it is desirable to cover the germline encoded parts of the V region as well. Generally, the Illumina sequencing platform has been established as the method of choice for Rep-Seq, due to its low error rate in comparison with other available methods. It has been used by far in most of the TCR studies [33, 34, 22, 35], while for IG Rep-Seq sometimes 454 sequencing has been the preferred method [36, 37, 38]. This is not only due to the fact that the study of SHMs requires coverage of the germline region, but also because the CDR3 of IG heavy chain genes is often longer than that of TCR β chains. However, with read lengths becoming longer with newer releases of Illumina sequencing kits, this argument is becoming less relevant.

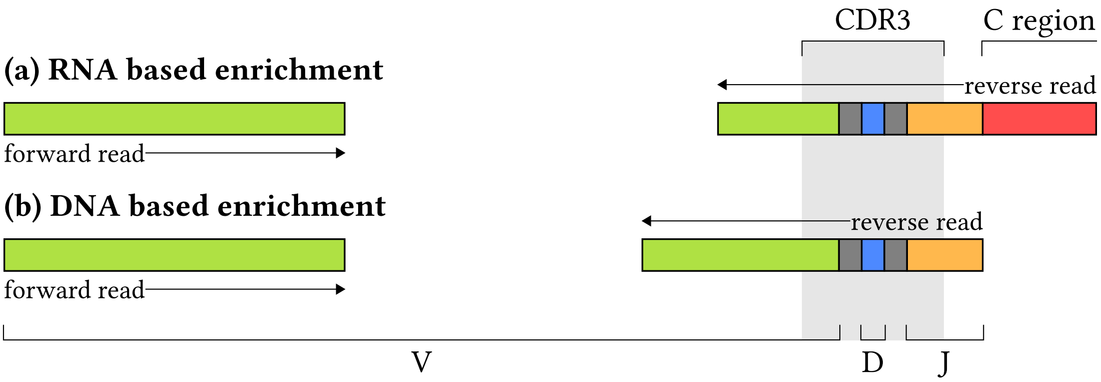


Figure 2.6.: The most common read configurations in targeted Rep-Seq experiments. The reverse read generally covers the CDR3 region with some extent of non-CDR3 V and J gene segment sequence. **(a)** In RNA based protocols with C segment primers, a fraction of the C fragment and the entire J gene segment is included in the reverse read. **(b)** In DNA based approaches using J primers, the read begins inside the J gene segment and a larger amount of V gene segment is included. In both cases, an optional second read (paired end sequencing) can additionally cover an upstream region of the V region, usually only parts of the V gene segment.

3. Preliminaries

In this chapter, we will define basic notations used throughout the thesis. Furthermore, we will introduce some well known sequence comparison problems and their algorithmic solutions, as they form the foundations for the methods described in the subsequent chapters. We will also define the notations we formally use to describe clonotypes and clonotype repertoires as previously described.

3.1. General Notations

3.1.1. Indicator Function

We will use $\mathbb{1}$ to denote an *indicator function* for boolean terms. Given the boolean term T, it is defined as

$$\mathbb{1}(T) := \begin{cases} 1 & \text{if } T \text{ is } \textit{true} \\ 0 & \text{if } T \text{ is } \textit{false}. \end{cases}$$

3.2. String Notations

We define a *string* to be a sequence of characters from a finite, non-empty alphabet Σ. The most relevant alphabets in computational biology are the DNA alphabet $\Sigma_{\text{DNA}} = \{A, C, G, T\}$ of size $|\Sigma_{\text{DNA}}| = 4$ and the canonical amino acid alphabet $\Sigma_{\text{AA}} = \{A, C, ..., Y\}$ of the $|\Sigma_{\text{AA}}| = 20$ amino acids that are encoded by the standard genetic code.

We furthermore define Σ^q to be the set of all $|\Sigma^q| = |\Sigma|^q$ strings of length q over the alphabet Σ. Such strings are also referred to as *q-grams* (equivalent terms found in literature are *k-mer*, which is also interchangeably used in this thesis, and *k-tuple*). Additionally, $\Sigma^* = \bigcup_{i \in \mathbb{N}^0} \Sigma^i$ denotes the set of all finite words over the alphabet Σ.

3.2.1. Substrings

In the following we will use a one-based indexing to describe *substrings* of strings. In particular, given a string $s \in \Sigma^L$ and two integers $i, j \in \{0, \ldots, L\}$ and $i < j$,

- $s[0]$ denotes the empty word ε,
- $s[i] \in \Sigma$ denotes the ith character in s,
- $s[i \dots j]$ denotes the *substring* of s starting at position i and ending at position j in s.

3.3. String Comparison and Sequence Alignments

A common problem when working with biological sequences is the *approximate string matching problem*, i.e. the comparison of sequences based on a similarity measure rather than identity. When comparing the sequences of genes or proteins, either against each other or against a reference database, both biological diversity as well as technical errors make exact string matching techniques not applicable.

In this section we will first discuss a common measurement to define the *similarity* of two sequences and furthermore describe algorithms in order to compute such similarities. The nomenclature used in this section is closely derived from Gusfield [39].

3.3.1. Edit Distance

One of the oldest and most important similarity measures for sequences is the *Edit Distance* or *Levenshtein Distance* [40]. It is based on the idea of a virtual transformation of one sequence into another, using a set of predefined operations.

Definition 3.1. An *edit transcript* representing the transformation of a string s_1 into another string s_2 is a sequence over the alphabet $\Sigma_T = \{I, D, S, M\}$, denoting character *insertions*, *deletions*, *substitutions* and *matches* respectively.

Note that the edit transcript for a given pair of strings is not necessarily unique.

Definition 3.2. Given two strings s_1 and s_2, the *edit distance* $\mathrm{dist}_E(s_1, s_2) = \mathrm{dist}_E(s_2, s_1)$ is given by the minimal number of edit operations $\{I, D, S\} \subset \Sigma_T$ required to perform a transformation of s_1 into s_2.

The *optimal edit transcript* corresponding to the edit distance of two sequences is not necessarily unique, however, their number is finite. An example for an edit transcript is shown in Figure 3.1(a).

3.3.2. Alignments

Another, more common way to represent a pairwise sequence transformation is a *sequence alignment.* It is an explicit representation of the character-wise relation of two sequences.

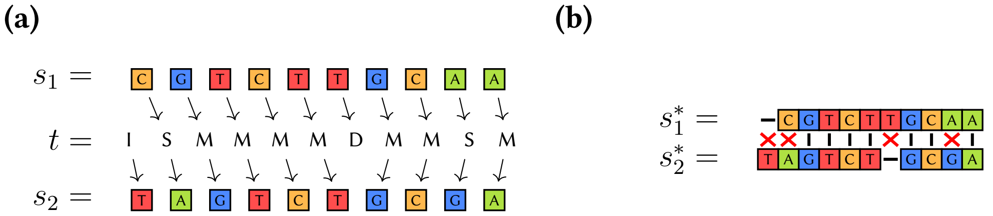


Figure 3.1.: (a) An example edit transcript t representing the transformation of s_1 to s_2. The characters corresponding to each operation are referenced by arrows. **(b)** The alignment representation equivalent to the edit transcript shown before. Matches are indicated with vertical bars, gaps with horizontal bars. Gaps and mismatches are highlighted with a red cross.

Definition 3.3. A (global) sequence alignment of two strings $s_1, s_2 \in \Sigma$ is a $2 \times L^*$ matrix, where the rows of the matrix contain versions of s_1 and s_2 interleaved with a *gap character* "-", i.e. $s_1^*, s_2^* \in \Sigma \cup \{\text{-}\}$ with $|s_1^*| = |s_2^*| = L^*$. The placement of the gap characters is constrained, such that no column in the alignment contains two gap characters.

Edit transcripts and alignments can be used equivalently to describe pairwise sequence transformations. In an alignment, columns without gap characters represent either matches, i.e. $s_1^*[i] = s_2^*[i]$, or mismatches, i.e. $s_1^*[i] \neq s_2^*$. Columns with a gap character in s_1^* describe an insertion and columns with a gap character in s_2 a deletion. The sequence alignment for the pair of sequences previously shown is depicted in Figure 3.1(b).

In contrast to edit transcripts, alignments directly reflect the symmetrical nature of sequence transformations, since insertions and deletions are equivalently denoted. An edit-distance based alignment between two strings s_1 and s_2 is *optimal*, if the corresponding edit transcript is optimal. Given such an optimal alignment $A = \{s_1^*, s_2^*\}$, we can derive the edit distance using

$$\text{dist}_E(s_1, s_2) = \sum_{i=1}^{L^*} \mathbb{1}(s_1^*[i] \neq s_2^*[i]).$$

3.3.3. Computing Sequence Similarity

Given this definition of a pairwise sequence similarity, we will now show how to obtain this similarity measure in a generalized form, based on a recursive description [41] of the cost function.

Definition 3.4. Given two sequences s_1 and s_2 of lengths $|s_1| = n$ and $|s_2| = m$, the $(n+1) \times (m+1)$ *edit matrix* D is defined s.t. $D(i, j)$ denotes the edit distance of the prefixes $s_1[0, \ldots, i]$ and $s_2[0, \ldots, j]$, where $s[0]$ denotes the empty string.

We will now describe how to obtain the edit matrix D, using *generic costs* for each edit operation, i.e. $\delta_G \in \mathbb{Z}$ for insertions and deletions (gaps) and $\delta_S : (x, y) \to \mathbb{Z}$ with $x, y \in \Sigma$ for substitutions and matches. The equivalent for the edit distance as described in Section 3.3.1 would be $\delta_G = 1$ and $\delta_S(x, y) := \mathbb{1}(x \neq y)$.

To obtain a prefix of length n from the empty string ε we have to insert n gaps, i.e.

$$D(i, 0) = i \cdot \delta_G \tag{3.1}$$

$$D(0, j) = j \cdot \delta_G. \tag{3.2}$$

The remaining entries of the matrix are then defined by the following recurrence:

$$D(i, j) = \min \begin{cases} D(i-1, j) + \delta_G \\ D(i, j-1) + \delta_G \\ D(i-j, j-1) + \delta_S(s_1[i], s_2[j]). \end{cases} \tag{3.3}$$

The distance between s_1 and s_2 is then defined as the value of the last computed entry of the matrix, i.e.

$$\operatorname{dist}_E(s_1, s_2) = D(n, m). \tag{3.4}$$

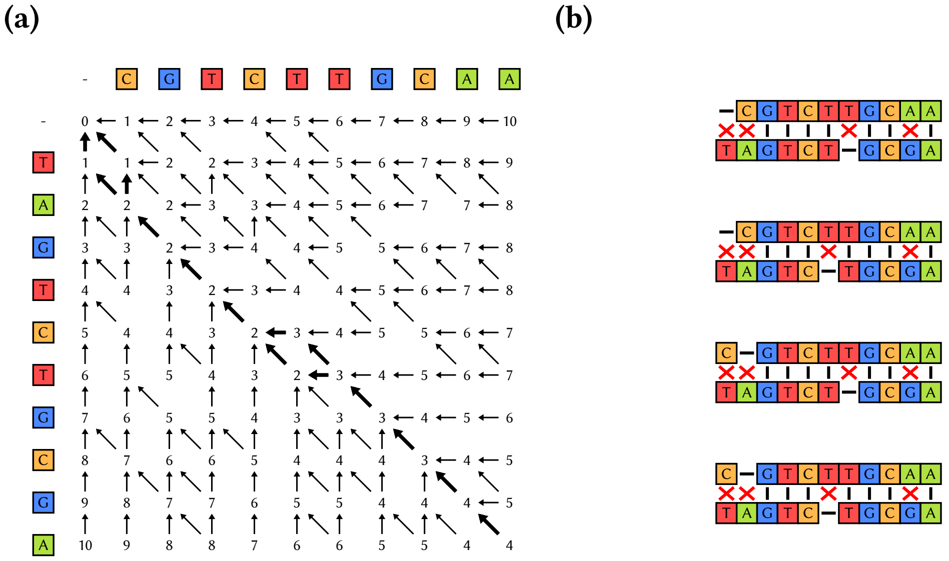
While the naïve, recursive solution to obtain $D(n, m)$ based on Equation 3.3 would be straight forward to implement, it is highly redundant and therefore computationally unnecessarily expensive. Instead, the edit matrix is computed using *dynamic programming*, i.e. in a bottom-up fashion starting with a matrix pre-initialized according to Equations 3.1 and 3.2. The remaining fields are then computed either row by row or column by column, making use of the fact that each value depends only on previously computed values. The edit matrix can therefore be computed in $\mathcal{O}(n \cdot m)$ time and space.

Further generalizations of this algorithm exist, most notably a formulation that enables the computation of *local similarities* between sequences and a generalized gap cost specification [42, 43]. A common approach is the differentiation of an initial gap (“gap open” cost δ_{GO}) and subsequent gaps following directly after the initial gap (“gap extension” cost, δ_{GE}).

3.3.4. Obtaining Optimal Alignments

Given the definition of the edit matrix, we now know how to obtain the edit distance, but not yet the corresponding optimal alignment(s). An efficient way to do so is to maintain pointers indicating the origin of each entry of the edit matrix, i.e. which of the cases in Equation 3.3 fulfills the minimality condition. Each matrix entry can therefore hold up to three pointers to previously computed adjacent entries.

Given an edit matrix and such *traceback pointers*, we can now enumerate all optimal alignments by inspecting all paths leading from $D(m, n)$ to $D(0, 0)$. Taking into



	-	C	G	T	C	T	T	G	C	A	A
-	0	1	2	3	4	5	6	7	8	9	10
T	1	1	2	2	3	4	5	6	7	8	9
A	2	2	2	3	3	4	5	6	7	7	8
G	3	3	2	3	4	4	5	5	6	7	8
T	4	4	3	2	3	4	4	5	6	7	8
C	5	4	4	3	2	3	4	5	5	6	7
T	6	5	5	4	3	2	3	4	5	6	7
G	7	6	5	5	4	3	3	3	4	5	6
C	8	7	6	6	5	4	4	4	3	4	5
G	9	8	7	7	6	5	5	4	4	4	5
A	10	9	8	8	7	6	6	5	5	4	4

Figure 3.2.: (a) The edit matrix for two strings s_1 and s_2. The arrows indicate the traceback pointers and bold arrows are part of optimal paths from $D(m,n)$ to $D(0,0)$. **(b)** The four optimal alignments corresponding to the four paths highlighted in the edit matrix.

account that horizontal edges correspond to gaps in s_1 and vertical edges correspond to gaps in s_2, we can recover the alignment rows s_1^*, s_2^* in $\mathcal{O}(n+m)$ time for each optimal alignment. Figure 3.2 shows the edit matrix for the pair of sequences previously used in Figure 3.1 as well as the traceback pointers and the four optimal alignments derived from them.

3.3.5. Distance vs. Score

The alignment cost minimization problem shown in Section 3.3.3 can be transformed into an equivalent score maximization problem by negating δ_G and δ_S and changing Equation 3.3 to maximize. We will use the notation

$$\mathrm{dist}(a,b)$$

when minimizing a cost function that rewards dissimilarity and punishes similarity and equivalently

$$\mathrm{score}(a,b)$$

when maximizing a cost function that rewards similarity and punishes dissimilarity.

3.3.6. Alternative Scoring Schemes

The edit distance scoring scheme has many advantages, in particular with respect to available algorithmic optimizations and filters (some of which we will discuss in Section 4.4.3 f.). However, there are also limitations. A binary match vs. mismatch relation does not reflect the nature of similarity and dissimilarity of the biochemical entities represented by characters from biological alphabets, such as DNA or protein alphabets. To better integrate this information into the comparison process, it is common practice to use more complex functions for δ_S, that take those properties into account [44].

Note that the edit distance scoring scheme treats matches neutrally. Pairwise distances are therefore only comparable, if they refer to sequences of equal lengths. In cases where longer alignments should yield a higher score than shorter alignments, matches have to be rewarded, e.g. by using $\delta_M(a, b) := 1 - 2 \cdot \mathbb{1}(a \neq b)$ in a score maximization based setting.

3.3.7. Banded Alignments

If we have prior knowledge about the optimal alignment trace and can therefore constrain its path, we can use a *banded alignment* in order to save computation time and space. A popular example is the case where we are only interested in the pairwise global alignment (similarity) of two sequences if their distance does not exceed a certain threshold k. Since the trace of the optimal alignment only deviates from the main diagonal of the edit matrix when insertions and deletions occur, we can limit our computation to the $2 \cdot k + 1$ diagonals surrounding (and including) the main diagonal. The required time and space is then $\mathcal{O}(k \cdot \min(n, m))$ and thus linear in the sequence length.

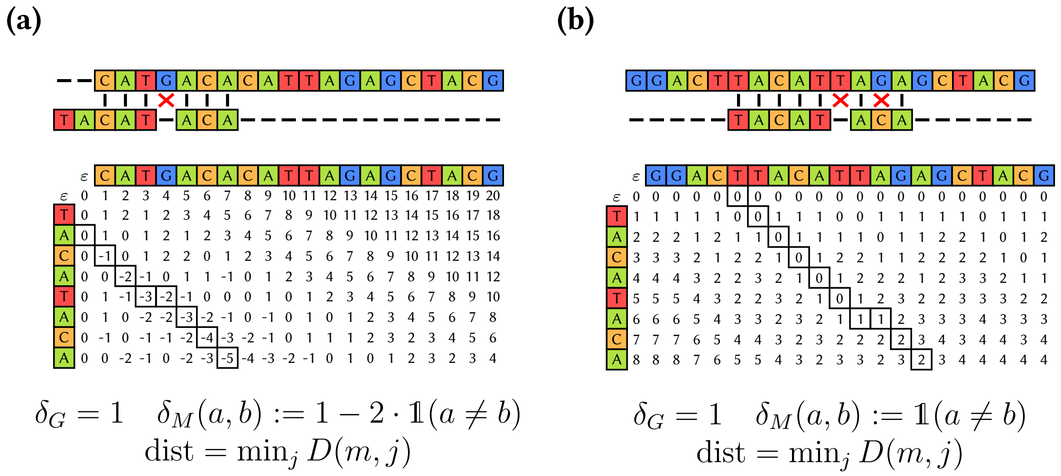
3.3.8. Overlap Alignment

An *overlap alignment* is a constrained form of the *free-end-gaps alignment*, which does not penalize both leading and trailing gaps flanking either sequence. Recall the nomenclature of the $n \times m$ edit matrix D for two sequences s_1 and s_2 of lengths n and m respectively, which was introduced in Section 3.3.2. Assume that we want to allow trailing gaps in s_1 and leading gaps in s_2. We can obtain the optimal overlap alignment by changing the DP algorithm to allow an entry into the leading sequence at any point without accounting for those gaps, i.e. changing Equation 3.1 to

$$D(i, 0) = 0 \qquad\qquad \forall\, i \in \{0, \ldots, m\}. \tag{3.5}$$

The distance is then defined by changing Equation 3.4 to

$$\operatorname{dist}(s, t) = \min_j (D(|s|, j)). \tag{3.6}$$



(a)

	ε	C	A	T	G	A	C	A	C	A	T	T	A	G	A	G	C	T	A	C	G
ε	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	0	1	2	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
A	0	1	0	1	2	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
C	0	-1	0	1	2	2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	0	0	-2	-1	0	1	1	-1	0	1	2	3	4	5	6	7	8	9	10	11	12
T	0	1	-1	-3	-2	-1	0	0	0	1	0	1	2	3	4	5	6	7	8	9	10
A	0	1	0	-2	-2	-3	-2	-1	0	-1	0	1	0	1	2	3	4	5	6	7	8
C	0	-1	0	-1	-1	-2	-4	-3	-2	-1	0	1	1	1	2	3	2	3	4	5	6
A	0	0	-2	-1	0	-2	-3	-5	-4	-3	-2	-1	0	1	0	1	2	3	2	3	4

$$\delta_G = 1 \quad \delta_M(a,b) := 1 - 2 \cdot \mathbb{1}(a \neq b)$$
$$\text{dist} = \min_j D(m,j)$$

(b)

	ε	G	G	A	C	T	T	A	C	A	T	T	A	G	A	G	C	T	A	C	G
ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T	1	1	1	1	1	0	0	1	1	1	0	0	1	1	1	1	1	0	1	1	1
A	2	2	2	1	2	1	1	0	1	1	1	1	0	1	1	2	2	1	0	1	2
C	3	3	3	2	1	2	2	1	0	1	2	2	1	1	2	2	2	2	1	0	1
A	4	4	4	3	2	2	3	2	1	0	1	2	2	2	1	2	3	3	2	1	1
T	5	5	5	4	3	2	2	3	2	1	0	1	2	3	2	2	3	3	3	2	2
A	6	6	6	5	4	3	3	2	3	2	1	1	1	2	3	3	3	4	3	3	3
C	7	7	7	6	5	4	4	3	2	3	2	2	2	2	3	4	3	4	4	3	4
A	8	8	8	7	6	5	5	4	3	2	3	3	2	3	2	3	4	4	4	4	4

$$\delta_G = 1 \quad \delta_M(a,b) := \mathbb{1}(a \neq b)$$
$$\text{dist} = \min_j D(m,j)$$

Figure 3.3.: Special cases of free-end-gaps alignments, **(a)** an overlap alignment and **(b)** a semi-global alignment. The figure shows an optimal alignment, the corresponding edit matrix with the entries comprising that alignment highlighted, as well as the scoring scheme used.

Note that with this reformulation of the DP algorithm, we can no longer use the edit-distance dist_E, since an overlap of length 0 would always yield an optimal alignment with $\text{dist}_E(s_1, s_2) = 0$. To obtain meaningful alignments, one either has do choose a different scoring scheme that rewards matches or to define an alignment band, i.e. define a minimal overlap or constrain the alignment even further. An example for an overlap alignment with a scoring scheme that rewards matches is shown in Figure 3.3 (a).

3.3.9. Semi-Global Alignment

Similar to the overlap alignment, a *semi-global* alignment is a special case of a free-end-gaps alignment, where both leading and trailing gaps in *one* of the two sequences are tolerated - in case of the pattern matching problem in the pattern. Again, this is achieved by a slight modification of the equations described in Section 3.3.2, namely Equation 3.2 which is changed to

$$D(0,j) = 0 \qquad \forall\, j \in \{0,\ldots,n\} \tag{3.7}$$

and Equation 3.4 which is again changed as shown in Equation 3.6. An example for an optimal, edit-distance based semi-global alignment is shown in Figure 3.3 (b).

3.4. Clonotypes and Repertoires

Recall the final configuration of antigen receptor genes as described in Section 2.3.3. Since, at least in the case of TCRs, all nucleotide modifications are contained in the CDR3, we can express a recombined antigen receptor gene by the incorporated gene segments and the sequence of the CDR3 region. Throughout this thesis and most importantly in the upcoming clonotyping chapter, we will use the following formal notation for the term *clonotype*. Given two sets of reference V and J segments

$$\begin{aligned} \mathcal{S}^V &= \left\{s_1^V, \ldots, s_{N_V}^V\right\} & s_i^V &\in \Sigma_{\text{DNA}}^* \\ \mathcal{S}^J &= \left\{s_1^J, \ldots, s_{N_J}^J\right\} & s_i^J &\in \Sigma_{\text{DNA}}^* \end{aligned}$$

we describe a clonotype as a tuple

$$\begin{aligned} c &= \left(c^V, c^J, c^{\text{CDR3}}\right), \text{where} \\ c^V &\subseteq \{1, \ldots, N_V\} \\ c^J &\subseteq \{1, \ldots, N_J\} \\ c^{\text{CDR3}} &\in \Sigma_{\text{DNA}}^*. \end{aligned}$$

With $c^V = \{n\}$ we indicate that the recombined gene contains the nth V gene segment s_n^V, allowing for multiple assignments to incorporate potential ambiguities in the data. Furthermore, we define a *clonotype repertoire* $(\mathcal{C}, \mathcal{F})$ as a collection of such clonotypes $\mathcal{C} = \{c_1, \ldots, c_N\}$ with associated absolute frequencies $\mathcal{F} = \{f_1, \ldots, f_N\}$. Ideally, in a measured clonotype repertoire every clonotype would have a unique V and J segment assigned and the associated frequencies would reflect the number of cells with that clonotype in the given sample. In practice one has to deal with gene segment ambiguities that cannot be resolved due to technical limitations and use abstract quantities such as read counts or UMI counts as clonotype frequencies, as described in Section 2.4.3.

In the case of immunoglobulins, this clonotype definition is insufficient, since the V and J segment component of the recombined gene cannot be described by the reference sequence due to somatic hypermutations. This can easily be accounted for by adding the modified gene sequences or a representation of the modifications such as a CIGAR string [45]. We will however neglect this, since it does not have any implications on the clonotype annotation method described in the following chapter.

Part II.

Clonotyping

4. Clonotyping & Repertoire Generation

The first post-experimental step of repertoire sequencing involves the handling of the raw data and its transformation into a dataset that allows for higher level clonotype repertoire analyses. In this chapter, we will define the task of clonotyping, i.e. exactly that initial data analysis step and name the challenges in properly solving that task. We then present an algorithmic workflow to solve the clonotyping problem. To evaluate the method, a simulation approach was developed in order to generate artificial Rep-Seq reads and finally an evaluation of the clonotyping method is performed based on both simulated as well as real datasets.

4.1. Introduction

In principle, the raw reads already represent the investigated antigen receptor repertoire. As described in Section 2.4.4, the underlying experimental protocol will result in identical reads (or read pairs) for every cell carrying the same antigen receptor gene under ideal conditions, i.e. assuming perfect primer performance and no other technical errors. Thus, associating every observed sequence with its frequency could be considered a valid representation of an immune repertoire.

In practice, analysis pipelines aim to identify the gene components, i.e. the V and J segments and the CDR3 sequence. We refer to this as *clonotyping*, i.e. we use this term to describe the process of correctly assigning these components to every read. This common practice has several reasons:

- **Technical errors**

  Even after thorough purification, the experimental steps undertaken to enrich the target gene will amplify non-target DNA to some extent, e.g. from cross reacting PCR primers. These fragments will be sequenced alongside the actual antigen receptor gene sequences. Annotating every sequence with its antigen receptor specific features facilitates the removal of sequences that do not show the expected sequence patterns. Additionally, in particular in the case of T cell receptors, defining the clonotype based on the most likely V and J segments eliminates false clonotype diversity originating from sequence errors in those regions of the read.

- **Non-functional recombinations**

  Especially DNA based protocols will potentially also enrich non-functional, silenced V(D)J recombined gene sequences in addition to the functional gene on the homologous chromosome. With a proper annotation of the sequences, one can filter out gene sequences that show frame shifts or stop codons and therefore drastically reduce the impact of the silenced genes.

- **Receptor similarity**

  As briefly discussed in Section 2.3.2, the different regions of the antigen receptor protein and thus their corresponding regions in the underlying, recombined gene, fulfill different functions. Therefore, depending on the application, it makes sense to group the observed receptor genes according to their gene features. In some studies, receptors are aggregated based solely on their CDR3 region [46, 47] or based on identical V and J segments and similar CDR3 regions [48].

- **Compression**

  A more practical advantage of handling clonotype information in an annotated fashion is the high degree of redundancy across even distinct gene recombinants. Most of the gene is conserved, even in the case of hypermutated IG genes. Therefore, storing IDs of the incorporated V and J gene segments, the CDR3 region and a record of the observed differences is significantly more compact.

We developed a method to solve the clonotyping task that not only annotates every read individually, but furthermore takes the information from all observed reads into account in order to generate an error corrected repertoire. Additionally, we are able to incorporate *paired end* reads (see Section 2.4.4), to improve the V segment assignment, as well as unique molecular identifiers (see Section 2.4.3) for the correction of sequence errors and the generation of bias normalized repertoires. Furthermore, some existing methods do not compute the alignments between the germline gene segments and recombined gene, but solely rely on aggregating index structures to report the most likely gene segment. In contrast, our method reports those alignments while still being highly performant with respect to the required computation times. This makes our method applicable to the analysis of IG genes, where somatic hypermutations are of interest, but also for the potential detection of gene segments or gene segment alleles not present in the used reference database. The details of this method are described in the following sections.

4.2. Method Overview

The interpretation of Rep-Seq data differs in many aspects from that of other common next generation sequencing data. When we investigate the genomic variation of an individual (*whole genome sequencing* (WGS), *whole exome sequencing* (WES)) or differential gene expression based on RNA abundance (*RNA-Seq*), we map the reads

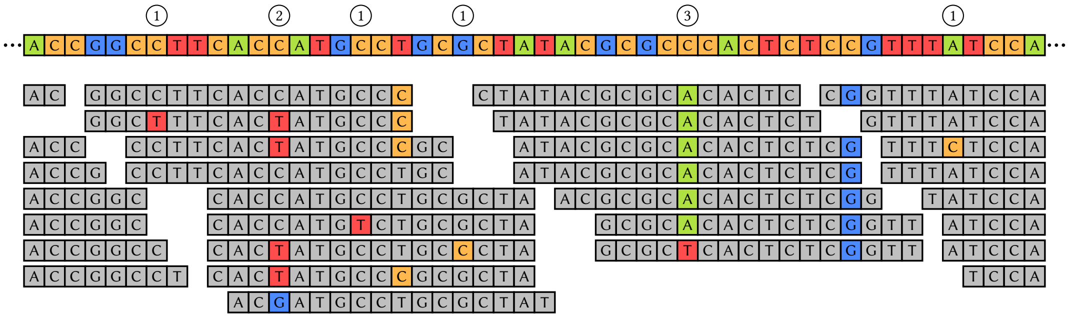


Figure 4.1.: A simplified illustration of a multiple sequence alignment, obtained by mapping reads of an individual against the human genome reference sequence in order to study polymorphisms. Single sequence deviations can be accounted for by sufficient redundancy (coverage), such that in a multiple sequence alignment positions that are in agreement with the reference (1) as well as homozygous (3) and heterozygous (2) variation can still be detected with a high degree of certainty, even though they contain errors.

against a known reference such as the genome or transcriptome of the investigated species using read mapping methods [49, 50, 51, 52, 53, 54]. With some exceptions, the large majority of produced sequence data is expected to be identical to a known reference. Where it isn't, we can usually make quite strong assumptions about the deviations. A human genomic variant, for example, should either occur on all (in the homozygous case) or in about half (in the heterozygous case) of the reads covering the affected location. Deviations that occur in just a few reads can safely be discarded as technical artifacts.

These assumptions usually enable us to handle technical artifacts, such as sequencing and PCR errors, by designing the experiment with a sufficient degree of redundancy, i.e. *read coverage*. Figure 4.1 shows a simplified example of a multiple sequence alignment of reads mapped against a reference as in WGS or WES experiments. We can clearly see how the redundancy available through the read coverage can account both for errors indicating deviations at positions that are actually in agreement with the reference as well as positions that show homozygous and heterozygous (single nucleotide) variations.

In antigen receptor repertoires, almost no assumptions can be made about the frequency distribution of clonotypes or about their CDR3 regions. Naïve cells are likely to occur as singletons, while subpopulations specific for an antigen originating from a recent infection or vaccination will probably be dominant in the repertoire - and many intermediate stages occur due to the complex behavior of proliferation and decline depending on various factors. Furthermore, biologically generated clonotypes that differ in a few or even just single nucleotides in the CDR3 region are expected to occur even though the theoretical space of clonotypes is extremely large. This is due to the selective pressure involved in the repertoire generation process in the thymus (see Section 2.3.6). We therefore cannot handle technical errors as in other

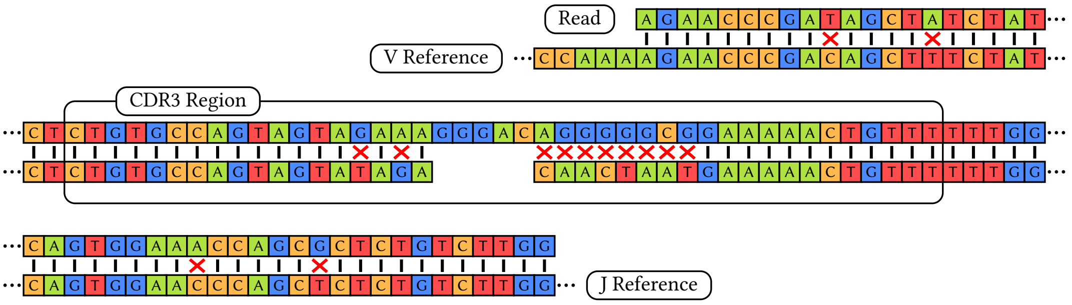

Read
AGAACCCGATAGCTATCTAT···
V Reference
···CCAAAAGAACCCGACAGCTTTCTAT···
CDR3 Region
···CTCTGTGCCAGTAGTAGAAAGGGACAGGGGGCGGAAAAACTGTTTTTTGG···
···CTCTGTGCCAGTAGTATAGA
CAACTAATGAAAAACTGTTTTTTGG···
···CAGTGGAAACCAGCGCTCTGTCTTGG
···CAGTGGAACCCAGCTCTCTGTCTTGG···
J Reference

Figure 4.2.: A TRB Rep-Seq read (top row) aligned to the best matching TRBV and TRBJ gene segments (bottom row). The mismatches in the non-CDR3 part of the read can be classified as technical artifacts, whereas the mismatches within the CDR3 region can either be technical or biological variation.

NGS experiments.

Figure 4.2 shows an example for a pairwise alignment between a TRB Rep-Seq read and the best matching TRBV and TRBJ gene segments and how technical errors at different positions might affect the clonotyping results. The mismatches located outside the CDR3 regions are likely to be caused by technical errors, which can be inferred because the correct sequence of those regions is known through the germline reference sequences. On the other hand, the mismatches located within the CDR3 region are expected due to the random removal and addition of nucleotides during the recombination process (see Section 2.3.3). We cannot assume that the germline sequence within this region is actually incorporated in the gene during the recombination and therefore have to find other means to account for technical errors within the CDR3 region than a reference comparison.

The challenge to infer quantitative information while we can make no assumptions about the individual clonotype abundances combined with the necessity to detect de-novo sequence content while the degree of sequence variation and the number of variants is unknown, makes this problem different from standard NGS applications. We therefore developed, implemented and evaluated a novel approach for the proper annotation of Rep-Seq reads, which consists of the following steps:

1. **Read preprocessing**

   Initially, a quality filter based on the quality score reported by the sequencing platform is applied. Reads with an average per base quality below a user defined threshold $q_{\min}$ are discarded. Identical reads are then collapsed, while the read count information is kept for the subsequent steps.

2. **UMI clustering of similar reads**

   If unique molecular identifiers were incorporated during the experimental preparation of the sample, the reads are clustered based on their barcode and sequence similarity. The process is described in the upcoming Section 4.3.

3. **V and J segment assignment**

   The main step of the clonotyping phase is the identification of the incorporated V and J segments by comparing each read sequence with the sets of reference sequences $\mathcal{S}^V$ and $\mathcal{S}^J$. The process is optimized to be feasible for a large number of reads while remaining fully sensitive and exact. It is described in detail in Section 4.4.

4. **Identification of the CDR3 sequence**

   Using the information about the best matching V and J gene segments that have been identified in the previous step, we define the CDR3 region based on where the germline encoded Cys_{104} and Phe_{118} triplets align to the read. After the CDR3 region is defined, another simple filtering step is applied, ensuring that both triplets are in-frame and that there are no stop codons in the CDR3 region. Unless explicitly requested otherwise by the user, those reads are discarded since they most likely originate from a non-functional recombination (see Section 2.3.3).

5. **Repertoire generation and error correction**

   After all unique reads have been clonotyped, a preliminary repertoire is defined based on the original number of non-unique reads that yield the same clonotype information, defined based on the incorporated V and J segments and the CDR3 region. This preliminary repertoire is then post-processed to account for ambiguous segment assignments and sequencing and PCR errors. The details of the repertoire post-processing are described in Section 4.5 f.

The main steps from the raw read data to a clonotype repertoire are described in the following sections.

4.3. UMI Based Read Clustering

As described in Section 2.4.3, the ability to correct sequence errors and PCR induced frequency biases can be greatly enhanced if the DNA fragments are labelled with unique molecular identifiers (UMIs) in an early stage of handling the templates.

We incorporated the ability to use UMI information into the clonotyping workflow. The preprocessed raw data prior to this step are unique pairs of reads and UMIs ($r_1, \ldots, r_n$ and $u_1, \ldots, u_n$) and the corresponding counts $f_1, \ldots, f_n$. We then inspect for every read r_{ref} every other more abundant read r_{tar}, i.e. $f_{\text{ref}} < f_{\text{tar}}$, starting from the most abundant target read r_{tar} descending to the least abundant. For every such pair we check whether the barcodes are identical or similar up to a hamming distance of δ_{UMI} and whether the read sequences have an error rate of at most ε_{UMI} (based on the number of edit operations). If both conditions are fulfilled, we consider r_{ref} an erroneous version of r_{tar}, re-assign the counts and inspect no further r_{tar} for this read.

We additionally constrain the maximum frequency ratio between r_{ref} and r_{tar}, i.e.

```
 1: function UMICorrection($\mathcal{R} = r_1, \ldots, r_N, \mathcal{F} = f_1, \ldots, f_N, \mathcal{U} = u_1, \ldots, u_N$)
 2:     $pairs \leftarrow \varnothing$
 3:     for $i \leftarrow 1, \ldots, N$ do                                  ▷ Parallel execution
 4:         $L \leftarrow$ Length($r_i$)
 5:         for $j \leftarrow N, \ldots, (i+1)$ do
 6:             if $f_i / f_j > r_{\text{UMI}}$ then
 7:                 Break                        ▷ Subsequent reads have lower counts
 8:             else if $d_H(u_i, u_j) > \delta_{\text{UMI}}$ then                ▷ UMI comparison
 9:                 Continue
10:             else if $\text{dist}_E(r_i, r_j)/L \leq e_{\text{UMI}}$ then            ▷ Read comparison
11:                 $pairs \leftarrow pairs \cup \{(i, j)\}$
12:                 Break                                           ▷ Match found
13:             end if
14:         end for
15:     end for
16:     SortAscending($pairs$, by = second component)
17:     for all $(i, j) \in pairs$ do
18:         $f_j = f_j + f_i$
19:         $f_i = 0$
20:     end for
21: end function
```

Algorithm 4.1: The UMI correction method. Given a list of reads $\mathcal{R}$ ordered from the least abundant to the most abundant, the corresponding frequencies $\mathcal{F} \in \mathbb{N}^+$ and the corresponding list of UMI sequences $\mathcal{U}$, the method modifies the frequency vector $\mathcal{F}$ to reflect the count reassignments performed during the correction. δ_{UMI} and e_{UMI} are user specified parameters.

${}^{f_{\text{ref}}}/_{f_{\text{tar}}} \leq r_{\text{UMI}}$ with $0 < r_{\text{UMI}} < 1$. $\delta_{\text{UMI}}, \varepsilon_{\text{UMI}}$ and r_{UMI} being user-defined parameters.

