Statistical Methods for Adaptive Immune Receptor Repertoire Analysis and Comparison

Branden J. Olson

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2020

Reading Committee:

Frederick A. Matsen IV, Chair

Yen-Chi Chen

Philip Bradley

Program Authorized to Offer Degree:
Department of Statistics

University of Washington

Abstract

Statistical Methods for Adaptive Immune Receptor Repertoire Analysis and Comparison

Branden J. Olson

Chair of the Supervisory Committee:
Affiliate Professor Frederick A. Matsen IV
Department of Statistics

B and T cell receptors, also known as adaptive immune receptors, perform key roles in adaptive immunity. These proteins identify and deal with foreign invaders like viruses or bacteria, allowing for robust and long-lasting immunological protection. The DNA sequences coding for these receptors arise by a complex stochastic recombination process followed by a series of productivity-based filters, as well as affinity maturation for B cells, allowing for immense diversity in the circulating pool of these sequences. Thus, proper analysis of adaptive immune receptor repertoire sequence (AIRR-seq) datasets as well as the immune context surrounding them presents a formidable but necessary challenge to computational biologists. In this dissertation, I present three projects that contribute to AIRR-seq analysis with an emphasis on statistical methods for repertoire comparison.

BCR sequences diversify through mutations introduced by purpose-built cellular machinery. A recent paper has concluded that templated mutagenesis, a hypothesized process in which mutations in the BCR locus are introduced by copying short segments from other BCR genes, is a major contributor to BCR diversification in mice and humans. If true, this would overturn decades of research and methodology involving B cell diversification. In joint work with Julia Fukuyama, I re-evaluate this hypothesis by directing the author's method at potential template donor genes not present in B cell genomes to obtain estimates of the methods's false positive rates. We find FPRs that are similar to or even higher than the

original inferences, resulting in little to no evidence that templated mutagenesis occurs at a substantial rate.

As AIRR-seq datasets are typically large and complex, it is non-trivial to characterize and compare them in precise yet interpretable ways. I introduce a comprehensive summary statistic framework that efficiently performs a wide variety of biologically-meaningful repertoire summaries and comparisons, and demonstrate how this framework can be used to perform general-purpose model validation. We find that summaries vary in their ability to differentiate between datasets, although many can distinguish between certain dataset covariates. Further, we show that recombination-based statistics tend to be more discriminative characterizations of a repertoire than those describing the amino acid composition of the CDR3 region. The framework also directly provides a convenient multidimensional scaling setup for visualizing dissimilarities between repertoires.

Current methods of TCR repertoire comparison often incur a high loss of distributional information by considering overly simplistic sequence- or repertoire-level characteristics. Optimal transport methods can be used to compare distributions given some distance or metric between values in the sample space, with appealing theoretical and computational properties. I formulate a nonparametric approach to TCR repertoire comparison driven by contemporary optimal transport methods and a recently-created distance on the space of TCRs. I describe a clustering algorithm based on our methodology and show that it can extract biologically meaningful regions of a target repertoire with respect to a source repertoire using several case studies, thus competing with more complicated methods despite minimal modeling assumptions and a simpler pipeline. I also establish a randomization test to identify TCRs that are significantly enhanced between repertoires, and validate it using a proxy null distribution based on biological replicates.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ACTIVATION-INDUCED CYTIDINE DEAMINASE (AID): an enzyme found in germinal centers responsible for introducing somatic hypermutation in humans and gene conversion in some other species

AIRR-SEQ: adaptive immune receptor repertoire sequence analysis

ANTIGEN: a foreign molecule whose presence in the body induces an immune response, resulting in the production of antibodies (antigen is short for *anti*body *gen*erator)

EPITOPE: a segment of an antigen protein which can be recognized by a B cell or T cell

GENE CONVERSION: a mechanism that diversifies antibodies in certain species like chickens, rabbits, and sheep in which mutations are introduced by the nonreciprocal transfer of homologous pseudo-IgHV genes into the variable region

OPTIMAL TRANSPORT: a class of methods which identify how to map one probability distribution to another by minimizing the cumulative distance that each mass particle must travel

SOMATIC HYPERMUTATION (SHM): a special mutation process that operates on B cells within the germinal center, allowing for further diversification and optimization of BCRs specific to a given antigen

TEMPLATED MUTAGENESIS (TM): a hypothesized gene conversion-like process for SHM in humans and mice in which mutations are introduced by the nonreciprocal transfer of homologous IgHV genes into the variable region

V(D)J RECOMBINATION: a process by which B cells and T cells stochastically assemble different gene segments along with random edge trimming and insertions to generate a unique receptor

ACKNOWLEDGMENTS

First and foremost, I am eternally grateful to my adviser Erick for being an exemplary mentor, colleague, and friend. His guidance and enthusiasm allowed me to make it this far, and I can look back on a very fun and productive few years in the lab. Between learning the remarkable science of immunology essentially from scratch, chalkboarding sessions in the clubhouse, a stochastically delicious weekly pizza lunch, and the many trips and excursions across Seattle and Washington, I can say with confidence that my years spent in the group have been some of the best of my life.

A special thank you to Philip Bradley, Yen-Chi Chen, and Chetan Seshadri for serving on my dissertation committee, and providing excellent feedback and support. I am particularly thankful for your flexibility in coordinating my general and final exams which both happened to occur during the COVID-19 pandemic and resultant quarantine.

I was surrounded by many amazing colleagues during my doctoral studies at the University of Washington. My PhD cohort was full of talented and friendly folks who I will miss dearly. Between cheering for opposite sports teams at breweries with Sheridan, discussing culture, politics, and spatial stats in and out of the office with Max, the video game chats and coursework survival with Daphne, and the many other friendships that developed, it will be difficult to say goodbye. Thank you to Peter Guttorp for offering me a research associateship during my first year to examine interesting cloud models in the context of statistical climatology. Thank you to Ellen Reynolds, Kristine Chan, Tracy Pham, and rest of the Statistics department for consistently providing assistance and answered questions as I slowly but surely progressed through the PhD requirements. And thanks to the many professors who truly pushed my intellectual abilities to their limits with some challenging

but fascinating courses.

I was fortunate to collaborate with many talented people throughout the course of my doctoral studies. I am grateful to the co-authors of the manuscripts which comprise the material presented in this dissertation, including: Julia Fukuyama for the templated mutagenesis project; Pejvak Moghimi, Chaim Schramm, Duncan Ralph, Jason Vander Heiden, Mikhail Shugay, Adrian Shepherd, and William Lees for the `sumrep` project; and Phil Bradley, Stefan Schattgen, and Paul Thomas for the optimal transport project. I am grateful to Trevor Bedford, Christian Busse, Gordon Dale, and Joshy Jacob for discussions that improved the analyses of the templated mutagenesis project, to Leng-Siew Yeap and Duncan Ralph for help analyzing the *gpt* data, and to Christian Busse who provided helpful comments on the manuscript. I also thank Misha Pogorelyy for kindly providing post-processed data, Quentin Marcou for help running `IGoR`, and other members of the AIRR Software WG for early stage discussion for `sumrep`, especially Christian Busse, Enkelejda Miho, Inimary Toby, and Jian Ye. Special thanks go out to Arman Bilge for introducing me to evolutionary biology and the Matsen group, and to Kristian Davidsen and Will Dewitt for facilitating my introduction to computational immunology. Thank you to Amrit Dhar who set an example as the first statistics PhD student to graduate from Erick's group, and for our fun chats about an assortment of topics. I also thank all of the other Matsen group members who helped to cultivate an engaging and enjoyable environment inside and outside the lab.

I would not have developed a serious interest in graduate school were it not for the many inspirational teachers that cultivated a passion for academics over the course of my schooling. During grade school, this includes Cinnamon Cain, Paul Black, Gretchen Friedman, Bobbi Faulkner, Ryan Triece, Daniel Sohl, Mary Ann Stavney, Bob Zimmerman, and Dr. Charles Vogel. My departmental adviser Anne Dougherty, my undergraduate adviser Juan Restrepo, and my M.S. adviser Will Kleiber were instrumental to my studies at CU Boulder as well as my transition into a PhD program. Jem Corcoran, Vanja Dukic, and Manuel Lladser also

helped catalyze my interest in probability and statistics.

Finally, thanks to my family and friends who supported me throughout my graduate career, particularly my mother Valorie who has done an amazing job raising and supporting me over the years, and invested in my education from early on, my stepfather Dowdie who also helped support me from an early age, my brother Brian who has been a great friend and huge influence since childhood, and my wonderful wife Michelle who is the perfect partner and somehow found the strength to stick beside me every step of the way.

DEDICATION

To my wife Michelle, for the endless love and support during my six and a half years of graduate school

Chapter 1

INTRODUCTION TO ADAPTIVE IMMUNE RECEPTOR REPERTOIRE ANALYSIS

The adaptive immune system makes a great case for itself as the most fascinating system in the human body. It is capable of defending us from nearly any possible foreign invader though a large, complex assortment of cellular and molecular interactions. Not only can it build a tailor-made response to new invaders which it has never previously encountered, but it "saves its progress" by creating a memory of this response to deal with similar invaders in the future. Nevertheless, while many advancements in the last century have helped us to understand adaptive immunology, much of it remains a mystery.

The effectiveness and mystery of the adaptive immune system can both be attributed to its incredibly stochastic nature. Our bodies produce lymphocytes known as B cells and T cells which perform the key roles within the immune response, largely due to their expression of surface receptors that are able to bind to foreign invaders called antigens. These specialized receptors arise through a complicated process involving the random generation of DNA sequences, various stages of selection to filter out the bad ones, and, in the case of B cell receptors, an additional Darwinian process that optimizes receptors to bind very well to specific antigens. As a result, probabilistic modeling and statistical inference are crucial to understanding adaptive immunology, as well as its application to the numerous important related fields like vaccine design, epidemiology, and cancer immunotherapy.

The arrival of high-throughput sequencing methods has given us the ability to query an individual's immune receptor repertoire with unprecedented precision [5]. Thus, we can obtain reliable samples from a population repertoire of interest, and apply computational and statistical techniques to make inferences about this population. Even so, adaptive immune

receptor repertoire sequence (AIRR-seq) analysis is a relatively new focus within the broader field of biology, leaving a lot of room for progress to advance the fields of statistical and computational adaptive immunology [26, 53]. This dissertation presents three projects that contribute to AIRR-seq analysis and pave the way for further progress.

In this chapter we will review the theory and practice of AIRR-seq analysis as it pertains to the material in this thesis. We start by giving a brief overview of adaptive immunology, focusing on the roles of B and T cell receptors. We discuss the formation and structure of these receptors and how the body can produce immensely effective receptors in response to virtually any antigen. We conclude with a pragmatic discussion of contemporary AIRR-seq data analysis.

1.1 The adaptive immune system

The immune system can be broadly categorized into two groups: the innate immune system and the adaptive immune system. Both of these systems exist to protect an organism from foreign substances and invaders like viruses and bacteria. The innate immune system is likely at least a billion years old and exists in most animals as well as other organisms such as insects, plants, fungi. This system is a first line of defense against pathogens, invoking relatively simple yet often effective defense mechanisms such as physical or chemical barriers, and phagocytic cells that can destroy many pathogens. The adaptive immune system is a much younger and more sophisticated system that originated in vertebrates within the last 500 million years. This system is invoked for peskier invaders that make it past the innate immune system, and can often eliminate these invaders through a very flexible system of cellular and chemical responses. While both systems are important in defending an individual from foreign attackers, we will focus only on the adaptive immune system for the purposes of this dissertation.

The adaptive immune system gets its name from its capability to adapt and respond to virtually any antigen that your immune cells encounter, even if the body has never seen anything like it before. This is possible to due a complex system of many different cells

and processes, but the two key players are B cells and T cells. These two types of cells allow for a robust immune response that is able to recognize that an infection is occurring, eliminate the infection with a tailor-made response, and memorize how to deal with any similar invaders if they ever cause a future infection. How the adaptive immune resolves the infection is situation-dependent, and sometimes T cells are able to orchestrate a sufficient response without the involvement of B cells. More commonly, a response is initiated when a B cell "recognizes" an unwanted invader and presents it to a T cell. The T cell is then signaled to stimulate more B cells that can recognize this invader, and these B cells undergo a process which results in highly-optimized antibodies that are eventually able to neutralize the infection. Indeed, the ability of B cells and T cells to execute such a dynamic response is a true scientific marvel.

The dynamic nature of these cells mostly results from the receptors expressed on their surfaces, aptly named *B cell receptors* (BCRs) and *T cell receptors* (TCRs). These receptors exist to bind or attach to many types of objects, such as antigens or other types of cells. A given receptor is randomly generated during the development of the B or T cell, and will only be able to bind to a small subset of possible antigens. However, when a B or T cell does happen to bind to an antigen, it is stimulated to proliferate, which allows the immune system to eventually overwhelm the infection. These dynamic receptors can thus be thought of as the key to adaptive immunity, and their study comprises a vast portion of the field of computational immunology as well as the majority of the material in this thesis. We therefore spend the next few sections discussing in detail how these receptors arise and interact within the immune response, starting with a discussion of a fundamental underlying process known as V(D)J recombination.

1.2 V(D)J recombination

V(D)J recombination, also called *V(D)J rearrangement*, is a process that occurs in developing B and T cells that produces a unique DNA sequence eventually used to create the full BCR or TCR protein. The process is essentially the same for both cell types, so we describe it in

abstract here before discussing the context-specific details pertaining to each cell type in the following sections.

We use the V(D)J notation because it includes two distinct processes: VJ recombination and VDJ recombination. These two processes operate on sets of gene segments found in B cells and T cells, known as V (variable) genes, D (diversity) genes, and J (junction) genes. VDJ recombination begins by randomly choosing one V gene, one D gene, and one J gene from each of the respective gene sets. The process then possibly trims, or deletes, a random amount of nucleotides from the edges of these genes, and concludes by joining these trimmed genes together, possibly inserting some random, non-templated nucleotides between each segment. The result is a unique DNA sequence which the B or T cell may eventually translate into a protein (see the discussion below). Figure 1.1 gives an illustration of VDJ recombination in the context of BCR generation.

VJ recombination is similar but does not involve a set of D genes. That is, it randomly chooses one V gene and one J gene from the same set of V genes and J genes as above, randomly trims their edges, and inserts a random amount of non-templated nucleotides between them. The result is another newly-formed DNA sequence which may also eventually be translated into a protein. When a viable VJ and VDJ sequence are formed, they will be translated into protein chains which together comprise the full BCR or TCR.

The protein chains resulting from V(D)J recombination have a conserved structure that immunologists have partitioned into four *framework regions* and three *complementarity-determining regions* (CDRs). Framework regions are so named because they contain highly-conserved regions that are integral to the receptor's function. Consequently, framework regions tolerate very few mutations. In contrast, CDRs are loops within the chain that are generally thought to be involved with binding, and thus contain much higher levels of variability to allow for a diverse pool of receptor binding specificities. The third CDR loop, simply called the CDR3, is thought to be particularly important to binding, in part because it contains the full site of V(D)J recombination, and is closer to the binding tip of the chain than the other CDRs. Indeed, the CDR3 is an iconic and heavily-studied object within the

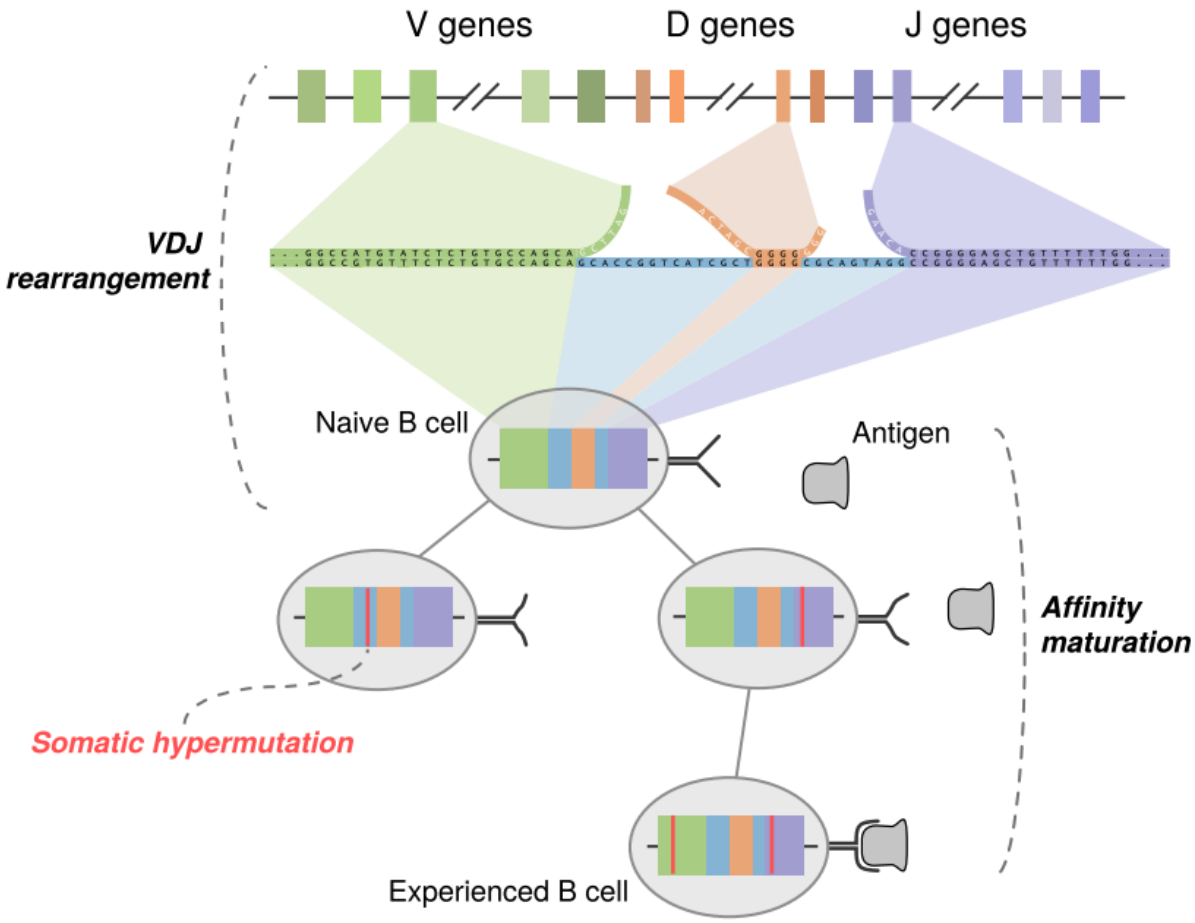


Figure 1.1: A depiction of V(D)J recombination followed by affinity maturation for B cell receptors. This figure is a modification of Figure 1 of [51].

field of adaptive immunology.