The approach is outlined in Algorithm 4.1. By separating the actual modifications from the computationally expensive comparisons we can easily parallelize the latter. Since we are not interested in the actual alignments between the reads but only in their edit distances, we can furthermore make use of an efficient bit parallel algorithm [55] to compute the distance, which will be described in more detail in Section 4.4.4. Also note that after the comparison step (Line 16 ff.) we iterate the pairs of unique reads from the least abundant to the most abundant target read, allowing subsequent corrections reflecting the branching nature of subsequent PCR errors to be taken into account.

4.4. V and J Segment Assignment

As described in Section 2.4.4, the primary clonotype information is contained in a read spanning parts of the V and J region as well as the intermediate CDR3 region of the

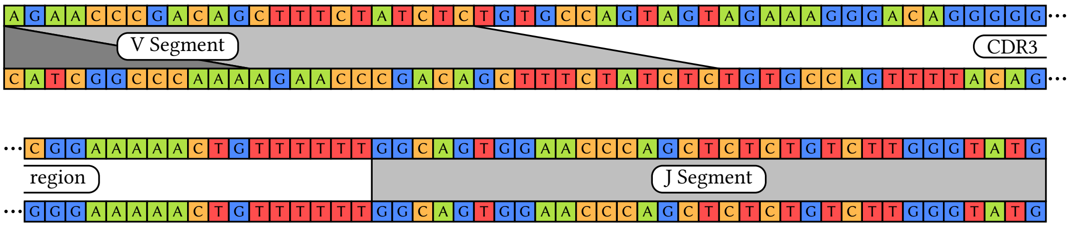


Figure 4.3.: Two TRB Rep-Seq reads, both encoding clonotypes incorporating the gene segments V19 and J1-4. The clonotype encoded in the upper read has a CDR3 length of 45 nt, the one encoded in the lower read of 33 nt. The lower read therefore contains additional V segment information, highlighted in dark gray.

gene. We defined this read as the V(D)J read which is optionally complemented by a second read covering an additional part of the V region from the 5' end of the gene referred to as the V-read. In this section we will describe how we identify the gene segments based on the V(D)J read using efficient filtering and alignment algorithms. We will see how the obtained information can be further refined using the additional V-read in Section 4.4.6.

4.4.1. Overview

In our method design, we assume that the V(D)J read does not contain any additional sequences, i.e. artifacts such as sequencing adapters have been clipped off with an appropriate method [56, 57, 58] if necessary. Also, if the protocol includes sequence originating from a neighboring intron (DNA based approach) or C-segment (RNA based approach) downstream of the J segment, it has been removed prior to the analysis.

Our aim is to identify the V and J segments incorporated in the recombined gene by comparing each read to the set of germline reference sequences. Recall that the V and J segments contain well conserved motifs embedding the triplets encoding for Cys_{104} and Phe_{118}, as described in Section 2.3.3. While these define the boundaries of the CDR3 region, the germline segments contain sequence beyond that point, i.e. which contributes to the genes CDR3 region. This sequence is of different lengths in different germline gene segments and additionally undergoes modifications to various degrees. In order to obtain a distance measure that does not prefer certain gene segments due to their longer germline CDR3 component, we only consider the non-CDR3 part of the gene segments when comparing the pairwise alignments. We refer to the V gene segment up until and including the Cys_{104} triplet as well as to the J gene segment from the Phe_{118} triplet on as the *CDR3-truncated gene segment* sequences.

We also have to keep in mind that two reads will mostly contain a different amount of V segment, even if the reads are of the same length and the underlying gene uses the same V segment. This is due to the varying length of the CDR3 region, as shown in Figure 4.3.

With the above assumptions in mind, aligning the CDR3-truncated gene segment sequences against the V(D)J read requires an overlap alignment as described in Section 3.3.8, i.e. we don't want to penalize trailing gaps on the V segment sequence where the read covers the CDR3 and J segment sequences and also not leading gaps on the read where the V region is not covered. The same applies inversely for the J segment.

A naïve solution to identify the V and J segment incorporated in a given V(D)J read would be to compute the pairwise overlap alignment score between every gene segment and every read and assign those to the read that yield the best alignment score. However, due to the high number of computationally expensive comparisons, this is a relatively inefficient approach. In the following sections we will devise a more efficient strategy.

4.4.2. Segment Core Fragments

To solve the V and J segment assignment problem, we divide it into a pattern matching problem and a banded overlap alignment problem. The *pattern matching* problem is to find approximate occurrences of a short sequence, the *pattern*, in a longer sequence, the *text*. More precisely, given a set of patterns we want to identify those patterns with a semi-global alignment distance (see Section 3.3.9) below a given threshold and their corresponding regions in the text. This problem is intensely studied in the context of biological sequences and therefore fast methods to enhance the performance of our segment identification problem already exist. Note that inversely to the common problem of mapping reads against a reference genome, we are looking for occurrences of short reference sequences inside the reads.

We design the pattern matching problem by defining a region in the reference gene segment sequences that we expect to find *as a whole* in any read sequence originating from a clonotype that incorporates it. We refer to these regions inside the V and J segments as *segment core fragments* (SCFs). The location of the SCF within the gene segment given a length $L_{\text{SCF}}^{V|J}$ is defined such that it contains the $L_{\text{SCF}}^{V|J}$ nucleotides adjacent to the CDR3 region, including the boundary triplets encoding for Cys_{104} and Phe_{118}:

$$\begin{aligned} \operatorname{scf}(s_i^V) &:= s_i^V[p_i^{\text{Cys}} - L_{\text{SCF}}^V + 3, p_i^{\text{Cys}} + 2] && \forall i \in \{1, \dots, |\mathcal{S}^V|\} \\ \operatorname{scf}(s_i^J) &:= s_i^J[p_i^{\text{Phe}}, p_i^{\text{Phe}} + L_{\text{SCF}}^J], && \forall i \in \{1, \dots, |\mathcal{S}^J|\} \end{aligned}$$

where $s_i^{V|J}$ denotes the ith V or J segment and $p_i^{\text{Cys|Phe}}$ denotes the Cys / Phe encoding triplet position of the ith V or J segment. An example for four V segments is shown in Figure 4.4.

Since the segment core fragments (i.e. our patterns) will all be of equal length, we eliminate the need for distance or score normalization and maintain the ability to use the edit distance scoring scheme. Note that the SCFs $\mathcal{S}_C^V$ and $\mathcal{S}_C^J$ are not necessarily

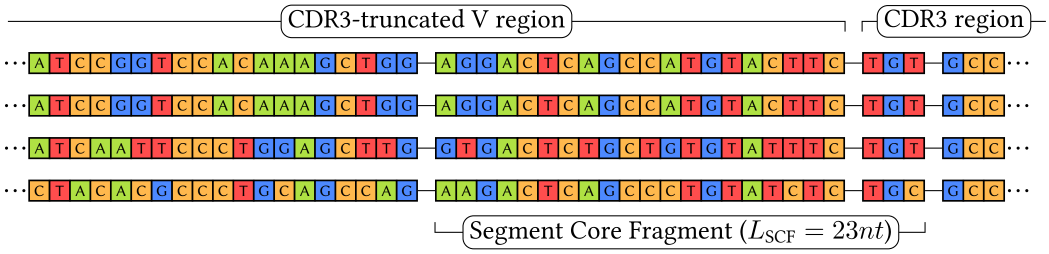

CDR3-truncated V region
CDR3 region
···ATCCGGTCCACAAAGCTGG–AGGACTCAGCCATGTACTTC–TGT–GCC···
···ATCCGGTCCACAAAGCTGG–AGGACTCAGCCATGTACTTC–TGT–GCC···
···ATCAATTCCCTGGAGCTTG–GTGACTCTGCTGTGTATTTC–TGT–GCC···
···CTACACGCCCTGCAGCCAG–AAGACTCAGCCCTGTATCTC–TGC–GCC···
Segment Core Fragment ($L_{\text{SCF}} = 23nt$)

Figure 4.4.: An excerpt from the four human TRBV gene segments 1, 2, 3-1 and 4-1 (from top to bottom). We define the segment core fragment to be a region of fixed length (in this example $L_{\text{SCF}} = 23nt$) starting inside the V segment and ending in the Cys$_{104}$ triplet.

unique especially for small values of L_{SCF}, i.e.

$$\begin{aligned} \mathcal{S}_C^V &= \{\text{scf}(s) \mid s \in \mathcal{S}\} & \qquad \left|\mathcal{S}_C^V\right| &\leq \left|\mathcal{S}^V\right| \\ \mathcal{S}_C^J &= \{\text{scf}(s) \mid s \in \mathcal{S}\}\,. & \qquad \left|\mathcal{S}_C^J\right| &\leq \left|\mathcal{S}^J\right| \end{aligned}$$

Based on the SCF definition, we now partition the gene segment assignment problem into the following steps, which are applied for every read to be clonotyped:

- **Filtering**

  Initially, we apply a filtering method that reduces the search space of SCF vs. read alignments that have to be computed.

- **Verification**

  Every SCF proposed by the filter is verified against the proposed area within the read and only those are kept that yield an alignment score within user specified parameters.

- **CDR3-truncated gene segment alignment**

  For every valid occurrence of an SCF within the read, we compute the full overlap alignment between the CDR3-truncated gene segment sequences defining the SCF and the read. We can use the identified location in combination with an error threshold to narrowly band this alignment.

The three stages of the gene segment assignment are described in more detail in the following sections.

4.4.3. Filtering

The goal of the filtering step is to reduce the search space for the semi-global alignments between the SCFs and the reads on two levels: It reduces the number of SCFs that have to be aligned against the read and furthermore, for each SCF, limits the area within the read that has to be considered. To accomplish this, we make use of *SWIFT*, a fully sensitive *alignment filter* developed by Rasmussen et al. [59]. The filter can be used to reduce the search space for the following problem:

Problem 4.1. Given a text t and a pattern p of length $L = |p| \leq |t|$, find all substrings t' of t, s.t. $\text{dist}_E(t', p) \leq e$.

We refer to t' as an e-match of p in t. To accomplish the search space reduction, the authors make use of the following q-gram counting lemma:

Lemma 4.1. Let t' be a substring of a text t, p a pattern of length $L = |p| \leq |t|$ and $q \in \mathbb{N}^+$. If $\text{dist}_E(t', p) \leq e$, there exist at least $(L+1) - q(e+1)$ q-hits within $e+1$ consecutive diagonals within the corresponding edit matrix.

Proof. Given a string p of length L, there are $L - q + 1$ possible starting positions for substrings of length q. For any position in p, at most q such q-grams overlap at that position. Thus, the distribution of e errors can affect at most $q \cdot e$ q-grams, leaving $(L+1) - q(e+1)$ q-grams intact. Therefore, if there is a e-match of p in a text t, the corresponding edit-matrix will contain at least that many q-hits. For every insertion or deletion, the q-hits continue on an adjacent diagonal, thus they are distributed across at most $e+1$ diagonals. □

A key accomplishment of the SWIFT method is the generalization of Lemma 4.1 to pairwise *local* matches of a minimum match length n_0 and with a relative maximum error rate ε. The algorithm then defines lower bounds for the number of q-hits within *parallelograms* in the edit-matrix, that do not necessarily span either dimension entirely.

The authors furthermore developed a method to enumerate such parallelograms based on a q-gram index built over one of the sequences. Given the alphabet Σ and the q-gram length q, the index consists of a lookup table $\mathcal{L}$ that points to a list of starting positions of g in the sequence for all $g \in \Sigma^q$. After the index was generated, a window is slid across the second sequence, keeping count of the number of q-hits in all overlapping parallelograms of the target size. Whenever a parallelogram reaches the threshold, a candidate region is reported.

Since we are interested in semi-global matches of the segment core fragments within the read, we omit the generalization to local matches and instead use a simplified version of SWIFT solely based on Lemma 4.1, as previously shown by Weese et al. [52] in the context of read mapping. Keeping track of the q-hit counts becomes simpler, as we are only interested in parallelograms spanning the pattern entirely, i.e. consecutive groups of full length diagonals as previously described. The computational overhead for opening and closing parallelograms (i.e. the respective q-hit counters) along the pattern can therefore be avoided.

Note that the number of adjacent diagonals given by Lemma 4.1 is a lower bound - obviously the same minimum number of q-hits is also found in a larger group of diagonals. As long as the inspected groups of diagonals overlap by at least $e+1$, the filter algorithm remains to be fully sensitive. The authors make use of this in order

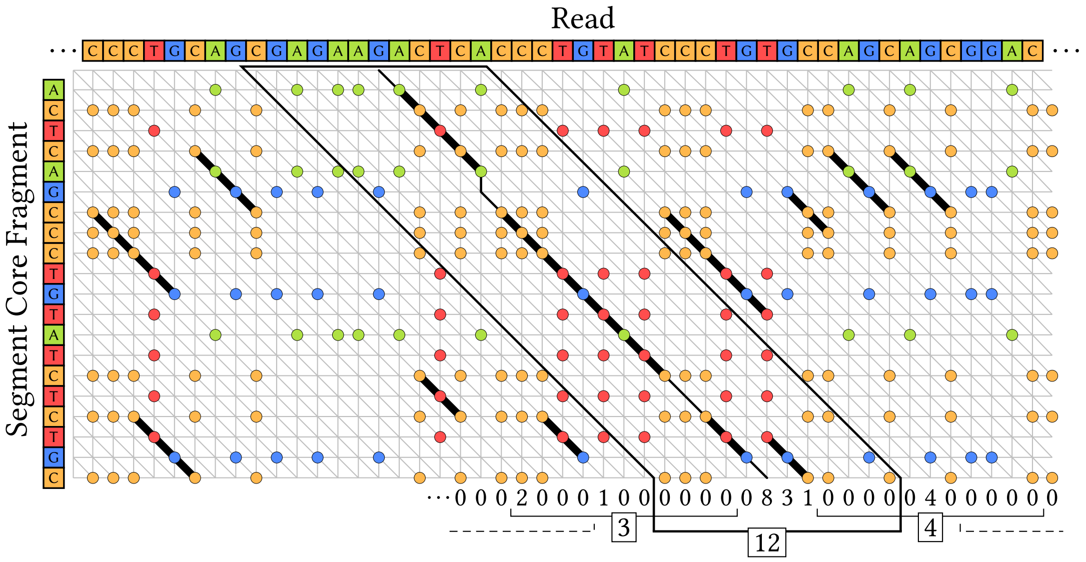

Read
··· C C C T G C A G C G A G A A G A C T C A C C C T G T A T C C C T G T G C C A G C A G C G G A C ···
Segment Core Fragment
A C T C A G C C C T G T A T C T C T G C
···0 0 0 2 0 0 0 1 0 0 0 0 0 0 0 8 3 1 0 0 0 0 0 4 0 0 0 0 0 0
3
12
4

Figure 4.5.: A simplified visualization of the edit matrix between an excerpt from an input read and the SCF for TRBV4-1. The SCF ($L^{V}_{\text{SCF}} = 20$) matches with an edit-distance of 3. Each dot represents a match, matches of size $q \geq 3$ are connected. Given Lemma 4.1, with a maximum error $e = 3$, the algorithm would look for windows spanning at least $e+1$ diagonals with $(L+1) - q(e-1) = 9$ or more q-hits. In this example, the window width is set to $11 \geq (e+1)$ and the only window exceeding the q-hit threshold is the one containing the desired alignment.

to reduce the number of q-hit counters that have to be maintained during the search. Instead of $e+1$, they inspect groups of size $\Delta + e + 1$. If Δ is chosen as a power of 2, the addressing of the counters can be further optimized using bit operations. These improvements trade filter specificity, but do significantly reduce the time and space requirements of SWIFT.

We initially build the required q-gram index structure over the set of SCF sequences and then slide along each read sequence to obtain the filter results. A visualization of the edit matrix between a read and an SCF with the corresponding q-hit counters per diagonal (for illustration) and per group (as actually implemented) is shown in Figure 4.5.

Segment core fragments that do not yield a group of diagonals exceeding the q-hit threshold are not investigated further. Otherwise, the corresponding region within the read is validated with respect to the given SCF as described in the next section. Note that the filter might yield multiple, disjoint regions within the read. If that is the case, we validate the entire region enclosed by the first and the last reported read position. This potentially reduces the specificity of the filter step, but does not have a major impact in practice due to the relatively high specificity of the SCF sequences as well as the high efficiency of the subsequently used verification algorithm.

4.4.4. Verification

Given the read sequence and the reduced set of SCFs, each associated with a potential target region within the read, we now need to validate whether each SCF yields a semi-global alignment score within the target region and the user specified score constraint. At this stage, we use an efficient bit-parallel string comparison method proposed by Myers [55]. The algorithm solves the optimization problem described in Section 3.3.2, but only provides the score and no option to trace back the corresponding alignment. When used in its semi-global form, one can obtain the alignment score for every end-position of the pattern in the text.

The algorithm computes the edit matrix column by column and bit-parallel over the rows. For that purpose, the columns of the edit matrix are encoded in multiple boolean vectors. The encoding exploits the fact that for the edit distance the differences between any entry and the three previously computed adjacent entries in the matrix are constrained. Given a fixed alignment column j, the algorithm uses the following bit vectors of length m:

$$\begin{aligned}
\text{Vertical positive difference} \quad & VP_j[i] := \mathbb{1}(D(i,j) - D(i-1,j) = 1) \\
\text{Vertical negative difference} \quad & VN_j[i] := \mathbb{1}(D(i,j) - D(i-1,j) = -1) \\
\text{Horizontal positive difference} \quad & HP_j[i] := \mathbb{1}(D(i,j) - D(i,j-1) = 1) \\
\text{Horizontal negative difference} \quad & HN_j[i] := \mathbb{1}(D(i,j) - D(i,j-1) = -1) \\
\text{Diagonal zero difference} \quad & D0_j[i] := \mathbb{1}(D(i,j) - D(i-1,j-1) = 0)
\end{aligned}$$

This representation of the edit matrix is based on the fact that the difference of horizontally or vertically adjacent cells is either -1, 0 or 1 [60, 61]. The bit vectors partially encode the same information redundantly and are only calculated as intermediate results in order to reduce the number of required operations. Additionally to the binary encoded DP matrix columns, the algorithm uses a pattern mask PM for every character $c \in \Sigma$, encoding its occurrences in the pattern p:

$$PM_c[i] = \mathbb{1}(p[i] = c) \qquad \forall\, c \in \Sigma, i \in \{1, \dots, m\}$$

The recurrence of the algorithm for the computation of the columns $D(i,1)$ to $D(i,n)$ is shown in Algorithm 4.2. The initialization of the bit vectors representing $D(i,0)$ is straight forward from Equations 3.1 and 3.7. The overall time complexity of the algorithm is in $\mathcal{O}(\lceil m/w \rceil \cdot n)$, where w is the machine word length of the underlying architecture, i.e. for patterns shorter than w the runtime is in $\mathcal{O}(n)$. For our application this means that the runtime remains linear in the length of the proposed target region within the read, as long as we choose $L^V_{\text{SCF}}, L^J_{\text{SCF}} \leq w$. Note that there is no critical constraint induced by the alphabet size here, since the bit operations are not executed on the strings but the virtual contents of the edit matrix.

```
for $j \in \{1, \dots, n\}$ do
    $D0_j \leftarrow (((PM_{t[j]} \,\&\, VP_{j-1}) + VP_{j-1}) \wedge VP_{j-1}) \mid PM_{t[j]} \mid VN_{j-1}$
    $HP_j \leftarrow VN_{j-1} \mid \sim (D0_j \mid VP_{j-1})$
    $HN_j \leftarrow D0_j \,\& VP_{j-1}$
    $VP_j \leftarrow (HN_j << 1) \mid \sim (D0_j \mid (HP_j << 1))$
    $VN_j \leftarrow D0_j \,\&\, (HP_j << 1)$
    if $HP_j \,\&\, 10^{m-1} \neq 0$ then
        $\mathrm{score}_j = \mathrm{score}_{j-1} + 1$
    else if $HN_j \,\&\, 10^{m-1} \neq 0$ then
        $\mathrm{score}_j = \mathrm{score}_{j-1} - 1$
    end if
end for
```

Algorithm 4.2: The bit-parallel recursion step of Myers algorithm to compute the jth column of the DP matrix, given a pattern of length m, a text t of length n and the initialized bit vectors.

4.4.5. CDR3-Truncated Gene Segment

For every matching SCF in $\mathcal{S}_{C*}^{V|J} \subseteq \mathcal{S}_{C}^{V|J}$ reported by the verification algorithm we now compute a narrowly banded overlap alignment (see Section 3.3.8) between the read and *any* CDR3-truncated gene segment that defines that SCF, i.e. for all

$$\begin{aligned} s^V \in \mathcal{S}_*^V &:= \left\{ s^V \;\middle|\; \mathrm{scf}(s^V) \in \mathcal{S}_{C*}^V \right\} \\ s^J \in \mathcal{S}_*^J &:= \left\{ s^J \;\middle|\; \mathrm{scf}(s^J) \in \mathcal{S}_{C*}^J \right\} . \end{aligned} \tag{4.1}$$

The band is defined by the location of the SCF within the segment reference (which is well defined as described in Section 4.4.2), the location of the SCF within the read as reported by the verification step and two additional parameters, ε_V and ε_J, defining the maximum error rate tolerated in the read to gene segment overlap alignment. Since the overlap length for each read and gene segment is defined after the SCF has been matched, the error rate can be transformed into an upper bound for the absolute number of errors and thus into an alignment band.

After all relevant gene segments have been overlapped with the read, the best scoring gene segment is reported as the one incorporated in the gene. If multiple gene segments score equally, they are all reported. Therefore, the output of the segment assignment step are two sets of gene segment IDs, c^V and c^J, for each read.

4.4.6. Non-Overlapping Paired End Sequencing

The V and J segment assignment procedure described up until here covers the analysis of the V(D)J read. As stated in Section 2.4.4, in a non overlapping paired end sequencing scenario we have an additional V read covering an upstream region of the V segment.

If such information is available, we use it to improve and speed up the V segment identification within the V(D)J read. We again use a combination of the SWIFT filter and the bit parallel verification algorithm to derive a set of V segments that score optimally, this time against the V read. The roles are switched in this case, i.e. the V read is the pattern and the V gene segments are the text to conduct the filtering and verification against, similar to the classical read mapping problem. Given a set of V segments identified based on the V read, c^{V*}, we then constrain the V(D)J read V segment identification. The filtering and verification steps are performed as previously described, but on limited sets of segment core fragments $\mathcal{S}_*^{CV}$ defined as

$$\mathcal{S}_*^{CV} = \left\{ \mathrm{scf}(s) \mid s \in c^{V*} \right\} .$$

The incorporated gene segments are then defined as described in Section 4.4.5, but instead of investigating all V segments that define the verified SCFs as shown in Equation 4.1, we only consider alignments against V segments previously identified by comparing the V read, c^{V*}.

Altogether the V read inclusion therefore reduces the V gene segment ambiguity, i.e. resolves cases where the V segments cannot be uniquely identified based on the V(D)J read. It furthermore improves the performance of the V(D)J read based segment identification by decreasing the search space size, however, at the additional cost of analysing the V read.

4.5. Repertoire Error Correction

As initially discussed in the introduction, we have two main types of errors that lead to false clonotyping, PCR and sequencing errors. Redundancy (coverage) cannot be used to avoid such errors in the same way as it could be done in many other sequencing applications (see Section 4.2). Additionally, filtering away reads with a low sequencing quality score can strongly disturb the quantitative information obtained from Rep-Seq samples, as shown in Figure 4.6, due to the biased nature of sequencing errors.

Unique molecular identifiers (see Section 4.3) are certainly the most powerful tool to avoid erroneous clonotyping and furthermore obtain bias-free frequencies for the identified clonotypes. However, UMIs are an additional experimental effort and furthermore take up a part of the available read length. In practice, a lot of samples are generated without UMI adapters. We therefore developed an independent strategy to correct sequence errors based on the preliminary clonotype repertoire derived from unfiltered (or less strictly filtered) reads. Reads that can be rescued using this strategy are then not discarded and contribute to the final repertoire. The method is based on a hierarchical clustering approach and is fully parametrizable with respect to the desired degree of correction. We refer to this as *standalone error correction*.

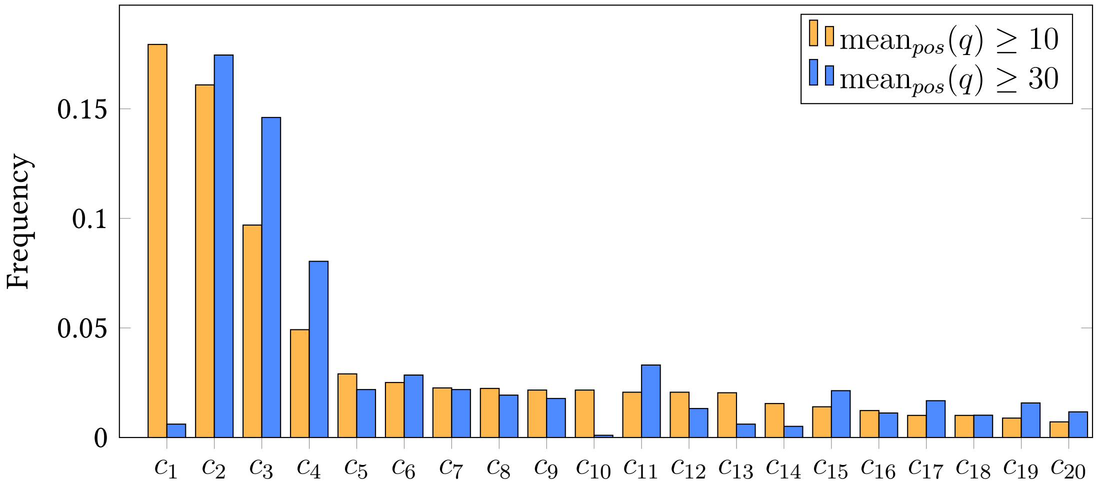


Figure 4.6.: The top 20 clonotypes and their frequencies obtained from a low quality Rep-Seq sample when applying an initial read quality filter including only those reads with an average quality ≥ 10 (orange) and ≥ 30 (blue). The stricter quality threshold of 30 strongly affects the clonotype order and almost removes the most dominant clonotype.

4.5.1. Basic Idea

Sequencing and PCR errors that occur outside the CDR3 region would decrease the pairwise alignment score between the gene segments and the read. They do however not necessarily falsify the identification of the gene segment, since the errors would have to systematically mimic the sequence of another gene segment to do so. The procedure therefore solely aims to correct errors inside the CDR3 region, i.e. targets clonotypes that share the same V and J segment information and have nearly identical CDR3 regions. We define the similarity of two CDR3 regions based on its *Hamming distance*, i.e. require the two sequences to be of the same length. Since clonotype calls with CDR3 regions out of frame are discarded, it is highly unlikely for sequence artifacts to cause a false clonotype call through insertions and deletions.

As described in Section 1.2.2, *sequencing errors* potentially come with a quality score indicating that a base call and thus a detected clonotype may be false. On the other hand, PCR error induced clonotypes cannot be distinguished from real, but low frequency ones. Our approach therefore distinguishes between the two error types and allows the user to decide how far the error correction should go with respect to either type of error. The method is furthermore guided by a *majority vote*, i.e. we assume that any experimental step (PCR, sequencing) will produce the desired, error-free outcome in *most* of the cases and that errors occur less often. The main idea of the error correction strategy is therefore to identify pairs of clonotypes that fulfill *clustering criteria* based on the ideas described above and to assume that the less frequent clonotype of the two is a technical artifact of the more frequent clonotype.

The exact formulation of the clustering criteria as well as the actual clustering strategy performed on the identified clonotype pairs that fulfill those criteria is described in the following sections.

4.5.2. Clustering Criteria

Given a major clonotype c_H with a count f_H and a minor clonotype c_L with a count f_L where $f_H > f_L$, we consider c_L to be an erroneous version of c_H and apply a correction if all of the following conditions are met.

Frequency ratio We expect the minor clonotype to occur substantially less often than the major clonotype. However, due to the systematic nature [62, 7, 63, 64] of both sequencing and PCR errors, a hard threshold on the abundance of the minor clonotype is not applicable. We therefore require that

$$\frac{f_L}{f_H} < r_{\max} \qquad\qquad 0 < r_{\max} < 1,$$

i.e. that the ratio of the frequencies of both clonotypes does not exceed $r_{\max}$.

Gene segment match The clonotypes have to match with respect to their identified V and J segments. In order to be maximally sensitive when choosing the cluster targets, we consider an overlap in the sets of identified gene segments to be sufficient, i.e.

$$c_L^V \cap c_H^V \neq \varnothing \ \wedge \ c_L^J \cap c_H^J \neq \varnothing.$$

CDR3 sequence similarity The critical choice is that of the CDR3 sequence similarity, i.e. whether we consider differences in the CDR3 sequence as true biological or falsely induced technical diversity. As initially stated, we want to allow the parameterized correction of both PCR and sequencing errors. To be able to evaluate the preliminary clonotypes based on the sequencing quality scores, we keep track of the *mean quality score* over all contributing reads of every position inside the CDR3 region. Given a clonotype c, we denote the mean quality score of the ith CDR3 position as $q(c_i^{\text{CDR3}})$. Given c_L and c_H with a CDR3 length of $|c_L^{\text{CDR3}}| = |c_H^{\text{CDR3}}| = \ell$, let

$$\mathrm{E}(c_L, c_H) = \left\{ i \mid c_L^{\text{CDR3}}[i] \neq c_H^{\text{CDR3}}[i],\ i \in \{1, \dots, \ell\} \right\}$$

be the positions at which the CDR3 sequences of c_L and c_H differ, i.e. $|E(c_L, c_H)| = d_H(c_L^{\text{CDR3}}, c_H^{\text{CDR3}})$. Furthermore, let

$$\begin{gathered} \mathrm{Q}_{\text{low}}(c) = \left\{ i \,\middle|\, q(c_i^{\text{CDR3}}) \leq t,\ i \in \{1, \dots, \ell\} \right\}, \text{with} \\ t = \operatorname*{median}_{i=1}^{\ell} \left(q(c_i^{\text{CDR3}}) \right) - s_{\min} \cdot \operatorname*{sd}_{i=1}^{\ell} \left(q(c_i^{\text{CDR3}}) \right) \end{gathered}$$

denote the positions within the CDR3 sequence of clonotype c, whose quality score is at least s_{min} standard deviations lower than the median quality score across the CDR3 sequence. Given two error thresholds e_q and e_s, the CDR3 sequence of the less abundant clonotype c_L^{CDR3} then has to fulfill

$$\max\left(0, |\mathrm{E}(c_L, c_H) \cap \mathrm{Q}_{\text{low}}(c_L)| - e_q\right) + |\mathrm{E}(c_L, c_H) \setminus \mathrm{Q}_{\text{low}}(c_L)| \leq e_s$$

for c_L and c_H to be considered for clustering. In words, we allow up to e_q errors in c_L^{CDR3} that are explained by a drop in quality and additionally e_s errors that are not explained by a drop in quality. If there are less than e_s errors with no quality drop but more than e_q quality related errors, the remaining “slots” for quality unrelated errors can be used for errors at low quality positions.

Note that the CDR3 sequence criterion does *not* take the quality of the major clonotype c_H into account. This choice was made to account for the often systematic nature of sequencing errors - if a position is likely to produce an erroneous call, many reads have a low quality (but correct) base call at that position. We therefore solely rely on the abundance when assessing whether a clonotype is more likely to be correct than another, and not on the quality of the more frequent clonotype.

4.5.3. Clustering Algorithm

With the clustering criteria defined, we can now inspect all pairs of clonotypes within the preliminary repertoire and identify those that fulfill the requirements on the frequency ratio, gene segment match and CDR3 sequence similarity. Initially, all pairs of clonotypes that fulfill the frequency ratio criterion are checked for the remaining criteria and, if they are met, stored for later processing as shown in Algorithm 4.3.

After all pairs are defined, the redistribution of counts is performed from the least frequent minor clonotype to the most frequent minor clonotype, allowing for a hierarchical clustering: a clonotype c will only be subject to redistribution after all clonotypes have been checked that might redistribute their reads to c. This hierarchical redistribution approach also resolves the potentially false redistributions towards other low-quality clonotypes discussed above, as the reads can be redistributed further towards the correct clonotype subsequently.

Example An example for this behaviour is shown based on three clonotypes taken from a real dataset in Figure 4.7. The figure shows clonotype c_L in comparison with two other clonotypes $c_{H,1}$ and $c_{H,2}$ that have been identified as cluster targets for c_L. The read counts fulfill $c_{H,1} \gg c_{H,2} > c_L$. The CDR3 sequences of c_L and $c_{H,1}$ disagree only at position 27, at which c_L has a low quality base. c_L and $c_{H,2}$ additionally disagree at position 12, again with a low quality score associated in c_L. The clustering algorithm would therefore treat c_L as erroneously generated from $c_{H,1}$ and $c_{H,2}$ and redistribute f_L between $f_{H,1}$ and $f_{H,2}$, maintaining their ratio. $c_{H,2}$ is probably not the true biological origin of c_L, however, this choice remains without effect since later on,

```
function ErrorCorrection($(\mathcal{C}, \mathcal{F})$)    ▷ $(\mathcal{C}, \mathcal{F})$ is sorted asc. by ct count
    $\mathcal{M}_i \leftarrow \varnothing \; \forall i \in \{1, \ldots, N\}$    ▷ The major clonotype IDs for minor CT $i$
    for $i \leftarrow 1, \ldots, N$ do    ▷ Parallel execution
        for $j \leftarrow N, \ldots, (i+1)$ do
            if $f_i / f_j > r_{\text{clust}}$ then
                Break    ▷ Subsequent reads have lower counts
            else if CheckCriteria($c_i, c_j$) then
                $\mathcal{M}_i \leftarrow \mathcal{M}_i \cup \{i\}$
            end if
        end for
    end for
    for $i \leftarrow 1, \ldots, N$ do
        $\mathcal{F}^* \leftarrow \{f_j \;| f_j \in \mathcal{F}, j \in \mathcal{M}_i\}$    ▷ Collect major ct counts
        RedistCounts($f_i, \mathcal{F}^*$)    ▷ Redistribute minor count to major counts
    end for
end function
```

Algorithm 4.3: The main error correction routine. The input of the routine is a repertoire $(\mathcal{C}, \mathcal{F})$ of size N, sorted by the clonotype read counts, i.e. $f_1 \leq f_2 \leq \cdots \leq f_N$. The clonotype pairs are processed in two stages. First, all pairs of clonotypes are checked for the clustering criteria. Second, the clonotype read counts are redistributed in a hierarchical fashion, from the least abundant to the most abundant clonotype.

```
function RedistCounts($f_L, \mathcal{F}_H = f_{H,1}, \ldots f_{H,N}$)
    SortDescending($\mathcal{F}_H$)    ▷ Sort major clonotypes largest to smallest
    for $i \leftarrow 1, \ldots, n$ do    ▷ First round of redistribution
        $f^{\text{rel}}_{H,i} = f_{H,i} / \sum_{j=0}^{N} f_{H,j}$    ▷ Relative major ct freq
        $v = \lfloor f^{\text{rel}}_{H,i} \cdot f_L \rfloor$
        $f_{H,i} = f_{H,i} + v; \quad f_L = f_L - v;$
    end for
    for $i \leftarrow 1, \ldots, f_L$ do    ▷ Second round of redistribution
        $f_{H,i} = f_{H,i} + 1; \quad f_L = c_f - 1;$
    end for
end function
```

Algorithm 4.4: The redistribution of clonotype read counts aims to maintain the frequency ratio of the major clonotypes receiving the clonotype read counts. If the ratios cannot be preserved perfectly, the redistribution favors larger clonotypes as a target.

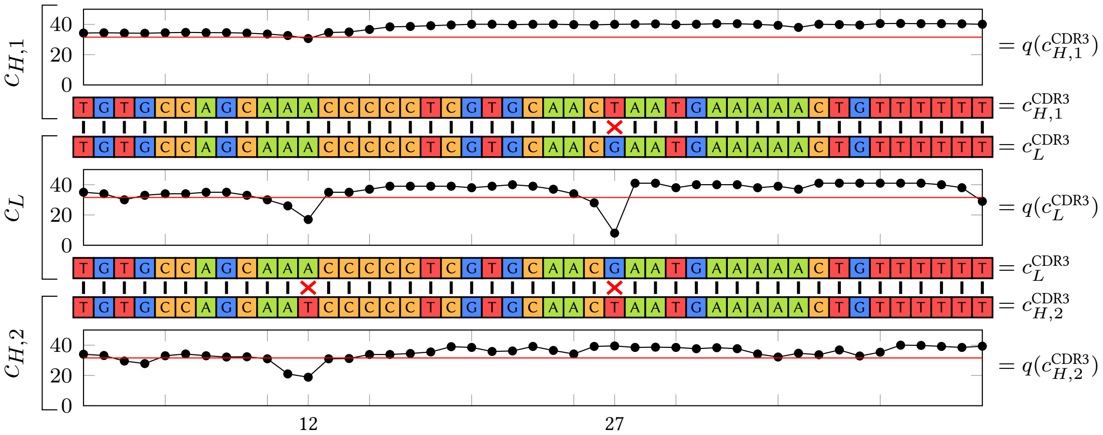


Figure 4.7.: An example for a minor clonotype c_L (center, CDR3 sequence shown twice) and two valid major clonotypes $c_{H,1}, c_{H,2}$ (top and bottom) taken from real sequencing data, with the corresponding frequencies fulfilling $f_{H,1} \gg f_{H,2} > f_L$. Alongside the pairwise alignments between the CDR3 sequences of c_L and both $c_{H,1}$ and $c_{H,2}$ the mean sequencing quality values for each base of each CDR3 sequence are shown. The threshold for low quality bases for $s_{\min} = 1$ is indicated in red.

when $c_{H,2}$ is inspected as a minor clonotype, its read count will be distributed further to $c_{H,1}$.

When read counts are redistributed from one minor clonotype to multiple major clonotypes, the read count ratios of the major clonotypes are preserved as good as possible. In borderline cases, more abundant major clonotypes are preferred. The exact redistribution strategy is shown in Algorithm 4.4.

4.6. Repertoire Finalization

Lastly, some final modifications to the repertoire generated so far are made, aiming to remove remaining artifacts from the preliminary clonotype repertoire $(\mathcal{C}, \mathcal{F})$.

4.6.1. Segment Ambiguity Resolution

As stated in Section 4.4.5, equally scoring gene segments lead to multiple gene segment assignments for single clonotypes. In many cases, this is unavoidable due to experimental constraints, e.g. depending on the primer design, read length and CDR3 length some sets of gene segments might differ only in regions that are not captured. In other cases, however, such ambiguities might be caused by technical artifacts or unspecific primer hybridization leading to a shift in the captured sequence. This only affects a fraction of the reads encoding the same clonotype and therefore might split the clonotype into multiple clonotypes with a different number of gene segment assignments. The same effect can also be caused by a fallback mode implemented for paired end data: If a V read is discarded due to a low average quality as described in Section 4.2, but the corresponding V(D)J read is not, the latter is analyzed with the standard single end routine. For some clonotypes this may result in a less precise V segment assignment and thus cause a split situation as well.

To capture those resolvable ambiguities, we apply a routine similar to that of the sequence error clustering described in Section 4.5.3, just that we now enumerate pairs of clonotypes $c_L, c_H \in \mathcal{C}$ with equal CDR3 sequences $c_L^{\text{CDR3}} = c_H^{\text{CDR3}}$ and a subset relation in both segment sets, i.e.

$$c_L^V \subseteq c_H^V \;\wedge\; c_L^J \subseteq c_H^J.$$

At most one of the terms can be fulfilled by equality, since all clonotypes $c \in \mathcal{C}$ are unique by definition. For every observed CDR3 sequence the corresponding clonotype pairs are then ordered from large to small gene segment sets and iteratively redistributed as previously described. The entire algorithm is described in Appendix A.1.

4.6.2. Removal of Low Quality Clonotypes

As initially stated in Section 4.5, the aim of the standalone error correction is to preferably reduce the number of false clonotype calls by correction rather than by filtering. This is of course not entirely possible, thus after we have exhausted all the described methods, we remove remaining clonotypes c with

$$\operatorname*{mean}_{i=1}^{|c^{\text{CDR3}}|} \left(q(c_i^{\text{CDR3}}) \right) < q_{\text{min}}^{\text{CDR3}},$$

where $q_{\text{min}}^{\text{CDR3}}$ is a user defined parameter. To properly complement the initial read filtering, the parameters should be chosen such that $q_{\text{min}}^{\text{read}} < q_{\text{min}}^{\text{CDR3}}$, i.e. to initially allow more reads to go into the analysis and discard those clonotype clusters with low quality CDR3 sequences after the correction has been exhausted.

4.7. Implementation

We implemented the clonotyping method described in this chapter in an application called IMSEQ. It was implemented in C++ 14 as a command line utility, designed for Unix-like systems. It uses SeqAn [65], a generic library for sequence analysis with a strong focus on biological data, which provides implementations of a large variety of string comparison algorithms such as the alignment filter and alignment computation methods discussed in this chapter. IMSEQ furthermore supports multithreading to execute time critical routines in parallel. In particular, the following methods are parallelized: the pairwise read comparisons computed for the UMI based error correction as indicated in Algorithm 4.1, the main per read clonotyping routine described in Section 4.4 and the standalone error correction as indicated in Algorithm 4.3.

The software was licensed under the *GNU General Public License (GPL) Version 3*, the source code and binaries are freely available from www.imtools.org. The tool features a variety of user-definable options and diagnostic outputs alongside the main clonotype frequency information.

4.7.1. Gene Segment References

The most common set of reference sequences for the various germline components of immunoglobulin and T cell receptor genes for a growing number of species is the one provided by the *International Immunogenetics Database (IMGT)* [66], IMGT/Gene-DB [67]. IMSEQ requires the reference sequences for the V and J gene segments of the species and receptor chain of interest in a FASTA [68] file following a particular annotation syntax within the FASTA IDs. Most importantly, the IDs have to specify the position of the Cys and Phe triplets within each sequence. The exact format is described in Appendix A.2. The publicly available IMSEQ releases already come with pre-compiled reference files based on the IMGT/Gene-DB entries for human T cell receptor α and β chains as well as for human immunoglobulin heavy and light chains.

4.8. Summary

In this chapter, we have described IMSEQ, an error-aware method to annotate reads from Rep-Seq experiments and generate clonotype repertoires. It uses fast and robust alignment algorithms to annotate the incorporated V and J segments to each input sequence. The gene segment alignment is highly parametrizable, allowing the user to configure the underlying algorithms depending on the experimental setup, data quality and desired performance regarding the computational speed. At the same time, the implementation comes with carefully chosen, thoroughly tested pre-defined or automatically tuned parameter settings.

The application is able to process both single end and non-overlapping paired end data. Data from protocols resulting in overlapping paired end data can easily be integrated by either preprocessing them with a fusion tool for overlapping read pairs [69] or by using the integrated trim function in IMSEQ to artificially reduce the read lengths. IMSEQ is able to handle antigen receptor gene sequence data from both T cells and immunoglobulins and from any species, as long as the corresponding germline gene reference data is available.

We have furthermore described the effect of technical artifacts such as fluctuating read abundances due to PCR amplification biases as well as sequence artifacts due to PCR and sequencing errors. Two types of approaches were discussed and are available in IMSEQ: a *standalone error correction*, that optionally takes into account single nucleotide quality scores from the underlying sequencing platform and a *UMI based error correction* that can make use of UMIs if available.

5. IMSEQ Evaluation

In this chapter we will briefly describe other clonotyping methods that have been published and evaluate the performance of IMSEQ in comparison to those methods. A main aspect of this evaluation will be an assessment of each method's error correction capabilities. For this purpose we developed a simulation framework, which we apply in order to generate Rep-Seq datasets under different conditions with respect to sequencing and PCR errors. We will furthermore evaluate the available methods using real, experimental data and assess the influence of some of IMSEQ's parameters on its performance.

5.1. Existing Methods

Multiple tools with partially different objectives exist in order to solve the clonotyping problem. Prominent reference implementations that serve as de facto gold standards for the sole per read annotation include those from the IMGT, i.e. V-QUEST [70], and IgBlast [71], the clonotyper provided and maintained by the *National Center for Biotechnology Information (NCBI)*.

In this section, we will give a brief overview over existing tools and methods related to the analysis of Rep-Seq data. We will then benchmark the performance of IMSEQ and those tools with a comparable scope for various tasks and datasets, both on simulated and real data. A summary of those methods and their features is additionally shown in Table 5.1.

5.1.1. IMGT/V-QUEST

IMGT/V-QUEST [70] is a web-based clonotyping tool, designed for the detailed analysis of single clonotypes. It is limited to 50 input sequences and can report various features of each recombined gene sequence in a number of output formats such as plain text, HTML pages or Microsoft Excel sheets. The reported features go beyond the V and J segment alignments or the CDR3 sequence identification. They furthermore contain a detailed analysis of the segment junctions, a mapping into the IMGT numbering space [16], individual mutation statistics, etc. Thus, it is primarily designed for the in-detail analysis of single or few gene sequences.

Some of the limitations were overcome when the high-throughput successor,

	IMSEQ	MITCR	MIXCR	Decombinator	TCRklass	VIDJIL	IgBlast	V-Quest	HighV-Quest
Supported data									
TCR	yes	yes	yes	no	no	yes	yes	yes	yes
IG	yes	no	yes	no	no	no	yes	yes	yes
split PE	yes	no	yes	no	yes	no	no	no	no
Performance									
HTS data	yes	yes	yes	yes	yes	yes	no	no	semi[1]
Multithread	yes	yes	yes	no	no	no	no	web	web
Error correction									
UMI based	yes	no[2]	no[2]	semi[3]	no	no	no	no	no
standalone	yes	yes	yes	no	semi	no	no	no	no

Table 5.1.: Feature comparison of available clonotyping tools. [1] HighV-Quest supports the submission of up to 500,000 input sequences. [2] The developers of MITCR and MIXCR provide a separate tool [47] for the UMI based correction of NGS reads which can be used to preprocess the data. [3] Decombinator only supports the exact UMI protocol used in-house by the authors without further modifications to the source code.

IMGT/HighV-QUEST [72], was made available. It again is implemented as a hosted web service, requiring a one-time registration and the submission of the data to the providers. Initially restricted to 150,000 sequences, since late 2013 up to 500,000 sequences can be submitted. Those limits still don't allow for standard, deep high-throughput repertoire capturing and the analysis times span between hours and days. The output is still very detailed and with a per-gene focus. Neither of the tools actually considers the repertoire as a whole, i.e. takes into account relationships between the detected clonotypes for error or frequency correction. Consequently, UMI sequences are also not supported. Every species for which the IMGT maintains a germline reference database can be analysed.

5.1.2. IgBlast

NCBI's IgBlast [71] is kind of the standalone, open-source pendant to the IMGT/V-QUEST tools. It too stems from the low-throughput sequencing era of clonotype analysis and reports detailed information for each input sequence without looking at the clonotype repertoire as a whole. Besides the standalone tool, a web service [73] is provided, which provides access to a subset of the IMGT maintained species. While the NCBI initially maintained its own database of germline TCR and IG sequences, it now frequently synchronizes with the IMGT reference database and adds a few pseudogenes not contained in the upstream IMGT reference. Despite the name, IgBlast supports the analysis of both T cell receptor and immunoglobulin sequences. IgBlast is implemented in C++, open source and public domain software as "United States Government Work" under the terms of the United States Copyright Act.

5.1.3. MITCR

MITCR [74] is a standalone application, limited to the analysis of T cell receptor gene sequences. It uses a fixed q-gram seed located at the Cys and Phe boundaries and extends alignments for all reference segments both away from and into the CDR3 region. Matches and mismatches are scored differently for high and low quality bases, with a fixed scoring scheme allowing values within $\{-2, -1, 1, 2\}$. The best matching segment and every segment with an alignment difference of at most 4 to the optimal one are reported. MITCR is furthermore capable of correcting technical errors. Regarding sequencing errors, reads are binarily classified as "high" or "low" quality reads. Depending on the user, the low quality clonotypes can either be discarded or mapped to the best matching high quality clonotype. For PCR errors, an edit-distance and frequency ratio based approach is implemented. MITCR defines clonotypes as identical when they share the same CDR3 sequence, i.e. it is not capable of distinguishing clonotypes with identical CDR3s but different V and / or J segments. MITCR is implemented in Java, open source and licensed under the GPL v3.

5.1.4. MIXCR

The successor of MITCR, MIXCR [75], is able to analyse both T cell receptor and immunoglobulin genes. It is also written in Java, freely available and open source. The V and J segment assignment now uses a seed based filter [76] and a refined error correction strategy that takes the exact error positions into account. Also the CDR3-defined clonotype restriction was lifted. MIXCR is open source and licensed under a custom license.

5.1.5. TCRklass

TCRklass [77] is yet another q-gram based clonotyper. It builds a q-gram index over the V and J reference sequences and then counts the number of matching q-grams in each read. The V and J segments that yield the most hits are assigned. The CDR3 region is then defined strictly based on the q-hit location for those q-grams that incorporate the Cys and Phe motifs. The error handling is limited to the removal of reads that are likely to yield a false CDR3 region. Reads with only high quality bases in the CDR3 region are always accepted, whereas reads with a low quality base in the CDR3 region are dropped if no identical sequence exists among the high quality cohort. Unique clonotypes are defined as unique combinations of V segments, J segments and CDR3 regions. As the name suggests, the tool is limited to the analysis of TCRs, most importantly because no alignments between the input sequences and the V and J segments are computed, making an assessment of somatic hypermutations impossible. TCRklass is implemented in Perl and C++, freely available and licensed under the GPL v3.

5.1.6. Decombinator

Decombinator [78] is a Python based clonotyper, freely available and open source. It performs the V and J segment matching using the exact pattern matching algorithm by Aho and Corasick [79]. The approach is based on unique substrings for each reference gene segment, which are searched for using the exact matching algorithm. To incorporate errors, the authors extended this approach to allow a hamming distance of at most one. The standard output comprises a custom clonotype definition that consists of five features, V ID, J ID, number of V errors, number of J errors and the sequences of the non-V, non-J parts of the CDR3 region. The software does, however, offer an option to postprocess the data and recover the traditional V, J, CDR3 clonotype information. Newer versions of Decombinator support the use of UMIs, however, strictly bound to the in-house UMI protocol used by the authors, who state that the method might work on other protocols after some modification to the python code. The sequencing and PCR error correction in Decombinator is limited to the germline encoded regions of the gene.

5.1.7. VIDJIL

Another tool that comes with its very own definition of a clonotype is VIDJIL [80, 81]. Its clonotyping method is based on a q-gram index which stores every q-gram that occurs among the reference gene segments as either *unique V*, *unique J* or *ambiguous V/J*. Every read is then inspected for matching q-grams from the pre built index and checked for consistency (e.g. if J q-grams match after V q-grams etc.). The *clonotype* is then defined as the sequence of length w (default $w = 40$) centered between the last matching V q-gram and the first matching J q-gram location within the read. The w-windows are not very robust against errors, in particular those that occur in the last (or first) q base pairs of the V (or J) region. To enhance the method's error robustness, the authors extended the approach to gapped q-grams and added a posterior clustering that identified w-windows that are likely to originate from the same clonotype. However, due to the definition of the w-window, this is again a method with a strong focus on the CDR3 region that is most likely not able to distinguish clonotypes with identical CDR3 sequences but different V and J segments.

The VIDJIL suite furthermore consist of an analysis server that can either be self-hosted or used as a service. The focus of the analysis tool is wider than just the task of clonotyping, i.e. offers a whole tool chain for biologists to study and visualize information gained from repertoire analysis. To provide access to the traditional clonotype information, the VIDJIL web interface features a direct interface to IgBlast and other traditional clonotypers to inspect the w-window clonotypes in more detail. VIDJIL is open source and GPL v3 licensed.

5.1.8. pRESTO

pRESTO [82] is a pre-processing tool for Rep-Seq data. It can perform various steps prior to and independently from the actual clonotyping, such as the removal of errors and duplicates based on UMIs, the fusion of overlapping reads, clipping of Rep-Seq specific primers and quality filtering. It does not perform any clonotyping itself though, i.e. is solely meant to complement an existing clonotyping tool. After the removal of PCR duplicates and collapsing erroneous reads based on UMIs, the preprocessed data could even be suitable for the analysis through IMGT/HighV-QUEST, as suggested by the authors. pRESTO additionally complements the sequence IDs with information about the aggregation process, enabling a subsequent analysis taking into account the original sequence abundances. It is implemented in Python, open source and licensed under the CC BY-SA 4.0 license.

5.1.9. Commercial Providers

Apart from publicly available standalone or hosted clonotyping solutions, a number of commercial providers have emerged providing both the experimental as well as the data analytical component of TCR and IG repertoire analysis. Since their clonotyping methods are not disclosed, we cannot benchmark and compare them to other methods. They do however play a major role in practice, as Rep-Seq is a translational field research and there is a large need for certified, ready to use solutions.

5.2. Simulation

In order to assess and compare the performance of clonotyping tools, we developed a simulation method called *IMSIM*, which can be used to generate recombined TCR and IG gene sequences, given a set of reference V, D and J segment sequences. The formation of the segment junctions is fully parametrizable, as described in the following section. Additionally, we added the possibility of simulating the processes of PCR amplification and applied already available methods to simulate the sequencing process.

5.2.1. V(D)J Recombination

To generate a recombined gene sequence given the sets of gene segment reference sequences $\mathcal{S}^V$, $\mathcal{S}^D$ and $\mathcal{S}^J$, the algorithm reproduces what is known about the biological process of V(D)J recombination (see Section 2.3.3). At first, the D and J segments are joined. The junction build process has 7 parameters:

$$\begin{aligned}
n_D^{D3}, n_D^{J5} \in \mathbb{N}^0 &:= \text{ the number of deletions at the 3' D and 5' J end}\\
n_P^{D3}, n_P^{J5} \in \mathbb{N}^0 &:= \text{ the number of palindromic insertions at the 3' D and 5' J end}\\
n_N^{D3}, n_N^{J5} \in \mathbb{N}^0 &:= \text{ the number of random insertions at the 3' D and 5' J end}\\
n_O^{DJ} \in \mathbb{N}^0 &:= \text{ the length of the DJ overlap.}
\end{aligned}$$

Equivalent parameters $n_D^{V3}, n_D^{D5}, n_P^{V3}, n_P^{D5}, n_N^{V3}, n_N^{D5}$ and n_O^{VD} exist for the V-DJ junction, which is formed after the DJ junction. For every recombination event, these parameters are drawn from a user specified discrete distribution. A built-in default, which is also used for the evaluations shown in the next section, was manually tuned to produce clonotypes with a CDR3 length distribution close to that observed in real datasets. The probabilities for each V segment as well as for each DJ segment combination can be manually specified as well, otherwise a uniform distribution is assumed.

To determine a junction sequence, at first the desired number of deletions is performed from either end. Since the different gene segments contain a different number of nucleotides beyond the Cys and Phe triplets into the CDR3 region, the drawn values for n_D cannot always be realized, i.e. they are constrained by the available number of nucleotides. After the deletion, the palindromic and random sequences are added according to n_P and n_N. After the modifications on either sequence have been performed, they are overlapped by n_O nucleotides and mismatches are resolved using a fair coin toss. Again, restrictions apply in order to prevent modifications that go beyond the Cys and Phe triplets.

5.2.2. PCR Amplification

Additionally, the amplification step of a Rep-Seq experiment is simulated using a basic PCR model. It starts given the set of sequences produced by the simulation. In every PCR cycle, every sequence has a probability of p_r to be replicated. During the replication, every nucleotide is replaced with a probability of p_e, while the probability of the replacement nucleotide is even among the three available choices. The user can choose between two alternative termination criteria, either a fixed number of PCR cycles n_C which will produce varying numbers of sequences or a target number of sequences n_Q, i.e. the PCR is terminated “mid cycle”.

5.2.3. Implementation

IMSIM was implemented in C++ 11 without any external dependencies, licensed under the GPL v3 and is publicly available under www.imtools.org.

5.2.4. Sequencing

The sequencing process was simulated using *Mason 2* [83], a free open source software for the simulation of Illumina, 454 and Sanger reads. For our evaluations we used Mason's "fragment sequencing" mode in the Illumina configuration, where an input file with multiple sequences is provided and Mason simulates equally long read pairs of a specified length from either end of the provided fragments.

5.3. Method Evaluation

In this section, we will evaluate IMSEQ's performance, both on its own regarding an assessment of parameters and their influence, as well as in comparison to other methods. The web based V-Quest tools and IgBlast are not designed for the analysis of high throughput data and also simply perform an independent, per read analysis of the data without looking at the repertoire as a whole. Thus they are excluded from the tool comparison, with the exception of IgBlast which we will initially use as a gold standard to proof the validity of the simulated data. While MITCR was evaluated in the original IMSEQ publication [84], we now evaluate its successor MIXCR, since the authors officially deprecated MITCR in favor of MIXCR. VIDJIL and pRESTO have a different scope than IMSEQ and in practice depend on the use of an additional clonotyper. They are therefore also not considered in the comparison.

At first we will look at each tools performance on simulated, error free, full length TRB gene sequences. We will then assess the impact of the segment core fragment length and the error rate parameters on the computational time to annotate the data with the clonotype information required by IMSEQ. Furthermore, we will simulate amplicon sequencing data with both PCR and sequencing errors. Lastly, we will compare each method's performance on real data both from in-house experiments as well as based on publicly available data.

Generally, we perform all evaluations based on V and J gene segment identity and the *nucleotide* sequence of the CDR3 region. The allele information for V and J segments is dropped, i.e. we only consider whether the segment has been correctly identified on the gene level. Since Decombinator does not provide an output format that contains the untranslated CDR3 sequences, the evaluation for this method is performed based on the amino acid sequence of the CDR3 region, i.e. false positive calls of Decombinator are not counted as long as they are silent mutations of a correct clonotype.

If not explicitly stated otherwise, IMSEQ was applied using its default settings.

Among other parameters, this includes a minimum average read quality of $q_{\text{min}} = 20$ in combination with a posterior threshold of $q_{\text{min}}^{\text{CDR3}} = 30$, a J SCF length of $L_{\text{SCF}}^{J} = 12$, a V SCF length automatically tuned as described in Appendix A.3, maximum gene segment error rates of $\varepsilon_V = 0.05$ and $\varepsilon_J = 0.15$ and a corresponding pair of maximum SCF errors $e_{\text{SCF}}^{V|J} = \lceil \varepsilon_{V|J} \cdot L_{\text{SCF}}^{V|J} \rceil$.

To avoid a potential bias and since not all methods support non-overlapping paired end data, all evaluations were run on single-end simulated or real data. All measurements were taken on a Intel Xeon X5675 3.06 GHz dual CPU system with 12 (+12) cores and 64 GB RAM running Mac OS X 10.11.

5.3.1. Evaluation Criteria

We perform the evaluation on two levels: On the *read level* we assess the extent to which the underling clonotype repertoire has been correctly reconstructed based on the read counts assigned to each clonotype. Let $\mathcal{F} = \{f_1, \ldots, f_N\}$ be the detected and $\mathcal{F}^{\text{ref}}\{f_1^{\text{ref}}, \ldots, f_N^{\text{ref}}\}$ the simulated absolute clonotype frequencies. We then define the number of false positives, false negatives and true positives as follows:

$$\#FP := \sum_{i=1}^{N} \min(f_i - f_i^{\text{ref}}, 0) \tag{5.1}$$

$$\#FN := \sum_{i=1}^{N} \min(f_i^{\text{ref}} - f_i, 0) \tag{5.2}$$

$$\#TP := \left(\sum_{i=0}^{N} f_i \right) - \#FP. \tag{5.3}$$

On the *clonotype level* we evaluate only whether a clonotype was detected or not, independent of its frequency. This evaluation gives us insights, for example in how far the clonotype diversity has been overestimated due to technical artifacts or whether a method fails to annotate certain recombinations entirely.

5.3.2. Error Free Simulated Data

We simulated 500,000 unique, error free, full length human TRB gene sequences using IMSIM as described in Section 5.2. We use this dataset to validate the different methods under perfect conditions, i.e. with no gene segment ambiguities due to partial sequences or PCR and sequencing artifacts. The gene sequences were then presented to each tool as reads, without actually simulating a sequencing step.

The results of the evaluation are shown in Table 5.2. We included IgBlast to confirm the validity of the V region sequences simulated and all sequences were clonotyped correctly. Since IgBlast correctly annotated all the provided sequences, we conclude

		Clonotype		**Read**			
Tool	**Settings**	**Precision**	**Recall**	**#assigned**	**Precision**	**Recall**	**Time [s]**
IgBlast	*default*	*1.0000*	*1.0000*	*500,000*	*1.0000*	*1.0000*	*> 3 h*
IMSEQ	no clust	**1.0000**	**1.0000**	**500,000**	**1.0000**	**1.0000**	**P 42**
IMSEQ	default	**1.0000**	**1.0000**	**500,000**	**1.0000**	**1.0000**	**P 50**
MIXCR	no clust	**1.0000**	**1.0000**	**500,000**	**1.0000**	**1.0000**	P 52
MIXCR	default	**1.0000**	**0.9999**	**500,000**	**0.9999**	**0.9999**	P 254
Decombinator	default	0.9598	0.9592	499,947	0.9593	0.9592	139
TCRklass	default	**1.0000**	**0.9988**	499,395	**1.0000**	**0.9988**	476

Table 5.2.: The tool performance comparison on simulated, error-free full length TRB V-region sequences. IGBLAST is included as a gold standard for the clonotyping of error-free data in order to confirm the validity of the simulated sequences. Tools that support multithreading were used with 24 threads and their running times marked with "P". Optimal values are highlighted in green, values less than 1% below the corresponding optimum are shown in boldface.

that our simulation framework is valid. We can furthermore observe that apart from not being able to perform a repertoire error correction, IgBlast is also unsuitable for high throughput data due to the extremely long computational time required - clonotyping the 500,000 unique sequences took more than three hours on the test system.

IMSEQ also clonotypes all input sequences correctly, both with the repertoire error correction explicitly disabled ("no clust") and with the default settings. Even though there are many clonotype pairs similar enough to be considered equal by the clustering criteria described in Section 4.5.2, the maximum clustering ratio prevents a false correction since all clonotypes are unique and thus equally abundant. Furthermore, also MIXCR properly clonotypes all input data when the repertoire error correction is explicitly disabled. With the default settings, MIXCR falsely collapses some clonotypes - even though the main clustering routine correctly reports that nothing can be clustered due to the missing hierarchy between the clonotypes. According to the application log, a "pre-clustering" routine though falsely corrects the data. This only affected a very small fraction of the clonotypes with two particular V gene segments. Since they are falsely detected as other, true clonotypes, the clonotype precision is still 100%, while it slightly drops at the read level. Since this effect seems to be limited to a well defined subset of the data with no obvious explanation, this is more likely a bug than a true methodological limitation.

The remaining two methods, Decombinator and TCRklass, actually falsely annotate a significant fraction of the data. Decombinator yields both a precision and a recall of $\sim 95\%$, on a read as well as on a clonotype level. A manual inspection of the produced data revealed problems related to certain gene segments that are apparently generally falsely assigned. Furthermore, some of the false annotations appear to be related to modifications of the germline V and J region inside, but close to the boundaries of the CDR3 region. Lastly, TCRklass does not falsely assign clonotypes to the simulated sequences, but fails to clonotype a small fraction of the data. The problem appears to affect clonotypes incorporating almost any V and J segment, a particular pattern could not be derived.

Regarding the computational time required for assigning the 500,000 unique clonotypes it is important to note that IMSEQ and MIXCR support multithreading, while the other tools do not. IMSEQ and MIXCR with disabled standalone error correction routines perform the raw clonotyping rather fast, IMSEQ processes the data in 42 s and MITCR in 52 s. When using the default settings, which in both cases include a repertoire error correction step, IMSEQ and MIXCR require 50 s and 254 s respectively - while in either case no clustering is performed due to the uniform distribution of clonotypes (with the exception of the "pre-clustering" step of MIXCR). Decombinator and TCRklass, the single threaded methods, terminate after 139 s and 476 s respectively. It should be noted that IMSEQ scales sub-linearly with the number of threads, i.e. using only a single thread the dataset was processed in 185 s.

5.3.3. IMSEQ Alignment Parameters

The runtime performance of IMSEQ is primarily determined by the SCF length ($L_{\text{SCF}}^{V|J}$) and error ($e_{\text{SCF}}^{V|J}$) parameters, as well as by the permitted error rate for the V and J segment overlap alignments $\varepsilon_{V|J}$.

- The *length of the segment core fragments* has a strong influence on how many gene segments have to be validated. Too short SCFs might not be unique across multiple gene segments and thus trigger a high number of overlap alignments. Too long segment core fragments on the other hand might exceed the amount of gene segment available in a read and thus lead to discarded reads.
- The *number of errors allowed within the segment core fragments* is directly bound to the specificity of the filter method described in Section 4.4.3. A high number of allowed errors in turn requires more SCFs to be validated, while a low number of allowed errors might not tolerate the technical artifacts present in the data.
- The *gene segment overlap alignment error rate* has a direct influence on the time required for each overlap alignment following an SCF match. A rather permissive error rate yields wider bands with higher running times, while a too strict error rate again might not account for errors in the data.

The applicable range for the segment core fragment lengths L_{scf}^{V} and L_{scf}^{J} are constrained by the number of V and J gene segment bases guaranteed to be present in each input sequence (see Section 4.4.2). The performance impact of those values is primarily related to the uniqueness of the SCFs regarding the corresponding gene segments. For the human TRBV gene segments this is illustrated in Figure 5.1, which shows the maximum number of V gene segments associated with the same SCF for different SCF

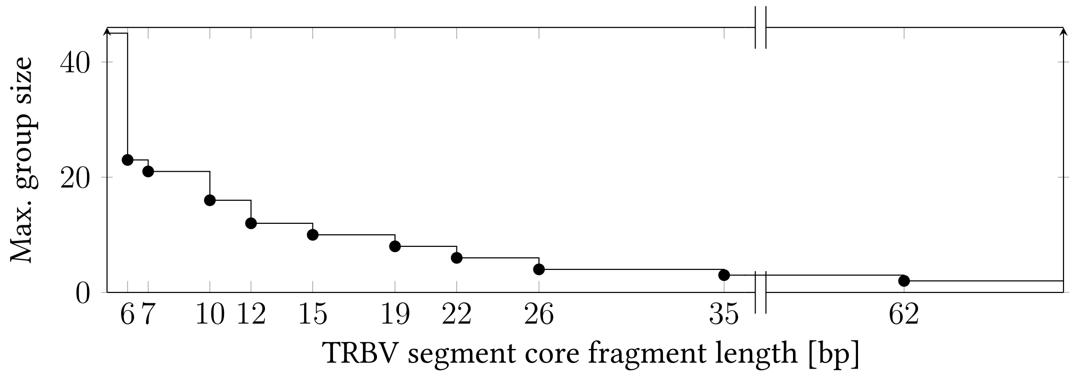


Figure 5.1.: The maximum number of human TRBV segments mapping to the *same* segment core fragment for different SCF lengths.

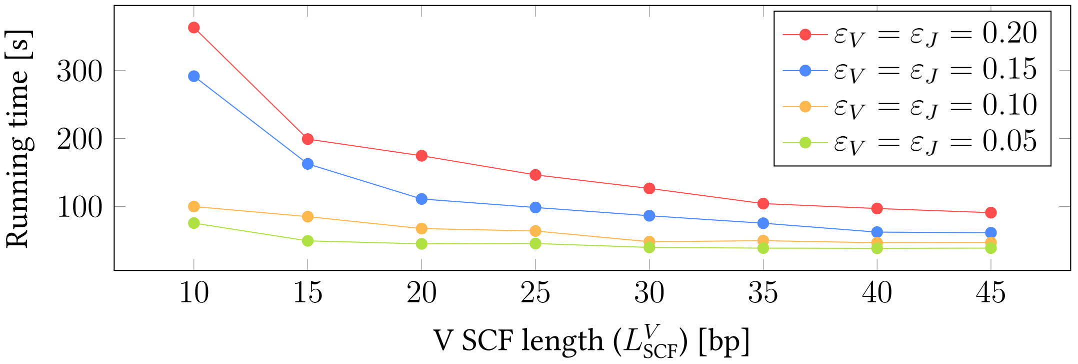


Figure 5.2.: The IMSEQ running time for the analysis of 500,000 unique unique clonotypes for various combinations of the V SCF length L^V_{SCF} and the maximum V and J segment error rate $\varepsilon_V = \varepsilon_J$.

lengths. While for $L^V_{\mathrm{SCF}} < 10$ more than 20 V gene segments might contain the same SCF, for $L^V_{\mathrm{SCF}} > 25$ the SCFs are almost unique.

The parametrization of the segment core fragments is more critical for the V gene than for the J gene. In contrast to the V fragment, the covered J area is usually well defined by the chosen primer locations. Furthermore, in the TRB gene there are fewer J than V gene segments and they are additionally more diverse. For $L^J_{\mathrm{SCF}} \geq 15$ the segment core fragments are unique for this particular gene.

To understand the impact of those parameters, we tested the method's runtime performance on the dataset of 500,000 unique clonotypes for different V SCF lengths L^V_{SCF}. Additionally, we varied the permitted error rates for the V and J gene segment overlap alignments $\varepsilon_{V|J}$. The maximum number of SCF errors $e^{V|J}_{\mathrm{SCF}}$ is coupled to the values of $\varepsilon_{V|J}$, i.e. set to the smallest integer value that ensures the same error rate on the SCFs. The results for four different error rates and V SCF lengths from 10 to 45 are shown in Figure 5.2. As expected the running time decreases with stricter error rate constraints and larger V SCF segment lengths. While for shorter SCFs the error rate has larger impact on the running times, the difference becomes smaller for longer SCF lengths. While larger error rate thresholds imply computational cost induced by wider bands in the overlap alignments, longer SCFs strongly reduce the number of those alignments that have to be computed in the first place, even for relatively high error rates.

While it is not possible to define a universal set of parameters, especially since the SCF lengths depend on the implemented protocol and the primer locations for the targeted enrichment PCR, we estimate a set of parameters depending on the provided reference data and the input data. The implemented heuristic for setting these parameters is described in Appendix A.3.

5.3.4. Simulated Errors

Next, we extended the simulation to incorporate technical artifacts as we would expect them to occur in real data and generate two datasets. Both datasets are based on an initial simulation of 1,500 human TRB gene sequences, which are additionally subjected to the PCR simulation routine using a PCR replication probability of $p_r = 0.8$ and a PCR base substitution error rate of $p_e = 10^{-4}$ to generate $n_Q = 1.5 \cdot 10^6$ sequences as described in Section 5.2.2. Dataset 1 is then truncated to a uniform sequence length of 150 bp, while Dataset 2 is generated by passing the generated gene sequences on to Mason in order to simulate 150 bp Illumina reads. Regarding the Illumina error model, Mason's default parameters were used. We again evaluated each tool's performance against the simulated ground truth on the clonotype and read level, which is shown in Table 5.3.

To illustrate the effect of technical artifacts on the data and the corresponding over estimation of diversity present in a sample we initially show the result acquired using IMSEQ without any posterior correction (denoted as "no clust"). Even though 98% of the sequences are clonotyped correctly, the remaining sequences containing artifacts inside the critical CDR3 region or in distinctive positions within the V and J segments lead to the detection of 1,925 (Dataset 1) and 14,610 (Dataset 2) clonotypes instead of the simulated 1,500. This results in a clonotype precision as low as 0.10 in the latter case, which includes sequencing errors.

In their default settings, which involve a standalone error correction, both IMSEQ and MIXCR reliably correct the sequence errors present in the data. On the clonotype level, IMSEQ reaches a precision of 0.99 with only PCR errors and 0.90 with PCR and sequencing errors, while being fully sensitive in either case. MIXCR reaches a clonotype precision of 0.95 and 0.68, being fully sensitive as well. Due to their limited ability to handle errors, the performance of Decombinator and TCRklass is relatively low. While they still yield a clonotype precision of $\sim$ 0.80 without sequencing errors, it drops to 0.11-0.16 when sequencing errors are added. In the latter case, TCRklass reports over 13,000 clonotypes instead of the simulated 1,500, which is almost as much as IMSEQ reports with no correction at all.

		Clonotype			**Read**		
Tool	**Settings**	**# total**	**Precision**	**Recall**	**Precision**	**Recall**	**Time [s]**
Dataset 1: PCR errors							
IMSEQ	*noclust*	*1,925*	*0.7792*	*1.0000*	*0.9897*	*0.9892*	*P 2*
IMSEQ	default	**1,512**	**0.9921**	**1.0000**	**0.9971**	**0.9966**	**P 2**
MIXCR	default	1,585	0.9464	**1.0000**	**0.9932**	**0.9892**	P 14
Decombinator	default	1,744	0.8217	0.9553	0.9495	0.9485	17
TCRKlass	default	1,866	0.8028	**0.9987**	**0.9907**	**0.9874**	140
Dataset 2: PCR and sequencing errors							
IMSEQ	*noclust*	*14,610*	*0.1027*	*1.0000*	*0.9016*	*0.8903*	*P 4*
IMSEQ	default	**1,658**	**0.9047**	**1.0000**	**0.9959**	**0.9835**	**P 5**
MIXCR	default	2,201	0.6815	**1.0000**	**0.9884**	0.8966	P 18
Decombinator	default	9,088	0.1578	0.9560	0.8968	0.8673	19
TCRklass	default	13,188	0.1137	**0.9993**	0.9110	0.8879	107

Table 5.3.: The tool performance comparison on 1,500 unique clonotypes with simulated posterior PCR amplification to $1.5 \cdot 10^6$ sequences (Dataset 1) and subsequent Illumina 2x150 paired end sequencing simulation (Dataset 2). Optimal values are highlighted in green, values less than 1% below the corresponding optimum are shown in boldface.

With respect to the running times, IMSEQ is by far the fastest of the tested methods, processing the dataset with PCR and sequencing errors in 5 s, compared to 18 s for MIXCR, 19 s for Decombinator and 107 s for TCRklass.

5.3.5. Real Data

In order to assess each method's standalone error correction abilities on real data, we make use of samples that were prepared using unique molecular identifiers. We initially analyse the data using IMSEQ *without* the standalone error correction ($e_s = e_q = 0$) but with the UMI based read correction enabled ($\delta_{\text{UMI}} = 1, \varepsilon_{\text{UMI}} = 0.05, r_{\text{UMI}} = 0.2$). We then use the result as a ground truth to evaluate each tool's performance regarding the UMI *independent* error correction.

Since clonotypes that are detected by another method but could not be detected by IMSEQ during the reference annotation would increase that other method's false positive count, we limit the sequences to those that IMSEQ was able to annotate. While this comparison is nevertheless not entirely bias-free, with each tool's performance on perfect data in mind it still gives us valuable insights in the relative error correction capabilities of the methods when compared to each other.

We perform the evaluation on two independent UMI labelled datasets that were prepared by different labs using different Rep-Seq protocols:

- **Dataset 1: In-house multiplex PCR data**

  An in-house dataset from a healthy individual. The sample was prepared based on DNA with an initial, 5 cycle linear PCR step using only unique J primers including a UMI 10-mer. Subsequently, a regular PCR step was performed, using multiple V specific forward primers and a single reverse primer targeting a binding site introduced during the linear PCR step. The dataset contains 11.7 m reads.

- **Dataset 2: External data produced using the template switch protocol**

  A publicly available external dataset from a sample obtained from an individual participating in a Leukemia study [85] at baseline (patient p744 at day 0). The sample was prepared using an RNA based template switch protocol as previously described by Shugay et al. [47], which does not require a multiplex PCR and is therefore less prone to PCR induced frequency biases. The dataset contains 5.7 m reads.

Each dataset was filtered for only those reads that show the intended primer + UMI signature, i.e. were likely successfully prepared according to the protocol. After generating the UMI corrected dataset, each tool was run using its default parameters, i.e. including the available standalone error correction measure but without taking the UMI sequences into account.

The results for both datasets are shown in Table 5.4. Since the results no longer show a clear ranking of the methods with respect to both precision and recall, we added the F_1 score to the evaluation, which is defined as the harmonic mean of the two:

$$F_1 := 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Again, we see that the standalone error correction abilities of Decombinator and TCRklass are limited - Decombinator's clonotype precision is only slightly above that of IMSEQ with no error correction. TCRklass performs better than Decombinator but still worse than IMSEQ and MIXCR, reaching a clonotype precision of $\sim$ 0.77 for either dataset.

IMSEQ and MIXCR perform similarly. For Dataset 1, MIXCR yields a higher precision but a lower recall than IMSEQ. When using the F_1 measure, IMSEQ clearly performs better on the clonotype level, whereas MIXCR performs slightly better on the read level. When looking at the second, template switch protocol based Dataset 2, IMSEQ clearly outperforms MIXCR with regard to the standalone error correction capabilities for all measures taken into account.