V(D)J recombination allows for a highly diverse set of possible BCR and TCR sequences for several reasons. First, there is a high amount of combinatorial diversity resulting from the various groupings of genes from their respective sets. For example, during BCR VDJ recombination in humans, there are approximately 40 V genes, 25 D genes, and 6 J genes from which to choose. This results in about $40 \times 25 \times 6 = 6,000$ unique combinations. Similarly, BCR VJ recombination in humans yields $40 \times 6 = 200$ unique combinations. Each of these combinations can be in principle combined with each of the others, leading to about $6,000 \times 200 = 1.2$ million potential choices.[1] On top of this combinatorial diversity, there is further diversity introduced from the random deletion and insertion of nucleotides during V(D)J recombination. Since these insertions and deletions arise from a non-templated stochastic process, this immensely amplifies the number of potential receptor sequences that could be generated; experts estimate that V(D)J recombination can theoretically produce at least 10^{11} different BCRs within an individual [50]. Finally, in the case of BCRs, there is yet another process that further diversifies the sequences called somatic hypermutation. We will discuss somatic hypermutation in detail in the section on B cell receptors below.

We conclude this section with some discussion about what it means for a regenerated V(D)J sequence to be a viable candidate for translation into a protein. This requires a review of the fundamental relationship between DNA, amino acids, and proteins. In a nutshell, DNA serves as a template for an organism to create proteins, which are large molecules that serve various functions or purposes within the organism. This happens by transcribing a given sequence of DNA to a sequence of messenger RNA, and then translating this messenger RNA into amino acids which make up the final protein product. Each amino acid in a protein sequence is comprised of exactly three nucleotides, and these nucleotide triples are called codons. Any possible codon maps to one of 20 particular amino acids, or a stop codon which signals the translation process to terminate. This means that the number of nucleotides in

[1] It should be noted the distribution of these potential BCRs is not uniform due to a variety of reasons such as biases in the gene selection process during recombination.

a DNA sequence must be a multiple of three for the sequence to be translated into a valid protein. In the context of V(D)J recombination, we call a rearranged V(D)J sequence non-functional if it results in a DNA sequence of length not divisible by three, or if it contains a premature stop codon in the middle of the sequence before the "true" designated stopping point is reached.[2]

Even after a pair of successful V(D)J rearrangement events, a cell will be destroyed if it fails to pass certain tests regarding its receptor's binding abilities. For example, during a process called thymic selection, T cells with receptors that are not able to bind to an important, related receptor molecule known as the major histocompatibility complex (MHC), or that bind to proteins in the body that they should not bind to, are signaled for destruction. B cells undergo similar selective steps during their development within bone marrow. This exacting set of trials provides our immune system with functional B and T cell receptors that are well-suited to safely fight off infections.

1.3 T cell receptors

T cells are often called the "quarterback" of the immune system as they oversee and orchestrate the immune response as a whole. There are several types of T cells that perform distinct roles, all of which help coordinate a quick and effective response against an infection. They interact with other cells mostly through their receptors, making the TCR an object of substantial scientific interest.

The main function of the TCR is to interact with two types of MHC molecules along with the peptides bound within these MHC molecules (this combination of an MHC with its bound peptide is also known as a peptide-MHC complex, or pMHC). Class I MHC molecules, which are expressed on the surfaces of most cells, present antigen peptides to killer T cells.

[2]Because of this criterion, we roughly expect only 1 in 3 rearrangement events to produce a functional protein. We need two valid rearrangement events to synthesize the full receptor which is made up of two protein chains. The details of how the body resolves this are nuanced, but the short story is that a failure to pair chains can result in a separate rearrangement event using the unused chromosome within the cell, whose failure results in destruction of the cell. For a B or T cell, V(D)J recombination is the gamble of a lifetime!

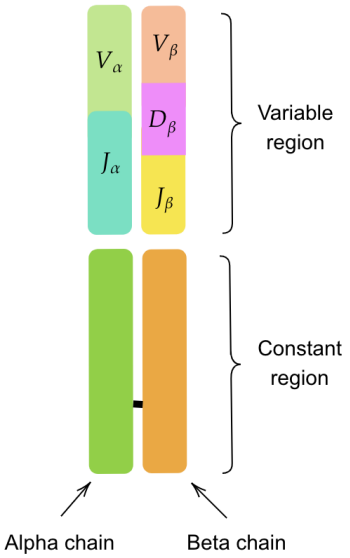
If the TCR can "recognize" the antigen peptide (i.e., if the TCR can bind to the pMHC complex), the T cell realizes there is a problem within the cell and signals it to be destroyed. Class II MHC molecules, which are expressed only by an exclusive group of cells called "antigen-presenting cells", present antigen peptides to helper T cells. If the TCR is able to bind to the pMHC in this case, the T cell is "activated", causing it to proliferate to yield many T cells with the same receptors, and allowing the coordination of further actions from the immune system to respond to this antigen. We call such a proliferation *clonal expansion*, a concept not only an important in the theory of immunology, but also, as we will see, in the methodology of TCR repertoire sequencing and analysis.

Each T cell expresses a unique TCR as a result of two particular V(D)J recombination events (although different cells can express common TCRs by chance). Almost all TCRs consist of an α chain and β chain, which are respectively encoded by the TRA (or TCRα) and TRB (or TCRβ) genes (Figure 1.2a). The remaining TCRs consist of γ and δ chains, respectively encoded by the TRG (TCRγ) and TRD (TCRδ) genes. These loci can be sequenced using modern technology, yielding the full V(D)J sequence corresponding to the variable domain of the receptor.

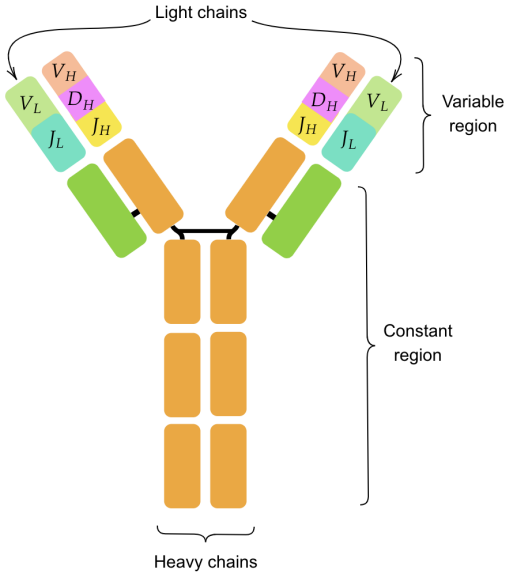
1.4 B cell receptors and antibodies

B cells are very similar to T cells in that they undergo V(D)J recombination followed by steps of selective filtering. Further, they both contain the same framework regions and CDRs, and exist to bind objects within the body. Despite this homology, BCRs exhibit several differences from TCRs with regard to structure and immunological function. For example, B cells mostly exist in order to bind antigen, sometimes extraordinarily well, and do not directly interact with other proteins like MHC.

Like helper T cells, B cells are activated if their receptor binds to an antigen, producing many more B cells than can bind to this antigen; this is another example of clonal expansion as previously discussed. B cells whose BCRs can bind to a given antigen are sent to a special area in the body called a *germinal center* to further optimize their binding capabilities



(a) The structure of a T cell receptor.



(b) The structure of a B cell receptor.

Figure 1.2: Cartoon depicting the structures of a T cell receptor (top) and B cell receptor (bottom). Both receptors include a variable region and a constant region. The variable regions both result from V(D)J recombination, although the V, D, and J gene sets differ between the receptors.

(discussed further in the next section). This results in a plethora of BCRs highly tailored to the antigen in question, leading to the neutralization of the infection in its entirety. The body recognizes that these highly-affine BCRs are valuable in case a similar infection were to arise in the future, and so these BCRs are "saved" via memory cells which can be readily invoked without the need to undergo this optimization process again.

Unlike T cells, whose TCRs can only be expressed on the cell's surface, certain B cells (called *plasma cells*) are able to secrete their receptors into the bloodstream; these secreted BCRs are what we call *antibodies* (Abs). Antibodies circulate throughout the bloodstream aiming to bind to foreign and unwanted objects, and tagging any such objects for an immune response by T cells and other immunological agents. Antibodies are also known as Immunoglobulins (Ig), and are characterized by their iconic Y-shape (Figure 1.2b).

Like T cells, each B cell expresses a unique BCR as a result of two particular V(D)J recombination events. All BCRs consist of two identical heavy chains and two identical light chains, where the heavy chain and light chain are analogous to the β and α chains in TCRs, respectively (the similarities are readily seen in Figure 1.2). The heavy chain is encoded by the IgH gene locus. There are two types of light chains: the kappa (κ) chain, encoded by the IgK gene locus and typically comprising about 67% of light chains, and the lambda λ chain, encoded by the IgL locus and comprising the remaining 33% of light chains. Like the loci for TCR chains, these loci can also be sequenced using modern technology.

1.5 Somatic hypermutation and affinity maturation

When a BCR is able to bind to an antigen, its B cell is taken to a germinal center within the lymphatic system. It starts in the "dark zone" of the germinal center, where it is stimulated to reproduce through multiple rounds of cell division. During this reproduction, a special process called *somatic hypermutation (SHM)* induces mutations via an enzyme called *activation-induced cytidine deaminase (AID)* specific to the germinal center, and yields a much higher mutation rate than observed in usual cell division. These mutated cells then travel to the "light zone", where they comptete with each other to bind to limited amounts

of the cognate antigen. The cells with the highest binding affinity are positively selected to survive, whereas the other cells are either sent back to the dark zone, or are destroyed. This Darwinian process continues for multiple rounds, ultimately generating BCRs with a very high affinity to the antigen. We refer to this overall process as *affinity maturation*, and it explains how the body is able to produce extremely effective antibodies to target previously unseen antigens. Affinity maturation is illustrated in the bottom section of Figure 1.1.

Note that affinity maturation occurs only for B cells, though helper T cells are involved in orchestrating affinity maturation within the germinal centers. As mentioned above, affinity maturation adds an extra layer of diversity to the possible set of BCR sequences within an individual.

1.6 Adaptive immune receptor repertoire sequence analysis

As we have seen, immunologists have uncovered quite a bit about how the adaptive immune system works. However, there are still a lot of important questions yet to be answered within the field of computational adaptive immunology. For example, given an arbitrary TCR or BCR, we cannot yet characterize its complete binding profile with respect to an arbitrary antigen based on sequence alone. This would be revolutionary for many applications like vaccine design and cancer immunotherapy as it would lead to faster and cheaper discoveries that do not rely on expensive and time-consuming experiments. Computational immunologists are thus blessed with a hearty set of questions that can comprise a lifetime of research. Between the importance of the field and potential for new research directions, there is active scientific interest in advancing the standards and technology of computational adaptive immunology.

AIRR-seq datasets provide snapshots of an individual's adaptive immune state and to some extent dictate how we construct our statistical and computational methods. These datasets can range from unprocessed sequence reads that span a subset or superset of the full variable region, to fully inferred and annotated sequence records. Inferring annotations involves handling dynamic properties that these sequences exhibit such as variable CDR3

length, as well as more stable properties such as the conserved amino acid positions that unambiguously define the CDR3. Nonetheless, efforts to standardize the nomenclature and definitions with regard to immune receptor data have greatly facilitated AIRR-seq analysis [41, 78].

High-throughput sequencing is not perfect, and we expect there to be some level of sequencing error present in any given dataset. Computational immunologists will sometimes attempt to model the generative process of these errors and incorporate this into their models, but often we simply view these errors as a nuisance that must be acknowledged but not explicitly dealt with. How this impacts the analysis is context-dependent and will determine the type of analysis that is most appropriate, but as is the case for most scientific applications where the data is subject to noise, this caveat should always be kept in mind when interpreting results.

1.7 Discussion

In summary, BCRs and TCRs are chiefly responsible for the recognition of and response to antigens. These receptors arise through a complex but incredibly effective stochastic process which gives our immune system the ability to address virtually any antigen we encounter. Immunologists can obtain samples of BCR and TCR repertoires to describe and make inferences about an individual's underlying immune system dynamics, such as in the context of an immune response. While these sample repertoires vary in their level of polish, the community has developed consistent standards and pipelines for their analysis for many typical research scenarios. Computational immunologists can thus enjoy a wealth of data ripe for analysis and scientific discovery.

Chapter 2

LACK OF EVIDENCE FOR A SUBSTANTIAL RATE OF TEMPLATED MUTAGENESIS IN B CELL DIVERSIFICATION

2.1 Introduction

Our immune systems generate a highly diverse set of antibodies to protect us from pathogens. An important part of this process is affinity maturation, which generates high-affinity antibodies for antigens encountered by the immune system. Affinity maturation is the result of multiple rounds of mutation and selection: mutations are introduced into the rearranged antibody gene by enzymatic processes, and mutations leading to higher-affinity antibodies are selected.

Two major processes are believed to underlie the mutation processes in B cells: classical somatic hypermutation (SHM) and gene conversion (GCV). Both processes depend on activation-induced cytidine deaminase (AID) [46], which creates U:G lesions in the DNA by deaminating deoxycytidine to deoxyuridine. In SHM, the lesion is resolved by recruiting error-prone repair machinery which can introduce non-templated point mutations at and around the AID-induced lesion. In GCV, the lesion is repaired using a homologous segment elsewhere in the genome as a donor template, resulting in the homologous tract being copied into the rearranged antibody gene.

In a recent paper, Dale *et al.* [17] propose that new mutation process, called "templated mutagenesis," is also an important contributor to B cell receptor diversification in mice and humans. In this process, an incompletely-understood mechanism uses the sequence of other germline genes to guide the mutation process at a rearranged germline gene. One candidate mechanism for this process is gene conversion. However, templated mutagenesis differs from previous descriptions of gene conversion (in species such as chicken) in that it does not

require long stretches of homology between donor and recipient sequences. Indeed, Dale *et al.* find that templated mutagenesis "extends into the somatically mutated non-Ig sequences, LAIR1, *gpt*, and β-globin, despite the lack of overt homology between these genes and the IgHV repertoire." For this paper we will simply refer to this newly-hypothesized process as templated mutagenesis.

Although much of the evidence presented by Dale *et al.* was in the form of statistically significant deviations from a simplified null model, the authors suggest that ∼50-65% of mutations in IgH from a collection of human and murine data sets are consistent with templated mutagenesis. If over half of mutations truly come from templated mutagenesis, B cell repertoire analysis methods will need to be rebuilt. For example, to estimate the likelihood of a group of mutations that match a template elsewhere in the Ig locus, one must incorporate the respective likelihoods that the group occurred from classical SHM or from templated mutagenesis. Therefore, any method that relies on estimating mutation probabilities would need to be updated. This includes all core methods for B cell sequence analysis: germline annotation, lineage tree estimation, selection strength estimation, and validation techniques such as repertoire simulation. In light of this dependence, accurate rate estimates of templated mutagenesis are crucial.

In this paper, we show that data from human samples [10] and data from a transgenic mouse model [85] analyzed by Dale *et al.* do not support a high rate of templated mutagenesis. We do so by re-implementing the software described in Dale *et al.*, called PolyMotifFinder, which identifies potential templated mutagenesis events by comparing mutated sequences to a pool of potential donor genes. Using this software, we run a control experiment absent from the original analysis: we calculate the rate of templated mutagenesis using a donor set of simulated genes *not* present in the organism from which the mutated sequences derived. In this way, we show that the PolyMotifFinder strategy for detecting templated mutagenesis via microhomology has a false positive rate very close to, and in some cases above, the reported positive rate. This yields an upper bound on the range of the true templated mutagenesis rate; this range is often zero. We also describe how Dale *et al.* conflate a non-trivial rate of

templated mutagenesis with significance estimates for a simplified null model. In addition, we argue that clustering of mutations is compatible with the classical Neuberger model of SHM and thus evidence of such clustering is not *prima facie* evidence of templated mutagenesis. We conclude that more evidence is needed if templated mutagenesis should be accepted as an important part of BCR diversification.

2.2 Materials and methods

Because the original software in Dale *et al.* was not made available as part of the publication, nor was it available upon personal request without a materials transfer agreement restricting its use, we re-implemented the algorithms described in Dale *et al.* as open-source software in Python, a widely available and free programming language.

The PolyMotifFinder algorithm relies on the creation of two matrices. Given a set of n mutated sequences all deriving from the same germline sequence, the length of which is p, and a window size k corresponding to the minimum allowable donor tract length for templated mutagenesis, we create matrices M and S, each having n rows and p columns. $S_{ij} = 1$ if (i) there is a mutation at position j in sequence i and (ii) there exists a window of size k around position j in the ith sequence that contains at least two mutations and is represented in the donor set. That is, S_{ij} is an indicator of position j in sequence i belonging to a pair of mutations consistent with templated mutagenesis. For M, we take $M_{ij} = 1$ if (i) there is a mutation at position j in sequence i, (ii) there is at least one other mutation in sequence i whose distance from j is $k - 1$ or less, and (iii) that mutation is not part of a pair of mutations that was seen in one of the previous sequences. In other words, M identifies the set of unique mutations within a length-k window from other mutations. In Dale *et al.*, S is referred to as the scoring matrix and M as the mutation matrix. The templated mutagenesis coverage is then computed as $\sum_{i=1}^{n}\sum_{j=1}^{p} M_{ij}S_{ij} / \sum_{i=1}^{n}\sum_{j=1}^{p} M_{ij}$.