It should be noted that the tools generally perform better in this evaluation compared to the simulated PCR and sequencing errors, even though the error rates in the simulation were not set particularly high. This is likely due to the used reference

		Clonotype			**Read**		
Tool	**Settings**	**Precision**	**Recall**	**F_1 Score**	**Precision**	**Recall**	**F_1 Score**
Dataset 1: In-house multiplex PCR test data							
IMSEQ	*no clust*	*0.6748*	*0.9993*	*0.8056*	*0.9755*	*0.9730*	*0.9743*
IMSEQ	default	0.9320	0.8936	**0.9124**	0.9819	0.9797	**0.9808**
MIXCR	default	**0.9525**	0.8081	0.8744	**0.9871**	**0.9779**	**0.9825**
Decombinator	default	0.7032	0.8585	0.7731	0.9019	0.8868	0.8943
TCRklass	default	0.7683	**0.9314**	0.8420	**0.9795**	**0.9710**	**0.9752**
Dataset 2: External data produced using a template switch protocol							
IMSEQ	*no clust*	*0.6200*	*0.9984*	*0.7650*	*0.9737*	*0.9692*	*0.9714*
IMSEQ	default	**0.9933**	**0.9848**	**0.9890**	**0.9966**	**0.9930**	**0.9948**
MIXCR	default	0.8926	0.9086	0.9005	0.9550	0.9427	0.9488
Decombinator	default	0.6404	0.7475	0.6898	0.8976	0.7227	0.8007
TCRklass	default	0.7746	0.8333	0.8028	0.9411	0.8115	0.8715

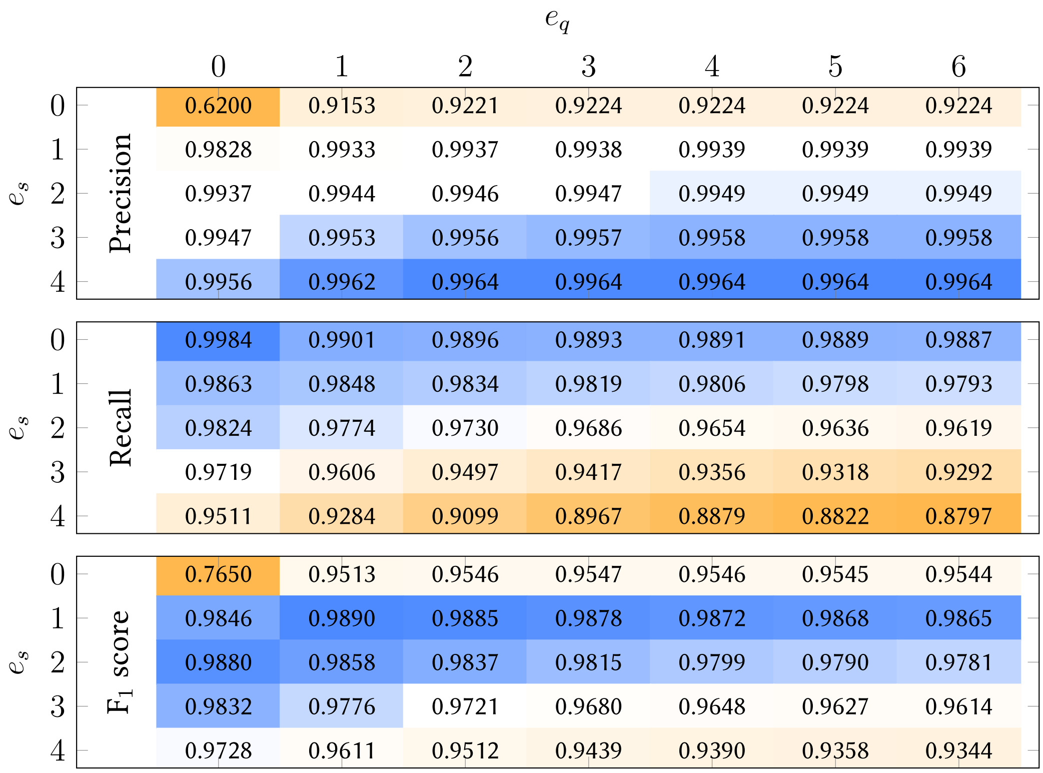
Table 5.4.: Standalone error correction evaluated on two real datasets. The reference dataset was generated using IMSEQ with UMI based and no standalone posterior error correction. Each tool was then run only with the standalone error correction enabled. Additionally, the results for running IMSEQ without any error correction is shown for reference. Optimal values are highlighted in green, values less than 1% below the corresponding optimum are shown in boldface.

dataset - the purely UMI based correction will likely not correct all errors present in the sample and thus present a “simpler” task to the competing tools. Nevertheless, it provides meaningful insight regarding the relative performance of each tool compared to another.

5.3.6. Error Correction Parameter Choice

To better understand the interplay between the error correction performance and the used parameters, namely the number of allowed quality related errors, e_q, and the number of allowed unrestricted errors, e_s, we tested a wider range of parameter combinations on the public dataset from the template switch protocol sample. The results for $e_s \in \{1, \ldots 4\}$ and $e_q \in \{1, \ldots, 6\}$ with a fixed maximum cluster ratio of $r_{\max} = 0.25$ are shown in Figure 5.3. One can clearly see that allowing a single error has the largest impact on improving the clonotyping precision. With $e_s = e_q = 1$, i.e. up to two errors in the CDR3 region, we already reach a precision of 0.99, which does not improve significantly for more permissive correction settings - while drastically reducing the recall at some point due to false correction events.

We can furthermore see that the quality score based error correction can indeed be



Precision

e_s \ e_q	0	1	2	3	4	5	6
0	0.6200	0.9153	0.9221	0.9224	0.9224	0.9224	0.9224
1	0.9828	0.9933	0.9937	0.9938	0.9939	0.9939	0.9939
2	0.9937	0.9944	0.9946	0.9947	0.9949	0.9949	0.9949
3	0.9947	0.9953	0.9956	0.9957	0.9958	0.9958	0.9958
4	0.9956	0.9962	0.9964	0.9964	0.9964	0.9964	0.9964

Recall

e_s \ e_q	0	1	2	3	4	5	6
0	0.9984	0.9901	0.9896	0.9893	0.9891	0.9889	0.9887
1	0.9863	0.9848	0.9834	0.9819	0.9806	0.9798	0.9793
2	0.9824	0.9774	0.9730	0.9686	0.9654	0.9636	0.9619
3	0.9719	0.9606	0.9497	0.9417	0.9356	0.9318	0.9292
4	0.9511	0.9284	0.9099	0.8967	0.8879	0.8822	0.8797

F_1 score

e_s \ e_q	0	1	2	3	4	5	6
0	0.7650	0.9513	0.9546	0.9547	0.9546	0.9545	0.9544
1	0.9846	0.9890	0.9885	0.9878	0.9872	0.9868	0.9865
2	0.9880	0.9858	0.9837	0.9815	0.9799	0.9790	0.9781
3	0.9832	0.9776	0.9721	0.9680	0.9648	0.9627	0.9614
4	0.9728	0.9611	0.9512	0.9439	0.9390	0.9358	0.9344

Figure 5.3.: The precision, recall and their harmonic mean (F_1 score) for various combinations of the e_q and e_s parameters, computed based on the real data error evaluation, Dataset 2 (template switch protocol).

used more confidently with respect to the distinction of technical vs. true biological diversity: allowing a single quality related correction ($e_s = 0, e_q = 1$) yields a precision gain 35-fold of the corresponding loss in recall, while allowing a single unverified correction ($e_s = 1, e_q = 0$) yields a precision gain 30-fold of the loss in recall.

The equivalent analysis based on the in-house multiplex PCR based sample does not show such a clear distinction between the two error types, which is likely due to the fact that the error landscape in that sample is dominated by PCR artifacts due to the higher number of PCR cycles. The results for both samples at different maximum cluster rates r_{max} are shown in Appendix A.4.

5.3.7. UMI Frequency Impact

Lastly, we quantified the impact of a frequency normalization based on unique molecular identifiers, i.e. in how far the frequency vector describing the clonotype repertoire differs when built from the number of reads vs. the number of UMIs contributing to each clonotype. For this purpose, we again used the two real datasets described in the previous section.

This time, we enabled IMSEQ's UMI mode as in a normal production scenario, i.e. while otherwise using the default settings including the posterior standalone error correction. When UMIs are incorporated, IMSEQ can produce two counts for each observed clonotype, the number of reads and the number of UMIs. In either case, the clonotyping routine uses the UMI information to correct for sequence artifacts in the data. The UMI based repertoire additionally accounts for the effects of biased PCR amplification, thus the comparison of the two frequency vectors can give us an insight how strong the read abundance for each clonotype is affected by the PCR bias.

The impact on the (according to the UMI corrected frequencies) top ranking clonotypes is shown in Figure 5.4. Each sample contains a few dominating clonotypes (comprising up to 6% of the repertoire, not shown) which are relatively robust against PCR induced rank shifts. The following clonotypes however (as shown from rank 6 onwards) suffer from major false quantifications when using the read counts to estimate the repertoire composition. Subpopulations with clonotypes of similar abundances appear highly heterogeneous in the read count derived repertoire. While not being representative, the data from the two different enrichment protocols also shows a trend for the template switch protocol to be more robust against such biases, as the rank shifts and frequency heterogeneity in the observed interval is lower in that dataset. This is indeed expected due to the fact that the template switch protocol unlike the DNA based approach does not require a PCR with multiple forward and reverse primers, potentially performing at different efficiencies. Nevertheless, also in a PCR with only a single primer pair some fragments are amplified with a higher efficiency than others, e.g. due to their size or sequence, and even leaving such biases aside, the stochastic nature of the PCR process will likely lead to a skewed result. Our evaluations underline the necessity for UMIs to be both used on the experimental end and supported on the computational end, i.e. by any method for the generation of repertoires.

5.4. Summary

After having introduced the IMSEQ method in the previous chapter, we now evaluated its performance in the context of other existing methods. We initially compared the methods on a descriptive feature level, before evaluating their performance on simulated data. To generate such data, we developed a workflow that simulates the V(D)J recombination and the PCR amplification and combined it with an existing method to simulate Illumina reads. We then simulated both error free data and data that contains PCR and sequencing errors in order to evaluate the standalone error correction capabilities of each method. Additionally, we assessed the standalone error correction mechanisms on real data that was prepared with UMIs, using the UMIs to define a ground truth. The evaluations revealed that IMSEQ solves the clonotype annotation task optimally among the compared methods, with MIXCR performing comparably on one of the real datasets.

Furthermore, we inspected IMSEQs performance when varying critical parameters.

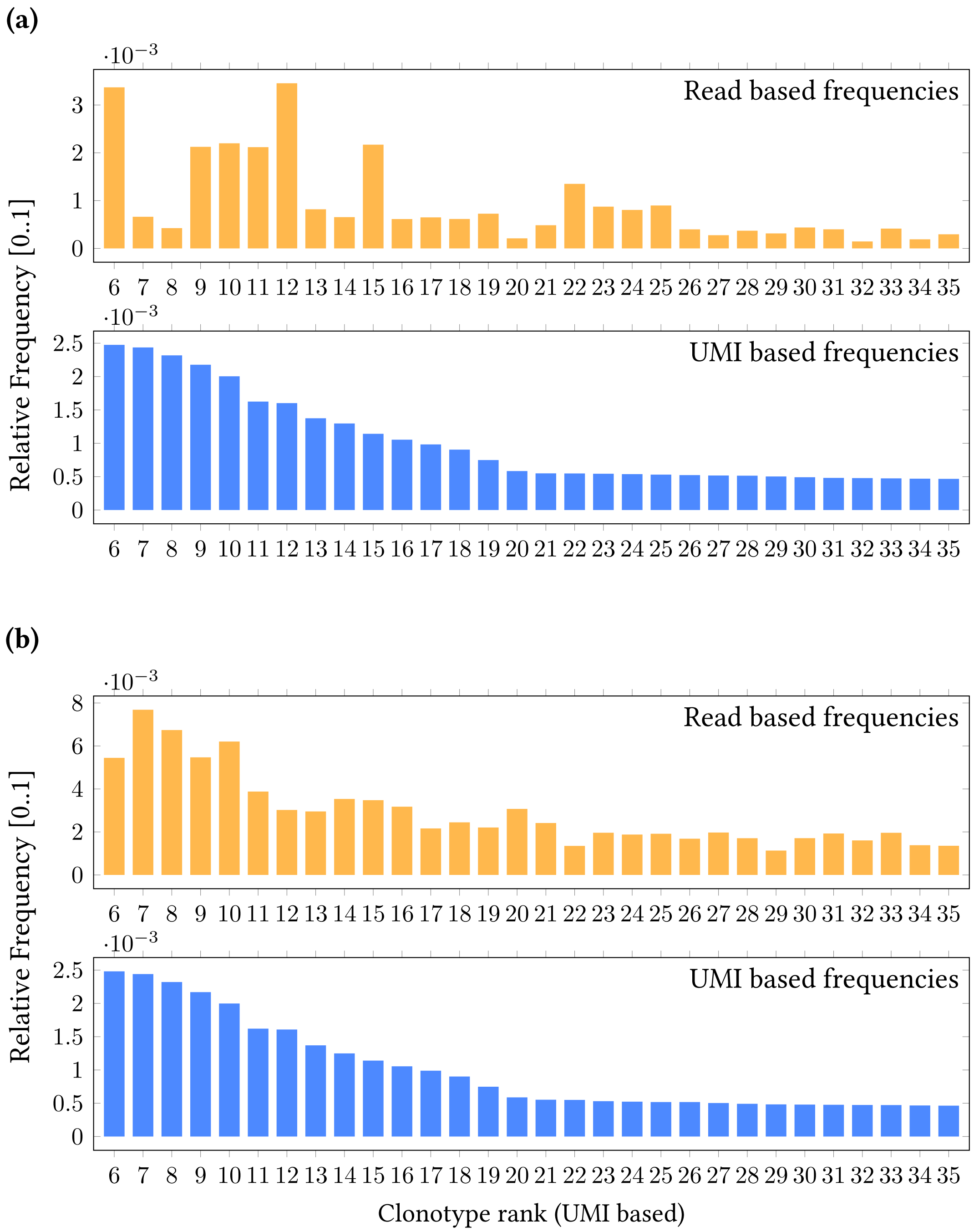


Figure 5.4.: The clonotypes on frequency ranks 6 to 35 and their relative frequencies **(a)** from an in-house sample prepared using a multiplex PCR protocol and **(b)** from a publicly available sample using a template switch protocol. In each case the clonotypes are ordered and selected based on the UMI based frequencies.

We evaluated the impact of the number of allowed errors during the V and J gene segment matching and the error thresholds for the standalone error correction to be able to define meaningful default settings and provide an insight how sensitive the method is to parameter changes.

Lastly, we showed that UMI based clonotype abundance estimates can deviate strongly from those derived from raw read counts. Our results show that the use of UMIs should be encouraged, if not become a standard operating procedure for Rep-Seq.

Part III.

Rep-Seq Applications

6. Clonotype Identity Based Applications

In this chapter, we will describe Rep-Seq applications that make use of the generated clonotypes as *identifiers* for subpopulations of antigen receptors. That is, these applications do not derive any functional information directly from the identified gene sequences, but use that sequence solely to estimate abundances of cells with identical function and to track them across different samples or points in time. In the first section of this chapter, we will talk about a clinical application that we assessed in the context of renal transplant patients, while the second section of this chapter gives a brief overview over other applications from literature.

6.1. Rep-Seq Driven Differential Diagnosis in Renal Transplantation

The adaptive immune system plays a major role in the context of *allograft transplantation*, i.e. the transplantation of tissue, primarily organs, from one individual to another. If there is a (partial) mismatch of HLA-haplotypes between the donor and the recipient, the graft cells will express MHC molecules that the recipients immune system has not been negatively selected against. Consequently, a cascade of immune processes is activated, causing a targeted response against the foreign tissue and leading to local and systemic inflammation. This can result in the deterioration of the graft tissue, a process known as *graft rejection*.

The recognition of foreign MHC molecules is both T and B cell driven [9], with the two factors being commonly referred to as *T cell-mediated rejection* (TCMR) and *antibody-mediated rejection* (ABMR) respectively. We further differentiate between two main types of graft rejection: *acute rejection* and *chronic rejection*. Acute rejection describes episodes of mostly T cell driven responses against the graft, which usually occur in the first months after transplantation. Chronic rejection occurs in highly vascularized grafts (such as kidney (renal), liver, heart or lung transplants), and is characterized by *fibrosis* of the grafts blood vessels [86]. The actual underlying mechanisms leading to long term graft failure due to chronic rejection are not well understood. It is believed to be caused by a complex of various factors, however, ABMR appears to play a major role [87, 88]. The advanced development of *immunosuppressant drug*s (ISDs) has substantially improved graft survival rates with respect to early acute rejection events, while chronic rejection still poses a major challenge and limits the graft survival time span [89].

The management of acute rejection episodes with ISDs does, however, cause other issues. Most importantly, the suppressed immune system is no longer able to protect the host from pathogens as effectively as under healthy conditions. If there are no effective medical strategies available against such pathogens, they pose a serious thread to the host and graft. In the following sections we will describe such a pathogen, the BK virus, its effect on renal graft recipients and in how far Rep-Seq can provide a solution for a critical differential diagnosis in affected patients.

6.1.1. BK Virus

The *BK virus* (BKV), named after the initials of the first patient it was observed in, was originally described by Gardner et al. [90] in 1971. It is a double stranded DNA virus with a genome size of approximately 5,300 bp which has a very high prevalence in the general population with seropositivity rates in the range of 60-100% [91]. Initial infection typically occurs during childhood and is usually accompanied by no or mild symptoms [91]. However, the virus remains persistent at a low level in the kidney in a significant number of cases [92]. This is typically not accompanied by any complications, as the immune system keeps the viral counts low. In immunodeficient individuals, however, the virus can reactivate from the dormant state and start replicating in the renal tissue. It can then cause a number of kidney related diseases [91], a situation referred to as *BK virus associated nephropathy* (BKVAN).

6.1.2. BKVAN as a Complication in Renal Transplantation

Due to the aforementioned treatment with ISDs in order to prevent and handle acute rejection episodes in organ recipients, these patients suffer from massive immunodeficiency and thus often from BKV reactivation. In individuals with renal grafts this poses a particularly complex situation, since the site of virus infection and the site of potential graft-specific immune response is the same.

From a clinical perspective, the condition is considered manifested when the graft function worsens. It is then crucial to differentiate between two potential causes: either, BKV reactivation or an episode of acute rejection. In the first case, the graft tissue is damaged due to the virus replication, while in the second case the damage occurs because of graft infiltrating T cells. Since there is no effective antiviral treatment against BKV available, the treatment options for either case are exactly complementary: in case of an acute rejection, ISD treatment would have to be *increased*, while an ongoing BKV reactivation would be treated by *decreasing* ISD administration in order to allow the hosts immune system to clear the virus infection.

As stated by Babel et al. [93], differentiating between these two potential causes of graft dysfunction is time critical. Not only does the ongoing damage to the graft tissue call for immediate action, but also an uprising BKV reactivation can only be countered effectively in an early stage. If ISD administration is reduced early enough, cytotoxic T cells can control the BKV reactivation and prevent a progression to BKVAN without causing a significant level of intra-graft inflammation in most cases. If, however, the BKV load rises above a certain threshold, the massive recruitment of T cells is not able to clear the viral infection soon enough before the T cells will also start infiltrating healthy graft cells in response to the foreign HLAs in addition to the BKV infected cells. The response that was originally targeted at the BKV infected cells has then turned against the graft and an ISD reduction is no longer advisable.

6.1.3. Differentiation of Acute Rejection and BK Virus Reactivation

The state of the art method to differentiate an episode of acute rejection from BKV reactivation is the histological examination of a tissue sample obtained through a renal biopsy in conjunction with electron microscopy and virological diagnostics. Tubular epithelial cells are inspected for cytopathic changes which serve as indicators for BKV infiltration of the observed tissue [94]. The procedure, however, has a relatively limited accuracy. Firstly, only a certain fraction of cells is affected by the virus, leading to a significant amount of false negative findings due to sampling when the biopsies are obtained [95]. Secondly, some of the visible changes caused by an acute rejection and those caused by BKV infiltration are similar in their morphological appearance, leading to false classifications [96]. In the following section we will describe a Rep-Seq based approach to differentiate an acute rejection episode from BKV infiltration, which is potentially more robust against these factors.

6.1.4. Rep-Seq Based Differentiation

The approach described here is based on Dziubianau et al. [97]. Instead of examining observable changes in the tissue itself, in the Rep-Seq based approach we aim to identify the cause of the graft tissue damage by examining the antigen specificity of the graft infiltrating T cells through Rep-Seq. Two alternative approaches were evaluated to acquire the subpopulation of T cells that is representative for the renal graft tissue infiltrating cells, cell extraction from a renal biopsy and from a urine sample. Apart from the graft infiltrating repertoire, two reference T cell samples are prepared to obtain repertoires comprising the recipients clonotypes specific for BKV and for the graft donors HLAs respectively. We then try to define the ongoing response in the graft based on these three repertoires.

The preparation of the three samples is outlined in Figure 6.1. The *graft infiltrating repertoire* is obtained directly either from urine or from a biopsy. To obtain the *BKV specific repertoire*, recipient *peripheral blood mononuclear cell*s (PBMCs), which can be

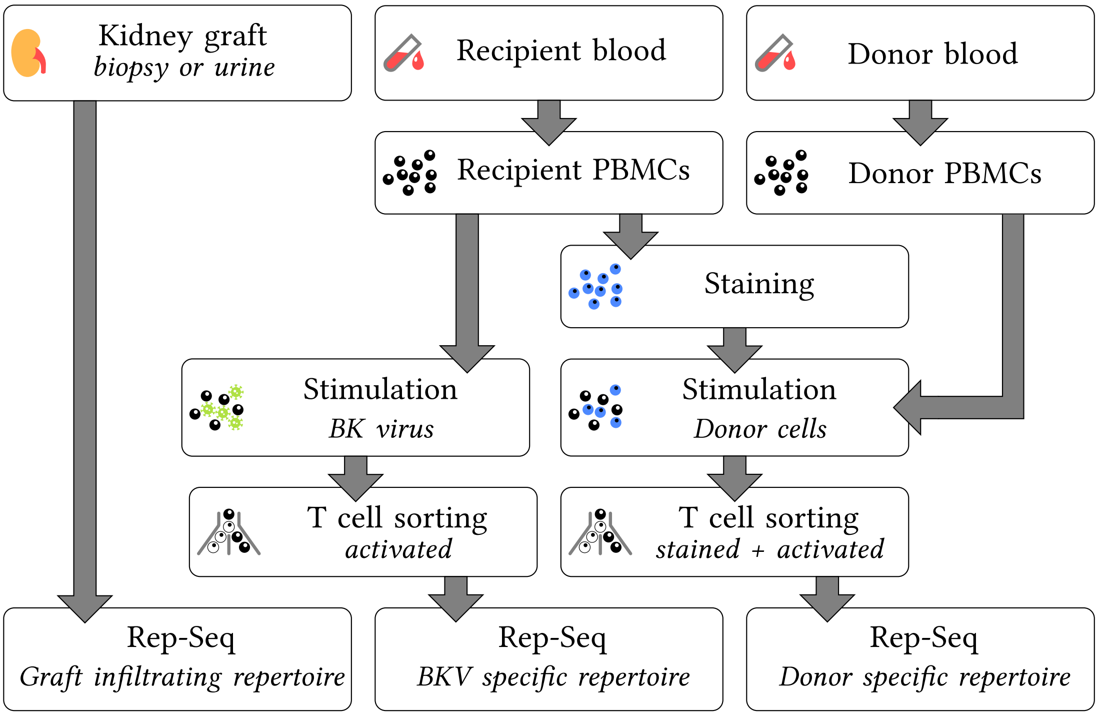


Figure 6.1.: The workflows to generate the three repertoires used in the Rep-Seq based diagnosis approach are obtained: the *graft infiltrating* repertoire in question as well as the two reference repertoires specific for *BKV* and *donor HLAs* respectively.

easily extracted from blood and include T cells, are stimulated with BKV antigens ex vivo. They are then sorted by surface markers specific for activated T cells, i.e. those that responded to the antigens. Lastly, a *graft specific repertoire* is measured by exposing recipient PBMCs to donor PBMCs ex vivo, triggering a response against the donor HLAs. To be able to subsequently distinguish the recipient and donor cells, the recipient cells are stained with membrane dyes prior to the donor cell exposure. The cells are then sorted again for activation markers and additionally for the recipient specific staining. Each of the three samples is then independently sequenced as previously described in Section 2.4.

The sorting step may include additional cell type refinements, such as CD4 or CD8. If cells from the donor are unavailable, e.g. due to a non-living donation, allograft specific T cells may be enriched using HLA type matched cells from an appropriate cell bank, as previously described by Landwehr-Kenzel et al. [98] in the context of allograft specific regulatory T cell enrichment.

6.1.5. Early Clinical Results

The workflow described in the previous section was applied for patients with acute cases of renal allograft dysfunction. A summary of the results for two patients is shown

	Patient 1		**Patient 2**	
Reference clonotypes	**Biopsy**	**Urine**	**Biopsy**	**Urine**
BKV CD4+	1.55%	0.00%	10.33%	14.81%
BKV CD8+	13.85%	1.10%	17.67%	33.83%
Allograft CD4+	n.a.	n.a.	0.00%	0.77%
Allograft CD8+	15.01%	0.62%	1.57%	0.00%

Table 6.1.: The cumulative relative frequencies of graft recipient CD4 and CD8 clonotypes that were identified as BKV specific or allograft specific within the repertoires generated from renal biopsies and urine. From Dziubianau et al. [97].

in Table 6.1. The reference clonotypes were defined as the predominant clonotypes in the BKV and allograft specific repertoires. The sorting step additionally differentiated between the CD4 and CD8 T cell subtypes. The table shows the cumulative frequencies of the reference clonotypes in the repertoires obtained from the renal biopsies and urine samples as well as the diagnosis provided by the histology lab.

Patient 1 showed a severely impaired renal function with a high BKV load in the blood (BK viremia). The histology report indicated BKVAN, but after a significant reduction of the ISD treatment the graft impairment continued to progress. The Rep-Seq based approach revealed that both BKV specific and allograft specific clonotypes were abundant in the graft, suggesting that the T cells infiltrating the graft had already progressed to react against the foreign tissue in addition to the BKV infected cells. This finding was unavailable through the histology based diagnosis and could explain the continuing graft failure. In contrast to the biopsy repertoire, the urine repertoire contained the reference clonotypes only at low frequencies or not at all.

In Patient 2 the BKVAN histology finding was confirmed by the Rep-Seq approach, as both BKV specific CD4 and CD8 clonotypes were abundant in the biopsy and urine repertoires, while allograft specific clonotypes were nearly absent.

Additional cases are shown in the supplement of the original publication [97]. While only a small number of patients was analyzed, the results indicate that the Rep-Seq based differential diagnosis for BKV infiltration vs. acute rejection in renal graft patients may be able to supplement or replace current histological methods. The mixed results in urine derived repertoires can most likely be explained by the low number of leucocytes in urine and their varying survival rates depending on the urine composition and related factors such as pH or electrolyte concentrations. If these can be controlled or adequately quantified in quality control procedures, urine based kidney repertoire sampling might provide a non-invasive proxy for a biopsy based diagnosis.

6.2. Other Rep-Seq Applications

To date there are few translational or already clinically applied Rep-Seq based methods. A prominent example exists in the context of hematologic malignancies, i.e. leukemias and lymphomas. A critical clinical parameter in these conditions is the *minimal residual disease* (MRD), which refers to a small number of malignant cells that remain in the patient after treatment. The assessment of the MRD, which is a major cause of relapse, has been shown to be a critical parameter to predict the clinical outcome of patients and to base treatment decisions on [99]. However, to date it has not been established as a standard diagnostic procedure, primarily because no practically applicable measurement method is available. The approaches that have been explored so far can be roughly divided into two categories: flow cytometry and *real-time quantitative PCR* (qPCR) based approaches, where qPCR is a method to detect or quantify a targeted DNA molecule using PCR [100].

Given a cell sample, the flow cytometry based approaches detect malignant cells based on surface markers that are known to be uniquely expressed in the affected cells. However, they feature a relatively low sensitivity in range of an affected cell concentration of 10^{-4} [101] and come with standard technical difficulties in the context of flow cytometry such as strong (operator dependent) variability. Also, it has been observed that the used tumor specific surface markers are to a large extent not expressed by tumor cells after chemotherapy, i.e. the cells are invisible to these detection methods after treatment [102].

qPCR based approaches involve the design of a tumor specific DNA probe, which is then used in the qPCR detection or quantification process. Such target sites are typically designed against rearranged genes, fusion gene transcripts or aberrant genes which are specific for the tumor [103]. Since the tumor develops from a single cell in most cases, in the case of hematologic tumors, i.e. malignant T and B cells, the V(D)J rearranged antigen receptor gene locus is also specific for the tumor and can therefore be used as a probe target. The method, however, requires sequencing the tumor before hand and the subsequent design and production of a specific probe, which makes this approach too labor intense to be applied rapidly in practice.

Rep-Seq based MRD detection and quantification approaches have therefore been suggested and implemented by multiple labs [104, 105, 106]. Since before treatment the cancer clonotype is by far more abundant than the healthy clonotypes, the identification within the sequenced repertoire can be performed solely based on the frequencies, i.e. without any experimental characterization of clonotypes. Due to the availability of sequencers that can rapidly sequence small DNA samples as they occur in clinical routine, such as the Illumina MiSeq or Life Technologies Ion Torrent, Rep-Seq can serve as a more rapid and less laborious alternative to qPCR approaches, while still being as sensitive, which was shown in the aforementioned studies.

Besides clinical applications, Rep-Seq has been applied to investigate fundamental immunological questions, such as the actual sizes of the T and B cell repertoires in humans, which have been estimated to range in the order of 10^6 unique clonotypes each, based on Rep-Seq studies [22, 104]. Also the number of unique clonotypes in certain functional subgroups of cells has been studied, e.g. it has been shown that memory T cell populations are more oligoclonal in comparison to naïve T cell populations [107], which agrees with the common understanding of the development of memory cell populations. Further studies have been conducted in the context of ageing, i.e. in how far clonotype diversity and distribution is different in elder individuals, or course of infection. Among others, Calis and Rosenberg [108] have reviewed the methodology that has emerged around Rep-Seq as well as more applications and results than mentioned here.

7. Functional Clonotype Characterization

In the previous chapter we have seen how Rep-Seq data can be used to derive information from clonotypes that have been characterized before by means of prior observations or additional experiments. In this chapter, we will discuss a potential approach to interpret clonotypes that haven't been previously characterized.

7.1. Introduction

So far we have used the clonotype information as an *identifier* for a subpopulation of the immune repertoire. Given that the underlying experimental and computational methods reconstruct the sampled immune repertoire accurately, we can e.g. track the abundance of pre-characterized clonotypes over time or in different tissues. What all such applications have in common is that one only uses the recovered antigen receptor gene sequence to make a statement whether two cells are functionally identical or not.

A far more challenging task is the question whether we can derive any kind of functional information from the observed clonotypes without a prior characterization. Due to the complex and highly individual nature of pMHC recognition by T cell receptors (see Section 2.3.2) and the randomness involved in the generation of the latter, it is hard to generalize such information from collected observations. Due to the polymorphism of MHC genes, even an identical clonotype found in two individuals might have a different function. On the other hand, two individuals that share a large number of MHC alleles or are even genetically identical may defeat the same pathogen with very different clonotypes, due to the independent development of each repertoire.

One approach to deal with these complications is the large scale collection of data, i.e. observed clonotype function under known MHC conditions. Both studies taking into account large cohorts [109] as well as suggestions for public databases designed to collect data from literature or researcher submissions [110, 111] have recently emerged. However, to date no comprehensive collection of clonotypes or widely accepted designs for such databases exist. Given the large number of TCR clonotypes and HLA allele combinations, the task of annotating all possible combinations is rather challenging.

Alternatively, one can attempt to *generalize* information from annotated clonotypes to make a statement regarding the functionality of other, previously unseen clonotypes. In this chapter we will describe and evaluate an approach based on supervised learning, more precisely on support vector machine classification.

7.2. Method Goals

The aim is to assign functional information to a previously unobserved clonotype based on training data that was presented to the model beforehand. As an evaluation environment we chose to classify T cell receptor β chain clonotypes based on their binary differentiation into either the CD4 or the CD8 subtype (see Section 2.3.2 and 2.3.6). The CD4-CD8 classification problem has several key advantages as a test environment for a clonotype learning model:

Binary nature of the biological system Rare CD4-CD8 double positive cells aside, the underlying biological mechanism is a binary differentiation pathway. Every cell, initially expressing both cofactors, looses either the CD4 or the CD8 phenotype during the early development in the thymus. On the computational end, multi-class prediction methods add an additional layer of complexity, thus working with a binary system at first yields the advantage of access to a variety of well established and less complex methods.

Class balance Other functionally defined groups of T cells, like the various subtypes of CD4 and CD8 T cells or subpopulations specific for a particular antigen, occur at significantly lower abundances. In the case of antigen specificity, a handful of T cell clonotypes may govern the response [112, 113], while the vast majority of the repertoire does not match. A strong imbalance between the classes that are to be classified makes the training and evaluation of a learning model more challenging, thus a balanced task is more suitable for an initial model evaluation. Since almost every T cell is of either the CD4 or CD8 subtype and both subtypes occur in the same order of magnitude (in healthy individuals the ratio is approximately 2:1 [114]), one can easily sample equally large subsets.

Data availability Since the CD4 and CD8 proteins are membrane bound coreceptors of the TCR, they can be directly stained with specific antibodies and separated using cell sorting methods. Due to their high abundance they can be easily sampled in high numbers from PBMCs, which are available as a byproduct from blood banks. Additionally, large amounts of data were available from in-house experiments, since cells have been sorted according to the CD4 and CD8 subtype for all transplant patients in the BKV study described in Chapter 6.

In the following sections we will first introduce the implemented methods and their background before we then describe how we evaluate their performance on the CD4-CD8 classification problem and discuss how they perform.

7.3. Binary Classification by Supervised Learning

The methods described in this chapter are based on the concept of *supervised machine learning*, which describes a range of approaches designed to estimate an unknown function

$$f : \mathcal{X} \to \mathcal{Y}$$

that assigns a non-observable value $f(x) \in \mathcal{Y}$ to any data point $x \in \mathcal{X}$ (often $\mathbb{R}^n$) comprising observable values. We refer to $\mathcal{X}$ as the *feature space*, to each $x \in \mathcal{X}$ as a *feature vector* and to each dimension of $\mathcal{X}$ as a *feature*, while $y \in \mathcal{Y}$ is the *output* residing in the *output space*. Generally, machine learning is applied when the existence of a relation between the input features and the output variable can be assumed, but is not understood, i.e. f is unknown. Instead, machine learning approaches try to derive an estimator g to represent the relation between the features and the output based on the data itself. In case of *supervised* machine learning, the function g is defined based on *labeled* input data, i.e. data for which the output is already known.

We will deal with *binary classification* problems based on n observable, real valued features in the following sections, thus the labeled input data of size N can be described as

$$\mathcal{D} := \left\{ (\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^n, y_i \in \{-1, 1\} \right\}_{i=1}^{N}.$$

7.3.1. Support Vector Machine Classification

*Support vector machine*s (SVMs) [115] (in their basic form) solve the above problem for *linearly separable* data, i.e. for data where an $n-1$ dimensional *hyperplane* that separates the data into the two classes exists. An example for linearly separable and non separable data is shown in Figure 7.1 (a) and (b) respectively. If the data is linearly separable, then the number of separating hyperplanes is infinitely large. The SVM method optimizes for the *maximum margin hyperplane*, i.e. the hyperplane with the largest Euclidean distance to the nearest data points, following the intuition that this solution is most robust against deviations from the training data. The problem of finding the maximum margin hyperplane is solved as an optimization problem.

Primal Form

Let the feature space be $\mathbb{R}^n$. A hyperplane in that feature space can then be described by

$$\mathbf{w}^\intercal \mathbf{x} + b = 0 \qquad\qquad \mathbf{w}, \mathbf{x} \in \mathbb{R}^n \text{ and } b \in \mathbb{R}, \tag{7.1}$$

i.e. the hyperplane comprises those points $\mathbf{x} \in \mathbb{R}^n$ that fulfill the equation. If $\mathbf{w}$ and b define a separating hyperplane, then let $(\mathbf{x}^*, y^*) \in \mathcal{D}$ be a labeled input data point

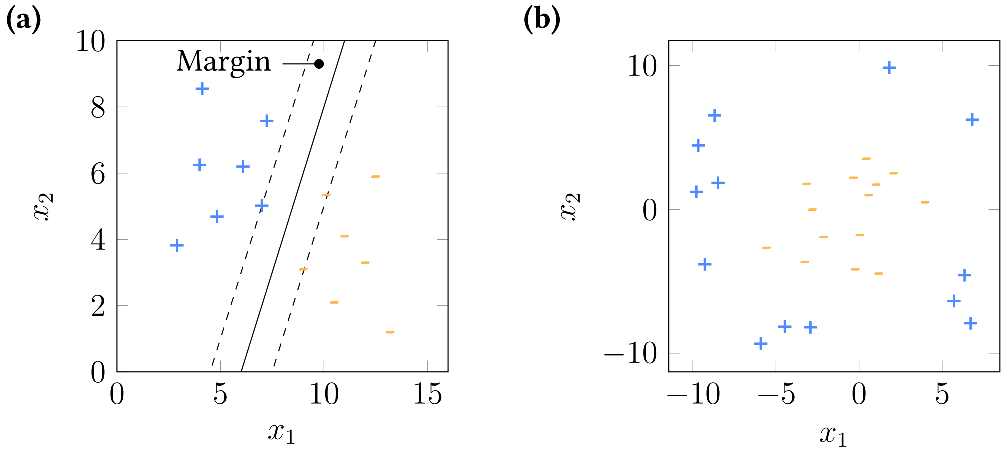


Figure 7.1.: Example data illustrating linear separability for $\mathbf{x} \in \mathbb{R}^2$ and $y \in \{-1, 1\}$ (visualized as + and -). **(a)** The data is linearly separable and the separating hyperplane (solid) corresponding to the maximum margins separating the data (dashed) is shown. **(b)** Data that is not linearly separable.

that is *closest* to the hyperplane. Since $\mathbf{w}$ and b are scale invariant with respect to the hyperplane they describe, they can be normalized by multiplication with a scalar. Thus, w.l.o.g. we can state that

$$|\mathbf{w}^\intercal \mathbf{x}^* + b| = 1. \tag{7.2}$$

Given that by definition the vector $\mathbf{w}$ is orthogonal to the plane, we can obtain the distance between the nearest point $\mathbf{x}^*$ and the plane (i.e. the size of the margin) by projecting $(\mathbf{x}^* - \mathbf{x})$ on $\mathbf{w}$, where $\mathbf{x}$ is any point on the hyperplane:

$$\left| \frac{\mathbf{w}}{\|\mathbf{w}\|}(\mathbf{x}^* - \mathbf{x}) \right| = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\intercal \mathbf{x}^* - \mathbf{w}^\intercal \mathbf{x}| = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\intercal \mathbf{x}^* + b - \mathbf{w}^\intercal \mathbf{x} - b| \overset{(1)}{=} \frac{1}{\|\mathbf{w}\|},$$

where (1) follows by substitution with Equations 7.1 and 7.2. Finding a solution for $\mathbf{w}$ and b that maximizes $1/\|\mathbf{w}\|$ can be formulated as an optimization problem. Since $\|\mathbf{w}\|$ is difficult to optimize, we equivalently minimize $\frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\mathbf{w}^\intercal \mathbf{w}$, resulting in the following quadratic optimization problem, given N training data points $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\mathbf{w}^\intercal \mathbf{w} \\ &\text{subject to} && y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1 && \forall i \in \{1, \ldots, N\} \end{aligned} \tag{7.3}$$

Dual Form

A standard approach to optimize a target function under equality constraints is its transformation into the *Lagrangian dual problem*. Using an extension that allows for inequality constraints, the *Karush-Kuhn-Tucker (KKT) conditions* [116, 117], the same approach can be used for the primal problem described above. The target function of the optimization problem is rewritten as follows:

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^\intercal\mathbf{w} - \sum_{i=1}^{N} \alpha_i (y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1), \tag{7.4}$$

where α_i are Lagrange multipliers. Using the partial derivatives for $\mathbf{w}$ and b

$$\begin{aligned} \nabla_{\mathbf{w}}\mathcal{L} &= \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = 0 \\ \frac{\delta\mathcal{L}}{\delta b} &= -\sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{7.5}$$

we can simplify Equation 7.4 and the optimization problem becomes

$$\begin{aligned} \text{maximize} \quad & \mathcal{L}(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^\intercal \mathbf{x}_j & (7.6) \\ \text{subject to} \quad & \alpha_i \geq 0 \qquad \forall i = \{1, \ldots, N\} & (7.7) \\ & \sum_{i=1}^{N} \alpha_i y_i = 0. \end{aligned}$$

The above problem can now be solved with respect to α using a *quadratic programming* (QP) solver. Given a solution, we can then use Equation 7.5 to obtain $\mathbf{w}$. To obtain b, we can use the original constraint from the primal form given by Equation 7.3, which has now become part of the target function. The equation is fulfilled with equality for data points with a corresponding $\alpha_i > 0$, the so-called *support vectors* which constrain the margin. It is guaranteed that $\exists\, i$ such that $\alpha_i > 0$, since the solution is otherwise not optimal. Given $\mathbf{w}$ and b we can now define the decision function as

$$g(\mathbf{x}) = \operatorname{sgn}(\mathbf{w}^\intercal\mathbf{x} + b) = \operatorname{sgn}\left(\left(\sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i\right)^\intercal \mathbf{x} + b\right).$$

The dual form has practical advantages over the primal form. The Lagrange multipliers α_i give us access to those data points that define the margin and consequently those data points that define the decision function. Their number is expected to be small compared to the total number of training vectors, which significantly decreases the complexity of the decision function. Furthermore, a major advantage of the dual problem formulation is that in order to define the optimization problem we only require the input data in form of all pairwise inner products $\mathbf{x}_i^\intercal\mathbf{x}_j$ as shown in Equation 7.6, which enables us to use kernel functions as will be described in Section 7.3.3.