We note in passing that calculation of the templated mutagenesis coverage depends on the order in which the sequences are processed. As an example, consider the very abbreviated case where the germline sequence is AAA, the donor set is the single sequence CT (so that

$k = 2$), and the mutated sequences are ATT and CTT. If the sequences are processed as ATT followed by CTT, we will have

$$S = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}, \quad M = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

Then $\sum_{i,j} M_{ij} S_{ij} = 1$, $\sum_{i,j} M_{ij} = 3$, for a templated mutagenesis coverage of 1/3.

If the sequences are processed in the opposite order, CTT followed by ATT, we have

$$S = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad M = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

Then $\sum_{i,j} M_{ij} S_{ij} = 2$, $\sum_{i,j} M_{ij} = 3$, for a templated mutagenesis coverage of 2/3. Nevertheless, we re-implemented this order-dependent procedure.

PyMotifFinder, our package including a re-implementation of PolyMotifFinder, is available on GitHub (https://github.com/matsengrp/PyMotifFinder), and our analysis scripts are available at https://github.com/matsengrp/TemplatedMutagenesis-1. Our implementation takes as input pairs of naive and mutated sequences along with a donor gene set, the set of potential templates for templated mutagenesis. It then computes a templated mutagenesis coverage according to the strategy defined above. Our implementation contains unit tests to verify the accuracy of the algorithm. The sequence data used in our analyses are available on Zenodo (https://doi.org/10.5281/zenodo.3572361). The complete analysis starting from preprocessed data, including generating the figures and tables included in this article, can be reproduced by copy-pasting a handful of commands into a provided Docker container [7] as described in the GitHub repository. All preprocessing scripts are included with the data on Zenodo.

2.2.1 Sequence data sets

We analyzed three sets of mutated sequences: one from human subjects, and two from a transgenic mouse model. The first set of sequences, described in [10], corresponds to antibodies to the membrane-anchored Ebola virus glycoprotein trimer. They were collected from

the peripheral B cells of a convalescent donor who survived the 2014 Ebola Zaire outbreak, and will be referred to as the anti-Ebola sequences. The sequences were downloaded from GenBank using the accession numbers corresponding to the heavy-chain sequences (taken from the supplemental material of [10]), and both the accession numbers and sequences used are available on Zenodo.

The second and third sets of mutated sequences come from a transgenic mouse model described in [85] which was also investigated by Dale *et al.*. Briefly, these sequences come from the B cells of mice that have been genetically engineered with a modified heavy-chain locus. One chromosome, with the "productive" allele, contains the pre-rearranged V region of the 4-hydroxy-3-nitrophenylacetyl (NP)-binding B1-8 antibody (VB1-8). The other chromosome, with the "passenger" allele, contains a sequence consisting of a VB1-8 promoter and leader, followed by a stop codon, followed by either the *E. coli gpt* gene or another copy of VB1-8. The sequences on both alleles accumulate mutations by somatic hypermutation following the immunization of the mice with NP-chicken gamma globulin. Since the sequences on the passenger allele cannot be expressed, the SHM patterns on these sequences are unaffected by natural selection, making the system particularly useful for studying SHM. The analyses presented here use only the passenger *gpt* or VB1-8 sequences from this system. The sequences come from B cells collected from either the Peyer's patches or the spleens of vaccinated mice (six samples from each), and we included sequences taken from both tissue types in our analysis. These sequences will be referred to as the *gpt* sequences and the VB1-8 sequences, respectively.

The *gpt* and VB1-8 sequences were downloaded from the Sequence Read Archive (SRP061422). pRESTO [77] was used to assemble the raw paired-end reads, filter reads to those with an average quality score of at least 20, remove PhiX contamination, and filter to sequences that were seen at least twice. The script used for this process is available with the data on Zenodo.

2.2.2 Donor gene sets

For each mutation tract, PolyMotifFinder looks for templated mutagenesis from a provided donor gene set, which contains potential templated mutagenesis donors: if a mutation tract in a mature sequence matches exactly to a region in the donor gene set, the mutation is explainable by templated mutagenesis from that donor set. We prepared five donor gene sets: a set of human IGHV genes, two sets of mouse IGHV genes, and two "mock" sets containing simulated genes homologous to the *E. coli gpt* gene. Because these simulated *gpt* homologs are not present in mouse, we use them as controls as described below. We refer to these sets as the human IGHV gene set, the mouse IMGT IGHV gene set, the mouse 129S1 IGHV gene set, and the mock *gpt* gene sets.

The human IGHV donor set was used to obtain the PyMotifFinder (PyMF) rate estimate of templated mutagenesis in the anti-Ebola sequences. This set consists of human IGHV gene segments downloaded from IMGT (http://www.imgt.org/vquest/refseqh.html) in September 2017. The IMGT label for this set was "F+ORF+all P," corresponding to all functional genes, all open reading frames, and all pseudogene alleles, yielding 466 total segments. A file containing these segments is available on Zenodo.

The mouse IMGT IGHV gene set was used to obtain the PyMF rate estimate of templated mutagenesis in the VB1-8 and *gpt* sequences. This set consists of mouse IGHV gene segments downloaded from IMGT (http://www.imgt.org/vquest/refseqh.html) in September 2017. The IMGT label for this set was "F+ORF+all P," corresponding to all functional genes, all open reading frames, and all pseudogene alleles, yielding 499 total segments. A file containing these segments is available on Zenodo.

The mouse 129S1 IGHV donor set was used primarily to describe what the mutation spectrum would look like under a templated mutagenesis model in the transgenic mouse system described in [85]. While these mice belonged to strain 129P2, the 129P2 heavy-chain locus has not been sequenced; instead, we used the set of IGHV genes present in the closely related strain 129S1, published in [66]. We note that the gene set is incomplete, with only

the 3' half of the IGHV locus sequenced. Since this gene set was used primarily to describe the likelihood of mutations in a model of templated mutagenesis and not to get at the rate of templated mutagenesis, we decided that a partial set of genes that match closely those found in in the actual system was an appropriate choice. In particular, we believe it is better than the alternative of using the mouse IGHV genes taken from IMGT, which include several times as many genes as are present the 129P2 genome. The sequenced and annotated region of the 129S1 genome was downloaded from GenBank (https://www.ncbi.nlm.nih.gov/nuccore/126349412). The V gene segments were extracted using a custom Python script available on Zenodo.

The two mock *gpt* donor sets were used to estimate the false positive rate of the PolyMotifFinder strategy with the human IMGT donor set and the false positive rate of the PolyMotifFinder strategy with the mouse IMGT donor set. For a mock donor set to give a good estimate of the false positive rate of the PolyMF strategy, the mock set should have approximately the same homology structure as the donor set used by PolyMF. We created one such set for the human IMGT IGHV donor set and one for the mouse IMGT IGHV donor set. In each case, we aligned the sequences in the gene set using MUSCLE version 3.8.31 [22] and inferred a phylogenetic tree on the sequences using FastTree version 2.1.7 [62]. We then used pyvolve [74] to simulate a new set of sequences from the estimated phylogenetic tree. In each case, the *gpt* sequence was the root, and the sequences were simulated according to a continuous-time Markov process along the estimated phylogeny. We used the GY94 codon model [28] with parameters $\alpha = .98$, $\beta = .65$.

All of the donor sets are available on Zenodo, along with the scripts used to extract the IGHV gene segments from the 129S1 genome, the scripts to align and create trees from the human IGHV genes and *gpt* genes, and the script to create the two mock *gpt* donor sets.

2.2.3 Germline annotation and mutation calling

To use PyMF on the anti-Ebola, VB1-8, and *gpt* sequences, we needed to identify the mutations and the naive sequences. For the anti-Ebola and VB1-8 sequences, germline sequences

and mutations were identified using partis version 0.13.0 [64] with default germline V, D, and J gene sets. These sets comprise curated subsets of the germline genes in IMGT: excluded are genes that are biologically implausible (e.g. on the wrong chromosome, non-functional, lacking the conserved cysteine) or otherwise considered inaccurate [79].

For the *gpt* sequences, partis was run using a modified set of germline genes. The reference sequence for the passenger *gpt* gene (obtained via personal communication with Dr. Leng-Siew Yeap) was

```
CTTTCTCTCCACAGGTGTCCACTCCCAGGTCCAACTGTAGTAGATGAGCGAAAAATACATCGTCACCTGGGACAT
GTTGCAGATCCATGCACGTAAACTCGCAAGCCGACTGATGCCTTCTGAACAATGGAAAGGCATTATTGCCGTAAG
CCGTGGCGGTCTGGTACCGGGTGCGTTACTGGCGCGTGAACTGGGTATTCGTCATGTCGATACCGTTTGTATTTC
CAGCTACGATCACGACAACCAGCGCGAGCTTAAAGTGCTGAAACGCGCAGAAGGCGATGGCGAAGGCTTCATCGT
TATTGATGACCTGGTGGATACCGGTGGTACTGCGGTTGCGATTCGTGAAATCTGCAGTGACGCGCCCACTCTCAC
AGTCTCCTCAGGTGAGTCCTTACAACCTCTCTCTT
```

In the reference sequence, positions 44 through 351 correspond to the first 308 nucleotides of the *gpt* gene and the remainder are linkers. To align and call mutations from the germline sequence, partis requires a set of germline V, D, and J gene segments. For the V gene segment, we used the first 308 nucleotides of the *gpt* gene, i.e., the portion of the *gpt* gene inserted into the mouse germline. For the D and J gene segments, we used arbitrary "fake" gene segments: `AAAAAAAAAA` for the D gene segment and `GGGGGGGGGG` for the J gene segment. We appended these segments to the end of each sequence so that each input sequence to partis ended with `AAAAAAAAAAGGGGGGGGGG`. When used this way, partis aligns the *gpt* portion of the sequence to the portion of the *gpt* gene included in the reference, aligns the added suffix to the fake D and J gene segments, and treats the linker region following the end of the *gpt* gene as a VD insertion. We validated the results by checking that each inferred "V" sequence length was correct, that the inferred mutation rate in the "V" gene region was not too high, that the inferred VD insertion lengths were correct, and that the VD insertion sequences corresponded to the linker portion of the naive *gpt* sequence (positions 352 through

410 in the sequence above). Therefore, all mutations identified by partis are located on the *gpt* portion of the sequence with none in the linker sequence. This is the desired behavior for our analysis because we are looking for regions of microhomology in the *gpt* sequence and we do not wish to analyze mutations that occurred in the linkers.

2.2.4 Model for mutation probabilities due to templated mutagenesis

To investigate whether templated mutagenesis could explain the observed mutations, we constructed a simple statistical model for templated mutagenesis. The model assumes a uniform probability distribution over the possible templated mutagenesis donors, so that each donor is equally likely to have provided the template for a given templated mutation event. For each mutated site we identified the three possible mutations: the mutation that actually occurred and the two mutations that did not occur. For each of the three mutations, we identified all ways of aligning a donor gene to a mutation-containing region so that the two match in at least k bases around the mutation. We defined this set as the set of potential templated mutagenesis donors, and we modeled mutation due to templated mutagenesis as a uniform draw from this set of donors. In this model, the probability of seeing the observed mutation from a templated mutagenesis event is the number of donors containing the observed mutation divided by the total number of donors. That is,

$$P(\text{observed mutation} \mid \text{templated mutation event}) = \frac{n_{\text{obs}}}{n_{\text{obs}} + n_{\text{unobs}}} \tag{2.1}$$

where n_{obs} is the number of donors matching the observed mutation and n_{unobs} is the number of donors matching one of the two possible unobserved mutations. We can compute these probabilities for any donor gene set to evaluate how well it explains the observed pattern of mutations.

2.2.5 Upper bound on the rate of templated mutagenesis

Given a bound on the false positive rate of PyMF and an estimate by PyMF of the rate of templated mutagenesis, we can compute an upper bound on the true rate of templated

mutagenesis. To do this, we use a simple mixture model in which we assume that mutations arise either by templated mutagenesis or by classical SHM. We first define the following quantities, all of which take values in $[0, 1]$:

1. The false positive rate, FPR, is the probability that PyMF classifies a mutation due to classical SHM as explainable by templated mutagenesis.

2. The true positive rate, TPR, is the probability that PyMF classifies a mutation due to templated mutagenesis as explainable by templated mutagenesis.

3. The positive rate, PR, is the overall probability that PyMF classifies a mutation as explainable by templated mutagenesis.

4. p_{shm} is the true proportion of classical SHM events. Since a mutation can be due to either SHM or templated mutagenesis but not both, $1 - p_{\text{shm}}$ is the true proportion of templated mutagenesis events.

Then the overall probability that PyMF classifies a mutation as explainable by templated mutagenesis is

$$\text{PR} = p_{\text{shm}} \times \text{FPR} + (1 - p_{\text{shm}}) \times \text{TPR}.$$

If we assume that TPR $= 1$ so that all true templated mutagenesis events are correctly classified by PyMF as due to templated mutagenesis (i.e., PyMF has sensitivity 1) and that FPR $\geq b$ for some lower bound b, we can rearrange the expression above to conclude that $p_{\text{shm}} \geq \frac{1-\text{PR}}{1-b}$. Equivalently, the rate of templated mutagenesis would be at most $1 - \frac{1-\text{PR}}{1-b}$. If we do not specify a true positive rate, the upper bound for the rate of templated mutagenesis becomes

$$1 - \frac{\text{TPR} - \text{PR}}{\text{TPR} - \text{FPR}}, \tag{2.2}$$

assuming that TPR $>$ FPR. If TPR $<$ FPR, we cannot obtain an upper bound. We apply (2.2) to obtain upper bounds on the rate of templated mutagenesis in mice and humans.

2.2.6 Hypothesis testing and confidence intervals

The *gpt* sequences have a grouped structure: each mutated sequence comes from one of 12 tissue samples from 6 different organisms, and so it is inappropriate to model them as independent and identically distributed. Hypothesis testing and confidence interval construction for the *gpt* sequences was therefore performed using a mixed effects model fit with the `lme4` package [3] in R [63]. For each value of k (the minimum donor tract length) and each reference sequence, we modeled the probability of a mutation given templated mutagenesis using a mixed model with a random effect for tissue sample. Confidence intervals were plotted as the fitted value in the mixed model plus or minus two standard errors. To test whether templating from the IGHV genes could explain the observed mutations better than templating from the *gpt* genes, we computed the probability of each mutation under the model of templated mutagenesis by IGHV genes and templated mutagenesis by *gpt* genes. We then computed the difference between the probability of the mutation under the IGHV templating model and the *gpt* templating model. Under the null hypothesis that the two models are equally good at explaining the observed mutations, these differences should have mean zero. As before, since the mutations have a grouped structure, we tested this null hypothesis using a mixed model with a random effect for tissue sample.

2.3 Results

2.3.1 PyMotifFinder identifies a high fraction of mutations as explainable by templated mutagenesis

To verify that our re-implementation of PolyMotifFinder was comparable to the version presented by Dale *et al.*, we ran PyMF on the *gpt* and VB1-8 sequences described in [85] with the mouse IGHV gene set, as well as the anti-Ebola sequences described in [10] with the human IGHV gene donor set. We found slightly higher but comparable rates of mutations explainable by templated mutagenesis in the *gpt* sequences: of the mutations within 8 nucleotides of each other, 60-75% had 8-mer templates in the mouse V genes (Figure 2.1), consistent with

the initial report. We found a higher rate of mutations explainable by templated mutagenesis in the anti-Ebola and VB1-8 sequences: of the mutations within 8 nucleotides of each other, 73% of the anti-Ebola sequences had 8-mer templates in the human V genes and 75% of the VB1-8 sequences had 8-mer templates in the mouse V genes. The discrepancy is likely due to differences in gene filtering, germline annotation, and mutation calling between our respective pipelines. Indeed, when we perform stricter filtering to our donor gene sets by removing open reading frame and pseudogene sequences, we obtain rates estimates of 42% for humans and 62% for mice, which are very close to the Dale *et al.* estimates. We discuss this in detail in Section "Consistent results using filtered donor gene sets".

2.3.2 gpt *sequences can be used to estimate the PolyMotifFinder false positive rate*

To investigate the false positive rate of the PolyMotifFinder strategy, we ran PyMF on the set of somatically mutated *gpt* sequences described in the "Sequence data sets" section using a mock donor set of simulated *gpt* homologs that are not present in B cells. Since mutations could not have arisen from copying over templates from our mock set, any inference of templated mutagenesis events identified by the method must be a false positive. For the rate of false positives provided by the mock donor gene set to provide a good estimate of the true false positive rate, the mock donor gene set should be constructed so that the probability that an SHM-induced mutation in a *gpt* sequence matches a member of the mock donor gene set is close to the probability that an SHM-induced mutation in a real antibody sequence matches one of the IGHV genes. For this to hold, the distribution of molecular divergences among the genes in the mock donor set should match the distribution of divergences among the real donor gene set.

Our mock donor gene sets were created to have these properties. To verify this, we estimated phylogenies for the two mock donor gene sets and the two IMGT IGHV gene sets and computed the divergences between the roots and the leaves in each. The divergence distribution in the mock *gpt* set based on the mouse IMGT gene set resembled the divergence distribution in the mouse IMGT gene set (mean divergence .48 and .48, respectively). The

same holds with the mock *gpt* set based on the human IMGT gene set and the human IMGT gene set (mean divergence .41 and .4, respectively). For a more complete description of the divergences, five-number summaries of the divergences in each of the four gene sets are given in Supplemental Table I.

Finally, we note that in the *gpt* system, we expect all of the mutations to be introduced by SHM. Although Dale *et al.* analyze these sequences and suggest that templated mutagenesis could be occurring, the lack of homology between the *gpt* gene and the V genes makes it *a priori* unlikely that these mutations are introduced by templated mutagenesis. We address the potential contribution from small micro-homologies (matches of fewer than 10 bases) between the *gpt* gene and the IGHV genes due to chance sequence similarity in Section "Evidence that mutations in *gpt* sequences are not due to templating from V genes".

2.3.3 gpt *analysis demonstrates that the MotifFinder methodology has a high false positive rate*

To estimate the false positive rate of PyMF with the human IMGT IGHV gene set and the false positive rate of PyMF with the mouse IMGT IGHV gene set, we ran the algorithm on the *gpt* sequences with the corresponding mock *gpt* donor gene set. Since the donor set of simulated *gpt* homologs is not present in the mouse, they cannot have been used as templated mutagenesis donors, and any mutation PyMF identifies as explainable by templated mutagenesis from this set is a false positive. We ran PyMF with minimum donor tract length ranging from 8 to 14, and we found false positive rates that were on the same order as the PolyMF rates obtained when mutated sequences were run against real donor sets. The absolute false positive rates were particularly high for minimum donor tracts of 8 and 9 (Figure 2.1). The average false positive rate was 83% for a donor tract of 8, 50% for donor tracts of size 9, and 25% for donor tracts of size 10. This rate falls dramatically as the minimum donor tract size increases, dropping to 5% for donor tracts of size 14. This suggests that the PolyMotifFinder strategy has a large false positive rate for small values of k, classifying more than 50% of mutations as explainable by templated mutagenesis from genes that were

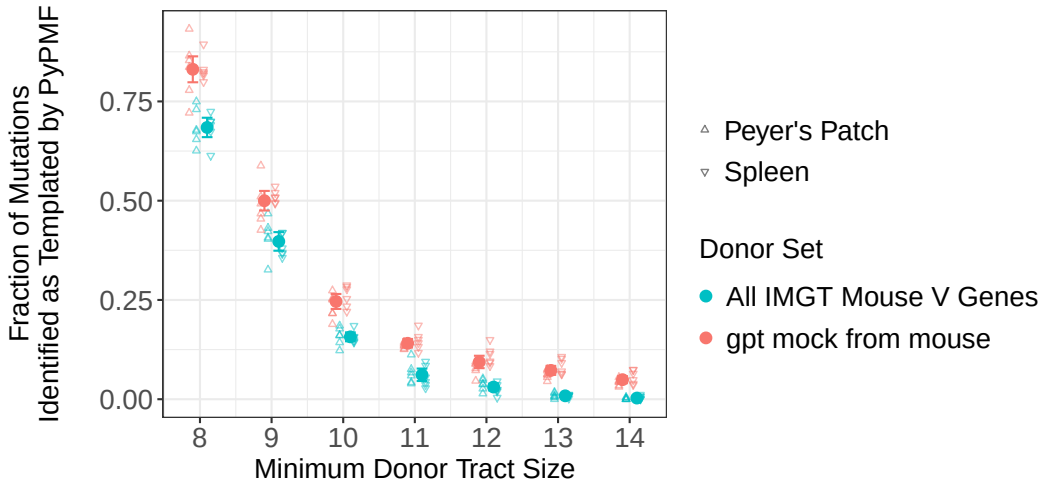


Figure 2.1: Red points represent the fraction of mutations in the *gpt* gene explainable by templates in the mock donor set of simulated *gpt* homologs, which are not present in the mouse germline and are not available as templates for templated mutagenesis (i.e., the FPR). Blue points represent the fraction of mutations in the *gpt* gene explainable by templates in the mouse IMGT IGHV gene donor set, which are potentially present in the mouse germline and available as templates. For each tract length and donor set, the filled circle and error bar represents the overall estimate of the probability of a mutation being explainable by templated mutagenesis plus or minus two standard errors. We see that the FPR of PyMF is larger on average than the PyMF estimate of the fraction of mutations explainable by templated mutagenesis. Points corresponding to samples from Peyer's patches and spleen are offset slightly to the left and right, respectively, to facilitate comparison and to avoid overplotting. This analysis was performed once on data from six individual mice, with two replicates per mouse corresponding to samples from Peyer's patches and spleen, yielding 12 total samples.

not present in the mouse when $k = 8$ or $k = 9$.

2.3.4 Evidence that mutations in gpt *sequences are not due to templating from V genes*

One might explain the high false positive rate of PolyMotifFinder by saying that in spite of the overall lack of homology between *gpt* genes and V genes, the mutations in the *gpt* sequence were actually introduced by templating from very small homologous tracts in the mouse V genes. To check this possibility, we ran PyMF on the *gpt* sequences with the mouse IMGT IGHV genes as a donor set. We found that many of the mutations could be explained by templating from very small homologous tracts in the mouse V genes, corresponding to the findings of Dale *et al.*. For example, nearly 60% of the mutations had a template of size 8 in the V gene set and about 40% of the mutations had a template of size 9. However, the percentage approaches zero as template size increases, and for every template size the average proportion of mutations explainable by templating from *gpt* sequences is higher than the average proportion of mutations explainable by templating from V genes (Figure 2.1). In fact, the proportion of mutations explainable by templating from the V genes is very close to zero for templating by tracts of size 11 or greater, while the proportion explainable by templating from the mock *gpt* donor set remains non-neglibible.

We next formally evaluated the plausibility that the mutations in the *gpt* sequences were introduced by templated mutagenesis from the IGHV genes present in the mouse. To do so, we computed the probabilities of the *gpt* mutations via templated mutagenesis from 129S1 IGHV donor gene set and the probabilities of the *gpt* mutations via templated mutagenesis from the mock donor set of simulated *gpt* homologs, using the uniform-across-donors probability model specified by Equation (2.1) for both cases. These models encode our intuition that if the mutations really were templated from a donor gene set, the observed mutation spectrum should be biased towards bases that are represented more frequently in potential donors from that gene set. As an example, suppose that we are considering one mutation in the *gpt* sequence from A to T. If all of the potential templated mutagenesis donors in the V gene set would lead to a mutation from A to T and all of the templated

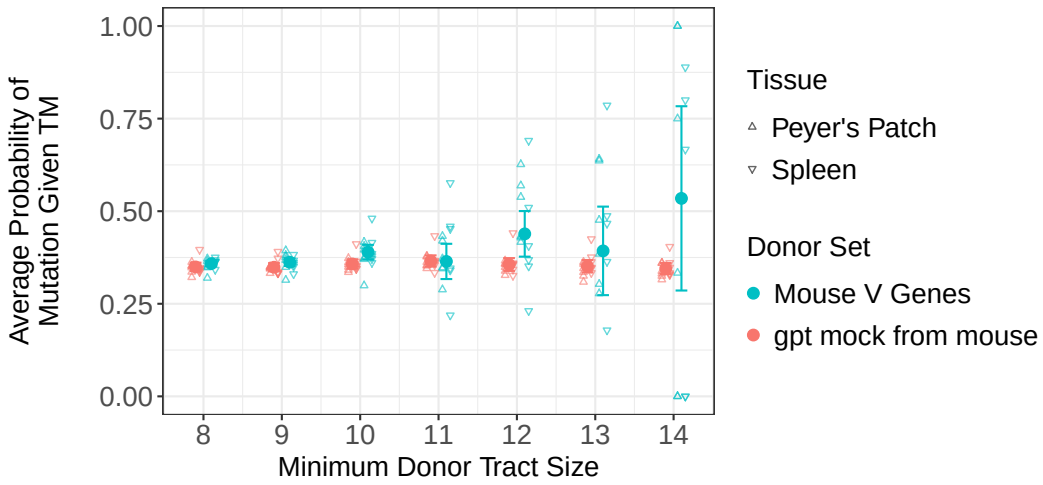


Figure 2.2: Average probability of the observed mutations under a templated mutagenesis model, either templating from *gpt* genes or templating from the set of 129S1 V genes. Each point corresponds to one sample taken from either spleen or Peyer's patches, so that the average is computed over all sequences in a given sample. This analysis was performed once on data from six individual mice, with two replicates per mouse corresponding to samples from Peyer's patches and spleen, yielding 12 total samples.

mutagenesis donors in the *gpt* gene set would lead to a mutation from A to C, the observed A to T mutation is explained better by templating from the V genes than by templating from the *gpt* genes. We can fit one model using donors from the mouse V genes and another model using donors from the *gpt* genes and compare how well each model explains the data: if templated mutagenesis from mouse V genes were really occurring, we would expect the V gene model to fit the data better than the *gpt* model. If this is not true, it suggests that both the V gene and the *gpt* inferences are spurious, as the *gpt* donor genes are not actually present in the mouse.

For each sample, we computed the average probability of the mutations in the *gpt* sequences given templated mutagenesis from the mouse IGHV gene donor set and the *gpt* donor set for tract sizes ranging from 8 to 14. We found that these numbers were comparable for the *gpt* donor set and the mouse IGHV gene donor set, as shown in Figure 2.2. As described in the Methods section, we used a mixed effects model to test for a difference in the expected probabilities of mutation due to templating from the *gpt* donor set and the mouse IGHV gene donor set. The resulting p-values were: $p = .059, .054, .115, .202, .001, .213$, and $.249$ for $k = 8$ through 14, respectively. This indicates that the mutations in the *gpt* sequences do not tend to look any more like the mouse IGHV gene donor set than they do like the *gpt* donor set. Because the *gpt* donor set was not present in the mouse, we believe that it is unlikely that the mutations in the *gpt* sequences were introduced by templating from the mouse IGHV genes.

To further investigate whether the mutations could have arisen due to templated mutagenesis from the mouse V genes, we asked whether mutations that had a higher probability under the templated mutagenesis model were observed more frequently. For each mutation from germline base b_1, we computed the probability of mutation from b_1 to any of the other three bases at that position under the templated mutagenesis model. We then asked whether target bases that had a higher probability under the templated mutagenesis model were observed more frequently.

We found that mutations with higher probabilities under the templated mutagenesis

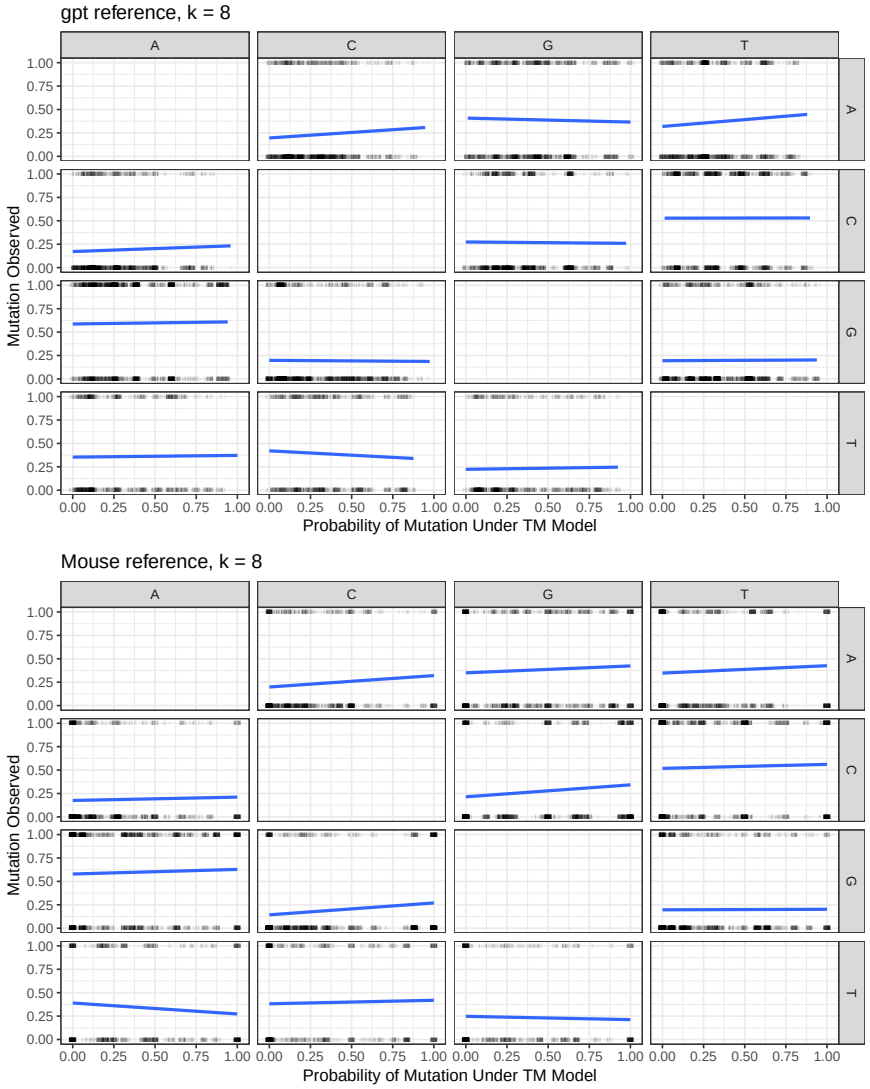


Figure 2.3: Each subplot displays whether a mutation was observed (on the y-axis) versus its probability under the templated mutagenesis model (on the x-axis). A y-value of one means the mutation was observed, and a y-value of zero means the mutation was not observed. For each mutation, the germline base is indicated by the row name, and the target base by the column name. The lines are linear smoothers. We do not observe any consistent and significant trend to these lines, indicating that templated mutagenesis has not contributed to the observed sequence changes in the *gpt* sequence data set.

model were not observed any more frequently than mutations with low probabilities under the model (Figure 2.3). This finding held true both for the model of templated mutagenesis from *gpt* genes and from the 129S1 IGHV genes. To formally test whether mutations with high probabilities occurred more frequently, we performed independent logistic regressions for each pair of germline and target base. The response variable was an indicator of whether the observed mutation was the target base, and the predictor variable was the probability of mutation to the target base under the templated mutagenesis model. In each case, we found that the slope in the model was non-significant at the .05 level, indicating that the templated mutagenesis model did not help to explain the observed pattern of mutations. This analysis provides further evidence that the mutations in the *gpt* sequences did not arise by templated mutagenesis from the IGHV genes present in the mouse.

2.3.5 Upper bounds on the rate of templated mutagenesis

We combined PyMF's estimate of the rate of templated mutagenesis with our estimate of PyMF's false positive rate to obtain an approximate upper bound on the true rate of templated mutagenesis in mice (using the VB1-8 sequences) and in humans (using the anti-Ebola sequences). Plugging in PyMF's estimates of the rate of templated mutagenesis in the VB1-8 sequences and our estimates of PyMF's false positive rates from the *gpt* sequences to Equation 2.2, we found upper bounds on the rate of templated mutagenesis in this system ranging from 0 (in cases where our estimate of the false positive rate exceeds the rate at which PyMF identified templated mutations) to .1, depending on the value of k and the assumed true positive rate (Table 2.1, top panel). The largest upper bounds were obtained at $k = 8$. For the human anti-Ebola sequences, we found upper bounds on the rate of templated mutagenesis ranging from 0 to .12, with the numbers again varying based on the value of k and the assumed true positive rate. In this case, the largest upper bounds is obtained at the largest value of k, $k = 14$, and in general the larger values of k correspond to larger upper bounds. However, note that, since these estimates are upper bounds of the true rates in both humans and mice, they are consistent with a rate of zero.

Mice:

k	PyMF rate	PyMF FPR	UB (1)	UB (.99)	UB (.95)	UB (.9)
8	0.79	0.83	0	0	0	0
9	0.54	0.5	0.08	0.08	0.09	0.1
10	0.28	0.25	0.05	0.05	0.05	0.05
11	0.15	0.14	0.01	0.01	0.01	0.02
12	0.11	0.09	0.01	0.01	0.01	0.02
13	0.07	0.07	0	0	0	0
14	0.06	0.05	0.01	0.01	0.01	0.01

Humans:

k	PyMF rate	PyMF FPR	UB (1)	UB (.99)	UB (.95)	UB (.9)
8	0.73	0.78	0	0	0	0
9	0.44	0.43	0.02	0.02	0.02	0.02
10	0.26	0.2	0.07	0.08	0.08	0.09
11	0.18	0.11	0.09	0.09	0.09	0.1
12	0.15	0.07	0.09	0.1	0.1	0.11
13	0.15	0.05	0.1	0.1	0.11	0.11
14	0.13	0.03	0.1	0.11	0.11	0.12

Table 2.1: Upper bounds (UB) on the rate of templated mutagenesis in the VB1-8 (top) and the anti-Ebola sequences (bottom) computed for a range of tract lengths k and sensitivities. k denotes tract length, PyMF rate is the naive PyMF estimate of the rate of templated mutagenesis, PyMF FPR is the PyMF false positive rate, UB denotes upper bound, and the number in parentheses denotes the assumed sensitivity (true positive rate) of PyMF.

We caution against taking these numbers as definitive as we do not know the true positive rate of PyMF, and they require that our estimate of the false positive rate of PyMF is a lower bound. However, they attempt to correct the observed rates using false positive rate estimates, and in particular show that templated mutagenesis does not occur at a high rate unless PyMF misses many true templated mutagenesis events.