Note that the size of Equation 7.6, which is passed on to the QP solver, is dominated by the $N \times N$ values generated from the pairwise inner products of the training feature vectors. Thus, the size of the problem for the solver depends on the number of training samples, but is independent of the number of observed features, i.e. the dimension of the feature vectors. The latter independence is exploited when implicit transformations into high-dimensional features spaces are performed (kernel functions), while the former dependence has practical implications with regard to the feasibility for larger training datasets. For cases where solvers will no longer be able to obtain an exact solution with applicable resources, heuristics can be used [118].

In practice, the linearity constraint on the data separation is usually not fulfilled. There are two main factors to be considered: outliers, i.e. rare exceptions that do not follow the general relation of the data, e.g. due to false measurements, and structural non-linearity of the separating function. Both cases can be dealt with within the SVM framework - outliers can be taken into account by relaxing the formulation to a soft-margin SVM and a non-linear relation of the data from the two classes can be dealt with using data transforms in the form of kernel functions. Both approaches are combined in practice and will be described in the following two sections.

7.3.2. Soft-Margin SVM

In the basic SVM formulation, no feature vector from the training data is allowed to violate the margin or even be misclassified, i.e. violate the hyperplane. Allowing for outliers to violate these constraints can be achieved by adding a slack variable ξ that relaxes the constraint relating the data points to the hyperplane. In the primal form, the problem is then defined as follows:

$$\begin{aligned} &\text{minimize} && \mathbf{x}^\intercal\mathbf{w} + C\sum_{i=1}^{N}\xi_i \\ &\text{subject to} && y_i(\mathbf{w}^\intercal\mathbf{x}_i + b) \geq 1 - \xi_i && \forall\, i \in \{1, \ldots, N\}, \end{aligned}$$

where C is a parameter to control the “softness” of the margin: $C \rightarrow \infty$ corresponds to the original, hard-margin SVM, while small values for C allow for a higher overall slack.

Conveniently, in the dual form the corresponding Lagrangian term drops out, thus the formulation of the objective function does not change, but the constraint on the single α_i in Equation 7.7 is changed to

$$0 \leq \alpha_i \leq C.$$

Note that the set of support vectors, i.e. feature vectors where the corresponding $\alpha_i > 0$, now comprises two types of support vectors: those that lie on the margin and those that violate the margin. The support vectors on the margin will be strictly smaller than C, i.e. fulfill $0 < \alpha_i < C$ and can again be used to obtain b as previously described.

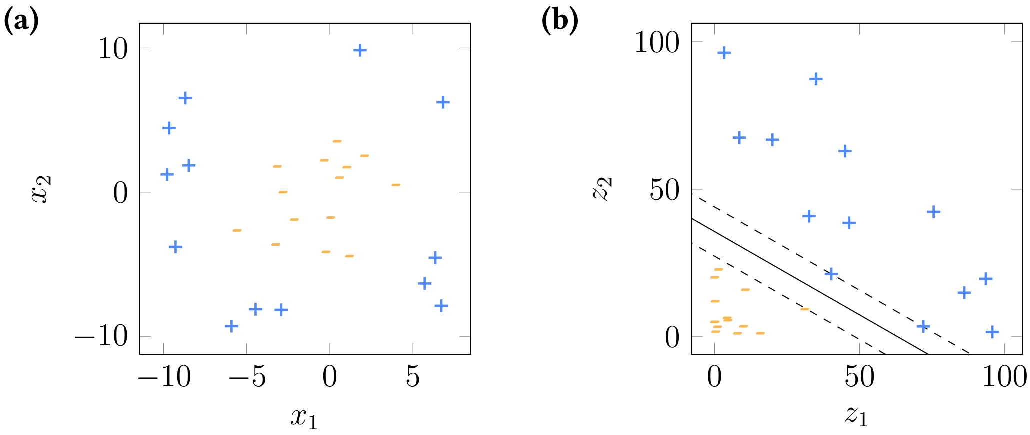


Figure 7.2.: Example data **(a)** before and **(b)** after an explicit transformation using $\mathbf{z} = \Phi(\mathbf{x}) := (x_1^2, x_2^2)$. After the transformation the data is linearly separable, as indicated by the maximum margin hyperplane.

7.3.3. Kernels

To handle data where the underlying, unknown function f is non-linear, data transformations of the form

$$\Phi : \mathcal{X} \to \mathcal{Z}$$

can be used to transform the data into a different feature space where a separating hyperplane exists. An example for such a transformation is shown in Figure 7.2 for the example data previously shown. The objective function of the dual optimization function is then defined based on the transformed data, i.e. Equation 7.6 becomes

$$\mathcal{L}(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \mathbf{z}_i^\intercal \mathbf{z}_j. \tag{7.8}$$

As previously stated, a convenient feature of the dual formulation of the optimization problem is that the input data appears solely in the form of an inner product. This gives rise to the *kernel trick*: If a closed form for the inner product in the $\mathcal{Z}$ space is known, the actual transformation of the data can be omitted. This enables us to use complex, high dimensional feature space transformations at potentially no extra cost. A *kernel function* is defined as

$$\mathrm{K}(\mathbf{x}, \mathbf{x}') := \Phi(\mathbf{x})^\intercal \Phi(\mathbf{x}') = \mathbf{z}^\intercal \mathbf{z}$$

and can be used as an equivalent substitution for the inner product of data points in Equation 7.8. From an abstract point of view, the kernel function can be seen as a similarity measure for any two data points $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. It is important to note that the corresponding optimization problem will provide a solution for various K, however that solution will only correspond to a maximum margin hyperplane in a transformed feature space $\mathcal{Z}$ if K corresponds to the inner product in $\mathcal{Z}$.

Polynomial Kernel

A popular choice as a kernel function is the *polynomial kernel*, which is defined as

$$\mathrm{K}^{\mathrm{poly}}_{d,c}(\mathbf{x}, \mathbf{x}') := (c + \mathbf{x}^{\intercal}\mathbf{x}')^{d},$$

where the degree d and the constant c are free parameters. The corresponding transformed feature space spans dimensions for monomials of all degrees up to d. For example, for $d = 2$, $c = 1$ and $\mathcal{X} = \mathbb{R}^2$ the equivalent explicit transformation would be $\Phi \ : \ \mathbb{R}^2 \rightarrow \mathbb{R}^6$ defined as

$$\Phi(x_1, x_2) := (1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_2x_2).$$

The computational complexity of the kernel function remains the same for all choices of d and depends on the dimensionality of the input space $\mathcal{X} = \mathbb{R}^n$ in the same way the original problem formulation does, i.e. through the computation of the inner product. Therefore, the kernel remains feasible even for large n and d, even though the (virtual) corresponding transformed feature space $\mathcal{Z}$ becomes very high-dimensional.

RBF Kernel

The *Gaussian radial basis function (RBF) kernel* is defined as

$$\mathrm{K}^{\mathrm{RBF}}_{\sigma}(\mathbf{x}, \mathbf{x}') := \exp\left(-\frac{\|\mathrm{x} - \mathrm{x}'\|^2}{2\sigma^2}\right),$$

where σ is a Gaussian RMS width parameter. While the kernel is compact and easy to compute, the corresponding transformed feature space $\mathbb{Z}$ is infinitely large and cannot be explicitly computed. The kernel function can in parts be expressed as an infinite sum of polynomial kernel functions with $d \in [0, \ldots, \infty]$ and is thus very flexible with respect to various order relations within the data. It has been established as a general purpose kernel for SVM classification and comes with the advantage that it adds only a single parameter that requires tuning.

Linear Kernel

A kernel that corresponds to the inner product of the original feature vectors

$$K^{\mathrm{lin}}(\mathbf{x}, \mathbf{x}') := \mathbf{x}^{\intercal}\mathbf{x}$$

is referred to as a *linear kernel*. It corresponds to the original SVM formulation using Equation 7.6, but since the SVM method is primarily seen as a kernel based approach, it has become standard terminology.

7.3.4. Multiple Kernel Learning

The Support Vector Machine concept can furthermore be extended to include multiple kernels into the same model, referred to as *multiple kernel learning* (MKL) as opposed to *single kernel learning* (SKL). The basic concept is to replace the kernel by a linear combination of kernels, i.e.

$$K(\mathbf{x},\mathbf{x}') = \sum_{k=1}^{K} \beta_k \cdot \mathrm{K}_k(\mathbf{x},\mathbf{x}')$$

where K_k denotes the kth kernel and β_k a corresponding weight coefficient. Since every linear combination of kernels is itself a valid kernel, this formulation is valid if the individual K_k correspond to inner products in a transformed feature space. There are two key ideas behind this concept: the incorporation of heterogeneous data sources and an improvement of the model's interpretability [119]. The latter can be achieved by observing the weights β_k, which become part of the learning process: if the input data is partitioned and distributed across different kernels, the weights can be used to infer in how far features or a particular feature encoding contribute to the solution.

The weight coefficients can either be obtained as part of the model selection process (which will be described in Section 7.3.6), or by incorporating them into the optimization problem that maximizes the separating hyperplane margin. Multiple solutions to accomplish the latter have been proposed, we will use the solution proposed by Sonnenburg et al. [119] and use the model selection only to tune the soft-margin SVM and kernel parameters.

7.3.5. Kernel Normalization

While we can in principle classify data in any real valued (transformed) feature space, it is often essential to *normalize* the data. One reason for this is that the original features may be defined on different value ranges and thus alter the meaning of the distance measure that is being optimized for. The normalization can be performed either in the original or in the transformed feature space. Since the transformed feature space is not explicitly known, the normalization has to be defined based on the kernel function. Given a kernel K, we will use the following kernel reformulation in order to perform a normalization in the transformed feature space:

$$\mathrm{K}_{\text{norm}}(\mathbf{x},\mathbf{x}') := \frac{\mathrm{K}(\mathbf{x},\mathbf{x}')}{\sqrt{\mathrm{K}(\mathbf{x},\mathbf{x}) \cdot \mathrm{K}(\mathbf{x}',\mathbf{x}')}}.$$

It has been shown that this particular normalization within the transformed feature space is superior to a data preprocessing step, since applying a kernel function often comes with a loss of the original normalization [120]. Additionally, we are able to use this type of normalization for kernels that perform a data type transformation, i.e. where the original feature space does not comprise real valued numbers, but for example sequences [121].

7.3.6. Model Selection

Both the soft-margin formulation of the SVM classification as well as most kernels involve parameters, i.e. describe a class of methods and the suitable method has yet to be defined. In order to do so, a set of parameter combinations can be evaluated based on the available training data using a n-fold *cross validation.* The training data $\mathcal{D}$ is partitioned into n equally large partitions

$$\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_n,$$

where $\mathcal{D}_i \cap \mathcal{D}_j = \varnothing$ for all $i \neq j$. The model is then trained n times, while each time leaving out one partition when training and using it to evaluate the trained model's performance. It is quantified based on standard statistical performance measures for binary classification such as the sensitivity, specificity and accuracy. Since we will deal with balanced classes, we will use the accuracy as the standard performance measure for our models. The overall model performance is defined as the mean performance over all n training data partitions.

The search space of parameters is traversed by performing a *grid search*, i.e. by an evaluation of all parameter combinations defined by a finite discrete range for each single parameter. For some parameters the values are not directly interpretable, but commonly applicable ranges of values exist and are initially applied. If the optimum performance of the model is reached at the boundary of a defined range, the grid is extended accordingly.

7.4. A Classification Framework for Antigen Receptor Genes

The aim of the classification method shown here is to be able to classify *single clonotype gene sequences* based on their function. In the following sections we will describe approaches to transform the gene sequence either explicitly or implicitly using kernels.

As previously described, the recombined gene sequence of each chain can be viewed as a composition of three components: the V and J segments and the CDR3 region. While each of them defines a substring of a sequence, the gene segments are limited by the segments available in the germline genome. We can therefore treat the gene segment components of the recombined gene as *categorical data*, while we have to consider more sophisticated methods to define a proper numerical feature space for the CDR3 region.

We will then combine the separate components of the data using the MKL approach described in Section 7.3.4. The data conversions and transformations for each component are described in the following sections. Note that we will use a nomenclature that assumes the original feature space to be the TRB clonotype repertoire, i.e. a kernel function will always be defined as $K : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$. Every subkernel will operate on a single component of the clonotype feature space, which we denote in the superscript of the kernel descriptor, e.g. K^{CDR3} denotes a kernel function that only takes into account the CDR3 region.

7.5. V and J Segment Encoding

To encode the categorical V and J segment information, we use a sparse feature vector with as many dimensions as there are gene segments where the values of all dimensions are 0 except for the one corresponding to the incorporated gene segment which is set to 1. Given the set of V gene segments $\mathcal{S}_V$ with $N_V = |\mathcal{S}_V|$, we define the transformed feature space as $\mathbb{R}^{N_V}$ and the feature vector corresponding to a clonotype c as

$$\Phi^V(c) := \mathbf{x} = (x_1, \ldots, x_{N_V}) \text{ where } x_i = \mathbb{1}_{i=c^V}.$$

In a kernel formulation, this corresponds to

$$\mathrm{K}^V(c_1, c_2) := \mathbb{1}_{c_1^V = c_2^V}.$$

The J gene segment is encoded equivalently.

7.6. CDR3 Region Encoding

In contrast to the V and J gene segments, the sequence of the CDR3 region is (for the most part) random. In the following sections we will describe kernels to map the CDR3 sequences into a numerical feature space. We will describe the kernels in the context of two clonotypes c_1 and c_2 with their corresponding CDR3 sequences c_1^{CDR3} and c_2^{CDR3} of lengths L_1 and L_2. Note that if not stated otherwise, the methods can potentially be applied either on the nucleotide or on the amino acid sequence of the CDR3 region. For a model aiming to assess clonotypes on a functional level it is however rather obvious to operate on the amino acid level, since clonotypes with silent substitutions are functionally indistinguishable.

7.6.1. K-Spectrum Kernel

Many approaches have been proposed to handle sequence information in kernelized learning methods. One of the sequence kernels developed for protein sequences is the *k-spectrum kernel*, initially used to predict protein homologies [121]. Given a k-mer length k, it is defined as

$$K_k^{\text{CDR3-KS}}(c_1, c_2) = \sum_{i=1}^{L_1-k+1} \sum_{j=1}^{L_2-k+1} \mathbb{1}_{c_1^{\text{CDR3}}[i \ldots i+k-1] = c_2^{\text{CDR3}}[j \ldots j+k-1]}.$$

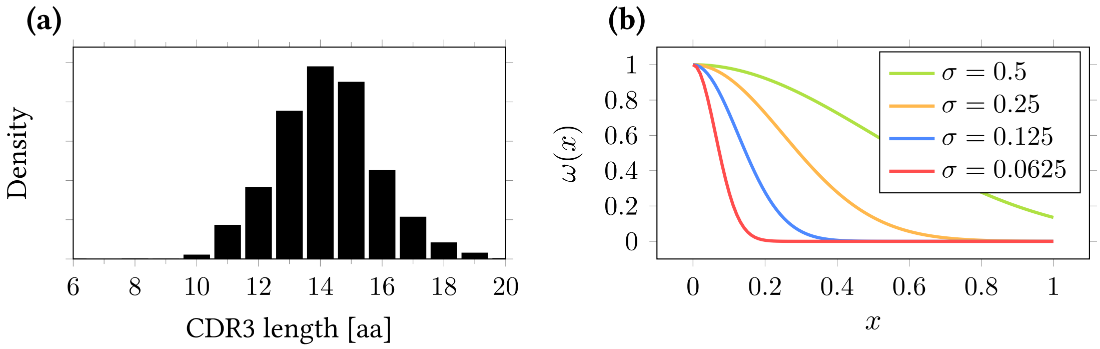


Figure 7.3.: (a) A typical TRB CDR3 length distribution across observed clonotypes, as observed in an nonspecific sample of human cells. **(b)** The Gaussian weight function used in the Weighted k-spectrum kernel for various values of σ.

This kernel formulation is equivalent to the explicit transformation $\Phi^{\text{CDR3-KS}} : \mathcal{C} \to \mathbb{R}^{\Sigma^k}$, where each dimension corresponds to a k-mer over the alphabet Σ and the feature vectors hold the absolute frequencies of those k-mers in the CDR3 sequence:

$$\Phi^{\text{CDR3-KS}}(c) := \left(\phi_a\left(c^{\text{CDR3}}\right)\right)_{a \in \Sigma^k}$$
$$\text{where } \phi_a(s) := \#\text{ times } a \text{ occurs in } s.$$

As the kernel only counts pairwise k-mer matches, the locality information is lost in the comparison. Two sequences are considered similar if they share many k-mers, even if they occur in an entirely different locations.

7.6.2. Weighted K-Spectrum Kernel

Several approaches have been proposed to include the locality information in string kernels. Among them is the *weighted degree kernel* [122], which is similar to the k-spectrum kernel but uses a range of k-mer lengths and is position specific, and the *locality improved kernel* [123], which is based on a mapping of each position in the sequence into multiple feature space dimensions. What these kernels have in common is that, unlike the k-spectrum kernel, they are defined only for sequences of the same length and are therefore not applicable to the CDR3 region without restricting the repertoire to clonotypes with equal CDR3 lengths.

While the lengths of the CDR3 regions are not identical, they are in a relatively restricted domain (see Figure 7.3(a)). We therefore extended the k-spectrum kernel to to the *weighted k-spectrum kernel* for input sequences of *similar* but not necessarily equal lengths. It weights in how far the *relative* positions of matching k-mers agree.

Given a k-mer length k it is defined as

$$K_{k,\varsigma}^{\text{CDR3-WKS}}(c_1, c_2) = \sum_{p_1=1}^{L_1-k+1} \sum_{p_2=1}^{L_2-k+1} w_\varsigma \left(\frac{p_1}{L_1} - \frac{p_2}{L_2} \right) \cdot \mathbb{1}_{c_1^{\text{CDR3}}[i \ldots i+k-1] = c_2^{\text{CDR3}}[j \ldots j+k-1]} \tag{7.9}$$

where $\omega_\varsigma : \mathbb{R} \to \mathbb{R}$ is a Gaussian weight function defined as

$$\omega_\varsigma(x) = \exp\left(-\frac{x^2}{2\varsigma^2} \right).$$

The function is shown for a selection of values for ς in Figure 7.3(b). For $\varsigma = 0$ the kernel only considers k-mer matches at identical (relative) positions, whereas for $\varsigma = \infty$ the kernel becomes position independent, i.e. equivalent to the k-spectrum kernel.

7.6.3. Weighted K-Spectrum Amino Acid Kernel

As previously stated, the clonotype gene sequence comparison should be performed on the amino acid sequence level. The kernels described in the preceding sections perform the character-wise comparison in a binary fashion, i.e. consider two compared positions in the observed sequences either as equal or unequal. This is somewhat inadequate for protein sequences, since some amino acids are known to be functionally more similar than others. To quantify the similarity of amino acids, both substitution or similarity matrices based on empirical observations [124, 125, 44, 126, 127, 128] as well as mappings of amino acids into multi-dimensional feature spaces based on biochemical properties [129, 130] have been published.

To incorporate the information of such feature maps $\Psi : \Sigma_{\text{AA}} \to \mathbb{R}^n$ that assign numerical feature vectors to amino acids into the previously described string kernels, we follow an approach suggested by Toussaint et al. [131]. They suggest to replace the substring comparison $\mathbb{1}_{s=s'}$ in the previous kernel formulations with a subkernel that takes the amino acid features into account. Given two substrings s and s' of equal length ℓ and an amino acid feature map Ψ, they propose the following formulation based on the polynomial kernel

$$\mathrm{K}_{\Psi,d}^{\text{AA-P}}(s, s') := \left(\sum_{i=1}^{\ell} \Psi(s[i])^\intercal \Psi(s'[i]) \right)^d$$

and the following kernel based on the RBF-kernel

$$\mathrm{K}_{\Psi,\sigma}^{\text{AA-RBF}}(s, s') := \exp\left(-\frac{\sum_{i=1}^{\ell} \|\Psi(s[i]) - \Psi(s'[i])\|^2}{\sigma^2} \right),$$

where d and σ are the parameters of the polynomial and RBF kernel as previously described. We use the latter formulation for the substring comparison in our weighted k-spectrum kernel, resulting in

$$\begin{aligned}K^{\text{WKS-AA}}_{k,\varsigma,\Psi,\sigma}(c_1, c_2) = &\sum_{p_1=1}^{L_1-k+1} \sum_{p_2=1}^{L_2-k+1} w_\varsigma \left(\frac{p_1}{L_1} - \frac{p_2}{L_2}\right)\\ &\cdot \mathrm{K}^{AA-RBF}_{\Psi,\sigma}(c_1^{\text{CDR3}}[i \dots i+k-1], c_2^{\text{CDR3}}[j \dots j+k-1]).\end{aligned}$$

7.6.4. Explicit Amino Acid Feature Mapping

A key advantage of the string kernels discussed in the previous sections over an explicit mapping of the characters to numerical features is the feasibility for long sequences. However, since the CDR3 sequences are relatively short in length, an explicit, character-wise application of an amino acid feature map remains computationally feasible in this case. Given a set of clonotypes $\mathcal{C}$ of length L and an amino acid feature map $\Psi : \Sigma_{\text{AA}} \to \mathbb{R}^p$, this results in a transformation function $\Phi : \mathcal{C} \to \mathbb{R}^{Lp}$. Let

$$\Psi(t) := (\psi_1(t), \dots, \psi_p(t))^\intercal$$

be the p-dimensional feature vector for a symbol $t \in \Sigma_{\text{AA}}$. Then

$$\begin{aligned}\Phi_\Psi(c) := \big(&\psi_1(c^{\text{CDR3}}[1]), \psi_2(c^{\text{CDR3}}[1]), \dots, \psi_p(c^{\text{CDR3}}[1]), \dots,\\ &\psi_1(c^{\text{CDR3}}[L]), \psi_2(c^{\text{CDR3}}[L]), \dots, \psi_p(c^{\text{CDR3}}[L])\big)^\intercal\end{aligned}$$

describes the resulting $p \cdot L$ dimensional feature vector for a clonotype c. We can then apply standard kernels such as the polynomial or the RBF kernel on the resulting feature space.

However, such an explicit transformation is only defined for clonotypes with equally long CDR3 regions. To overcome this limitation, we define a target length L^* and insert uniformly distributed dummy features with the mean feature values of two adjacent positions if the original sequence is too short. If the original sequence is too long, adjacent positions are merged, again using the mean feature values of the two positions. Let $L \neq L^*$ be the length of c^{CDR3} and $\Delta = |L - L^*|$. We then partition c^{CDR3} into $\Delta + 1$ ideally equally large partitions. Let $\mathcal{Q} = \{q_1, \dots, q_\Delta\}$ be the last positions of the first Δ partitions. We then define as

$$\Psi^q_{\text{agg}}(c^{\text{CDR3}}) := \frac{\Psi(c^{\text{CDR3}}[q]) + \Psi(c^{\text{CDR3}}[q+1])}{2}$$

the mean feature vector of the two positions adjacent to the partition. If $|c^{\text{CDR3}}| > L^*$ we shorten the resulting feature vector by replacing $(\dots, \Psi(c^{\text{CDR3}}[q], \Psi(c^{\text{CDR3}}[q+1], \dots)$ with $(\dots, \Psi^q_{\text{agg}}(c^{\text{CDR3}}), \dots)$. If $L < L^*$, the feature vector is elongated by replacing $(\dots, \Psi(c^{\text{CDR3}}[q], \Psi(c^{\text{CDR3}}[q+1], \dots)$ with $(\dots, \Psi(c^{\text{CDR3}}[q], \Psi^q_{\text{agg}}(c^{\text{CDR3}}), \Psi(c^{\text{CDR3}}[q+1], \dots)$, in either case for all $q \in \mathcal{Q}$. While this approach comes with a certain degree of information loss, it follows the intuition of amino acids that are at a similar location in either CDR3 region are more likely to bind to a similar region within the target and thus share a functional domain.

7.7. Related Work

In 2004, Thomas et al. [46] presented an approach for the aggregation and classification of *entire repertoires* of T cell receptor β chain CDR3 sequences. They perform an *explicit transformation* of the repertoires into a numerical feature space using a *bag of words* approach. Initially, for a fixed q a set of q-grams is sampled from the pool of all CDR3 sequences across all samples. The q-grams are transformed into a $q \cdot m$ dimensional feature vector defined by a character-wise application of an amino acid feature map $\Psi : \Sigma_{\text{AA}} \rightarrow \mathbb{R}^m$. The resulting vectors were subjected to k-means clustering, defining a codebook of k reference vectors. With the codebook defined, the authors then encode a repertoire of CDR3 sequences by an $\mathbb{R}^k$ dimensional feature vector: p q-grams are sampled from the repertoire, transformed into $q \cdot m$ dimensional vectors using the same transformation as described above and assigned to the nearest vector in the codebook. The feature vector of the repertoire is then defined as the spectrum of assignments, i.e. the ith dimension denotes the number of assignments to the ith codebook vector. The authors used the repertoire feature vectors to discriminate TRB repertoires of mice prior and post immunization with an antigen using either hierarchical clustering or an SVM. Their results showed that their models were able to distinguish repertoires before and after the immunization quite well, while a distinction of samples taken 5, 14 or 60 days after immunization in a multi-class approach showed mixed results.

In contrast, Li et al. [132] performed a *clonotype based* classification of TRB CDR3 sequences using SVMs. They as well performed an explicit feature mapping, similar to the approach described in Section 7.6.4, but without any length correction, i.e. restricted to a subset of sequences of a fixed length. They too apply their model to the CD4 vs. CD8 subtype classification problem and report prediction accuracies of approximately 88%.

7.8. Data Selection and Preparation

We will now briefly describe the in-house data that was used to evaluate the performance of the different models and their configurations, how it was generated and which preprocessing steps were applied.

7.8.1. Sample Preparation

The experiments were performed by the lab of Prof. Dr. Nina Babel at the Berlin Brandenburg Center for Regenerative Therapies (BCRT). PBMCs from patients blood samples were stained with the following antibodies from Biolegend: CD3 App-Cy7 (Clone HIT3a), CD4 Alexa Fluor 700 (Clone RPA-T4) and CD8 FITC (Clone RPA-T8). The cells were then flow cytometrically sorted on a BD Biosciences Aria II cell sorter. Propidium iodide staining was used to exclude dead cells. After cell lysis and DNA isolation the TRB gene was enriched in a multiplex PCR from genomic DNA as described by Dziubianau et al. [97]. The target fragments were isolated and prepared for pooled, paired end Illumina sequencing using indexed adapters both on the 3' and the 5' end (dual barcoding). The dual barcoding was performed in order to avoid clonotype "contamination" between any pair of samples sequenced on the same lane that is induced by residual contamination across the adapter oligonucleotides (see Section 8.2). The samples were then sequenced on an Illumina sequencing machine and the raw data subsequently processed by IMSEQ (see Chapter 4).

7.8.2. Data and Data Preparation

The pool of available CD4-CD8 dataset pairs comprises samples from 16 donors, partially from various points in time. Since the datasets originated from samples with different cell numbers and sequencing depths, as well as of varying quality, we preselected and preprocessed the datasets in order to fit the needs for a classification evaluation setting.

Beyond the general data quality assessment, the proper handling of duplicates and highly similar clonotypes that originate from technical artifacts is crucial. In principle, similarities between feature vectors are what we want to detect and utilize when applying machine learning methods. However, when dealing with clonotype repertoires, we face the situation that two distinct feature vectors might actually originate from the same clonotype - due to technical artifacts as described in Section 4.2. This can lead to an overestimation of the model performance if measured using cross validation: When the data is shuffled and partitioned, the artifacts are likely to be separated into different subsets. Thus, when evaluated, the model is likely to perform well on those feature vectors, generalizing the technical similarity between artifacts rather than the biological similarity between true clonotypes. For a robust method assessment, it is therefore crucial to limit the repertoire to high confidence clonotypes.

Furthermore, we want to compare the performance of models operating on different subsets of features, i.e. models that take only the CDR3 region in to account vs. models that also include the V and J gene segments. Feature vectors are not necessarily unique under all subsets of features. To make the model performances comparable, we therefore limit the repertoires to those clonotypes that are unique with respect to the CDR3 sequence, even if they have different V and J gene segments incorporated.

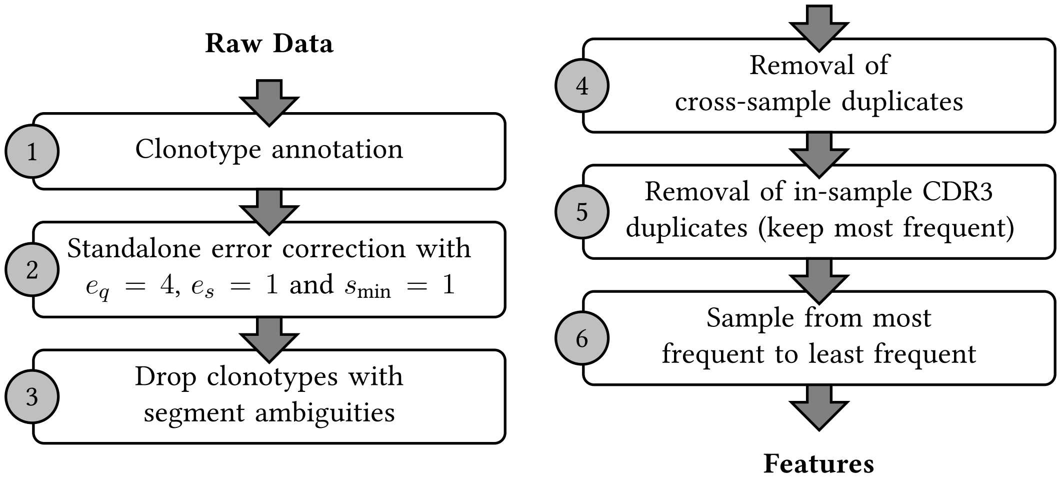

Raw Data
1
Clonotype annotation
2
Standalone error correction with $e_q = 4$, $e_s = 1$ and $s_{\min} = 1$
3
Drop clonotypes with segment ambiguities
4
Removal of cross-sample duplicates
5
Removal of in-sample CDR3 duplicates (keep most frequent)
6
Sample from most frequent to least frequent
Features

Figure 7.4.: A schematic overview of the data processing performed on the raw sequencing data prior to the model evaluation.

This enables us to evaluate models that take into account the V and J gene segments without being biased by CDR3 duplicates. In this clonotype reduction step, only the most frequent clonotype is kept.

An overview of the implemented data processing steps is shown in Figure 7.4. Initially, the raw data is annotated and the standalone error correction is performed as described in Section 4.5 using IMSEQ. We then remove all clonotypes with ambiguous gene segment assignments and cross-sample duplicates, i.e. those clonotypes that occur both in the CD4 and the CD8 samples. Such cases do occur at a low frequencies due to sorting errors or double positive cells. After the removal of CDR3 duplicates as described above, the samples are sorted from the most frequent to the least frequent clonotype. When sampling from clonotype repertoires for model evaluation, the clonotypes are obtained top-down, i.e. when performing a cross validation on n clonotypes from each sample, the n most frequent clonotypes are used. Since low quality clonotypes are expected to occur at lower frequencies, this maximizes the number of high confidence clonotypes.

To ensure that we do not sample clonotypes from the “tail” of each repertoire, we select datasets for evaluation with a minimum number of unique CDR3 clonotypes. Given that we will operate with balanced training and evaluation samples of size 1,000-5,000, i.e. up to 2,500 clonotypes per class, we limit the datasets to those where at least double, i.e. 5,000 unique CDR3 clonotypes after error correction are available per sample and thus never exhaust a sample to more than 50%. The used datasets and the corresponding clonotype counts are shown in Table 7.1.

		Error correction				No error correction			
		# CDR3s		# V-J-CDR3s		# CDR3s		# V-J-CDR3s	
Donor	Date	CD4	CD8	CD4	CD8	CD4	CD8	CD4	CD8
A	2013-09-13	9,671	7,414	11,886	9,652	10,422	8,429	12,721	10,782
A	2013-10-24	11,252	8,195	14,954	12,734	12,996	10,673	16,851	15,589
A	2015-01-21	19,422	8,354	22,829	11,025	22,537	9,863	26,799	13,046
B	2013-03-28	11,251	5,101	15,174	8,189	12,914	7,129	16,970	10,457
C	2013-03-25	13,170	5,058	18,758	8,581	16,670	8,500	23,344	12,638

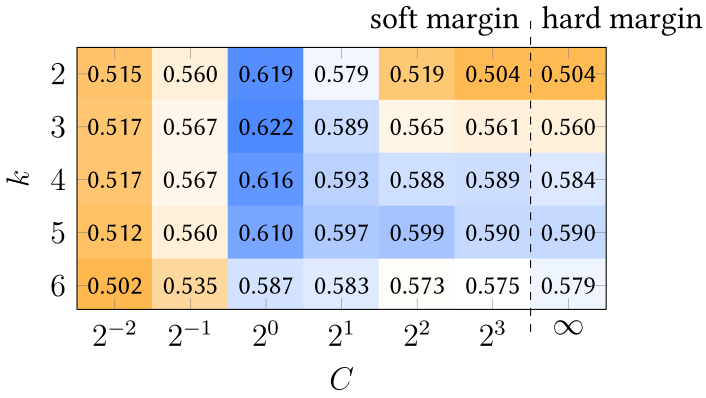
Table 7.1.: The five datasets selected for the evaluation of different data encodings and kernels for the differentiation of the CD4 and CD8 T cell subtype. The datasets originate from three unique donors. For each dataset, the number of unique "complete", i.e. V-J-CDR3 defined clonotypes and the number of unique CDR3 sequences is given.

7.9. Implementation

After the raw data processing and clonotype repertoire generation with IMSEQ, the data was pre-filtered in R (removal of ambiguous gene segment assignments) and then further processed in a native application written in C++. The application is based on the Shogun [133] machine learning library's implementation of the SVM framework and utility functionality such as cross validation.

7.10. Model Parameters

In the following we will explore the parameter spaces of all CDR3 kernels together with the soft margin SVM parameter C in order to identify optimal parameter combinations for each model. The parameter space was explored using a grid search and each parameter combination was evaluated using a 10-fold cross-validation (as described in Section 7.3.6), based on the 2×500 highest ranking clonotypes of the previously described five dataset pairs. The cross validation was performed on each dataset individually. The average accuracies over all datasets and all cross validation partitions are then built and used for the parameter selection. After we defined the optimal parameters for each CDR3 based model, we will evaluate them comparatively in conjunction with the V and J gene segment information using the identified parameters and a larger dataset.



k \ C	2^{-2}	2^{-1}	2^{0}	2^{1}	2^{2}	2^{3}	∞
2	0.515	0.560	0.619	0.579	0.519	0.504	0.504
3	0.517	0.567	0.622	0.589	0.565	0.561	0.560
4	0.517	0.567	0.616	0.593	0.588	0.589	0.584
5	0.512	0.560	0.610	0.597	0.599	0.590	0.590
6	0.502	0.535	0.587	0.583	0.573	0.575	0.579

Figure 7.5.: The evaluated model selection space for the *k-spectrum kernel*, i.e. the grid of values for the soft-margin SVM parameter C and the kernel parameter k with the corresponding mean accuracies (over all samples and cross validation partitions).

7.10.1. K-Spectrum Kernel

The parameter space that was explored for the k-spectrum kernel is shown in Figure 7.5. The explored set of parameters was built over all combinations of the soft margin parameter $C \in [2^{-2}, 2^{-1}, \ldots, 2^{3}, \infty]$ and the k-mer length kernel parameter $k \in [2, 3, \ldots, 6]$. The parameter combination of $C = 1$ and $k = 3$ performed best in the given analysis and there is a generally strong trend towards the soft margin parameter value $C = 1$. For harder margins, i.e. larger C, the performance gradually decreases while higher values for k performing better than lower values. In the hard margin configuration, the optimally performing choice for the k-mer length is $k = 5$, allowing the SVM to properly separate all training data the cost of generalization.

7.10.2. Weighted K-Spectrum Kernel

For the weighted k-spectrum kernel, the soft margin performance optimum for $C = 1$ remains. We therefore only show the results for $C = 1$ and $C = \infty$. Figure 7.6 depicts the parameter performances for k-mer lengths $k \in [2, 3, \ldots, 6]$ and widths for the Gaussian weight function $\varsigma \in [2^{-3}, 2^{-3}, \ldots, 2^{1}]$. Generally the impact of the weight function, i.e. whether we consider k-mer matches that are (relatively) more distant with respect to their positions in the two compared CDR3 sequences, does not strongly affect the model performance. Regarding k, we again see a tendency towards smaller k in the soft margin model and higher k in the hard margin model, with an overall optimum at $k = 3$ and $\varsigma = 0.5$ for $C = 1$.