2.3.6 Consistent results using the reverse complementary strand

We also tested whether templated mutagenesis could be occurring from the reverse complementary strand. To this end, we repeated all the analyses with the reverse complements added to the donor gene sets. The results are shown in Supplemental Figures 1 and 2 and Supplemental Table II, and were qualitatively similar to those with the original gene sets. The rate estimates were slightly higher because of the larger size of the donor sets (Supplemental Figure 1). The average probability of the observed mutations given templated mutagenesis from the *gpt* genes and their reverse complements remained about the same as the average probability of the observed mutations given templated mutagenesis from the IGHV genes and their reverse complements (Supplemental Figure 2). The upper bounds on the rate of templated mutagenesis also remained low when the reverse complements were included in the donor gene sets (Supplemental Table II).

2.3.7 Consistent results using filtered donor gene sets

We obtained positive rate estimates of 73% for humans and 79% for mice (Table 2.1) when applying the PolyMotifFinder strategy with $k = 8$ using our chosen donor gene sets and BCR sequence datasets as discussed in the Materials and Methods section. In their analysis, Dale *et al.* estimate this positive rate to range to be approximately $50 - 65\%$ when applying the same strategy to their chosen donor gene sets and BCR sequence datasets. While they describe the five different BCR sequence datasets used to obtain these estimates, two things remain unclear. First, it is not obvious how they extracted the $50 - 65\%$ range from the data displayed in their Figure 5(I), which seems to show positive rate estimates roughly

ranging from $\approx 30\%$ to $\approx 90\%$ and whose rate estimates seem to depend on the dataset in question. Secondly, they do not describe the exact donor gene sets obtained from IMGT. The construction of the donor sets is crucial since the number of genes in the donor set influence the positive rate estimates: adding more templates to the donor set can only increase the number of PolyMotifFinder hits since there will be more chances to observe a match.

To address these discrepancies, we re-ran both of the VB1-8 and anti-Ebola analyses using a more restricted donor gene set in each case. Specifically, we filtered out all open reading frame (ORF) and pseudogene (P) sequence reads from the respective IMGT sets, which led to a 31.7% decrease in potential donors for the VB1-8 sequences and a 44.9% decrease in potential donors for the anti-Ebola sequences. We obtained positive rate estimates of 42% for humans and 62% for mice for $k = 8$. Detailed tables of rate estimates using the filtered donor sets, analogous to Table 2.1, can be found in the main Github respository (`https://git.io/Jfl6t`). Between the collective full and restricted analyses for mice and humans, our positive rate estimates range from $42 - 79\%$, which contains the $50 - 65\%$ range proposed by Dale *et al.*. More importantly, our estimates on the upper bound of templated mutagenesis events remain highly similar between the full and restricted analyses, demonstrating the robustness of our methodology to the particular choice of donor gene set.

2.3.8 A small p-value for a simplified null model does not imply a non-trivial effect size for the rate of templated mutagenesis

Finally, we point out that our estimates of templated mutagenesis occurring at a low rate are in fact compatible with the large values of Stouffer's Z and the correspondingly small p-values obtained in Dale *et al.*. These authors compare the rate of templated mutagenesis to the rate obtained using a simplified null model (called RandomCheck) in which, conditional on the locations of the mutations, the mutation identity at each location is independent of the other locations and follows a fixed distribution taken from previous studies. This model is a simplification of the classical Neuberger model of somatic hypermutation in many ways. In the Neuberger model, lesions introduced by AID can be resolved by one of three pathways,

each of which leads to a repair by a different set of enzymes. The likelihood of each pathway being recruited to repair the lesion depends on nucleotide context, and each pathway is assumed to have its own unique, context-dependent mutation profile [67]. The result is that the mutations are not independent and identically distributed conditional on the germline base, in contrast with the assumption of RandomCheck, which was used to compute p-values and Stouffer's Z in Dale *et al.*. Aside from issues of independence, the overall mutation profile taken from the literature is exceedingly unlikely to be *exactly* correct, and, given enough samples, any consistent hypothesis testing framework will confidently identify even small differences between the true mutation profile and the one drawn from the literature.

To demonstrate that even a slightly misspecified model can lead to extreme values of Stouffer's Z and highly significant p-values, we performed a small simulation study. We suppose that the fraction of mutations explainable by templated mutagenesis in the "true" model is drawn from a beta distribution with mean .518 and variance .048, shown as a dashed line in Figure 2.4. In the null model, the fraction of mutations explainable by templated mutagenesis is drawn from a beta distribution with mean .5 and variance .05, shown as a solid line in Figure 2.4. We simulate 2,000 values (corresponding to 2,000 *gpt* sequences analyzed) for the fraction of mutations explainable by templated mutagenesis, construct Z-values from the hypothesis test that these values come from the null distribution, and finally compute Stouffer's Z from the collection of 2,000 Z values. We performed this procedure 10,000 times, yielding a distribution of 10,000 Stouffer's Z values.

In this simulation, the values of Stouffer's Z were centered around 3.65 with a standard deviation of .97. The corresponding p-values had a median value of 1.3×10^{-4}. 10% of the p-values were smaller than 4.2×10^{-7}, and 90% were smaller than 7.9×10^{-3}. The full distributions of both the p-values and Z statistics are shown in Figure 2.4. These numbers are comparable to those reported in Dale *et al.*, and they show that even a very small amount of misspecification in the null model could lead to very small p-values in the hypothesis testing framework.

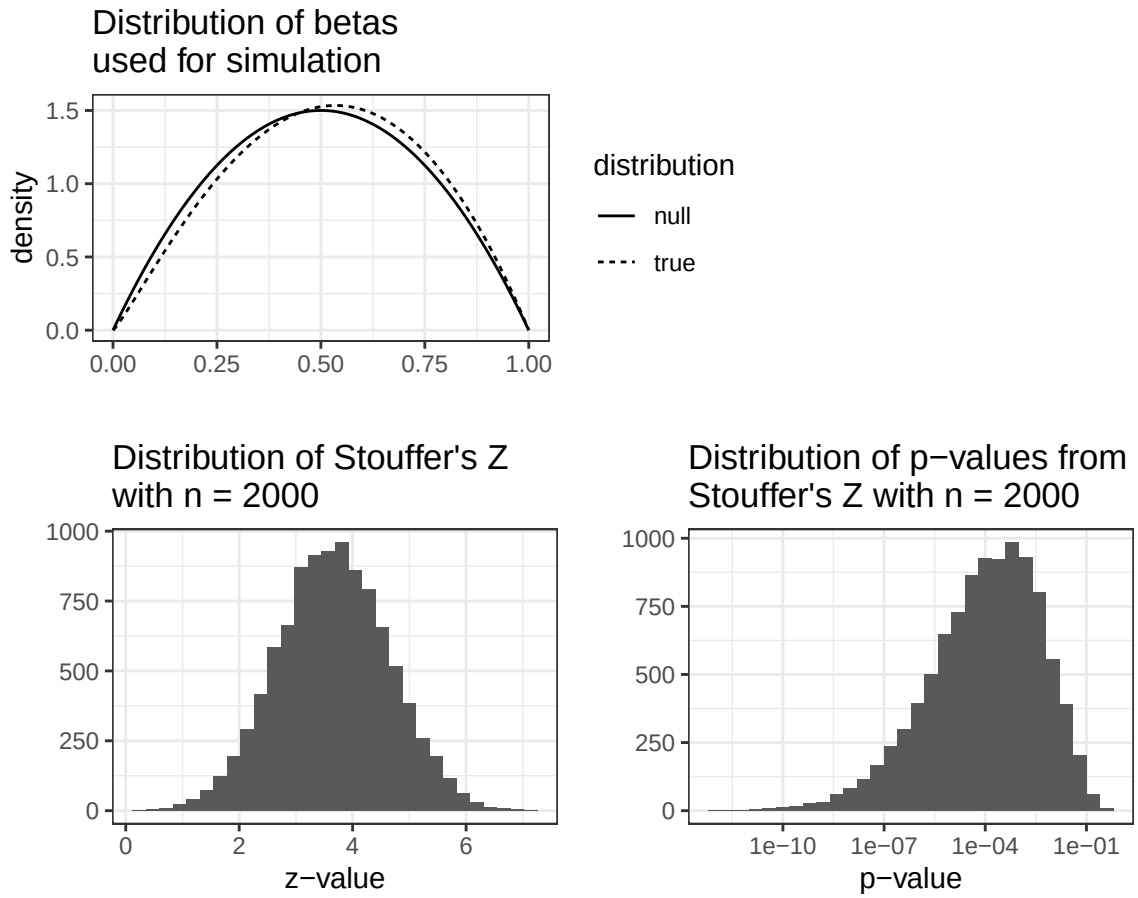


Figure 2.4: Top: Densities of the distributions used in the simulations of Stouffer's Z. Samples coming from the "true" distribution (dashed line) are tested against the hypothesis that they come from the null distribution (solid line). Bottom: Distributions of Stouffer's Z statistics (left) and p-values (right) for the true and null distributions in the top panel for 10,000 simulation trials. In each trial, the Stouffer's Z value is aggregated over 2,000 independent tests, which is about the same as the number of trials aggregated by Dale *et al.*

2.4 Discussion

Species rely on a variety of pathways for secondary antibody diversification, and the reasons for this variety remain an immunological puzzle. The current understanding is that chickens, rabbits, and some other species use a combination of gene conversion and somatic hypermutation during affinity maturation, while humans and mice use only somatic hypermutation. A recent paper by Dale *et al.* suggests that humans and mice also use extensive templated mutagenesis to diversify their repertoires, which may happen by a mechanism similar to gene conversion. This finding was based on a novel method, PolyMotifFinder, for identifying templated mutagenesis via microhomology, and in this article we studied its properties.

We were interested in the false positive rate of the PolyMotifFinder strategy and developed a novel way of estimating this rate. We ran the algorithm on two sets of mutation observations, derived from mouse and human respectively, using two corresponding sets of simulated donor genes not present in the subject in question; any inferences of templated mutagenesis in this case must be spurious. The homology structure of these "mock" donor genes mimicked that of the set of potential templated mutagenesis donors present in the subject. Using this method, we found that although the PolyMotifFinder strategy is quite sensitive to templated mutagenesis, it also has a false positive rate exceeding 50% for the donor tract sizes considered in Dale *et al.*. We used our estimates of the false positive rates of the PolyMotifFinder strategy along with the naive PolyMotifFinder estimates of the rate of templated mutagenesis to obtain upper bounds on the true rate in mice and humans. In each case, we obtain upper bounds ranging from zero to around 10%, although because these are upper bounds, the true rate may also be zero.

Many of the results in Dale *et al.* were based on findings of a statistically significant deviation from a null model instead of an estimate of the rate of templated mutagenesis. The results of the PolyMotifFinder/RandomCheck strategy were presented in terms of a Stouffer's Z score, which describes deviation from a simplified null hypothesis about the way the mutations arise. We showed that the observed Stouffer's Z values and p-values in Dale

et al. are not proof of templated mutagenesis, but merely reflect the fact that the specified null model is incorrect, and given thousands of samples we have enough power to detect even small departures from it.

The same considerations apply to the findings of linkage disequilibrium in the mutated sequences: a statistically significant amount of linkage disequilibrium does not imply templated mutagenesis, and is in fact entirely consistent with the Neuberger model. In particular, if a mutation-generating process satisfies

- mutation at one site implies a higher probability of mutation at nearby sites, and
- not every base has an equal probability of being chosen as the new base for mutation,

then sites that are close together will be in linkage disequilibrium, even though the mutations are not introduced by templated mutagenesis. One of the potential pathways posited by the Neuberger model to resolve AID lesions has exactly the properties described above. In that pathway, an exonuclease strips out several nucleotides around the AID-induced lesion, and the resulting single-stranded sequence is patched by Pol η, an error-prone polymerase. Thus, a mutation at one position is likely to be accompanied by mutations at neighboring positions, since Pol η might have introduced multiple errors in the same patch of nucleotides. In addition, we do not expect Pol η to replace nucleotides uniformly at random, since we expect bias in the nucleotide misincorporation rate [67]. We accordingly expect this pathway to cause linkage disequilibrium between sites, particularly those that are close together. Therefore, the observed significant linkage disequilibrium is not *prima facie* evidence of templated mutagenesis.

Next, we describe several limitations of the analysis presented here to be considered when interpreting the results. First of all, our bounds depend on our estimate of the false positive rate being an underestimate of the true false positive rate. We have two main reasons for believing that this is true, particularly for the human sequences. The first is that our mock donor sets of simulated *gpt* homologs are slightly smaller than the corresponding IMGT donor

gene sets. The mock *gpt* set based on the mouse IMGT IGHV genes has 462 unique genes, compared with 499 in the mouse IMGT IGHV gene set. The corresponding numbers for the mock *gpt* set based on the human IMGT IGHV genes and the human IMGT IGHV genes are 404 and 466. The mock gene sets have slightly smaller numbers of genes than the gene sets they were based on because of the simulation method: not all of the branches in the inferred tree actually lead to a mutation in the simulations, and so there are fewer unique genes than leaves in the tree. Our second reason for believing our estimate of the false positive rate is conservative involves the correspondence between diversity in variable regions and mutation hotspots: in real antibody sequences, mutations are more likely to occur in the CDRs, and there is also more variability in the IGHV genes in the CDRs. This is not the case for the *gpt* sequences: as demonstrated in [85], there are mutation hotspots in the *gpt* genes as well, but these hotspots do not correspond to regions of higher variability in the mock *gpt* gene sets. Since mutations are more likely to occur in regions with more templated mutagenesis templates in the antibody gene sequences than in the *gpt* sequences, we believe that the false positive rate estimate based on the *gpt* sequences is lower than the true false positive rate.

We emphasize that we have obtained bounds on, not estimates of, the rate of templated mutagenesis, and that these bounds depend on assumptions about the sensitivity of PyMF and on our estimate of the false positive rate being conservative. For humans, the quality of the bound also depends on how well our estimate of the PyMF false positive rate translates from mice to humans. We were only able to estimate the false positive rate of PyMF in the mouse because of the transgenic system set up in [85], and that estimate translates to humans to the extent that the somatic hypermutation processes of the two species coincide. We expect the processes to be similar enough that the false positive rate is valid for both species, but any differences that do exist mean that the bounds for mice are more reliable than those for humans.

It is still possible that templated mutagenesis occurs at a low rate. If so, characterizing its properties is important because, even if templated mutagenesis events occur infrequently, they could increase the rate of certain mutation patterns immensely. This has important

implications for estimation procedures (phylogenetic estimation, germline annotation, etc) as well as translational applications such as rational vaccine design. Thus, we do not view our work as closing the book on the interesting possibility that templated mutagenesis could play a role in B cell diversification.

2.5 Supplementary materials

Donor set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
gpt mock from Human	0.18	0.30	0.38	0.41	0.50	0.83
IMGT Human	0.23	0.31	0.36	0.40	0.46	0.74
gpt mock from mouse	0.18	0.39	0.49	0.48	0.58	0.79
IMGT Mouse	0.21	0.38	0.50	0.48	0.57	0.74

Table 2.2: Five-number summaries of the set of divergences between genes and root for four donor gene sets. The divergences for the *gpt* human mock set are similar to the divergences for the IMGT human set, and the divergences for the *gpt* mouse mock set are similar to the divergences for the IMGT mouse set.

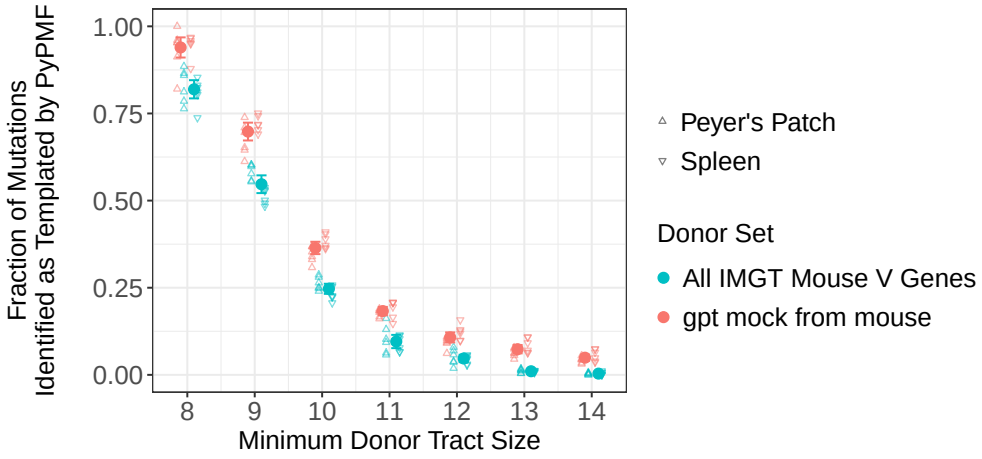


Figure 2.5: Hollow triangles represent the fraction of mutations explainable by templated mutagenesis in each sample, with upward-pointing triangles corresponding to samples from Peyer's patches and downward-pointing triangles corresponding to samples from the spleen. Reverse complements are included in each donor set. For each tract length, the filled circle and error bar represents the overall estimate of the probability of a mutation being explainable by templated mutagenesis plus or minus two standard errors. Points corresponding to samples from Peyer's patches and spleen are offset slightly to the left and right, respectively, to facilitate comparison and to avoid overplotting. This analysis was performed once on data from six individual mice, with two replicates per mouse corresponding to samples from Peyer's patches and spleen, yielding 12 total samples.

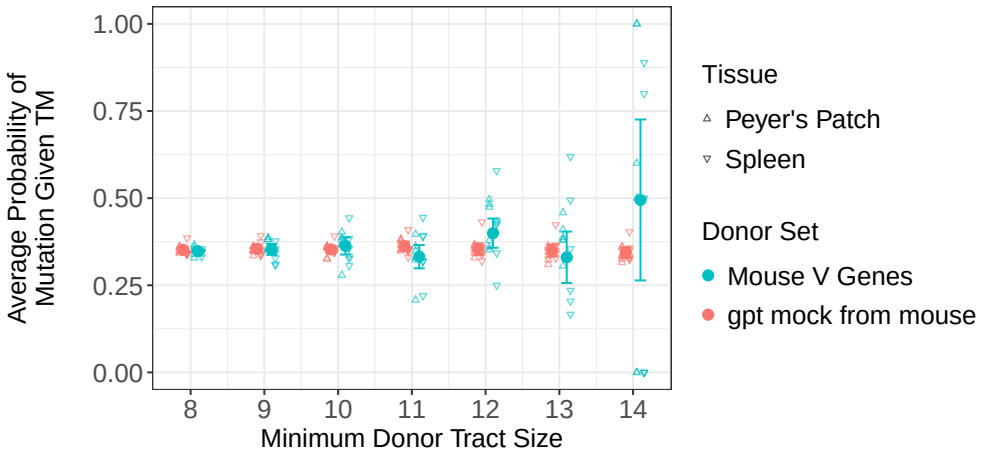


Figure 2.6: Average probability of the observed mutations under a templated mutagenesis model, using *gpt* genes and their reverse complements (red) as well as the set of 129S1 V genes and their reverse complements (blue). Each point corresponds to one sample taken from either spleen or Peyer's patches. This analysis was performed once on data from six individual mice, with two replicates per mouse corresponding to samples from Peyer's patches and spleen, yielding 12 total samples.

Mice:

k	PyPMF rate	PyPMF FPR	UB (1)	UB (.99)	UB (.95)	UB (.9)
8	0.9	0.94	0	0	0	—
9	0.7	0.7	0.02	0.02	0.03	0.03
10	0.46	0.36	0.15	0.15	0.16	0.18
11	0.22	0.18	0.04	0.04	0.04	0.04
12	0.13	0.11	0.03	0.03	0.03	0.03
13	0.08	0.07	0.01	0.01	0.01	0.01
14	0.06	0.05	0.01	0.01	0.01	0.01

Humans:

k	PyPMF rate	PyPMF FPR	UB (1)	UB (.99)	UB (.95)	UB (.9)
8	0.88	0.91	0	0	0	—
9	0.6	0.65	0	0	0	0
10	0.34	0.29	0.07	0.07	0.08	0.08
11	0.22	0.14	0.09	0.09	0.1	0.1
12	0.16	0.08	0.09	0.09	0.1	0.1
13	0.15	0.05	0.1	0.1	0.11	0.11
14	0.14	0.03	0.1	0.11	0.11	0.12

Table 2.3: Upper bounds (UB) on the rate of templated mutagenesis in the VB1-8 (top) and the anti-Ebola sequences (bottom) computed for a range of tract lengths k and sensitivities when including reverse complements in the donor set. k denotes tract length, PyPolyMF rate is the naive PyPolyMF estimate of the rate of templated mutagenesis, PyPolyMF FPR is the PyPolyMF false positive rate, UB denotes upper bound, and the number in paretheses denotes the assumed sensitivity (true positive rate) of PyPolyMF.

Chapter 3

A SUMMARY STATISTIC FRAMEWORK FOR IMMUNE RECEPTOR REPERTOIRE COMPARISON AND MODEL VALIDATION

3.1 Introduction

B cells and T cells play critical roles in adaptive immunity through the cooperative identification of, and response to, antigens. The random rearrangement process of the genes that construct B cell receptors (BCRs) and T cell receptors (TCRs) allows for the recognition of a highly diverse set of antigen epitopes. We refer to the set of B and T cell receptors present in an individual's immune system as their immune receptor repertoire; this dynamic repertoire constantly changes over the course of an individual's lifetime due to antigen exposure and the effects of aging.

Although immune receptor repertoires are now accessible for scientific research and medical applications through high-throughput sequencing, it is not necessarily straightforward to gain insight from and to compare these datasets. Indeed, if these datasets are not processed, they are simply a list of DNA sequences. After annotation one can compare gene usage [33, 44, 15, 25, 11, 9] and CDR3 sequences. This can be a highly involved task, and so it is common to simply compare the gene usage frequencies and CDR3 length distributions of repertoire [47, 39], leaving the full richness of the CDR3 sequence and potentially interesting aspects of the germline-encoded regions unanalyzed.

An alternative strategy is to transform a repertoire to a more convenient space and compare the transformed quantities according to some distance. For example, several studies reduce a set of nucleotide sequences to $k$mer distributions for classification of immunization status or disease exposure [75, 55, 31], where a $k$mer is a nucleotide subsequence of size

k. These $k$mer distributions can then be compared via sequence-based distances, but still comprise a large space and lose important information about where the $k$mer appears along the sequence. One can perform other dimension reduction techniques such as t-SNE to project repertoires down to an even smaller space [86], but these projections also discard a lot of information and can be difficult to interpret biologically.

While many biologically interpretable summaries such as physiochemical properties exist and have been widely applied [13, 57, 82, 81], these are often examined at the sequence level rather than the repertoire level.

We wish to facilitate the use of biologically interpretable summary statistics to capture many different aspects of AIRR-seq data. In addition to enabling comparison of different sequencing datasets, summary statistics can also be used to compare sequencing datasets to probabilistic models to which they have been fitted. Namely, one can use a form of model checking that is common in statistics: after fitting a model to data, one assesses the similarity of the model-generated data to the real data. In this case, we generate a repertoire of sequences from models and compare this collection to a real-data repertoire of sequences via summary statistics.

We are motivated to perform such comparison because these probabilistic models are used as part of inference, and because they are used for inferential tool benchmarking. Such generative models are used to simulate sequences as a "ground truth" for benchmarking inferential software [65, 29, 43], and thus the accuracy of such benchmarks depends on the realism of the generated sequences. Simulation tools can also be used to generate a null distribution used to test for a specific effect, such as natural selection [83].

Currently, there are no unified packages dedicated to the task of calculating and comparing summary statistics for AIRR-seq datasets. While the Immcantation framework (which includes the `shazam` and `alakazam` R packages) contains many summary functions for AIRR-seq data [30], it does not have general functionality for retrieving, comparing, and plotting these summaries. Many summaries of interest are implemented in one package or another, but differences in functionality and data structures make it troublesome to compute and

compare summaries across packages. Some summaries of interest, such as the distribution of positional distances between mutations, are not readily implemented in any package.

In this paper, we gather dozens of meaningful summary statistics on repertoires, derive efficient and robust summary implementations, and identify appropriate comparison methods for each summary. We present `sumrep`, an R package that computes these summary distributions for AIRR-seq datasets and performs repertoire comparisons based on these summaries. We investigate the effectiveness of various summary statistics in distinguishing between different experimental repertoires as well as between simulated and experimental data. We show that many summaries differentiate between various covariates by which the datasets are stratified. Further, we demonstrate how `sumrep` can be used for model validation through case studies of two state-of-the-art repertoire simulation tools: `IGoR` [43] applied to TRB sequences, and `partis` [64, 65] applied to IGH sequences.

3.2 Results

3.2.1 Implementation

The full `sumrep` package along with the following analyses can be found at https://github.com/matsengrp/sumrep. It supports the IGH, IGK, and IGL loci for BCR datasets, and the TRA, TRB, TRD, and TRG loci for TCR datasets. It is open-source, unit-tested, and extensively documented, and uses default dataset fields and definitions that comply with the Adaptive Immune Receptor Repertoire (AIRR) Community Rearrangement schema [78]. A reproducible installation procedure of `sumrep` is available using Docker [8].

Table 3.1 lists the summary statistics currently supported by `sumrep`, and includes the default assumed degree of annotation, clustering, and phylogenetic inference for each summary. The first group of statistics only requires the input or query sequences to be aligned to their inferred germline sequences (e.g. IMGT-aligned) and constrained to the variable region; this coincides with the presence of the `sequence_alignment` and `germline_alignment` fields in the AIRR schema. (We note that some of these statistics, such as GC content, do not

Summary statistic	Annotations	Clustering	Phylogeny	Implementation
Pairwise distance distribution	No	No	No	`stringdist` [76]
kth nearest neighbor distribution	No	No	No	`stringdist`
GC-content distribution	No	No	No	`ape` [59]
Hotspot motif count distribution	No	No	No	`Biostrings` [58]
Coldspot motif count distribution	No	No	No	`Biostrings` [58]
CDR3 length distribution	Yes	No	No	Tool-provided
Joint distribution of germline gene use	Yes	No	No	`sumrep`
Pairwise CDR3 distance distribution	Yes	No	No	`stringdist`
Atchley factor distributions	Yes	No	No	`HDMD` [45]
Kidera factor distributions	Yes	No	No	`Peptides` [45]
Aliphatic index distribution	Yes	No	No	`Peptides`
G.R.A.V.Y. index distribution	Yes	No	No	`alakazam` [30]
Polarity distribution	Yes	No	No	`alakazam`
Charge distribution	Yes	No	No	`alakazam`
Basicity distribution	Yes	No	No	`alakazam`
Acidity distribution	Yes	No	No	`alakazam`
Aromaticity distribution	Yes	No	No	`alakazam`
Bulkiness distribution	Yes	No	No	`alakazam`
Per-gene substitution rate	Yes	No	No	Tool-provided + `sumrep`
Per-gene-per-position substitution rate	Yes	No	No	Tool-provided + `sumrep`
Per-base substitution model	Yes	No	No	`shazam`
Per-base mutability model	Yes	No	No	`shazam` [30]
Positional distance between mutations distribution	Yes	No	No	`sumrep`
Distance from germline to sequence distribution	Yes	No	No	`stringdist`
V gene 3' deletion length distribution	Yes	No	No	Tool-provided
V gene 5' deletion length distribution	Yes	No	No	Tool-provided
D gene 3' deletion length distribution	Yes	No	No	Tool-provided
D gene 5' deletion length distribution	Yes	No	No	Tool-provided
J gene 3' deletion length distribution	Yes	No	No	Tool-provided
J gene 5' deletion length distribution	Yes	No	No	Tool-provided
VD (or VJ) insertion length distribution	Yes	No	No	Tool-provided
DJ insertion length distribution	Yes	No	No	Tool-provided
VD (or VJ) insertion transition matrix	Yes	No	No	`sumrep`
DJ insertion transition matrix	Yes	No	No	`sumrep`
V/J in-frame percentage	Yes	No	No	Tool-provided + `sumrep`
Cluster size distribution	Yes	Yes	No	Custom
Hill numbers (diversity indices)	Yes	Yes	No	`alakazam`
Selection estimates (using the BASELINe method)	Yes	Yes	No	`shazam`
Sackin index distribution	Yes	Yes	Yes	`CollessLike` [48]
Colless-like index distribution	Yes	Yes	Yes	`CollessLike`
Cophenetic index distribution	Yes	Yes	Yes	`CollessLike`

Table 3.1: Currently supported summary statistics grouped by their respective degrees of assumed post-processing. Annotation denotes whether annotation of the V(D)J germline segment is required, Clustering denotes whether clonal clustering is required, and Phylogeny denotes whether lineage tree inference is required. "Tool-provided" means that the summary can be directly computed from the output of an annotation tool.

require an alignment in principle. However, we wished to encourage meaningful analyses and comparisons with our software, and thus require an alignment to avoid accidental comparison of non-corresponding sequence regions.) The second group requires standard sequence annotations, such as inferred germline ancestor sequences for Ig loci, germline gene assignments, and indel statistics. The third group requires clonal family cluster assignments. The fourth group requires a inferred phylogeny for each clonal family of an Ig dataset. `sumrep` itself does not perform any annotation, clustering, or phylogenetic inference, but rather assumes such metadata are present in the given dataset; in principle, one can use any tool which performs these tasks as expected.

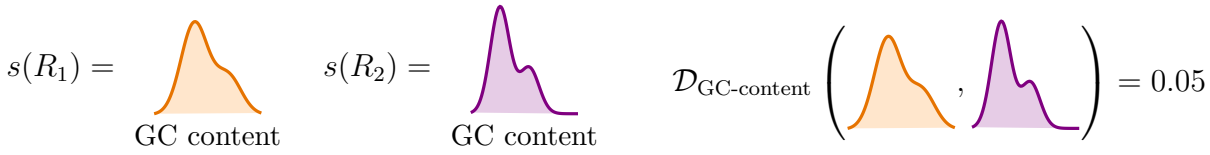
`sumrep` contains many types of summaries, including nucleotide sequence-level summaries (pairwise distances, hotspot motif counts, etc.), rearrangement summaries like insertion and deletion lengths, and many physiochemical properties applicable to the amino acid sequences of particular receptor regions. The Atchley factors are a set of five numerical descriptions of amino acids derived using a statistical technique called factor analysis from a larger pool of 494 descriptors of amino acid biochemical properties [1]. The Kidera factors are a similarly-constructed set of ten numerical descriptions of amino acids, which were derived using dimension reduction techniques [38]. `sumrep` also includes summaries to be applied at the clonal family level (e.g. cluster size distribution) and the phylogenetic level in the case of BCR sequences (e.g. Sackin index distribution).

`sumrep` makes it easy to compare summary statistics between two repertoires by equipping each summary with an appropriate divergence, or measure of dissimilarity, between instances of a summary. For example, the `getCDR3LengthDistribution` function returns a vector of each sequence's CDR3 length, and the corresponding `compareCDR3LengthDistributions` function takes two repertoires and returns a numerical summary of the dissimilarity between these two length distributions. The comparison method depends on the summary, which is discussed further in the Methods section. `sumrep` also includes a `compareRepertoires` function which takes two repertoires and returns as many summary comparisons as befit the data.

Figure 3.1 illustrates the general framework of comparing summary statistics between two repertoires R_1 and R_2. A given summary s is applied separately to R_1 and R_2, which for most summaries yields a distribution of values (Figure 3.1a). These two resultant distributions can be compared using a divergence $\mathcal{D}$ that is tailored to the nature of s (Figure 3.1b). We use Jenson-Shannon (JS) divergence to compare scalar distributions (e.g. GC content, CDR3 length), which is a symmetrized version of KL-divergence, a weighted average log-ratio of frequencies widely-used in statistics and machine learning. We use the similarly popular ℓ_1 divergence to compare categorical distributions (e.g. gene call frequencies, amino acid frequencies), which is a sum of absolute differences of counts.

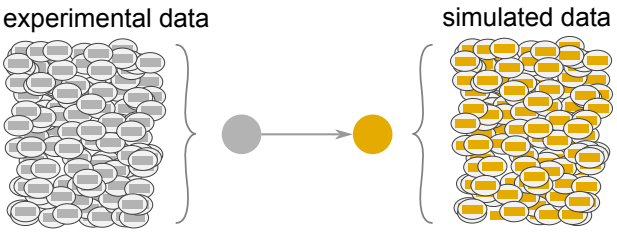
We have designed `sumrep` to efficiently approximate computationally intensive summaries. When the target summary is a distribution, we can gain efficiency by repeatedly subsampling from the distribution until our estimate has stabilized. The result is an approximation to the full distribution; by introducing slight levels of noise, we can gain very substantial runtime performance improvements for large datasets. This in turn allows fast, accurate divergence estimates between dataset summaries. We outline a generic distribution approximation algorithm as well as a modified version for the nearest neighbor distance distribution in the Methods section, and conduct extensive empirical validation of these algorithms in Appendices A and B.

`sumrep` additionally contains a plotting function for each univariate summary distribution. For example, the `getCDR3LengthDistribution` comes with a companion plotting function called `plotCDR3LengthDistribution`. `sumrep` also includes a master plotting function, `plotUnivariateDistributions`, which shows a gridded figure of all univariate distribution plots relevant to the locus in question which can be computed from the input dataset. Currently, these plotting functions support frequency polygons and empirical cumulative distribution functions (ECDFs). Examples of these plots can be found throughout later sections of this report.

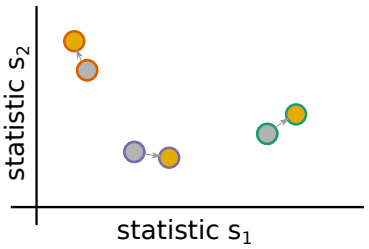


(a) Most summary statistics s, e.g. GC content, yield a distribution of values when applied to each of the sequences in a given repertoire R.

(b) We can compare summary distributions using a statistical divergence $\mathcal{D}$, which takes two distributions and outputs a nonnegative scalar.



(c) For a given experimental dataset, we use simulator tools to generate a corresponding set of synthetic sequences.



(d) We can compute many summaries of these repertoires yielding distributions for comparison.

Figure 3.1: Cartoon of our summary statistic and divergence framework, and how this can be applied to validation of repertoire simulators. Steps (a) and (b) can be applied to compare arbitrary datasets, while (c) and (d) show how sumrep can be used for model comparison.

3.2.2 Application of summary statistics to experimental data

To examine the ability of various summary statistics to distinguish among real repertoires, we applied `sumrep` to TCR and BCR datasets performed a multidimensional scaling (MDS) analysis of summary divergences. In particular, we computed divergences of each summary between each pair of repertoires, stratified by covariates such as individual, timepoint, and cell subset to form a dissimilarity matrix. We then mapped these dissimilarity matrices to an abstract Cartesian space using MDS.

For TCR repertoires, we used datasets from two individuals and five timepoints post-vaccination, with two replicate per donor-timepoint value, from [61]. Figure 3.2 displays plots of the first two coordinates of each replicate grouped by donor and timepoint. We see that for almost all summaries, these points cluster according to donor identity, with the CDR3 pairwise distance distribution being the only summary that does not decisively cluster by donor. Many summaries additionally cluster according to timepoint in the second dimension, although the tightness of clustering varies by summary, with some summaries (e.g. DJ insertion length distribution) being tightly clustered by a given donor/timepoint value and some summaries (e.g. Kidera factor 4) not obviously clustering by donor/timepoint. Moreover, the D gene usage distribution for each individual splits into two distinct groups which do not correlate with timepoint, though the import of this is more difficult to assess. Although these patterns would require further exploration in a particular research context, these `sumrep` divergences show interesting patterns when TCR datasets are stratified by covariates.