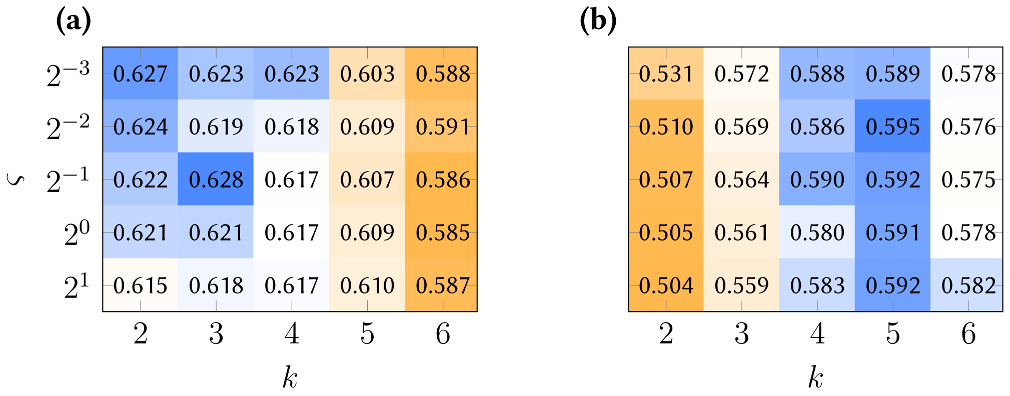


Figure 7.6.: A subset of the evaluated model selection space for the *weighted k-spectrum kernel*, i.e. the grid of values for the k-mer distance weight parameter ς and the k-mer length k with the corresponding mean accuracies (over all samples and cross validation partitions). The soft-margin parameter is fixed either as **(a)** $C = 1$ or as **(b)** $C = \infty$ (hard margin).

7.10.3. Weighted K-Spectrum Amino Acid Kernel

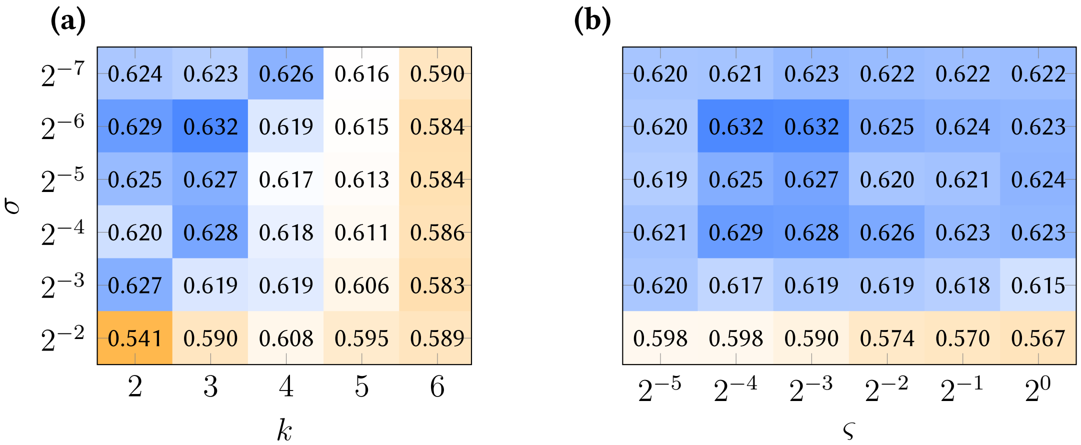
If we additionally integrate amino acid properties as described in Section 7.6.3, there are two additional parameters, the used amino acid property map Ψ and the Gaussian RBF subkernel weight σ. As previously done by Thomas et al. [46], we use the numerical feature vectors provided by Atchley et al. [130] to map single amino acids (see Appendix A.5).

The results for a subset of the explored parameter space are shown in Figure 7.7(a) for $\sigma \in [2^{-2}, 2^{-3}, \dots, 2^{-7}]$ and $k \in [2, 3, \dots, 6]$, while the parameters $C = 1$ and $\varsigma = 2^{-3}$ are fixed. The k-mer length $k = 3$ remains optimal, while with respect to σ the best performance is achieved for values $\leq 2^{-3}$. When we fix the k-mer length to $k = 3$ and measure the performance of the k-mer distance weight ς against σ as shown in Figure 7.7(b), we again see that the model performance is rather invariant for wide ranges of parameter choices, with an optimum at $\varsigma \simeq 2^{-3}$ and $\sigma \simeq 2^{-6}$.

7.10.4. Explicit Amino Acid Feature Mapping with RBF Kernel

Furthermore, we evaluated a model based on the length corrected, explicit feature mapping described in Section 7.6.4. We again used the amino acid feature map by Atchley et al. [130] and applied a Gaussian RBF kernel on the resulting numerical feature space. For the transformation, a target length of $L^* = 17$ was used in order to preferentially insert dummy positions rather than deleting positions and thus loose information. The length difference was restricted to be $\Delta \leq 5$, i.e. a small fraction of clonotypes that could not be mapped within these boundaries was discarded.

The model selection results for a parameter range of $C \in [0.5, 1, 2]$ and $\sigma \in$



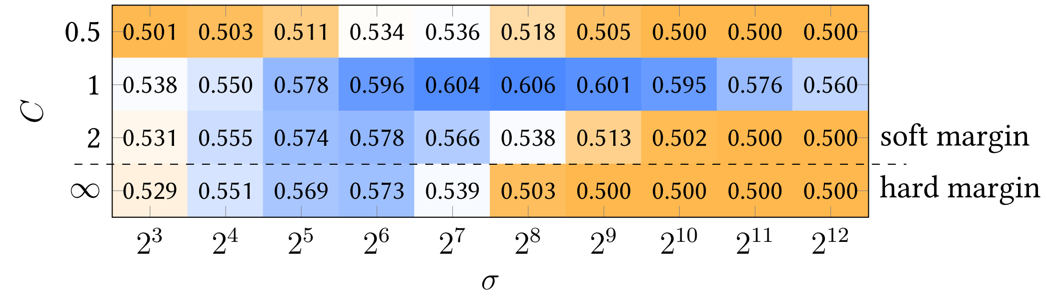
(a)

σ \ k	2	3	4	5	6
2^{-7}	0.624	0.623	0.626	0.616	0.590
2^{-6}	0.629	0.632	0.619	0.615	0.584
2^{-5}	0.625	0.627	0.617	0.613	0.584
2^{-4}	0.620	0.628	0.618	0.611	0.586
2^{-3}	0.627	0.619	0.619	0.606	0.583
2^{-2}	0.541	0.590	0.608	0.595	0.589

(b)

σ \ ς	2^{-5}	2^{-4}	2^{-3}	2^{-2}	2^{-1}	2^{0}
2^{-7}	0.620	0.621	0.623	0.622	0.622	0.622
2^{-6}	0.620	0.632	0.632	0.625	0.624	0.623
2^{-5}	0.619	0.625	0.627	0.620	0.621	0.624
2^{-4}	0.621	0.629	0.628	0.626	0.623	0.623
2^{-3}	0.620	0.617	0.619	0.619	0.618	0.615
2^{-2}	0.598	0.598	0.590	0.574	0.570	0.567

Figure 7.7.: A subset of the evaluated model selection space for the *weighted k-spectrum amino acid kernel*, with the corresponding mean accuracies (over all samples and cross validation partitions), where either **(a)** the k-mer distance weight is fixed to $\varsigma = 2^{-3}$ or **(b)** the k-mer length is fixed to $k = 3$. In both cases the soft margin parameter is set to $C = 1$.



C \ σ	2^{3}	2^{4}	2^{5}	2^{6}	2^{7}	2^{8}	2^{9}	2^{10}	2^{11}	2^{12}
0.5	0.501	0.503	0.511	0.534	0.536	0.518	0.505	0.500	0.500	0.500
1	0.538	0.550	0.578	0.596	0.604	0.606	0.601	0.595	0.576	0.560
2	0.531	0.555	0.574	0.578	0.566	0.538	0.513	0.502	0.500	0.500
∞	0.529	0.551	0.569	0.573	0.539	0.503	0.500	0.500	0.500	0.500

Figure 7.8.: The evaluated feature space for the *explicit amino acid feature mapping* combined with a Gaussian RBF kernel, where C is the soft margin SVM parameter and σ the width parameter of the kernel.

$[2^3, 2^4, \ldots, 2^{12}]$ is shown in Figure 7.8. With respect to the soft margin parameter, the model again performs best for $C = 1$, while the Gaussian RBF width parameter yields the best average result for $\sigma = 2^8$.

7.11. Model Comparison

Lastly, we compared each model in a larger scale cross-validation with respect to their accuracies as well as the number of support vectors as an estimator for the generalization performance. Each model was evaluated in a 10-fold cross-validation using balanced datasets based on the 2×2,500 highest ranking CD4 and CD8 clonotypes separately for each of the five samples. Since the model selection as shown in the previous section often revealed multiple parameters close to the optimal combination, multiple parameter sets were evaluated, but again revealed similar performance results. We therefore show a single parameter combination for each model in Figure 7.9, while the extended results are shown in Appendix A.6. The configurations shown here are the k-spectrum kernel with $k = 2$, the weighted k-spectrum kernel with $k = 2, \varsigma = 0.125$, the weighted k-spectrum amino acid kernel with $k = 3, \varsigma = 0.125, \sigma = 2^{-6}$ and the RBF kernel based on an explicit Atchley mapping with $\sigma = 2^{8}$. The soft margin SVM parameter was set to $C = 1$ for all models. For the weighted and unweighted k-spectrum kernels a value of $k = 2$ was selected, since this configuration resulted a lower number of support vectors while yielding comparable accuracies.

Initially, each CDR3 kernel was evaluated as a single kernel learning model. Resulting in a median accuracy of approximately 64%, the weighted and unweighted k-spectrum kernels perform comparably. The amino acid version and the explicit mapping of the CDR3 sequences into length normalized, Atchley factor based feature vectors range slightly below that.

If we additionally incorporate the V and J segment information in form of categorical data handled by a linear kernel as described in Section 7.5, the performance is consistently improved to over 72%. The performance of the individual models becomes indistinguishable, with each model performing more or less similarly. To further investigate the role of the gene segment information, we additionally evaluated the performance of an MKL model using only those features, i.e. an MKL model with two linear kernels encoding the V and J gene segment information. It ranges slightly below the full MKL models with a median performance of approximately 69% as shown in the right panel in Figure 7.9.

The results for the number of support vectors are shown in Figure 7.9(b) and confirm the results as defined based on the classification accuracy. The numbers of support vectors are generally high, ranging consistently above 2,500 in the MKL models and in the case of the SKL models partially even close to 4,000, indicating a suboptimal fit in concordance with the accuracies.

While the performances are close, the traditional k-spectrum kernel and the weighted k-spectrum kernel appear to yield the best results with respect to the highest accuracies combined with the lowest number of support vectors.

7.12. Summary and Conclusion

We have described and evaluated an approach for the binary classification of single T cell receptor β chain clonotypes and evaluated our methods on the biological binary CD4-CD8 subtype system. We primarily focused on the encoding of the hypervariable CDR3 and presented four alternative methods to encode and use the sequence information of that region in a learning model based on Support Vector Machines: (1) the k-spectrum kernel as previously described by Leslie et al. [121], to which we

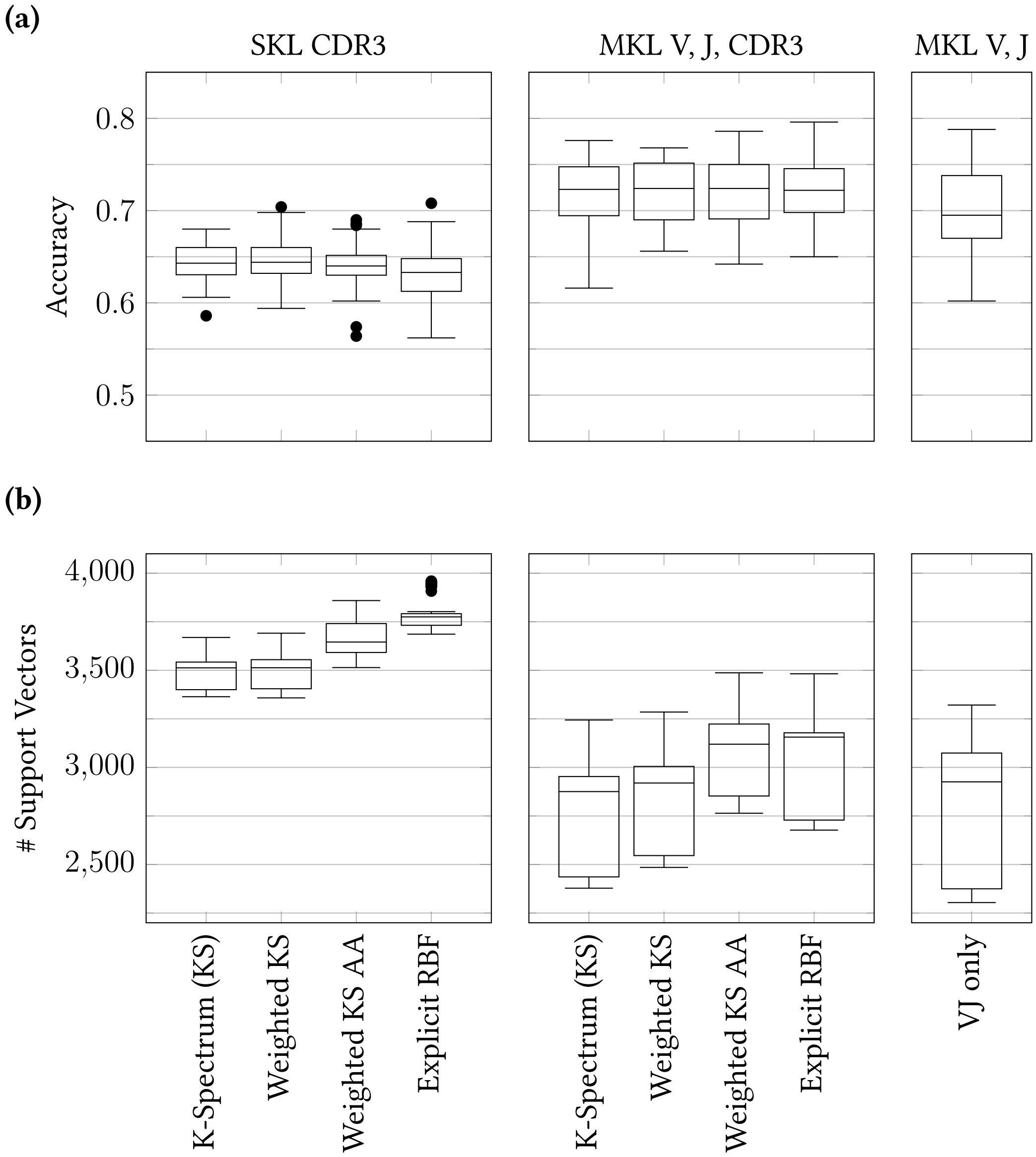


Figure 7.9.: The cross validation results comparing all models based on balanced datasets with 5,000 unique clonotypes. Each model was evaluated based on the five datasets previously described and the box plots comprise the **(a)** accuracies and **(b)** numbers of support vectors measured over all samples and all cross validation iterations. The left panels show the results based on SKL, the center panels based on MKL which additionally include the V and J segments and the right panels based on MKL using only the V and J information.

suggested two modifications - (2) a version that takes into account locality information in sequences of *similar* length and (3) a version that additionally takes into account amino acid properties. Lastly, we attempted to (4) explicitly map the CDR3 sequence information into a numerical feature space, again based on amino acid properties.

We then evaluated the performance of each CDR3 kernel in a standalone, single kernel learning configuration as well as in a multiple kernel learning adding two linear kernels encoding the incorporated V and J gene segments. The evaluations revealed a prediction accuracy of 64% for the SKL, 69% for the V-J MKL and 73% for the V-J-CDR3 MKL models, with each CDR3 kernel performing more or less similarly. The assessment of the number of support vectors as a measure for the generalization error depicts a similar trend, i.e. a limited generalization with slightly better values for the MKL models. The results, in particular the comparison between the V-J and V-J-CDR3 MKL models, suggest that all the tested models perform primarily based on the V and J segment information. It is important to recall that the V and J segments are not entirely disjoint with the CDR3 sequence, but overlap with it to a variable degree as shown in Figure 2.5. Therefore, the CDR3 sequence is to some extent informative with respect to the incorporated gene segments and thus partially discriminative for classes that primarily depend on the V and J segments. We do, however, see a small but reproducible improvement when all features, i.e. also the CDR3 is taken into account, which demonstrates the ability of the models to exploit discriminative information in CDR3 which is not part of the germline encoded segments.

The limited capabilities of the evaluated models on the CD4-CD8 discrimination problem can be either due to the fact that the available features do not encode the necessary information or because the models fail to sufficiently generalize available information. For the given data and evaluation results, both components are likely to play a role:

The CD4-CD8 evaluation system As initially described, the CD4-CD8 discrimination problem has properties that are beneficial for a model evaluation task, such as the binary nature of the problem and the biological balance of the classes. On the other hand, however, the biological background suggests that it might not serve well as a test environment for models based on the CDR3, since the regions of the readily folded receptor protein that interact with the MHC ligand are assumed to lie within the germline encoded region of the protein. Our findings stand in contrast to those of Li et al. [132], who claim that the CD4 vs. CD8 fate of T cells correlates with the CDR3 sequence and report classification accuracies of approximately 88% using a CDR3 kernel. However, our results are in agreement with what is known about the biological system. Classifying cells based on the germline encoded V and J segments of the gene sequence appears to be possible up to a certain degree, with the missing link possibly being the α chain. After all, when working with β chain sequencing data, we are only observing data encoding one of two chains of the receptor. Therefore one could speculate that with the models presented here, a more accurate classification may be possible based on paired α-β repertoire sequencing data, which is currently not available.

CDR3 kernel performance Due to the comparable performances of the tested CDR3 kernels and the limited overall performance it is hard to make a statement whether the CDR3 kernels perform well or not - if the assumption is that CD4-CD8 discrimination problem is not suitable for the CDR3 kernel evaluation, we are simply unable to assess that. However, other data that may be more suitable for the evaluation task, particularly a classification by antigen specificity, comes with the initially stated problems: very low clonotype numbers and, at the time of writing, insufficient availability. Nonetheless, even if we may not be able to estimate the CDR3 kernel performances with the data at hand, there is also room for future improvements on the data encoding and handling. Treating a complexly shaped and folded protein as a linear sequence of amino acids is a major generalization or aggregation step. Put differently, if we present the TCR protein to a computational model this way, the relations to be detected in order to solve the classification problem are hidden in deeper layers in the data. More sophisticated approaches that preprocess the sequence e.g. to obtain (partial) protein structures would expose some of the relations more directly to the model and might therefore perform better and be an interesting subject for future investigations.

Some of the issues regarding data availability can be expected to be (to some degree) improved in the near future. With paired chain repertoire sequencing techniques, which we will briefly discuss in Section 8.3.1, becoming available at increasingly high scale, these methods may soon lift the restrictions that currently apply due to the widely used single chain sequencing approaches. When databases based on large scale studies with paired chain clonotypes annotated with the HLA genotype of the donor and known functionalities become available, it would be interesting to reevaluate and optimize the methods discussed here.

Part IV.

Conclusion and Appendices

8. Conclusion and Outlook

In this thesis, we have discussed the computational and analytical questions that arise from T and B cell receptor repertoire sequencing. We started off from the processing of the raw sequence data and discussed our method IMSEQ, which we developed and evaluated with the various pitfalls, biases and errors in mind that can make Rep-Seq data processing difficult. Continuing from properly annotated Rep-Seq repertoires, we differentiated between two types of applications: those that use the identified clonotypes solely as identifiers of a subpopulation of T cells with identical antigen specificity, and those that aim to derive functional information from the recombined antigen receptor genes. Regarding the former, we described a clinical application by our collaborators in the context of renal transplant patients as well as other examples from literature. We then presented a learning method based on support vector machines that we developed to functionally classify antigen receptor clonotypes.

We will conclude this thesis with some general remarks regarding the ongoing challenges in the context of Rep-Seq, which were not discussed in detail in the previous chapters. Furthermore, we will give an outlook in the form of a brief insight into recent developments: the ongoing efforts to capture both receptor chains using latest single cell technologies and new findings in the context of the sequence based characterization of antigen receptor genes.

8.1. General Remarks

Since the discovery of the somatic gene recombination process driving antigen receptor diversity in the early 1980s, research on the adaptive immune system has made significant steps forward. Early studies estimated repertoire features using first generation sequencing technologies [134], which poses a serious limitation given the enormous diversity of immune repertoires. Now, with high throughput sequencing technologies at our hands, we have all the experimental tools needed for comprehensive studies of immune repertoires. We are solely bound by the limits of sampling, since we can obviously not obtain the entire T or B cell repertoire from a living individual.

Consequently, the key challenges have shifted towards the interpretation of the generated data. While new Rep-Seq data is continuously being generated, some of the key parameters of interest cannot be derived from the data in a satisfactory fashion. A prominent example is the *diversity* of a repertoire, a parameter often asked for as it is believed to give insights into the current activity of the immune system or may serve as a predictive parameter for disease or vaccine outcome when measured in antigen specific repertoires. The proposed solutions range from simply counting the number of observed clonotypes to diversity measures developed in the context of ecology [135], where one observes counts of species and is interested in biodiversity. The problem in Rep-Seq is, that we cannot clearly identify the species, i.e. clonotypes, in the same sense as ecological species. Due to the technical artifacts introduced by PCR and sequencing, we may observe multiple species instead of one true species, and consequently overestimate the sample diversity. Even if we can contain the effect of these errors, as shown in this thesis, i.e. we assume that we only observe true biological entities, the question of species definition remains open to some degree. That is, it may not be desired to treat two very similar clonotypes as different species in the same way as we treat two clearly distinct clonotypes as different species when computing a measure such as diversity.

8.2. Sensitivity to Contamination

Another important aspect that is rarely addressed in publications is contamination. PCR is a process highly prone to contamination [136], which, as long as contained to low levels, does not necessarily pose a threat to regular sequencing applications. When preparing genomes, exomes or transcriptomes, the results will not be impacted by cross contamination of a few molecules. In repertoire sequencing, however, it is very likely that we observe a number of clonotypes only in single cells, i.e. have only one template for amplification. Thus, we cannot simply discard all sequences that occur at very low frequencies, as they might be true biological entities. To some degree, this problem can be addressed with UMIs, i.e. if template is contaminated with product. Aside from contamination during the PCR preparation, another source of contamination is the pooling of samples in the same sequencing pool using index primers to demultiplex them after sequencing. Even without contamination between the libraries, we have often observed a certain degree of contamination between the index primers upon delivery in our in-house experiments. In fact, in some cases we could clearly identify low level dilutions of one sample in another sample in a pattern that allowed us to deduce in which order the primer oligonucleotides were synthesized by the manufacturer using the same machine. Again, the observed degree of contamination would usually not pose a problem to most other sequencing applications. Nonetheless, in the context of Rep-Seq it should be taken into account. While the issue has been mentioned in a few publications [24, 137, 138, 32, 139], in the majority of cases handling of contamination is not mentioned, which at least rises questions about many of the comparative studies that report sharing of clonotypes between individuals.

8.3. Outlook

8.3.1. Paired Chain Repertoire Sequencing

Most Rep-Seq data produced to date is from bulk, single chain sequencing experiments and from those, most are focused on the T cell receptor β chain or on the immunoglobulin heavy chain. This is due to the fact that the α and β (or heavy and light) chains are encoded on separate chromosomes, thus the bulk experimental setup as described in Section 2.4.2 is not suitable for capturing both chains in a *linked* fashion. While we can indeed amplify both genes with suitable PCR primers, the information which pairs of genes originated from which cell is lost. Therefore, for a long time methods that maintain the link information between the two chains were restricted to low throughput approaches where every cell is treated and sequenced separately. Such an approach is obviously unsuitable for samples of millions of cells.

Recently, several approaches have been made to develop paired chain repertoire sequencing methods that achieve higher throughputs. They generally aim to minimize the experimental effort that has to be applied on the individual cell level. In 2013, DeKosky et al. [140] proposed an approach where the individual cells are separated onto high density microwell plates. Inside the individual compartments the cells are then lysed and the RNA captured. Subsequently, the target RNAs are fused in a *linkage PCR* step, which targets the genes of the two chains as described in Section 2.4.2 and additionally achieves that the two products are linked to one. Throughout all these steps, a separation of the cells is continuously achieved either through the well compartments or emulsion. Only after the fusion of the two genes the reaction mix is joined and regularly sequenced. In 2014, Georgiou et al. [137] mention the development of a similar, improved method separating the cells in microdroplets. More recently, in 2016, Stubbington et al. [141] have described an approach named *TraCeR* that is based on *single cell RNA-Seq (scRNA-Seq)*. Single cell transcriptomics significantly advanced in recent years [142] and TraCeR can directly operate on that type of data without any experimental requirements specific for capturing immune repertoires. Additionally, this approach has the advantage that the transcriptome of each clone can be studied. However, the efficiency is far below that of bulk methods, i.e. currently an α chain clonotype can be annotated in 74-96% and a β chain clonotype in 70-93% of the cells - while only a few hundred cells can be studied in an experiment. Despite the efforts to increase the throughput, the total number of cells that can be analyzed by the aforementioned methods is also orders of magnitude below that of bulk Rep-Seq, ranging from hundreds to up to 10^5 cells.

A quite different approach has been suggested by Howie et al. [143] in 2015. They too compartmentalize the cells within the sample on a well plate, but not on the single cell level. Instead, the authors rely on the multiplicity of the clones, i.e. that one can expect multiple clones of each clonotype within a sample. Each compartment is then individually sequenced using a regular Rep-Seq approach, resulting in two repertoires per compartment, one per chain. The pairing is then solved using combinatorial methods: A pair of genes originating from the same cell will always occur together across different compartments. If the sample splits into enough compartments such that clones from one clonotype are unlikely to occupy the same subset of compartments as those from another clonotype, a high degree of pairing can be achieved. The method achieves a similar throughput as the aforementioned ones, however, with only a minimal additional experimental overhead in comparison to bulk Rep-Seq.

As the throughput and accuracy of paired Rep-Seq methods is expected to improve in the near future, these methods will certainly boost the field of immune repertoire analysis as they fill the yet missing piece to accurately genotyping T and B cells with respect to their antigen specificity.

8.3.2. Advances in Functional Clonotype Interpretation

In mid 2017, Dash et al. [144] published a distance metric for paired chain clonotypes, which they refer to as *TCRdist*. The distance measure is based on the amino acid sequences of the CDR1, CDR2 and the center part of the CDR3 of each chain. Additionally, a region between the CDR2 and CDR3, which was shown to be facing the pMHC in ternary structures, is included. The regions are mapped into a fixed alignment column space using the IMGT numbering scheme [16] followed by a position-wise scoring derived from the BLOSUM62 [44] matrix and using a gap penalty for positions that are covered by only one sequence.

The authors generated datasets from a pool of T cells from 32 humans and 78 mice. The cells were selected against 10 single MHC bound peptides separately using *tetramer staining*, an experimental method originally developed by Altman et al. [145]. They calculated the pairwise clonotype distances within the 10 repertoires and formed clonotype clusters using either a dimensionality reduction method or a hierarchical clustering algorithm. To identify the CDR3 residues determining the peptide specificity, the authors then represented each cluster in terms of their V and J segment frequencies and the positional amino acid frequencies within the CDR3. For each cluster, using a random background clonotype set with the same V and J segments, they reduced the CDR3 to those residues which were overrepresented *independently* of the germline gene segments, hypothesizing that these are the residues of interest. For pMHCs of known TCR bound ternary structure, the authors were able to confirm that the identified CDR3 residues were indeed in direct contact with the pMHC. They therefore concluded that they had found a method able to derive the TCR gene components mediating epitope recognition solely based on the clonotype repertoire sequence information.

Additionally, the authors introduced the concept of a *nearest neighbor distance (NN-distance)*, which, given a repertoire, is defined for every clonotype as a weighted average TCRdist distance to the closest n other clonotypes, where n was chosen as 10% of the total repertoire. This measure was then used to assess the predictive power of TCRdist by removing one donor from the pool and assigning each clonotype from that donor to the repertoire where it has the lowest NN-distance. The authors state that they were able to correctly assign 78% of the mouse clonotypes and 81% of the human clonotypes to their source repertoire.

At the very same time, Glanville et al. [146] published a slightly different approach, coming to a similar conclusion. The authors use tetramer stained repertoires with a single peptide specificity as well, but solely focus on the center parts of CDR3s. The notion of a *significant motif* is introduced as an amino acid k-mer ($k \in \{2, 3, 4\}$) which is at least 10-fold enriched over naïve repertoires and has a probability < 0.001 to occur in the latter. Clonotypes are then clustered if they are either *globally similar* within the CDR3, i.e. differ by at most 2 amino acids, or *locally similar*, i.e. share at least one significant motif. The authors found that, when applying their method on a mixed clonotype population of 8 specificities, 94% of the clustered clonotypes were correctly grouped with other clonotypes of the same specificity. However, only 14% of the clonotypes were clustered at all.

The authors also propose a classification approach based on their clustering methods by generating *positional weight matrices* (PWMs) from the identified clusters and assigning new clonotypes by scoring the CDR3 regions against the cluster PWMs. They claim positive classification results, without explicitly presenting an extensive evaluation. They did, however, experimentally generate de novo TCRs by sampling from the distribution of TCRs provided by the PWMs. The authors were able to show that their newly designed T cells were specific for the targeted antigen in 8 out of 10 tries and that two of those showed an even higher activation than the biological templates.

Interestingly and in contrast to the findings of Dash et al. [144], Glanville et al. [146] concluded from their comparison of single and paired chain data that the TCR α chain did not show any clear sequence motif and thus focus solely on the β chain in their analyses.

Collectively, these recent methods provide an improved insight into T cell repertoires on the sequence level, as obtained by Rep-Seq. The clustering methods improve our understanding of parameters of interest such as repertoire diversity and potentially help to overcome some of the limitations initially described in this chapter. It is furthermore interesting to see that a certain degree of function prediction can be achieved solely on the sequence level, at least for narrowly defined classes as achieved by single peptide tetramer stimulation. At the same time, the identification of multiple distinct sequence based clonotype clusters specific for the same pMHC with low similarity across clusters also clearly shows that the very same peptide can be recognized by rather dissimilar TCRs, as also stated by the authors. Therefore, it still remains crucial to explore learning methods that are able to generalize beyond the raw amino acid sequence and allow more complex relations between clonotype features, as we proposed in Chapter 7.

Appendices

A.1. Gene Segment Ambiguity Resolution

In the following, we describe the *gene segment ambiguity resolution* routine used to resolve cases, where the same clonotype is partially called with a more ambiguous set of V or J gene segments. The concept is described in Section 4.6.1.

Let $(\mathcal{C}, \mathcal{F})$ with $\mathcal{C} = \{c_1, \dots, c_N\}, \mathcal{F} = \{f_1, \dots, f_N\}$ be a subpopulation of a clonotype repertoire, where all clonotypes have *identical* CDR3 sequences. For every $c_i \in \mathcal{C}$ we then denote

$$\text{submatch}(c_i) := \left\{ j \,\middle|\, (c_j \neq c_i) \wedge (c_j^V \subseteq c_i^V) \wedge (c_j^J \subseteq c_i^J), 1 \leq j \leq N \right\}$$

as the set of indices of those clonotypes in $\mathcal{C}$ different from c, whose V and J segments are a subset or equal to those of c. Let $(\mathcal{C}, \mathcal{F})$ furthermore be sorted in descending order by the total number of gene segments assigned to each clonotype, i.e.

$$\left|c_i^V\right| + \left|c_i^J\right| \geq \left|c_j^V\right| + \left|c_j^J\right| \qquad\qquad \forall 0 \leq i < j \leq N.$$

We then redistribute the clonotype counts from every clonotype c_i to its more refined matching clonotypes:

```
for $i \leftarrow 1, \dots, N$ do
    $\mathcal{I} \leftarrow \textsc{submatch}(c_i)$
    $\mathcal{F}^* \leftarrow \{f_j \mid f_j \in \mathcal{F}, j \in \mathcal{I}\}$
    $\textsc{RedistCounts}(f_i, \mathcal{F})$
end for
```

where RedistCounts is equivalently implemented as shown in Algorithm 4.4. By iterating through the clonotypes from the most ambiguous to the least ambiguous we capture as many subsequent corrections as possible, i.e. design the clustering hierarchically.

A.2. IMSEQ Reference FASTA Format

The reference files for IMSEQ have to contain the V and J gene sequences of the gene (e.g. TRB or IGH) and species of interest. The FASTA IDs have to comprise five fields separated by the "|" character:

1. The gene name
2. The segment type, either "V", "J" or "D"
3. The segment identifier
4. The allele number and
5. The position of the Cys_{104} and Phe_{118} encoding triplet. "-1" for D segments. The counting is zero-based and points to the first character of the triplet.

IMSIM parses the same reference format as IMSEQ. D segment sequences are ignored by IMSEQ.

Example

```
...
>TRB|V|9|02|270
gattctggagtcacacaaaccccaaagcacctgatcacagcaactggacagcgagtgacg
ctgagatgctcccctaggtctggagacctctctgtgtactggtaccaacagagcctggac
cagggcctccagttcctcattcactattataatggagaagagagagcaaaaggaaacatt
cttgaacgattctccgcacaacagttccctgacttgcactctgaactaaacctgagctct
ctggagctgggggactcagctttgtatttctgtgccagcagcgtag
>TRB|V|9|03|270
gattctggagtcacacaaaccccaaagcacctgatcacagcaactggacagcgagtgacg
ctgagatgctcccctaggtctggagacctctctgtgtactggtaccaacagagcctggac
cagggcctccagttcctcattcaatattataatggagaagagagagcaaaaggaaacatt
cttgaacgattctccgcacaacagttccctgacttgcactctgaactaaacctgagctct
ctggagctgggggactcagctttgtatttctgtgccagcagc
>TRB|J|1-1|01|17
tgaacactgaagctttctttggacaaggcaccagactcacagttgtag
>TRB|J|1-2|01|17
ctaactatggctacaccttcggttcggggaccaggttaaccgttgtag
...
```

A.3. V Segment Core Fragment Length Auto Tuning

As described in Section 4.4.2, the segment core fragments are searched semi-globally in the read sequence, i.e. they have to be contained as a whole. As illustrated in Figure 4.3, depending on the incorporated J gene segment and the length of the CDR3 region, the part of the read covering non-CDR3 V gene segment sequence varies in length between clonotypes. While the length of the V SCF, L^V_{SCF}, is a parameter that can be freely configured by the user when invoking IMSEQ, by default it is set using the heuristic

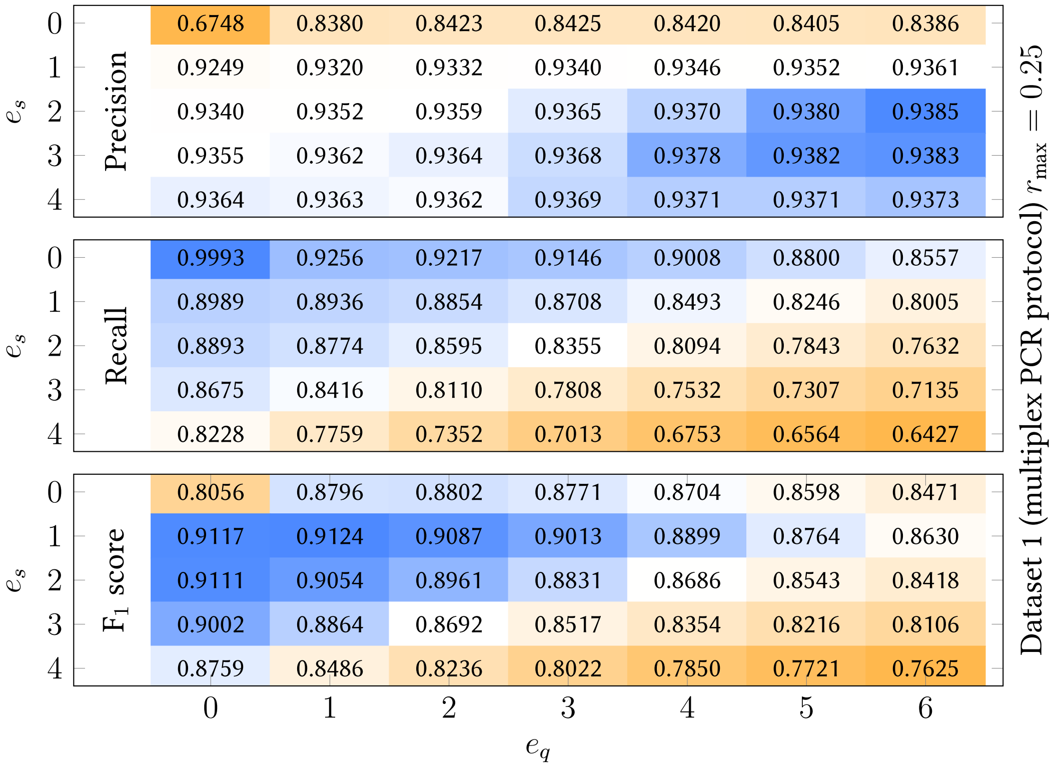
$$L^V_{\text{SCF}} = \begin{cases} 10 & \text{if } L^{\text{read}}_{\min} \leq 120 \\ \min \begin{cases} 60 \\ L^{\text{read}}_{\min} - 110 \end{cases} & \text{otherwise} \end{cases},$$

where $L^{\text{read}}_{\min}$ is the shortest V(D)J-read length observed in the input data. That is, we use a minimum SCF length of 10, which is motivated by the fact that data which does not contain at least 10 V gene segment bases outside the CDR3 region can most likely not be reliably annotated, even when using paired-end data. Furthermore, we use a maximum SCF length of 60, as we do not expect any performance improvements for higher values as shown in Section 5.3.3. We start expanding the V SCF length from a minimum V(D)J-read length of 120 and larger, based on the assumption that this ensures sufficiently many bases of V gene segment, considering cases of long CDR3 regions.

It is important to note that these values are all based on the human TRB gene with the Rep-Seq protocol as described in Section 2.4. While the characteristics are likely to be slightly different in other genes, species or different protocols, an SCF length value that is not perfectly adjusted to the data will most likely only have an impact on the performance and not on the outcome of the analysis.

A.4. Additional Error Correction Parameters

The precision, recall and their harmonic mean (F1 score) for various combinations of the e_q and e_s parameters, computed based on the real data error evaluation (see Section 5.3.5 for details). The following figures show the results for both Dataset 1 and 2 as well as for $r_{\max} \in \{0.25, 0.50, 0.75\}$.