We performed a similar MDS analysis of summary divergences of BCR repertoires stratified by covariate, using data from [68]. We computed divergences of each summary between each pair of a collection of datasets stratified by five pairs of twins as well as B cell classification as memory or naive to form a dissimilarity matrix. We then mapped these dissimilarity matrices to an abstract Cartesian space using MDS. Figure 3.3 displays plots of the first two coordinates of each donor grouped by twin pair identity and cell type. We see that

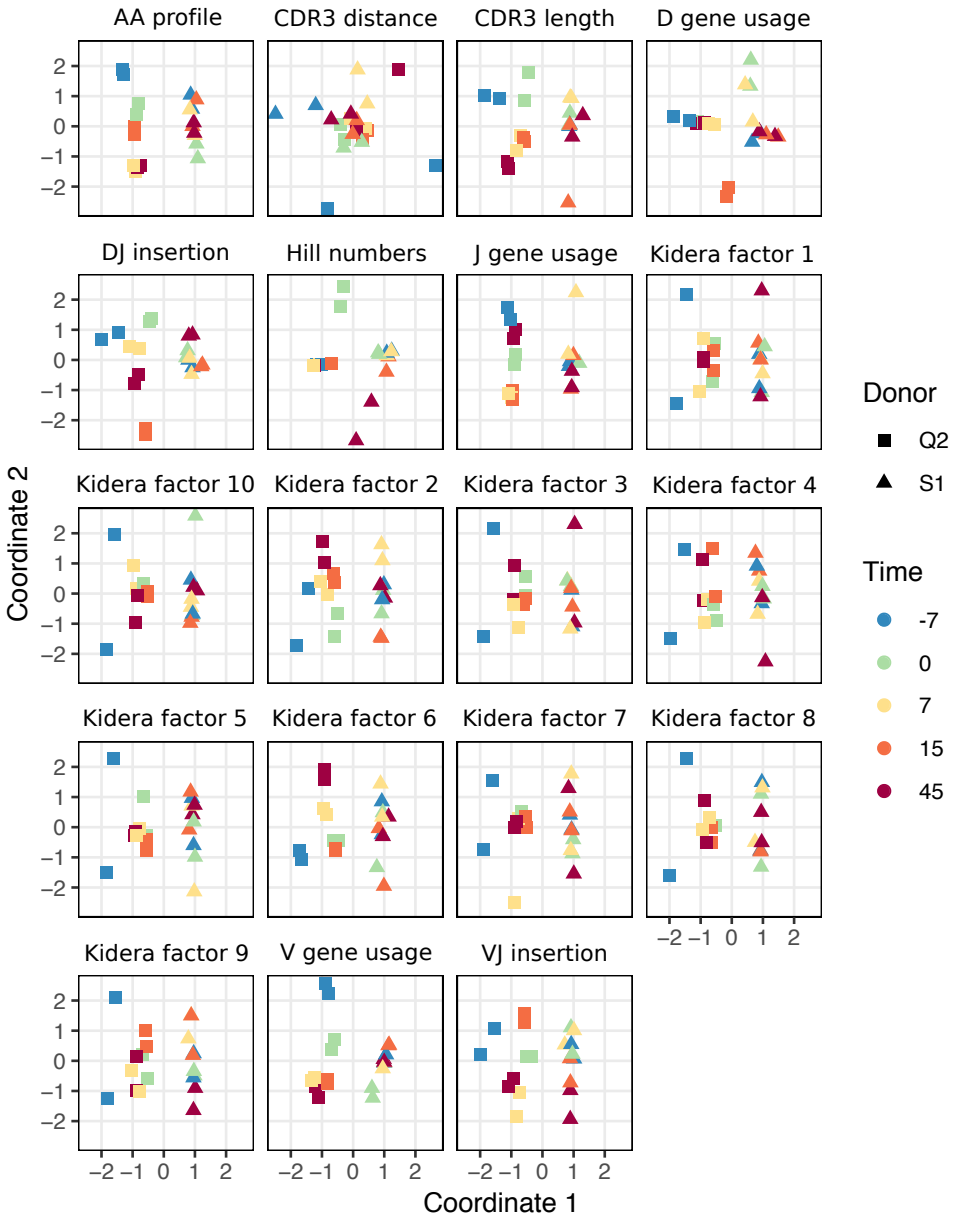


Figure 3.2: Plots of summary divergence MDS coordinates for data from Pogorelyy et al, 2018, grouped by donor and timepoint

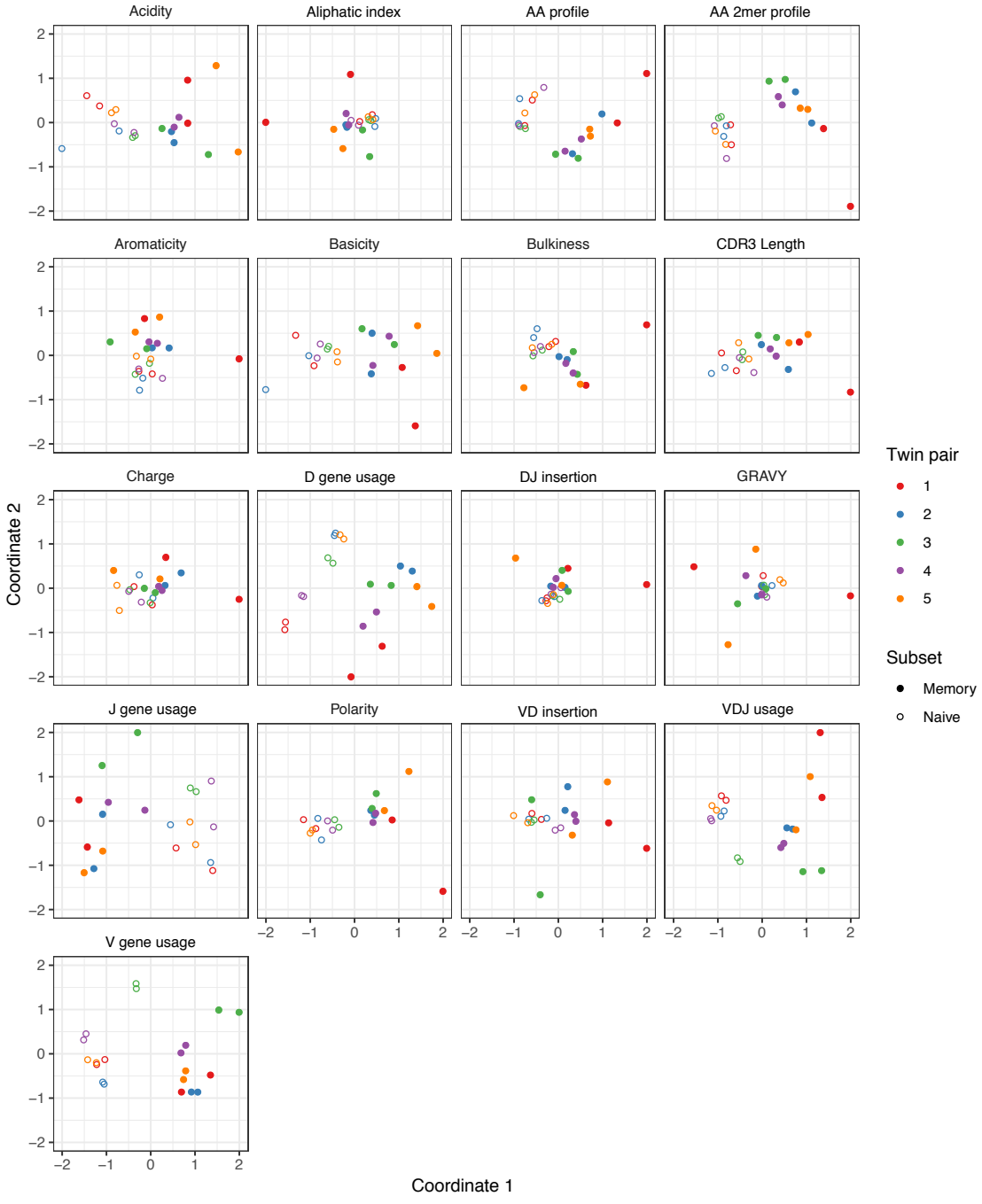


Figure 3.3: Plots of summary divergence MDS coordinates for data from Rubelt et al, 2016, grouped by twin pair identity and cell type (memory vs naive).

for each summary, points can be separated according to cell subset, with some summaries (e.g. V gene usage, AA frequencies, acidity) clustering more tightly among cell subset, and others (e.g. GRAVY index, DJ insertion length) clustering more loosely. In addition, the naive repertoires appear to be more tightly clustered than the memory repertoires for each summary. Finally, for the gene usage statistics, there is a strong tendency for twins to have higher similarity than unrelated donors, although this tendency is not consistently observed for other statistics. For example, points for the amino acid 2mer frequency distribution divergences tend to have high similarity between twins, but the GRAVY index distribution divergences do not. Thus, there seem to also be interesting dynamics underlying `sumrep` divergences when BCR datasets are stratified by covariates, and the observed patterns merit further investigation.

3.2.3 Ranking summary statistic informativeness

Due to the large number of summary statistics supported by `sumrep`, many of which are correlated, we sought an approach to identify a set of maximally-informative statistics that provide complimentary information to one another. To address this, we employed a lasso multinomial regression treating certain sequence-level summaries as covariates and dataset identity as the response. The basic idea is that this regression method cuts out all but a few predictor variables to find a smaller collection of informative summary statistics, as a coefficient is "allowed" to be nonzero only when the lasso deems it a relatively meaningful predictor. As the regularization parameter λ is decreased, more and more coefficients become nonzero, leading to a natural ordering of summaries as the order in which their coefficient "branches off" from zero. Then a resultant maximally-informative set of k summaries is the set of summaries with the k best ranks. We formalize this approach in the Methods section (Algorithm 3).

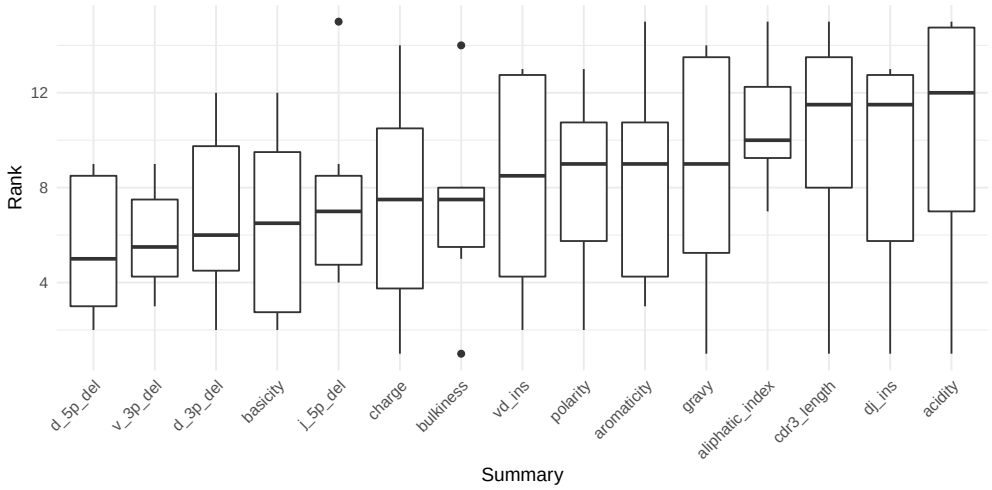
One caveat to this approach is that we can only use sequence-level summary statistics as covariates in order to have a well-defined regression procedure. However, the majority of summaries considered in this report are applied at the sequence level. Thus, between the

subset of informative sequence-level statistics and the remaining non-sequence-level statistics, we arrive at a considerably smaller set. Besides non-sequence-level summaries, we also omit Kidera Factors and Atchley factors from our analyses as these sets of statistics are orthogonal by construction according to particular measures of amino acid composition in their respective original contexts. This also leads to a much smaller design matrix and a substantially decreased runtime.

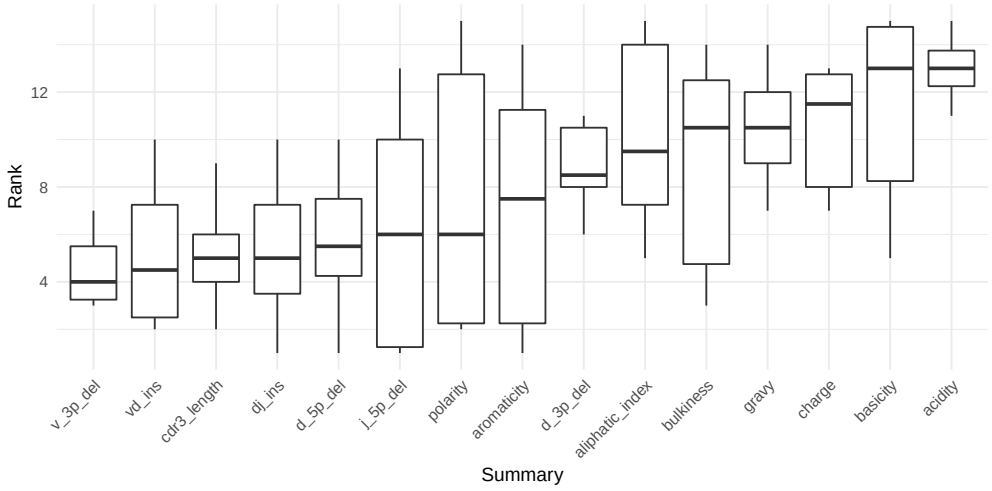
Figure 3.4a displays the results of applying Algorithm 3 to `IGoR` annotations of TRB sequences from datasets A4_i107, A4_i194, A5_S9, A5_S10, A5_S15, and A5_S22 from Britanova et al, 2016 [12]. We see that recombination-based deletion lengths comprise four of the top five summaries, with recombination-based insertion lengths, CDR3 length, and various physiochemical CDR3 properties scattered over the remaining positions. There appears to be high variability throughout the range of rankings, with the bottom three statistics all having a ranking of one for at least one coefficient vector.

Figure 3.4b displays the results of applying Algorithm 3 to `partis` annotations of IGH sequences from donors FV, GMC, and IB at timepoints -8d and -1h from Gupta et al, 2017 [29], downsampled to unique clonal families to avoid clonal abundance biases and decrease algorithmic runtime. We see that deletion lengths, insertion lengths, and CDR3 length comprise the top six summaries, with physiochemical CDR3 properties mostly in the bottom half of rankings. In contrast to the TCR result, there appears to be less overall variability throughout the range of rankings, with variability highest for the moderate ranking positions and notably lower for the top and bottom positions.

While it's difficult to say exactly the level of correlation of each summary by the lasso result alone, since the lasso is a regularized version of least-squares, our intuition is that the nice properties of least-squares combined with the lasso's ability to eliminate less relevant coefficients leads to a subset of covariates that are generally informative. To validate this intuition, we can examine distributions of particularly ranked summaries applied to a test set of annotated repertoires not used in the model fitting. Figure 3.5 displays ECDFs of the the acidity (bottom-ranked), aromaticity (middle-ranked), and V 3' deletion length (top-ranked)



(a) Summary informativeness rank boxplots using six `IGoR`-annotated Britanova (2016) datasets of TRB sequences.



(b) Summary informativeness rank boxplots using six `partis`-annotated Gupta (2017) datasets of IGH sequences.

Figure 3.4: Boxplots of summary rank values taken over each dataset, in order of informativeness, as determined by the median order in which the summary branches off from the lasso paths in Figure 3.17, taken over each of the six paths.

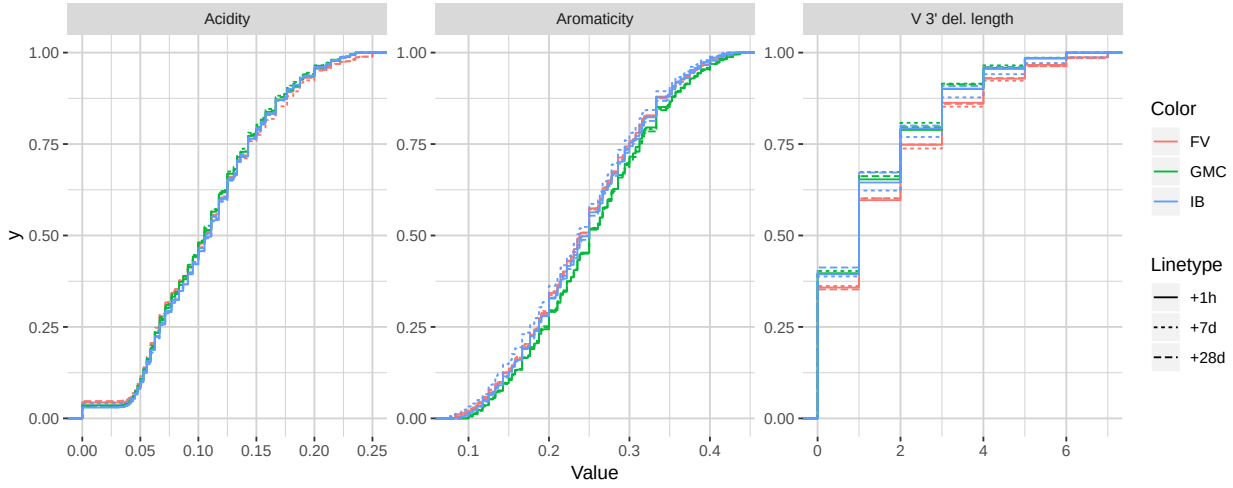


Figure 3.5: Empirical cumulative distribution functions for the bottom-, middle-, and top-ranked statistics for `partis`-annotated IGH repertoires, as determined by Figure 3.4b.

distributions for the FV, GMC, and IB donors at timepoints +1h, +7d, and +28d following an influenza vaccination (which differ from the −1h and −8d timepoints used for fitting), where the ranks are as determined by Figure 3.4b for `partis`-annotated IGH repertoires. Visually, we see that the acidity curves do not vary much among donors or timepoints; the aromaticity curves have slightly more variation but are still highly similar; and the V 3' deletion length curves are more distinguished between some donors (e.g. FV and GMC) as well as some donor-timepoint interactions (e.g. +7d and +28d timepoints for IB). Thus, there is visual evidence that the lasso scores can identify some degree of informativeness among summaries.