Dataset 1 (multiplex PCR protocol) $r_{\max} = 0.25$

Precision

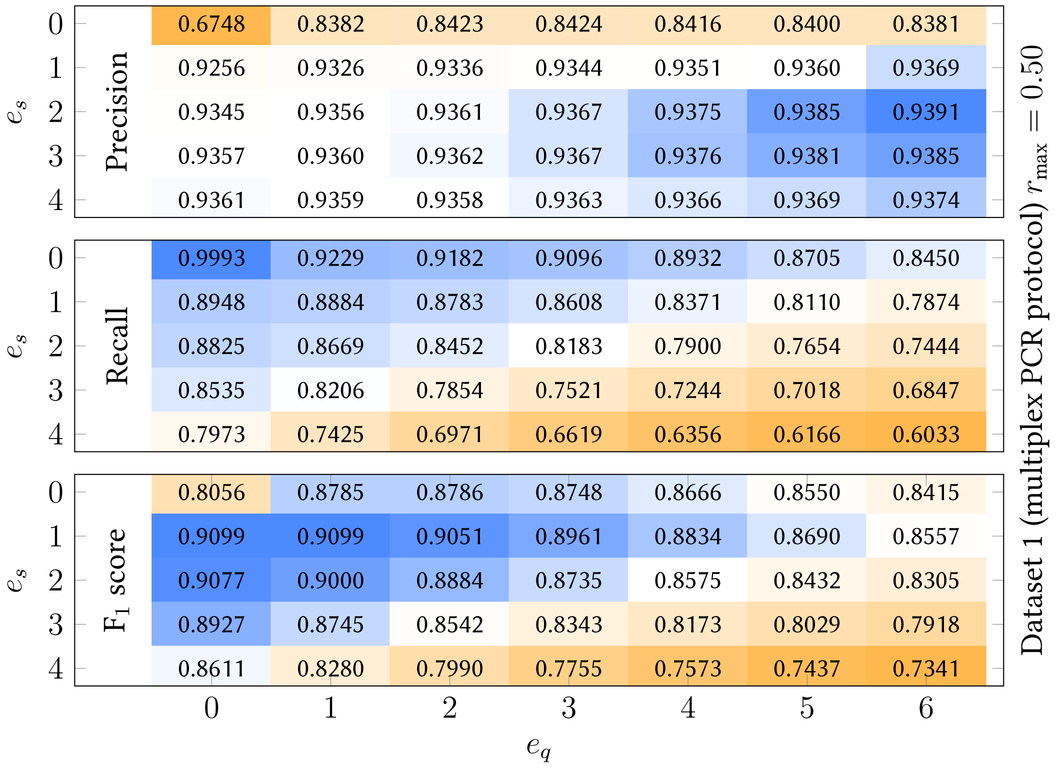
e_s \ e_q	0	1	2	3	4	5	6
0	0.6748	0.8380	0.8423	0.8425	0.8420	0.8405	0.8386
1	0.9249	0.9320	0.9332	0.9340	0.9346	0.9352	0.9361
2	0.9340	0.9352	0.9359	0.9365	0.9370	0.9380	0.9385
3	0.9355	0.9362	0.9364	0.9368	0.9378	0.9382	0.9383
4	0.9364	0.9363	0.9362	0.9369	0.9371	0.9371	0.9373

Recall

e_s \ e_q	0	1	2	3	4	5	6
0	0.9993	0.9256	0.9217	0.9146	0.9008	0.8800	0.8557
1	0.8989	0.8936	0.8854	0.8708	0.8493	0.8246	0.8005
2	0.8893	0.8774	0.8595	0.8355	0.8094	0.7843	0.7632
3	0.8675	0.8416	0.8110	0.7808	0.7532	0.7307	0.7135
4	0.8228	0.7759	0.7352	0.7013	0.6753	0.6564	0.6427

F_1 score

e_s \ e_q	0	1	2	3	4	5	6
0	0.8056	0.8796	0.8802	0.8771	0.8704	0.8598	0.8471
1	0.9117	0.9124	0.9087	0.9013	0.8899	0.8764	0.8630
2	0.9111	0.9054	0.8961	0.8831	0.8686	0.8543	0.8418
3	0.9002	0.8864	0.8692	0.8517	0.8354	0.8216	0.8106
4	0.8759	0.8486	0.8236	0.8022	0.7850	0.7721	0.7625



Precision

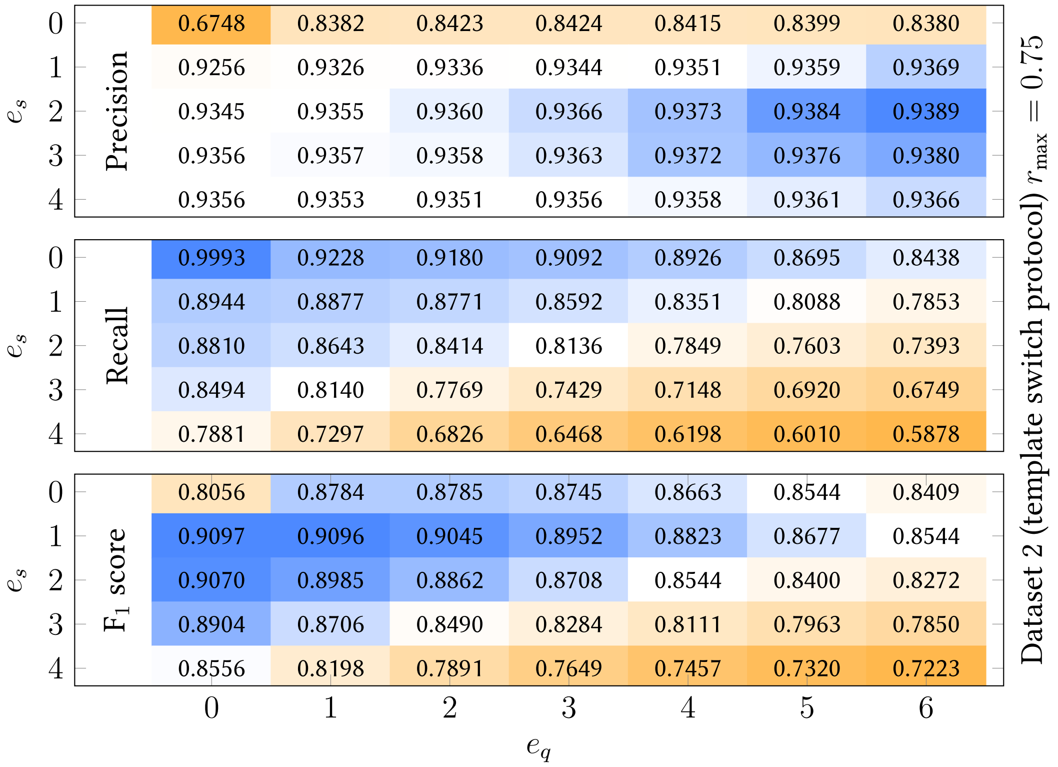
e_s \ e_q	0	1	2	3	4	5	6
0	0.6748	0.8382	0.8423	0.8424	0.8416	0.8400	0.8381
1	0.9256	0.9326	0.9336	0.9344	0.9351	0.9360	0.9369
2	0.9345	0.9356	0.9361	0.9367	0.9375	0.9385	0.9391
3	0.9357	0.9360	0.9362	0.9367	0.9376	0.9381	0.9385
4	0.9361	0.9359	0.9358	0.9363	0.9366	0.9369	0.9374

Recall

e_s \ e_q	0	1	2	3	4	5	6
0	0.9993	0.9229	0.9182	0.9096	0.8932	0.8705	0.8450
1	0.8948	0.8884	0.8783	0.8608	0.8371	0.8110	0.7874
2	0.8825	0.8669	0.8452	0.8183	0.7900	0.7654	0.7444
3	0.8535	0.8206	0.7854	0.7521	0.7244	0.7018	0.6847
4	0.7973	0.7425	0.6971	0.6619	0.6356	0.6166	0.6033

F_1 score

e_s \ e_q	0	1	2	3	4	5	6
0	0.8056	0.8785	0.8786	0.8748	0.8666	0.8550	0.8415
1	0.9099	0.9099	0.9051	0.8961	0.8834	0.8690	0.8557
2	0.9077	0.9000	0.8884	0.8735	0.8575	0.8432	0.8305
3	0.8927	0.8745	0.8542	0.8343	0.8173	0.8029	0.7918
4	0.8611	0.8280	0.7990	0.7755	0.7573	0.7437	0.7341



Precision

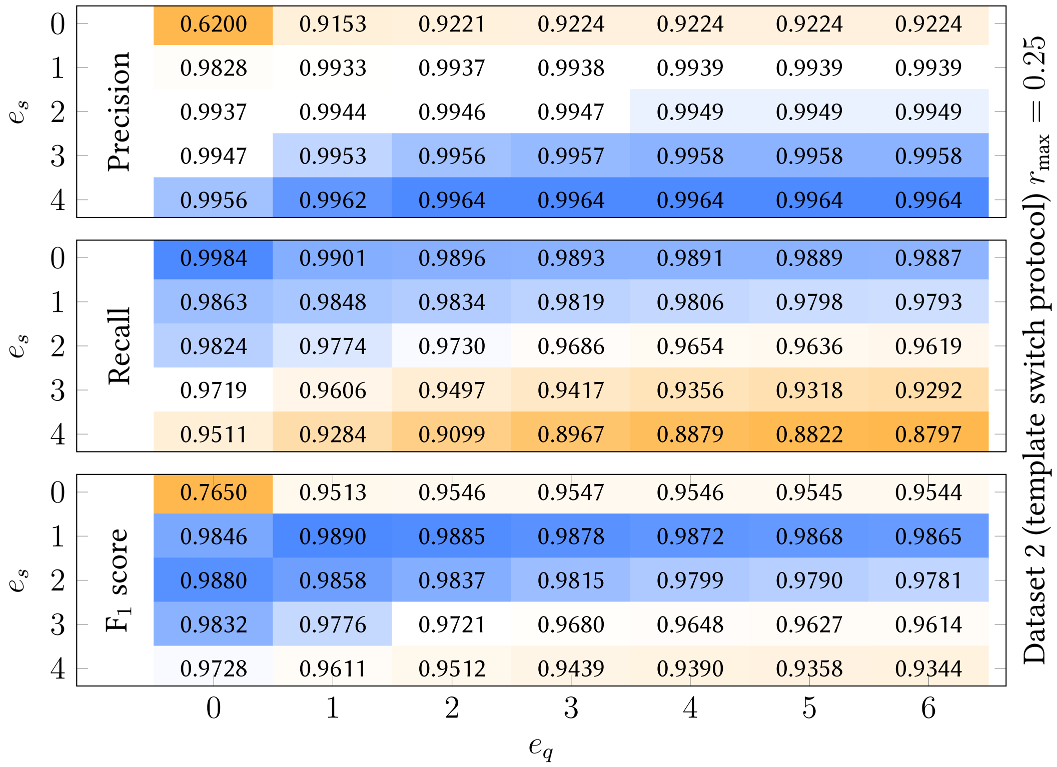
e_s \ e_q	0	1	2	3	4	5	6
0	0.6748	0.8382	0.8423	0.8424	0.8415	0.8399	0.8380
1	0.9256	0.9326	0.9336	0.9344	0.9351	0.9359	0.9369
2	0.9345	0.9355	0.9360	0.9366	0.9373	0.9384	0.9389
3	0.9356	0.9357	0.9358	0.9363	0.9372	0.9376	0.9380
4	0.9356	0.9353	0.9351	0.9356	0.9358	0.9361	0.9366

Recall

e_s \ e_q	0	1	2	3	4	5	6
0	0.9993	0.9228	0.9180	0.9092	0.8926	0.8695	0.8438
1	0.8944	0.8877	0.8771	0.8592	0.8351	0.8088	0.7853
2	0.8810	0.8643	0.8414	0.8136	0.7849	0.7603	0.7393
3	0.8494	0.8140	0.7769	0.7429	0.7148	0.6920	0.6749
4	0.7881	0.7297	0.6826	0.6468	0.6198	0.6010	0.5878

F_1 score

e_s \ e_q	0	1	2	3	4	5	6
0	0.8056	0.8784	0.8785	0.8745	0.8663	0.8544	0.8409
1	0.9097	0.9096	0.9045	0.8952	0.8823	0.8677	0.8544
2	0.9070	0.8985	0.8862	0.8708	0.8544	0.8400	0.8272
3	0.8904	0.8706	0.8490	0.8284	0.8111	0.7963	0.7850
4	0.8556	0.8198	0.7891	0.7649	0.7457	0.7320	0.7223



Dataset 2 (template switch protocol) $r_{\max} = 0.25$

Precision

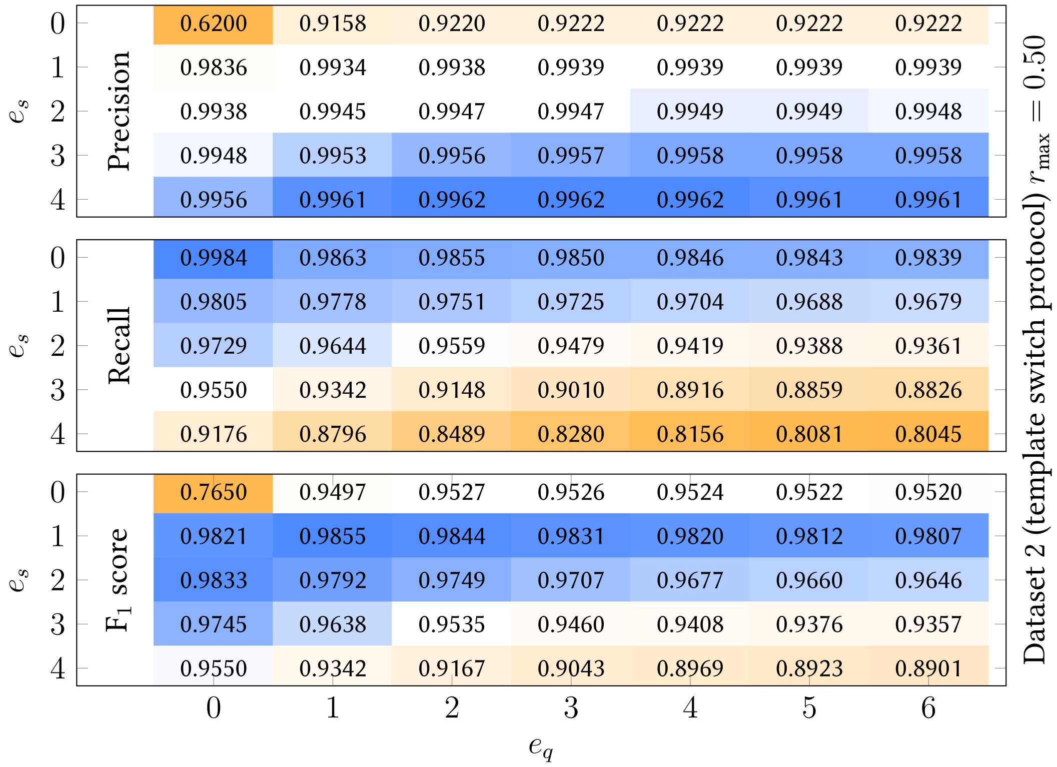
e_s \ e_q	0	1	2	3	4	5	6
0	0.6200	0.9153	0.9221	0.9224	0.9224	0.9224	0.9224
1	0.9828	0.9933	0.9937	0.9938	0.9939	0.9939	0.9939
2	0.9937	0.9944	0.9946	0.9947	0.9949	0.9949	0.9949
3	0.9947	0.9953	0.9956	0.9957	0.9958	0.9958	0.9958
4	0.9956	0.9962	0.9964	0.9964	0.9964	0.9964	0.9964

Recall

e_s \ e_q	0	1	2	3	4	5	6
0	0.9984	0.9901	0.9896	0.9893	0.9891	0.9889	0.9887
1	0.9863	0.9848	0.9834	0.9819	0.9806	0.9798	0.9793
2	0.9824	0.9774	0.9730	0.9686	0.9654	0.9636	0.9619
3	0.9719	0.9606	0.9497	0.9417	0.9356	0.9318	0.9292
4	0.9511	0.9284	0.9099	0.8967	0.8879	0.8822	0.8797

F_1 score

e_s \ e_q	0	1	2	3	4	5	6
0	0.7650	0.9513	0.9546	0.9547	0.9546	0.9545	0.9544
1	0.9846	0.9890	0.9885	0.9878	0.9872	0.9868	0.9865
2	0.9880	0.9858	0.9837	0.9815	0.9799	0.9790	0.9781
3	0.9832	0.9776	0.9721	0.9680	0.9648	0.9627	0.9614
4	0.9728	0.9611	0.9512	0.9439	0.9390	0.9358	0.9344



Dataset 2 (template switch protocol) $r_{\max} = 0.50$

Precision

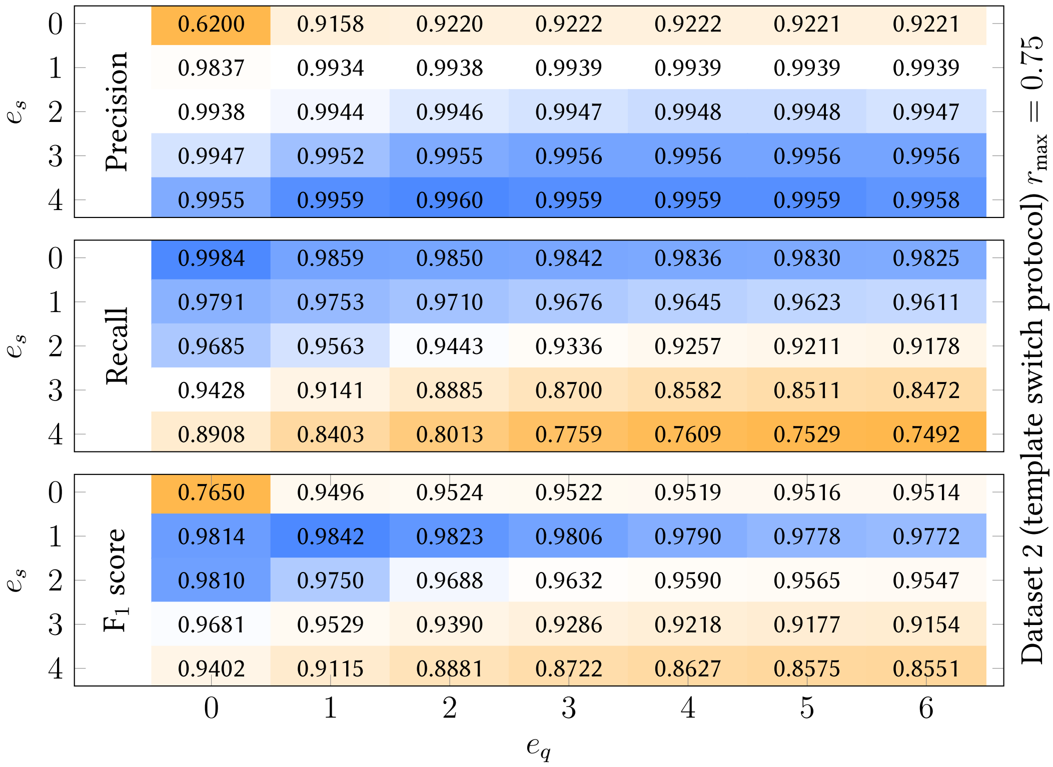
e_s \ e_q	0	1	2	3	4	5	6
0	0.6200	0.9158	0.9220	0.9222	0.9222	0.9222	0.9222
1	0.9836	0.9934	0.9938	0.9939	0.9939	0.9939	0.9939
2	0.9938	0.9945	0.9947	0.9947	0.9949	0.9949	0.9948
3	0.9948	0.9953	0.9956	0.9957	0.9958	0.9958	0.9958
4	0.9956	0.9961	0.9962	0.9962	0.9962	0.9961	0.9961

Recall

e_s \ e_q	0	1	2	3	4	5	6
0	0.9984	0.9863	0.9855	0.9850	0.9846	0.9843	0.9839
1	0.9805	0.9778	0.9751	0.9725	0.9704	0.9688	0.9679
2	0.9729	0.9644	0.9559	0.9479	0.9419	0.9388	0.9361
3	0.9550	0.9342	0.9148	0.9010	0.8916	0.8859	0.8826
4	0.9176	0.8796	0.8489	0.8280	0.8156	0.8081	0.8045

F_1 score

e_s \ e_q	0	1	2	3	4	5	6
0	0.7650	0.9497	0.9527	0.9526	0.9524	0.9522	0.9520
1	0.9821	0.9855	0.9844	0.9831	0.9820	0.9812	0.9807
2	0.9833	0.9792	0.9749	0.9707	0.9677	0.9660	0.9646
3	0.9745	0.9638	0.9535	0.9460	0.9408	0.9376	0.9357
4	0.9550	0.9342	0.9167	0.9043	0.8969	0.8923	0.8901



Dataset 2 (template switch protocol) $r_{\max} = 0.75$

Precision

e_s \ e_q	0	1	2	3	4	5	6
0	0.6200	0.9158	0.9220	0.9222	0.9222	0.9221	0.9221
1	0.9837	0.9934	0.9938	0.9939	0.9939	0.9939	0.9939
2	0.9938	0.9944	0.9946	0.9947	0.9948	0.9948	0.9947
3	0.9947	0.9952	0.9955	0.9956	0.9956	0.9956	0.9956
4	0.9955	0.9959	0.9960	0.9959	0.9959	0.9959	0.9958

Recall

e_s \ e_q	0	1	2	3	4	5	6
0	0.9984	0.9859	0.9850	0.9842	0.9836	0.9830	0.9825
1	0.9791	0.9753	0.9710	0.9676	0.9645	0.9623	0.9611
2	0.9685	0.9563	0.9443	0.9336	0.9257	0.9211	0.9178
3	0.9428	0.9141	0.8885	0.8700	0.8582	0.8511	0.8472
4	0.8908	0.8403	0.8013	0.7759	0.7609	0.7529	0.7492

F_1 score

e_s \ e_q	0	1	2	3	4	5	6
0	0.7650	0.9496	0.9524	0.9522	0.9519	0.9516	0.9514
1	0.9814	0.9842	0.9823	0.9806	0.9790	0.9778	0.9772
2	0.9810	0.9750	0.9688	0.9632	0.9590	0.9565	0.9547
3	0.9681	0.9529	0.9390	0.9286	0.9218	0.9177	0.9154
4	0.9402	0.9115	0.8881	0.8722	0.8627	0.8575	0.8551

A.5. Atchley Amino Acid Feature Map

Atchley et al. [130] developed a set of five numerical features per amino acid. Initially, they obtained 494 numerical amino acid indices, i.e. numerical descriptors for single amino acid features $\Omega : \Sigma_{\text{AA}} \to \mathbb{R}$. After performing a feature reduction to eliminate redundancies, 54 amino acid properties remained in their analysis, among them basic features such as polarity, size or charge but also more complex features related to their preference to occur in certain protein substructures. Based on these 54 amino acid indices, they used factor analysis [147] and determined five descriptive factors for each amino acid. The factors show a strong linkage to meaningful subgroups of the original feature set, thus allow for a certain degree of interpretability: Factor I reflects features related to polarity and free energy, Factor II is related to secondary structure, Factor III to size, weight and volume, Factor IV to mutation probability and number of codons and Factor V corresponds to features related to the charge of the molecule.

	I	**II**	**III**	**IV**	**V**
A	-0.59145974	-1.30209266	-0.7330651	1.5703918	-0.14550842
C	-1.34267179	0.46542300	-0.8620345	-1.0200786	-0.25516894
D	1.05015062	0.30242411	-3.6559147	-0.2590236	-3.24176791
E	1.35733226	-1.45275578	1.4766610	0.1129444	-0.83715681
F	-1.00610084	-0.59046634	1.8909687	-0.3966186	0.41194139
G	-0.38387987	1.65201497	1.3301017	1.0449765	2.06385566
H	0.33616543	-0.41662780	-1.6733690	-1.4738898	-0.07772917
I	-1.23936304	-0.54652238	2.1314349	0.3931618	0.81630366
K	1.83146558	-0.56109831	0.5332237	-0.2771101	1.64762794
L	-1.01895162	-0.98693471	-1.5046185	1.2658296	-0.91181195
M	-0.66312569	-1.52353917	2.2194787	-1.0047207	1.21181214
N	0.94535614	0.82846219	1.2991286	-0.1688162	0.93339498
P	0.18862522	2.08084151	-1.6283286	0.4207004	-1.39177378
Q	0.93056541	-0.17926549	-3.0048731	-0.5025910	-1.85303476
R	1.53754853	-0.05472897	1.5021086	0.4403185	2.89744417
S	-0.22788299	1.39869991	-4.7596375	0.6701745	-2.64747356
T	-0.03181782	0.32571153	2.2134612	0.9078985	1.31337035
V	-1.33661279	-0.27854634	-0.5440132	1.2419935	-1.26225362
W	-0.59533918	0.00907760	0.6719274	-2.1275244	-0.18358096
Y	0.25999617	0.82992312	3.0973596	-0.8380164	1.51150958

A.6. Additional Model Parameters

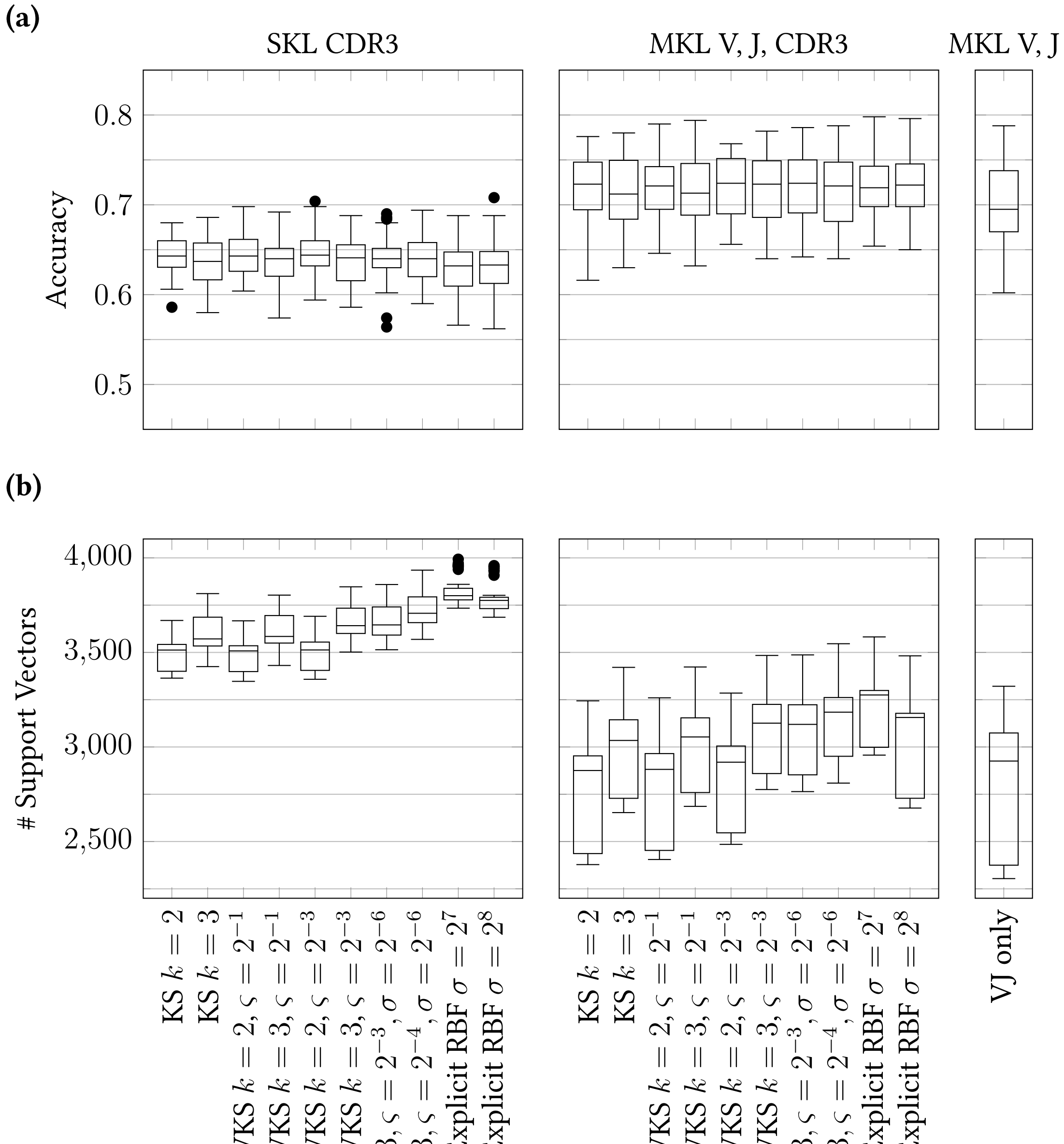


Figure A.1.: Additional parameters used during cross-validation. See Figure 7.9.

Zusammenfassung in deutscher Sprache

Die Fähigkeit von Wirbeltieren, Pathogene abzuwehren, basiert auf einer Reihe von Mechanismen, die sich in zwei Bereiche unterteilen lassen: Das *adaptive* und das *angeborene* Immunsystem. Während angeborene Immunität auf generischen Mechanismen beruht, welche z.B. das Vorhandensein von Bakterienzellen anhand von allgemeinen Parametern erkennen, sind die adaptiven Mechanismen in der Lage, neue Wege zu „erlernen“, bisher unbekannte Pathogene zu erkennen und zu bekämpfen. Vereinfacht gesagt werden immer neue Strategien auf zufällige Weise generiert, wobei das einzige Kriterium ist, dass sie nicht gegen den Wirtsorganismus selbst reaktiv sind.

Der dem adaptiven Charakter zugrundeliegende Prozess ist eine einzigartige, somatische Rekombination der Gene, welche für die Proteine kodieren, die diese pathogenen Strukturen erkennen: die *Antigen-Rezeptoren*. Durch die mittlerweile verfügbaren Hochdurchsatz-DNA-Sequenziermethoden ist es uns heute möglich, das Repertoire an Antigen-Rezeptor Genen, welches ein Individuum im Laufe der Zeit gebildet hat, ausgehend von einer Zell-Probe sichtbar zu machen („Immun-Repertoire-Sequenzierung“). Dies ermöglicht uns, das adaptive Immunsystem auf eine neue Art und Weise zu untersuchen, woraus sich eine Reihe möglicher medizinischer Anwendungen ergeben.

Im Kontext der Immun-Repertoire-Sequenzierung wurde im Rahmen dieser Arbeit zunächst eine Methode entwickelt, um die Rohdaten, die bei dieser Methode anfallen möglichst fehlerfrei zu annotieren. Hierbei wurde ein besonderes Augenmerk auf die verschiedenen technischen Fehlerquellen gelegt, sowohl auf solche, die allgemein im Kontext von DNA-Sequenzierung auftreten, als auch auf solche, die spezifisch für die Immun-Repertoire-Sequenzierung sind. Die Methode wird in dieser Arbeit zunächst inhaltlich beschrieben, bevor anschließend im Rahmen einer Evaluation ihre Überlegenheit im Vergleich zu zuvor veröffentlichten Methoden dargestellt wird.

Des Weiteren wurde ein auf maschinellem Lernen basierter Workflow entworfen, um die annotierten Daten zu interpretieren. Ziel hierbei ist es, unter Verwendung eines zuvor trainierten Modells eine gemessene Gensequenz funktional zu klassifizieren. Innerhalb des Workflows wurden verschiedene Modelle implementiert, welche in dieser Arbeit zunächst formal beschrieben werden.

Anhand von realen Daten aus dem Kontext eines binären Merkmals von T-Zellen, der erfolgten Differenzierung in *T-Helferzellen* und *zytotoxische T-Zellen*, werden anschließend die Fähigkeiten der Modelle, korrekte Klassifikationen vorzunehmen, evaluiert.

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet habe. Ich erkläre außerdem, die Arbeit nicht in einem früheren Promotionsverfahren eingereicht zu haben.

Berlin, den 19. Januar 2018
Léon Kuchenbecker

Bibliography

[1] J. E. Krebs, E. S. Goldstein, and S. T. Kilpatrick. *Lewin's Genes XII.* Jones & Bartlett Learning, 2017. ISBN 978-1284104493.

[2] J. D. Watson and F. H. C. Crick. Molecular structure of nucleic acids, 1953. ISSN 0028-0836.

[3] R. Saiki, D. Gelfand, S. Stoffel, S. Scharf, R. Higuchi, G. Horn, K. Mullis, and H. Erlich. Primer-directed enzymatic amplification of DNA with a thermostable DNA polymerase. *Science*, 239(4839):487–491, 1988. ISSN 0036-8075. doi: 10.1126/science.2448875.

[4] J. Cline. PCR fidelity of pfu DNA polymerase and other thermostable DNA polymerases. *Nucleic Acids Research*, 24(18):3546–3551, 1996. ISSN 13624962. doi: 10.1093/nar/24.18.3546.

[5] F. Sanger, S. Nicklen, and A. R. Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences*, 74(12):5463–5467, 1977. ISSN 0027-8424. doi: 10.1073/pnas.74.12.5463.

[6] M. L. Metzker. Sequencing technologies — the next generation. *Nature Reviews Genetics*, 11(1):31–46, 2010. ISSN 1471-0056. doi: 10.1038/nrg2626.

[7] K. Nakamura, T. Oshima, T. Morimoto, S. Ikeda, H. Yoshikawa, Y. Shiwa, S. Ishikawa, M. C. Linak, A. Hirai, H. Takahashi, M. Altaf-Ul-Amin, N. Ogasawara, and S. Kanaya. Sequence-specific error profile of Illumina sequencers. *Nucleic Acids Research*, 39(13), 2011. doi: 10.1093/nar/gkr344.

[8] F. Krueger, S. R. Andrews, and C. S. Osborne. Large Scale Loss of Data in Low-Diversity Illumina Sequencing Libraries Can Be Recovered by Deferred Cluster Calling. *PLoS ONE*, 6(1):e16607, 2011. ISSN 1932-6203. doi: 10.1371/journal.pone.0016607.

[9] K. Murphy and C. Weaver. *Janeway's Immunobiology, 9th edition.* 2016. ISBN 9781315533247.

[10] M. F. Flajnik and M. Kasahara. Origin and evolution of the adaptive immune system: genetic events and selective pressures. *Nature Reviews Genetics*, 11(1):47–59, 2010. ISSN 1471-0056. doi: 10.1038/nrg2703.

[11] J. L. Jorgensen, U. Esser, B. Fazekas de St. Groth, P. A. Reay, and M. M. Davis. Mapping T-cell receptor–peptide contacts by variant peptide immunization of single-chain transgenics. *Nature*, 355(6357):224–230, 1992. ISSN 0028-0836. doi:

10.1038/355224a0.

[12] D. N. Garboczi and W. E. Biddison. Shapes of MHC Restriction. *Immunity*, 10(1): 1–7, 1999. ISSN 10747613. doi: 10.1016/S1074-7613(00)80001-1.

[13] B. A. Cobb, Q. Wang, A. O. Tzianabos, and D. L. Kasper. Polysaccharide processing and presentation by the MHCII pathway. *Cell*, 117(5):677–687, 2004. ISSN 00928674. doi: 10.1016/j.cell.2004.05.001.

[14] S. Tonegawa. Somatic generation of antibody diversity. *Nature*, 302(5909): 575–581, 1983. doi: 10.1038/302575a0.

[15] M.-P. Lefranc and G. Lefranc. *The T cell receptor FactsBook*. Gulf Professional Publishing, 2001. ISBN 978-0-12-441352-8.

[16] M.-P. Lefranc, C. Pommié, M. Ruiz, V. Giuducelli, E. Foulquier, L. Truong, V. Thouvenin-Contet, and G. Lefranc. IMGT unique numbering for immunoglobulin and T cell receptor variable domains and Ig superfamily V-like domains. *Developmental and Comparative Immunology*, 27(1):55–77, 2003. ISSN 0145305X. doi: 10.1016/S0145-305X(02)00039-3.

[17] Z. Li, C. J. Woo, M. D. Iglesias-Ussel, D. Ronai, and M. D. Scharff. The generation of antibody diversity through somatic hypermutation and class switch recombination. *Genes and Development*, 18(1):1–11, 2004. doi: 10.1101/gad.1161904.

[18] B. L. Brady, N. C. Steinel, and C. H. Bassing. Antigen Receptor Allelic Exclusion: An Update and Reappraisal. *The Journal of Immunology*, 185(7):3801–3808, 2010. ISSN 0022-1767. doi: 10.4049/jimmunol.1001158.

[19] E. Padovan, G. Casorati, P. Dellabona, S. Meyer, M. Brockhaus, and A. Lanzavecchia. Expression of two T cell receptor alpha chains: dual receptor T cells. *Science (New York, N.Y.)*, 262(5132):422–4, 1993. ISSN 0036-8075.

[20] M. G. Macey. Principles of flow cytometry. *Flow Cytometry: Principles and Applications*, 16(4):1–15, 2007. ISSN 09553886. doi: 10.1007/978-1-59745-451-3_1.

[21] J. J. M. van Dongen, A. W. Langerak, M. Brüggemann, P. A. S. Evans, M. Hummel, F. L. Lavender, E. Delabesse, F. Davi, E. Schuuring, R. García-Sanz, J. H. J. M. van Krieken, J. Droese, D. González, C. Bastard, H. E. White, M. Spaargaren, M. González, A. Parreira, J. L. Smith, G. J. Morgan, M. Kneba, and E. A. Macintyre. Design and standardization of PCR primers and protocols for detection of clonal immunoglobulin and T-cell receptor gene recombinations in suspect lymphoproliferations: Report of the BIOMED-2 Concerted Action BMH4-CT98-3936. *Leukemia*, 17(12):2257–2317, 2003. doi: 10.1038/sj.leu.2403202.

[22] H. S. Robins, P. V. Campregher, S. K. Srivastava, A. Wacher, C. J. Turtle, O. Kahsai, S. R. Riddell, E. H. Warren, and C. S. Carlson. Comprehensive assessment of T-cell receptor β-chain diversity in $\alpha\beta$ T cells. *Blood*, 114(19):4099–4107, 2010. doi: 10.1182/blood-2009-04-217604.

[23] M. F. Polz and C. M. Cavanaugh. Bias in template-to-product ratios in multitem-

plate PCR. *Applied and environmental microbiology*, 64(10):3724–30, 1998. ISSN 0099-2240.

[24] I. Z. Mamedov, O. V. Britanova, I. V. Zvyagin, M. A. Turchaninova, D. A. Bolotin, E. V. Putintseva, Y. B. Lebedev, and D. M. Chudakov. Preparing unbiased T-cell receptor and antibody cDNA libraries for the deep next generation sequencing profiling. *Frontiers in Immunology*, 4:456, 2013. ISSN 16643224. doi: 10.3389/fimmu.2013.00456.

[25] M. Matz, D. Shagin, E. Bogdanova, O. Britanova, S. Lukyanov, L. Diatchenko, and A. Chenchik. Amplification of cDNA ends based on template-switching effect and step-out PCR. *Nucleic Acids Research*, 27(6):1558–1560, 1999. ISSN 03051048. doi: 10.1093/nar/27.6.1558.

[26] J. A. Casbon, R. J. Osborne, S. Brenner, and C. P. Lichtenstein. A method for counting PCR template molecules with application to next-generation sequencing. *Nucleic Acids Research*, 39(12), 2011. doi: 10.1093/nar/gkr217.

[27] T. Kivioja, A. Vähärautio, K. Karlsson, M. Bonke, M. Enge, S. Linnarsson, and J. Taipale. Counting absolute numbers of molecules using unique molecular identifiers. *Nature Methods*, 9(1):72–74, 2011. ISSN 1548-7091. doi: 10.1038/nmeth.1778.

[28] S. Islam, A. Zeisel, S. Joost, G. La Manno, P. Zajac, M. Kasper, P. Lönnerberg, and S. Linnarsson. Quantitative single-cell RNA-seq with unique molecular identifiers. *Nature Methods*, 11(2):163–166, 2013. ISSN 1548-7091. doi: 10.1038/nmeth.2772.

[29] J. König, K. Zarnack, G. Rot, T. Curk, M. Kayikci, B. Zupan, D. J. Turner, N. M. Luscombe, and J. Ule. iCLIP reveals the function of hnRNP particles in splicing at individual nucleotide resolution. *Nature Structural & Molecular Biology*, 17(7):909–915, 2010. doi: 10.1038/nsmb.1838.

[30] Q. He, J. Johnston, and J. Zeitlinger. ChIP-nexus enables improved detection of in vivo transcription factor binding footprints. *Nature Biotechnology*, 33(4):395–401, 2015. doi: 10.1038/nbt.3121.

[31] K. Karlsson, E. Sahlin, E. Iwarsson, M. Westgren, M. Nordenskjöld, and S. Linnarsson. Amplification-free sequencing of cell-free DNA for prenatal non-invasive diagnosis of chromosomal aberrations. *Genomics*, 105(3):150–158, 2015. doi: 10.1016/j.ygeno.2014.12.005.

[32] C. Vollmers, R. V. Sit, J. A. Weinstein, C. L. Dekker, and S. R. Quake. Genetic measurement of memory B-cell recall using antibody repertoire sequencing. *Proceedings of the National Academy of Sciences*, 110(33):13463–13468, 2013. ISSN 0027-8424. doi: 10.1073/pnas.1312146110.

[33] R. L. Warren, J. D. Freeman, T. Zeng, G. Choe, S. Munro, R. Moore, J. R. Webb, and R. A. Holt. Exhaustive T-cell repertoire sequencing of human peripheral blood samples reveals signatures of antigen selection and a directly measured

repertoire size of at least 1 million clonotypes. *Genome Research*, 21(5):790–797, 2011. doi: 10.1101/gr.115428.110.

[34] J. D. Freeman, R. L. Warren, J. R. Webb, B. H. Nelson, and R. a. Holt. Profiling the T-cell receptor beta-chain repertoire by massively parallel sequencing. *Genome Research*, 19(10):1817–1824, 2009. ISSN 1088-9051. doi: 10.1101/gr.092924.109.

[35] H. S. Robins, S. K. Srivastava, P. V. Campregher, C. J. Turtle, J. Andriesen, S. R. Riddell, C. S. Carlson, and E. H. Warren. Overlap and effective size of the human CD8+ T cell receptor repertoire. *Science translational medicine*, 2(47):47ra64, 2010. ISSN 1946-6242. doi: 10.1126/scitranslmed.3001442.

[36] J. A. J. A. Weinstein, N. N. Jiang, R. A. R. A. White, D. S. D. S. Fisher, and S. R. S. R. Quake. High-throughput sequencing of the zebrafish antibody repertoire. *Science*, 324(5928):807–810, 2009. ISSN 1095-9203. doi: 10.1126/science.1170020.

[37] S. D. Boyd, E. L. Marshall, J. D. Merker, J. M. Maniar, L. N. Zhang, B. Sahaf, C. D. Jones, B. B. Simen, B. Hanczaruk, K. D. Nguyen, K. C. Nadeau, M. Egholm, D. B. Miklos, J. L. Zehnder, and A. Z. Fire. Measurement and Clinical Monitoring of Human Lymphocyte Clonality by Massively Parallel V-D-J Pyrosequencing. *Science Translational Medicine*, 1(12):12ra23–12ra23, 2009. ISSN 1946-6234. doi: 10.1126/scitranslmed.3000540.

[38] J. Glanville, W. Zhai, J. Berka, D. Telman, G. Huerta, G. R. Mehta, I. Ni, L. Mei, P. D. Sundar, G. M. R. Day, D. Cox, A. Rajpal, and J. Pons. Precise determination of the diversity of a combinatorial antibody library gives insight into the human immunoglobulin repertoire. *Proceedings of the National Academy of Sciences*, 106(48):20216–20221, 2009. doi: 10.1073/pnas.0909775106.

[39] D. Gusfield. Algorithms on strings, trees, and sequences: computer science and computational biology, 1997. ISSN 01635700.

[40] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals, 1966. ISSN 00385689.

[41] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970. ISSN 00222836. doi: 10.1016/0022-2836(70)90057-4.

[42] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981. ISSN 00222836. doi: 10.1016/0022-2836(81)90087-5.

[43] O. Gotoh. An improved algorith for matching biological sequences. *Journal of Molecular Biology*, 162:705–708, 1982. ISSN 00222836. doi: 10.1016/0022-2836(82)90398-9.

[44] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 89(22):10915–10919, 1992. doi: 10.1073/pnas.89.22.10915.

[45] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, and R. Durbin. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079, 2009. ISSN 13674803. doi: 10.1093/bioinformatics/btp352.

[46] N. Thomas, K. Best, M. Cinelli, S. Reich-Zeliger, H. Gal, E. Shifrut, A. Madi, N. Friedman, J. Shawe-Taylor, and B. Chain. Tracking global changes induced in the CD4 T-cell receptor repertoire by immunization with a complex antigen using short stretches of CDR3 protein sequence. *Bioinformatics*, 30(22):3181–3188, 2014. ISSN 14602059. doi: 10.1093/bioinformatics/btu523.

[47] M. Shugay, O. V. Britanova, E. M. Merzlyak, M. a. Turchaninova, I. Z. Mamedov, T. R. Tuganbaev, D. a. Bolotin, D. B. Staroverov, E. V. Putintseva, K. Plevova, C. Linnemann, D. Shagin, S. Pospisilova, S. Lukyanov, T. N. Schumacher, and D. M. Chudakov. Towards error-free profiling of immune repertoires. *Nature methods*, 11(6):653–5, 2014. ISSN 1548-7105. doi: 10.1038/nmeth.2960.

[48] C. Wang, Y. Liu, M. M. Cavanagh, S. Le Saux, Q. Qi, K. M. Roskin, T. J. Looney, J.-Y. Lee, V. Dixit, C. L. Dekker, G. E. Swan, J. J. Goronzy, and S. D. Boyd. B-cell repertoire responses to varicella-zoster vaccination in human identical twins. *Proceedings of the National Academy of Sciences of the United States of America*, 112(2):500–5, 2015. ISSN 1091-6490. doi: 10.1073/pnas.1415875112.

[49] B. Langmead and S. L. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature methods*, 9(4):357–9, 2012. ISSN 1548-7105. doi: 10.1038/nmeth.1923.

[50] H. Li and R. Durbin. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, 25(14):1754–1760, 2009. ISSN 13674803. doi: 10.1093/bioinformatics/btp324.

[51] D. Weese, M. Holtgrewe, and K. Reinert. RazerS 3: Faster, fully sensitive read mapping. *Bioinformatics*, 28(20):2592–2599, 2012. ISSN 13674803. doi: 10.1093/bioinformatics/bts505.

[52] D. Weese, A. K. Emde, T. Rausch, A. Döring, and K. Reinert. RazerS - Fast read mapping with sensitivity control. *Genome Research*, 19(9):1646–1654, 2009. ISSN 10889051. doi: 10.1101/gr.088823.108.

[53] E. Siragusa, D. Weese, and K. Reinert. Fast and accurate read mapping with approximate seeds and multiple backtracking. *Nucleic Acids Research*, 41(7), 2013. ISSN 03051048. doi: 10.1093/nar/gkt005.

[54] A. Dobin and T. R. Gingeras. Mapping RNA-seq Reads with STAR. *Current Protocols in Bioinformatics*, pages 11.14.1–11.14.19, 2015. ISSN 1934-340X. doi: 10.1002/0471250953.bi1114s51.

[55] G. Myers. A fast bit-vector algorithm for approximate string matching based on dynamic programming. *Journal of the ACM*, 46(3):395–415, 1999. ISSN 00045411. doi: 10.1145/316542.316550.

[56] M. Martin. Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal*, 17(1):10, 2011. ISSN 2226-6089. doi: 10.14806/ej.17.1.200.

[57] A. M. Bolger, M. Lohse, and B. Usadel. Trimmomatic: A flexible trimmer for Illumina sequence data. *Bioinformatics*, 30(15):2114–2120, 2014. ISSN 14602059. doi: 10.1093/bioinformatics/btu170.

[58] M. Dodt, J. Roehr, R. Ahmed, and C. Dieterich. FLEXBAR—Flexible Barcode and Adapter Processing for Next-Generation Sequencing Platforms. *Biology*, 1(3): 895–905, 2012. ISSN 2079-7737. doi: 10.3390/biology1030895.

[59] K. R. Rasmussen, J. Stoye, and E. W. Myers. Efficient q-gram Filters for Finding All ϵ-Matches over a Given Length. *Journal of Computational Biology*, 13(2): 296–308, 2006. ISSN 1066-5277. doi: 10.1089/cmb.2006.13.296.

[60] W. J. Masek and M. S. Paterson. A faster algorithm computing string edit distances. *Journal of Computer and System sciences*, 20(1):18–31, 1980.

[61] E. Ukkonen. Finding approximate patterns in strings. *Journal of Algorithms*, 6 (1):132–137, 1985. ISSN 01966774. doi: 10.1016/0196-6774(85)90023-9.

[62] J. C. Dohm, C. Lottaz, T. Borodina, and H. Himmelbauer. Substantial biases in ultra-short read data sets from high-throughput DNA sequencing. *Nucleic Acids Research*, 36(16), 2008. ISSN 03051048. doi: 10.1093/nar/gkn425.

[63] A. M. Dunning, P. Talmud, and S. E. Humphries. Errors in the polymerase chain reaction. *Nucleic Acids Research*, 16(21):10393, 1988. ISSN 03051048. doi: 10.1093/nar/16.21.10393.

[64] J. Brodin, M. Mild, C. Hedskog, E. Sherwood, T. Leitner, B. Andersson, and J. Albert. PCR-Induced Transitions Are the Major Source of Error in Cleaned Ultra-Deep Pyrosequencing Data. *PLoS ONE*, 8(7), 2013. ISSN 19326203. doi: 10.1371/journal.pone.0070388.

[65] A. Döring, D. Weese, T. Rausch, and K. Reinert. SeqAn An efficient, generic C++ library for sequence analysis. *BMC Bioinformatics*, 9(1):11, 2008. ISSN 1471-2105. doi: 10.1186/1471-2105-9-11.

[66] M.-P. Lefranc. IMGT, the international ImMunoGeneTics database®. *Nucleic Acids Research*, 31(1):307–310, 2003. ISSN 13624962. doi: 10.1093/nar/gkg085.

[67] V. Giudicelli, D. Chaume, and M.-P. Lefranc. IMGT/GENE-DB: a comprehensive database for human and mouse immunoglobulin and T cell receptor genes. *Nucleic acids research*, 33(Database issue):D256–61, 2005. ISSN 1362-4962. doi: 10.1093/nar/gki010.

[68] W. R. Pearson and D. J. Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the United States of America*, 85(8):2444–8, 1988. ISSN 0027-8424. doi: 10.1073/pnas.85.8.2444.

[69] J. Zhang, K. Kobert, T. Flouri, and A. Stamatakis. PEAR: A fast and accurate

Illumina Paired-End reAd mergeR. *Bioinformatics*, 30(5):614–620, 2014. ISSN 13674803. doi: 10.1093/bioinformatics/btt593.

[70] V. Giudicelli, X. Brochet, and M. P. Lefranc. IMGT/V-QUEST: IMGT standardized analysis of the immunoglobulin (IG) and T cell receptor (TR) nucleotide sequences. *Cold Spring Harbor Protocols*, 6(6):695–715, 2011. ISSN 15596095. doi: 10.1101/pdb.prot5633.

[71] J. Ye, N. Ma, T. L. Madden, and J. M. Ostell. IgBLAST: an immunoglobulin variable domain sequence analysis tool. *Nucleic acids research*, 41(Web Server issue):34–40, 2013. ISSN 13624962. doi: 10.1093/nar/gkt382.

[72] S. Li, M.-P. Lefranc, J. J. Miles, E. Alamyar, V. Giudicelli, P. Duroux, J. D. Freeman, V. D. A. Corbin, J.-P. Scheerlinck, M. A. Frohman, P. U. Cameron, M. Plebanski, B. Loveland, S. R. Burrows, A. T. Papenfuss, and E. J. Gowans. IMGT/HighV QUEST paradigm for T cell receptor IMGT clonotype diversity and next generation repertoire immunoprofiling. *Nature Communications*, 4(1):2333, 2013. ISSN 2041-1723. doi: 10.1038/ncomms3333.

[73] IgBlast Online (accessed 2017-07-05). URL https://www.ncbi.nlm.nih.gov/igblast/.

[74] D. A. Bolotin, M. Shugay, I. Z. Mamedov, E. V. Putintseva, M. A. Turchaninova, I. V. Zvyagin, O. V. Britanova, and D. M. Chudakov. MiTCR: software for T-cell receptor sequencing data analysis. *Nature methods*, 10(9):813–4, 2013. ISSN 1548-7105. doi: 10.1038/nmeth.2555.

[75] D. A. Bolotin, S. Poslavsky, I. Mitrophanov, M. Shugay, I. Z. Mamedov, E. V. Putintseva, and D. M. Chudakov. MiXCR: software for comprehensive adaptive immunity profiling. *Nature Methods*, 12(5):380–381, 2015. ISSN 1548-7091. doi: 10.1038/nmeth.3364.

[76] Y. Liao, G. K. Smyth, and W. Shi. The Subread aligner: Fast, accurate and scalable read mapping by seed-and-vote. *Nucleic Acids Research*, 41(10), 2013. ISSN 03051048. doi: 10.1093/nar/gkt214.

[77] X. Yang, D. Liu, N. Lv, F. Zhao, F. Liu, J. Zou, Y. Chen, X. Xiao, J. Wu, P. Liu, J. Gao, Y. Hu, Y. Shi, J. Liu, R. Zhang, C. Chen, J. Ma, G. F. Gao, and B. Zhu. TCRklass: a new K-string-based algorithm for human and mouse TCR repertoire characterization. *Journal of immunology (Baltimore, Md. : 1950)*, 194(1):446–54, 2015. ISSN 1550-6606. doi: 10.4049/jimmunol.1400711.

[78] N. Thomas, J. Heather, W. Ndifon, J. Shawe-Taylor, and B. Chain. Decombinator: A tool for fast, efficient gene assignment in T-cell receptor sequences using a finite state machine. *Bioinformatics*, 29(5):542–550, 2013. ISSN 13674803. doi: 10.1093/bioinformatics/btt004.

[79] A. V. Aho and M. J. Corasick. Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6):333–340, 1975. ISSN 00010782. doi: 10.1145/360825.360855.

[80] M. Giraud, M. Salson, M. Duez, C. Villenet, S. Quief, A. Caillault, N. Grardel, C. Roumier, C. Preudhomme, and M. Figeac. Fast multiclonal clusterization of V(D)J recombinations from high-throughput sequencing. *BMC Genomics*, 15(1): 409, 2014. ISSN 1471-2164. doi: 10.1186/1471-2164-15-409.

[81] M. Duez, M. Giraud, R. Herbert, T. Rocher, M. Salson, and F. Thonier. Vidjil: A Web Platform for Analysis of High-Throughput Repertoire Sequencing. *PLOS ONE*, 11(11):e0166126, 2016. ISSN 1932-6203. doi: 10.1371/journal.pone.0166126.

[82] J. A. Vander Heiden, G. Yaari, M. Uduman, J. N. H. Stern, K. C. O'connor, D. A. Hafler, F. Vigneault, and S. H. Kleinstein. PRESTO: A toolkit for processing high-throughput sequencing raw reads of lymphocyte receptor repertoires. *Bioinformatics*, 30(13):1930–1932, 2014. ISSN 14602059. doi: 10.1093/bioinformatics/btu138.

[83] M. Holtgrewe. Mason – A Read Simulator for Second Generation Sequencing Data. *Life Sciences*, (October):18, 2010.

[84] L. Kuchenbecker, M. Nienen, J. Hecht, A. U. Neumann, N. Babel, K. Reinert, and P. N. Robinson. IMSEQ—a fast and error aware approach to immunogenetic sequence analysis. *Bioinformatics*, 31(18):2963–2971, 2015. ISSN 1367-4803. doi: 10.1093/bioinformatics/btv309.

[85] I. V. Zvyagin, I. Z. Mamedov, O. V. Tatarinova, E. A. Komech, E. E. Kurnikova, E. V. Boyakova, V. Brilliantova, L. N. Shelikhova, D. N. Balashov, M. Shugay, A. L. Sycheva, S. A. Kasatskaya, Y. B. Lebedev, A. A. Maschan, M. A. Maschan, and D. M. Chudakov. Tracking T-cell immune reconstitution after TCR$\alpha\beta$/CD19-depleted hematopoietic cells transplantation in children. *Leukemia*, 31(November):1–26, 2016. ISSN 0887-6924. doi: 10.1038/leu.2016.321.

[86] P. Libby and J. S. Pober. Chronic rejection. *Immunity*, 14(4):387–397, 2001. ISSN 10747613. doi: 10.1016/S1074-7613(01)00119-4.

[87] G. Einecke, B. Sis, J. Reeve, M. Mengel, P. M. Campbell, L. G. Hidalgo, B. Kaplan, and P. F. Halloran. Antibody-Mediated Microcirculation Injury Is the Major Cause of Late Kidney Transplant Failure. *American Journal of Transplantation*, 9(11):2520–2531, 2009. ISSN 16006135. doi: 10.1111/j.1600-6143.2009.02799.x.

[88] P. F. Halloran, J. P. Reeve, A. B. Pereira, L. G. Hidalgo, and K. S. Famulski. Antibody-mediated rejection, T cell–mediated rejection, and the injury-repair response: new insights from the Genome Canada studies of kidney transplant biopsies. *Kidney International*, 85(2):258–264, 2014. ISSN 00852538. doi: 10.1038/ki.2013.300.

[89] J. R. Chapman. What are the key challenges we face in kidney transplantation today? *Transplantation Research*, 2(1):S1, 2013. ISSN 2047-1440. doi: 10.1186/2047-1440-2-S1-S1.

[90] S. D. Gardner, A. M. Field, D. V. Coleman, and B. Hulme. New human papovavirus (B.K.) isolated from urine after renal transplantation. *Lancet (London, England)*,

1(7712):1253–7, 1971. ISSN 0140-6736. doi: 10.1016/S0140-6736(71)91776-4.

[91] M. Reploeg, G. Storch, and D. Clifford. BK virus: A clinical review. *Clinical Infectious Diseases*, 33(2):191–202, 2001. ISSN 10584838. doi: 10.1086/321813.

[92] J. Heritage, P. M. Chesters, and D. J. McCance. The persistence of papovavirus BK DNA sequences in normal human renal tissue. *Journal of Medical Virology*, 8(2):143–150, 1981. ISSN 01466615. doi: 10.1002/jmv.1890080208.

[93] N. Babel, H.-D. Volk, and P. Reinke. BK polyomavirus infection and nephropathy: the virus–immune system interplay. *Nature Reviews Nephrology*, 7(7):399–406, 2011. ISSN 1759-5061. doi: 10.1038/nrneph.2011.59.

[94] D. L. Bohl and D. C. Brennan. BK virus nephropathy and kidney transplantation. *Clinical Journal of the American Society of Nephrology*, 2(SUPPL. 1), 2007. ISSN 15559041. doi: 10.2215/CJN.00920207.

[95] C. B. Drachenberg, J. C. Papadimitriou, H. H. Hirsch, R. Wali, C. Crowder, J. Nogueira, C. B. Cangro, S. Mendley, A. Mian, and E. Ramos. Histological Patterns of Polyomavirus Nephropathy: Correlation with Graft Outcome and Viral Load. *American Journal of Transplantation*, 4(12):2082–2092, 2004. ISSN 1600-6135. doi: 10.1046/j.1600-6143.2004.00603.x.

[96] D. N. Howell, S. R. Smith, D. W. Butterly, P. S. Klassen, H. R. Krigman, J. L. Burchette, and S. E. Miller. Diagnosis and management of BK polyomavirus interstitial nephritis in renal transplant recipients. *Transplantation*, 68(9):1279–88, 1999. ISSN 0041-1337.

[97] M. Dziubianau, J. Hecht, L. Kuchenbecker, A. Sattler, U. Stervbo, C. Rödelsperger, P. Nickel, A. U. Neumann, P. N. Robinson, S. Mundlos, H.-D. Volk, A. Thiel, P. Reinke, and N. Babel. TCR Repertoire Analysis by Next Generation Sequencing Allows Complex Differential Diagnosis of T Cell-Related Pathology. *American Journal of Transplantation*, 13(11):2842–2854, 2013. ISSN 16006135. doi: 10.1111/ajt.12431.

[98] S. Landwehr-Kenzel, F. Issa, S.-H. Luu, M. Schmück, H. Lei, A. Zobel, A. Thiel, N. Babel, K. Wood, H.-D. Volk, and P. Reinke. Novel GMP-Compatible Protocol Employing an Allogeneic B Cell Bank for Clonal Expansion of Allospecific Natural Regulatory T Cells. *American Journal of Transplantation*, 14(3):594–606, 2014. ISSN 16006135. doi: 10.1111/ajt.12629.

[99] T. Szczepariski, A. Orfão, V. H. van der Valden, J. F. S. Miguel, and J. J. van Dongen. Minimal residual disease in leukaemia patients. *The Lancet Oncology*, 2(7):409–417, 2001. ISSN 14702045. doi: 10.1016/S1470-2045(00)00418-6.

[100] R. Higuchi, C. Fockler, G. Dollinger, and R. Watson. Kinetic PCR Analysis: Real-time Monitoring of DNA Amplification Reactions. *Nature Biotechnology*, 11(9): 1026–1030, 1993. ISSN 1087-0156. doi: 10.1038/nbt0993-1026.

[101] M. J. Borowitz, M. Devidas, S. P. Hunger, W. P. Bowman, A. J. Carroll, W. L. Carroll,

S. Linda, P. L. Martin, D. J. Pullen, D. Viswanatha, C. L. Willman, N. Winick, and B. M. Camitta. Clinical significance of minimal residual disease in childhood acute lymphoblastic leukemia and its relationship to other prognostic factors: a Children's Oncology Group study. *Blood*, 111(12):5477–5485, 2008. ISSN 0006-4971. doi: 10.1182/blood-2008-01-132837.

[102] M. Roshal, J. R. Fromm, S. Winter, K. Dunsmore, and B. L. Wood. Immaturity associated antigens are lost during induction for T cell lymphoblastic leukemia: Implications for minimal residual disease detection. *Cytometry Part B: Clinical Cytometry*, 78B(3):139–146, 2010. ISSN 15524949. doi: 10.1002/cyto.b.20511.

[103] V. H. J. van der Velden, A. Hochhaus, G. Cazzaniga, T. Szczepanski, J. Gabert, and J. J. M. van Dongen. Detection of minimal residual disease in hematologic malignancies by real-time quantitative PCR: principles, approaches and laboratory aspects. *Leukemia*, 17(6):1013–1034, 2003. ISSN 0887-6924. doi: 10.1038/sj.leu.2402922.

[104] S. D. Boyd, E. L. Marshall, J. D. Merker, J. M. Maniar, L. N. Zhang, B. Sahaf, C. D. Jones, B. B. Simen, B. Hanczaruk, K. D. Nguyen, K. C. Nadeau, M. Egholm, D. B. Miklos, J. L. Zehnder, and A. Z. Fire. Measurement and Clinical Monitoring of Human Lymphocyte Clonality by Massively Parallel V-D-J Pyrosequencing. *Science Translational Medicine*, 1(12):12ra23–12ra23, 2009. ISSN 1946-6234. doi: 10.1126/scitranslmed.3000540.

[105] A. C. Logan, H. Gao, C. Wang, B. Sahaf, C. D. Jones, E. L. Marshall, I. Buno, R. Armstrong, A. Z. Fire, K. I. Weinberg, M. Mindrinos, J. L. Zehnder, S. D. Boyd, W. Xiao, R. W. Davis, and D. B. Miklos. High-throughput VDJ sequencing for quantification of minimal residual disease in chronic lymphocytic leukemia and immune reconstitution assessment. *Proceedings of the National Academy of Sciences*, 108(52):21194–21199, 2011. ISSN 0027-8424. doi: 10.1073/pnas.1118357109.

[106] D. Wu, A. Sherwood, J. R. Fromm, S. S. Winter, K. P. Dunsmore, M. L. Loh, H. A. Greisman, D. E. Sabath, B. L. Wood, and H. Robins. High-Throughput Sequencing Detects Minimal Residual Disease in Acute T Lymphoblastic Leukemia. *Science Translational Medicine*, 4(134):134ra63–134ra63, 2012. ISSN 1946-6234. doi: 10.1126/scitranslmed.3003656.

[107] C. Wang, C. M. Sanders, Q. Yang, H. W. Schroeder, E. Wang, F. Babrzadeh, B. Gharizadeh, R. M. Myers, J. R. Hudson, R. W. Davis, and J. Han. High throughput sequencing reveals a complex pattern of dynamic interrelationships among human T cell subsets. *Proceedings of the National Academy of Sciences*, 107(4): 1518–1523, 2010. ISSN 0027-8424. doi: 10.1073/pnas.0913939107.

[108] J. J. Calis and B. R. Rosenberg. Characterizing immune repertoires by high throughput sequencing: strategies and applications. *Trends in Immunology*, 35 (12):581–590, 2014. ISSN 14714906. doi: 10.1016/j.it.2014.09.004.

[109] R. O. Emerson, W. S. DeWitt, M. Vignali, J. Gravley, J. K. Hu, E. J. Osborne,

C. Desmarais, M. Klinger, C. S. Carlson, J. A. Hansen, M. Rieder, and H. S. Robins. Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire. *Nature Genetics*, 49(5):659–665, 2017. ISSN 1061-4036. doi: 10.1038/ng.3822.

[110] N. Tickotsky, T. Sagiv, J. Prilusky, E. Shifrut, and N. Friedman. McPAS-TCR: a manually curated catalogue of pathology-associated T cell receptor sequences. *Bioinformatics*, 33(18):2924–2929, 2017. ISSN 1367-4803. doi: 10.1093/bioinformatics/btx286.

[111] M. Shugay, I. V. Zvyagin, and D. M. Chudakov. VDJdb. URL https://vdjdb.cdr3.net/.

[112] Y. Yanagi, A. Tishon, H. Lewicki, B. A. Cubitt, and M. B. Oldstone. Diversity of T-cell receptors in virus-specific cytotoxic T lymphocytes recognizing three distinct viral epitopes restricted by a single major histocompatibility complex molecule. *Journal of virology*, 66(4):2527–31, 1992. ISSN 0022-538X.

[113] J. L. Maryanski, C. Jongeneel, P. Bucher, J.-L. Casanova, and P. R. Walker. Single-Cell PCR Analysis of TCR Repertoires Selected by Antigen In Vivo: A High Magnitude CD8 Response Is Comprised of Very Few Clones. *Immunity*, 4(1): 47–55, 1996. ISSN 10747613. doi: 10.1016/S1074-7613(00)80297-6.

[114] J. A. Owen, J. Punt, J. Kuby, and S. A. Stranford. *Kuby Immunology: International Edition*. Macmillan Learning, 2013. ISBN 9781464137846.

[115] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual workshop on Computational learning theory - COLT '92*, pages 144–152, 1992. ISSN 0-89791-497-X. doi: 10.1145/130385.130401.

[116] W. Karush. Minima of functions of several variables with inequalities as side conditions. *Master thesis, University of Chicago*, 1939.

[117] H. W. Kuhn and A. Tucker. Nonlinear Programming. *Proceedings of the Second Symposium on Mathematical Statistics and Probability*, pages 481–492, 1951.

[118] L. Bottou and C.-J. Lin. Support Vector Machine Solvers. *Large Scale Kernel Machines*, pages 301–320, 2007. doi: 10.1.1.127.511.

[119] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large Scale Multiple Kernel Learning. *Journal of Machine Learning Research*, 7:1531—-1565, 2006. ISSN 1532-4435.

[120] A. B. A. Graf and S. Borer. Normalization in Support Vector Machines. *Neural Computation*, 13:277–282, 2001. doi: 10.1007/3-540-45404-7_37.

[121] C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: a string kernel for SVM protein classification. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 575:564–575, 2002. ISSN 2335-6936. doi: 10.1142/9789812799623_0053.

[122] G. Rätsch and S. Sonnenburg. Accurate Splice Site Detection for Caenorhabditis elegans. In *Kernel Methods in Computational Biology*, pages 277–298. MIT press, 2004. ISBN 0262195097.

[123] A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, and K.-R. Müller. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807, 2000. ISSN 1367-4803. doi: 10.1093/bioinformatics/16.9.799.

[124] M. Dayhoff, R. Schwartz, and B. Orcutt. A model of evolutionary change in proteins, 1978. ISSN 17494613.

[125] G. Gonnet, M. Cohen, and S. Benner. Exhaustive matching of the entire protein sequence database. *Science*, 256(5062):1443–1445, 1992. ISSN 0036-8075. doi: 10.1126/science.1604319.

[126] D. T. Jones, W. R. Taylor, and J. M. Thornton. The rapid generation of mutation data matrices from protein sequences. *Bioinformatics*, 8(3):275–282, 1992. ISSN 13674803. doi: 10.1093/bioinformatics/8.3.275.

[127] J. Adachi and M. Hasegawa. Model of amino acid substitution in proteins encoded by mitochondrial DNA. *Journal of Molecular Evolution*, 42(4):459–468, 1996. ISSN 0022-2844. doi: 10.1007/BF02498640.

[128] T. Müller, R. Spang, and M. Vingron. Estimating Amino Acid Substitution Models: A Comparison of Dayhoff's Estimator, the Resolvent Approach and a Maximum Likelihood Method. *Molecular Biology and Evolution*, 19(1):8–13, 2002. ISSN 1537-1719. doi: 10.1093/oxfordjournals.molbev.a003985.

[129] M. Venkatarajan and W. Braun. New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physical-chemical properties. *Journal of Molecular Modeling*, 7(12):445–453, 2001. ISSN 16102940. doi: 10.1007/s00894-001-0058-5.

[130] W. R. Atchley, J. Zhao, A. D. Fernandes, and T. Drüke. Solving the protein sequence metric problem. *Proceedings of the National Academy of Sciences of the United States of America*, 102(18):6395–400, 2005. ISSN 0027-8424. doi: 10.1073/pnas.0408677102.

[131] N. C. Toussaint, C. Widmer, O. Kohlbacher, and G. Rätsch. Exploiting physico-chemical properties in string kernels. *BMC Bioinformatics*, 11(Suppl 8):S7, 2010. ISSN 1471-2105. doi: 10.1186/1471-2105-11-S8-S7.

[132] H. M. Li, T. Hiroi, Y. Zhang, A. Shi, G. Chen, S. De, E. J. Metter, W. H. Wood, A. Sharov, J. D. Milner, K. G. Becker, M. Zhan, and N.-p. Weng. TCR repertoire of CD4+ and CD8+ T cells is distinct in richness, distribution, and CDR3 amino acid composition. *Journal of Leukocyte Biology*, 99(3):505–513, 2016. ISSN 0741-5400. doi: 10.1189/jlb.6A0215-071RR.

[133] S. Sonnenburg, H. Strathmann, S. Lisitsyn, V. Gal, F. J. I. García, W. Lin, C. Zhang,

S. De, Frx, Tklein23, E. Andreev, JonasBehr, Sploving, P. Mazumdar, C. Widmer, A. Kislay, S. Mahindre, K. Hughes, R. Votyakov, Khalednasr, S. Sharma, A. Novik, A. Panda, E. Anagnostopoulos, L. Pang, Serialhex, A. Binder, E. Sørig, M. Uřičář, and B. Esser. Shogun 5.0.0 - Ōtomo no Yakamochi. 2016. doi: 10.5281/zenodo.164882. URL http://www.shogun-toolbox.org/.

[134] T. P. Arstila. A Direct Estimate of the Human T Cell Receptor Diversity. *Science*, 286(5441):958–961, 1999. ISSN 00368075. doi: 10.1126/science.286.5441.958.

[135] G. Yaari and S. H. Kleinstein. Practical guidelines for B-cell receptor repertoire sequencing analysis. *Genome Medicine*, 7(1):121, 2015. ISSN 1756-994X. doi: 10.1186/s13073-015-0243-2.

[136] M. C. Longo, M. S. Berninger, and J. L. Hartley. Use of uracil DNA glycosylase to control carry-over contamination in polymerase chain reactions. *Gene*, 93(1): 125–128, 1990. ISSN 03781119. doi: 10.1016/0378-1119(90)90145-H.

[137] G. Georgiou, G. C. Ippolito, J. Beausang, C. E. Busse, H. Wardemann, and S. R. Quake. The promise and challenge of high-throughput sequencing of the antibody repertoire. *Nature Biotechnology*, 32(2):158–168, 2014. ISSN 1087-0156. doi: 10.1038/nbt.2782.

[138] A. Murugan, T. Mora, A. M. Walczak, and C. G. Callan. Statistical inference of the generation probability of T-cell receptors from sequence repertoires. *Proceedings of the National Academy of Sciences*, 109(40):16161–16166, 2012. ISSN 0027-8424. doi: 10.1073/pnas.1212755109.

[139] W. H. Robinson. Sequencing the functional antibody repertoire—diagnostic and therapeutic discovery. *Nature Reviews Rheumatology*, 11(3):171–182, 2014. ISSN 1759-4790. doi: 10.1038/nrrheum.2014.220.

[140] B. J. DeKosky, G. C. Ippolito, R. P. Deschner, J. J. Lavinder, Y. Wine, B. M. Rawlings, N. Varadarajan, C. Giesecke, T. Dörner, S. F. Andrews, P. C. Wilson, S. P. Hunicke-Smith, C. G. Willson, A. D. Ellington, and G. Georgiou. High-throughput sequencing of the paired human immunoglobulin heavy and light chain repertoire. *Nature Biotechnology*, 31(2):166–169, 2013. ISSN 1087-0156. doi: 10.1038/nbt.2492.

[141] M. J. T. Stubbington, T. Lönnberg, V. Proserpio, S. Clare, A. O. Speak, G. Dougan, and S. A. Teichmann. T cell fate and clonality inference from single-cell transcriptomes. *Nature Methods*, 13(4):329–332, 2016. ISSN 1548-7091. doi: 10.1038/nmeth.3800.

[142] O. Stegle, S. A. Teichmann, and J. C. Marioni. Computational and analytical challenges in single-cell transcriptomics. *Nature Reviews Genetics*, 16(3):133–145, 2015. ISSN 1471-0056. doi: 10.1038/nrg3833.

[143] B. Howie, A. M. Sherwood, A. D. Berkebile, J. Berka, R. O. Emerson, D. W. Williamson, I. Kirsch, M. Vignali, M. J. Rieder, C. S. Carlson, and H. S. Robins. High-throughput pairing of T cell receptor and sequences. *Science Trans-*

lational Medicine, 7(301):301ra131–301ra131, 2015. ISSN 1946-6234. doi: 10.1126/scitranslmed.aac5624.

[144] P. Dash, A. J. Fiore-Gartland, T. Hertz, G. C. Wang, S. Sharma, A. Souquette, J. C. Crawford, E. B. Clemens, T. H. O. Nguyen, K. Kedzierska, N. L. La Gruta, P. Bradley, and P. G. Thomas. Quantifiable predictive features define epitope-specific T cell receptor repertoires. *Nature*, 547(7661):89–93, 2017. ISSN 1476-4687. doi: 10.1038/nature22383.

[145] J. D. Altman, P. A. H. Moss, P. J. R. Goulder, D. H. Barouch, M. G. McHeyzer-Williams, J. I. Bell, A. J. McMichael, and M. M. Davis. Phenotypic Analysis of Antigen-Specific T Lymphocytes. *Science*, 274(5284):94–96, 1996. ISSN 0036-8075. doi: 10.1126/science.274.5284.94.

[146] J. Glanville, H. Huang, A. Nau, O. Hatton, L. E. Wagar, F. Rubelt, X. Ji, A. Han, S. M. Krams, C. Pettus, N. Haas, C. S. L. Arlehamn, A. Sette, S. D. Boyd, T. J. Scriba, O. M. Martinez, and M. M. Davis. Identifying specificity groups in the T cell receptor repertoire. *Nature*, 547(7661):94–98, 2017. ISSN 1476-4687. doi: 10.1038/nature22976.

[147] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis*. New Jersey: Prentice-Hall., 5th edition, 2002. ISBN 0-13-092553-5.

List of Figures

List of Tables

Acronyms

ABMR antibody-mediated rejection 85

BKV BK virus 86

BKVAN BK virus associated nephropathy 86

CDR complementary determining region 20

DNA deoxyribonucleic acid 9

dNTP deoxyribose nucleoside triphosphate, a single DNA nucleotide 12

HLA human leucocyte antigen 19

HTS high throughput sequencing 14

IG immunoglobulin 19

IGH immunoglobulin heavy chain 18

IGL immunoglobulin light chain 18

ISD immunosuppressant drug 85

MHC major histocompatibility complex 19

MKL multiple kernel learning 101

MRD minimal residual disease 90

NGS next generation sequencing 14

PBMC peripheral blood mononuclear cell 87

PCR polymerase chain reaction 12

pMHC peptide MHC complex 19

PWM positional weight matrix 125

QP quadratic programming 97

qPCR real-time quantitative PCR 90

RBF radial basis function 100

RNA ribonucleic acid 10

RSS recombination signal sequences 22

SCF segment core fragment 48

SHM somatic hypermutation 24

SKL single kernel learning 101

SVM support vector machine 95

TCMR T cell-mediated rejection 85

TCR T cell receptor 19

TRA T cell receptor α chain 18

TRB T cell receptor β chain 18

UMI unique molecular identifier 28

WES whole exome sequencing 42

WGS whole genome sequencing 42