3.2.4 Comparing experimental observations to model simulations

`sumrep` can be used to validate BCR/TCR generative models, i.e. models from which one can generate (simulate) data, through the following approach. First, given a collection of AIRR-seq datasets, model parameters are inferred using the modeling software tool for each repertoire, and then these parameters are used to generate corresponding simulated datasets

(Figure 3.1c). Next, `sumrep` is used to compute the summary statistics listed in Table 3.1 for each dataset and compare these summaries between each pair of datasets (Figure 3.1d). Then, a score is calculated for how well the software's simulation replicates a given summary based on how small the divergences of observed/simulated dataset pairs are compared to divergences between arbitrary observed/observed or simulated/simulated pairs.

Applying this methodology using many datasets should give a clear view of which characteristics the model captures well, as well as areas for improvement. As described in the introduction, we are motivated to do this because models are often benchmarked on simulated data, and it is important to understand discrepancies between simulated and observed data in order to properly interpret and extrapolate benchmarking results. We emphasize that validating the model in this way is different than the usual means of benchmarking model performance: rather than benchmarking the inferential results of the model, we benchmark the model's ability to generate realistic sequences.

We illustrate this approach with two case studies: an analysis of `IGoR` [43] applied to TRB sequences, and an analysis of `partis` [64, 65] simulations applied to IGH sequences. Both tools are applied to separate sets of experimental repertoires, yielding model-based annotations for each repertoire, as well as simulated datasets from the inferred model parameters for each experimental set. Summary divergences are applied to each dataset, allowing for scores for each summary to be computed for each tool.

3.2.5 Assessing summary statistic replication for `IGoR`

We apply the methodology discussed in the previous section to TRB sequences from datasets A4_i107, A4_i194, A5_S9, A5_S10, A5_S15, and A5_S22 from Britanova et al, 2016 [12]. Although `IGoR` is typically applied to non-productive sequences in order to capture the pre-selection recombination process, for this example application we wished to understand `IGoR`'s ability to fit the complete repertoire directly without the need for an additional selection model (e.g. [23]). Thus, we fit the `IGoR` model with all sequences (which we expect to be dominated by productive sequences) and restricted the simulation to productive sequences.

Figure 3.6 contains frequency polygons of each summary distribution for each experimental and simulated repertoire.

Observation-based summary scores are computed using a log ratio of average divergences (referred to as LRAD-data, and defined in (3.10)) for a variety of TRB-relevant summaries (Figure 3.7a). The LRAD-data score of a summary will be high when simulations look like their corresponding observations with respect to that summary, and low when observations look more like other observations than their corresponding simulations. We exclude summaries based on `sequence_alignment` values (e.g. pairwise distance distributions) since `IGoR` does not currently have an option to output the full variable region nucleotide sequences for experimental reads.

`IGoR` simulations were able to recapitulate gene usage statistics of an empirical repertoire well, with J gene usage frequency being the most accurately replicated, followed by various recombination-based indel statistics. V, D, and joint VDJ gene usage are all also well-replicated, as well as both VD and DJ insertion matrices. Conversely, the CDR3 length distribution was the least accurately replicated statistic among rearrangement statistics. The Kidera factors of the CDR3 region were also replicated well, despite CDR3 length being one of the least accurately replicated statistics. Scores for other CDR3-based statistics besides Kidera factors ranged from mildly good to mildly bad, with the GRAVY index distribution being the best CDR3-based statistic (excluding Kidera factors) and charge distribution being the worst.

We also computed simulation-based summary scores (LRAD-sim, defined in Equation (3.11)) for the same datasets and simulations (Figure 3.7b). The LRAD-sim score of a summary will be high when simulations look like their corresponding observations with respect to that summary, and low when simulations look more like other simulations than their corresponding observations. We still saw high scores for gene usage and indel statistics, although the CDR3 length distribution and various Kidera factor and GRAVY index distributions had much lower scores. This suggests that while the average `IGoR` simulation yields Kidera factor and GRAVY index distributions that look more like the observed repertoire's distributions

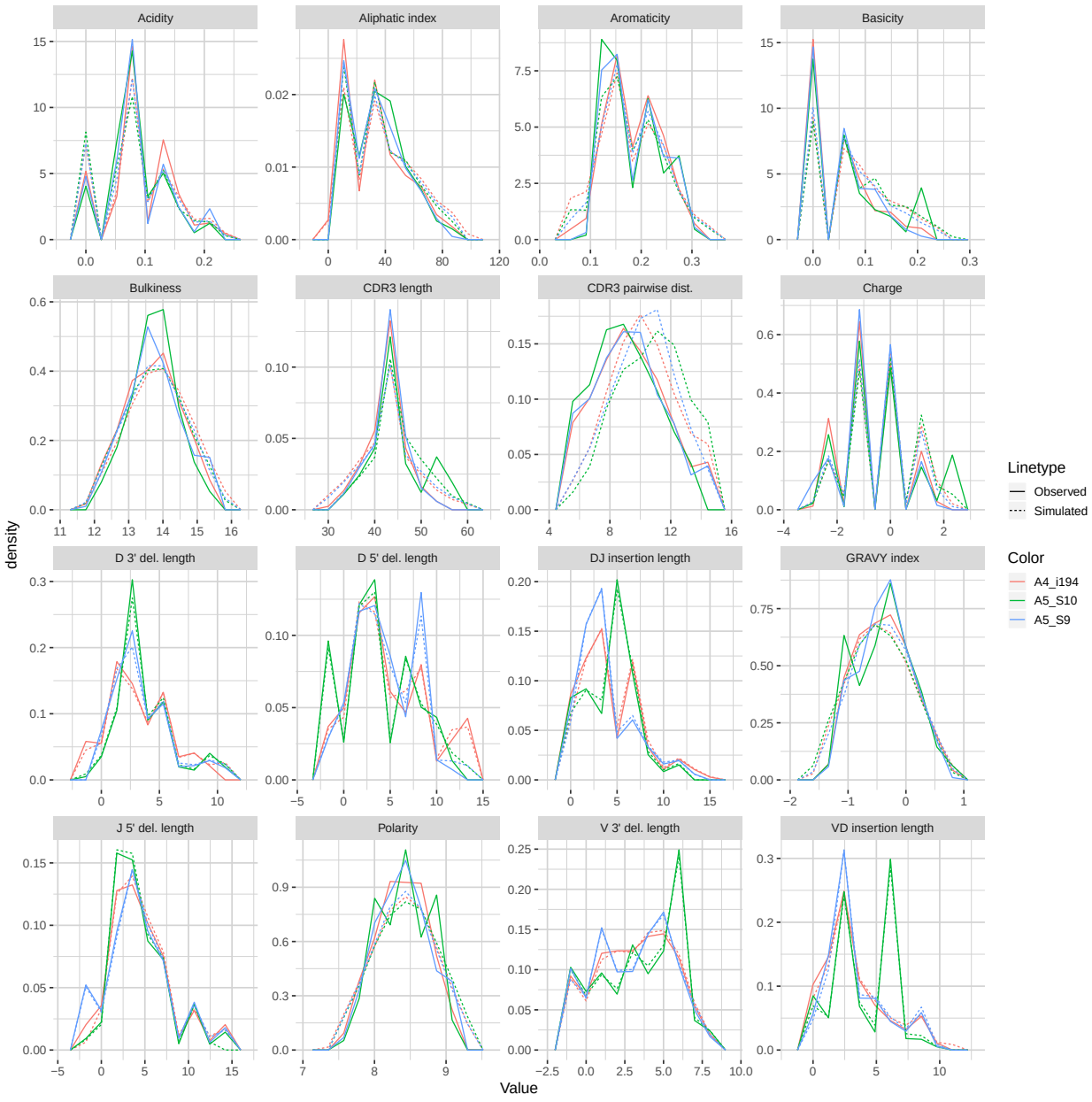
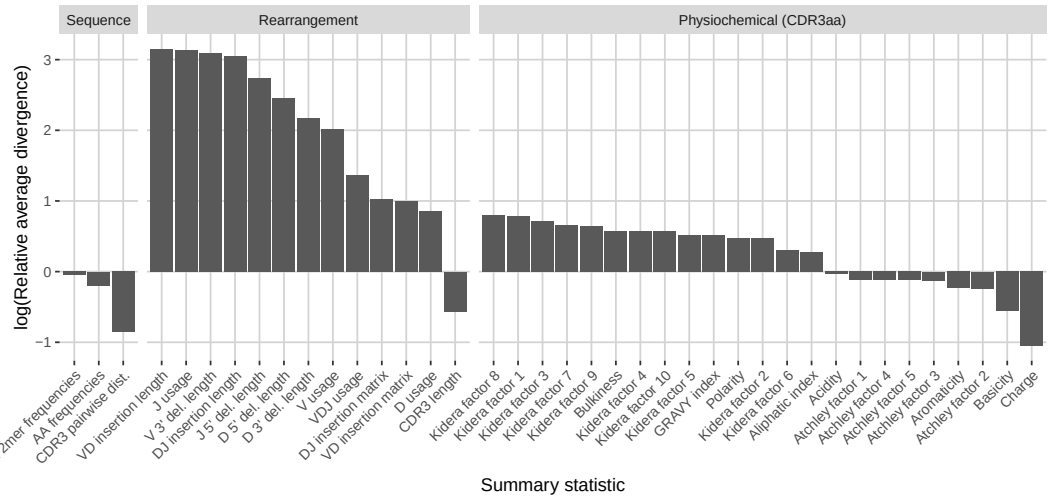
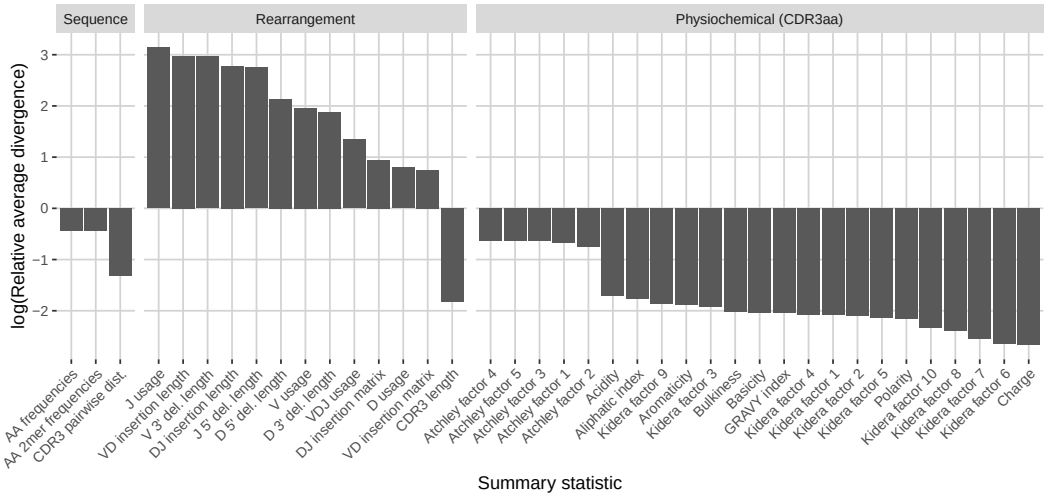


Figure 3.6: Frequency polygon plots of each univariate summary distribution for the IGoR datasets.



(a) LRAD-data values for each relevant TRB statistic available from `IGoR` or `IgBLAST`.



(b) LRAD-sim values for each relevant TRB statistic available from `IGoR` or `IgBLAST`.

Figure 3.7: Summary scores, denoted as "log(Relative average divergence)" or "LRAD," for each statistic in the `IGoR` model validation experiment. For both cases, a high score indicates a well-replicated statistic by the simulations with respect to their corresponding experimental repertoires of functional TRB sequences.

than other observed repertoires do, these simulated repertoires still tend to produce more similar distributions to each other than to their observed counterparts. In turn, this provides an avenue of future research for TCR generative models in which certain CDR3aa properties are incorporated and expressed in simulated data.

3.2.6 Assessing summary statistic replication for `partis`

We applied the same methodology to IGH sequences from Gupta et al, 2017 [29], using datasets corresponding to the -1h and -8d timepoints for each of the FV, GMC, and IB donors. Figure 3.8 displays frequency polygons of each summary distribution for each experimental and simulated repertoire.

Observation-based summary scores were computed using the LRAD-data equation (3.10) for a variety of IGH relevant summaries (Figure 3.9a).

Like `IGoR`, we see that `partis` simulations also excelled at replicating gene usage and recombination statistics, while additionally replicating CDR3 length distributions well. However, `partis` struggled to recapitulate VD and DJ insertion matrices, which it does not explicitly include in its model. This contrasts with `IGoR` which incorporates these insertion matrices during model fitting, and thus recapitulates these matrices well. The other statistics yielded scores ranging from slightly to very negative, with many mutation-based summaries like positional distance between mutations and hot and cold spot counts being poorly captured. The low scores of mutation-based summaries may arise from the decision to select a single representative from each clonal family, which itself arises from the complications in matching clonal family abundance distributions of simulations to data. This makes it difficult to identify the exact contributions of these factors to the summary discrepancies. Nonetheless, this suggests that these sorts of quantities may need to be more explicitly accounted for in BCR generative models if more realistic simulations are desired.

We also computed simulation-based summary scores (LRAD-sim, defined in (3.11)) for the same datasets and simulations (Figure 3.9b). The scores are highly similar to those seen in Figure 3.9a, with some summaries seeing a moderate drop.

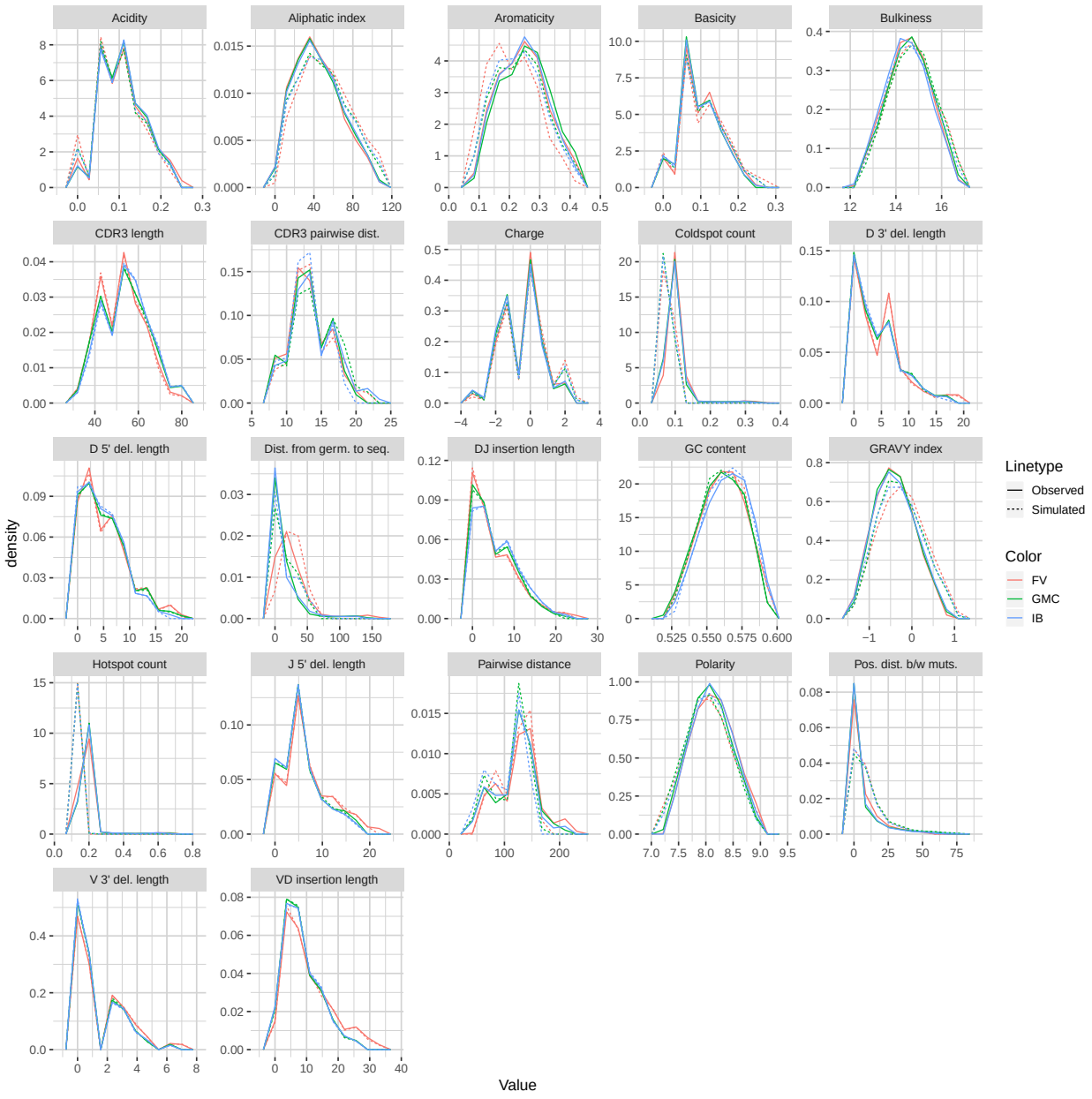
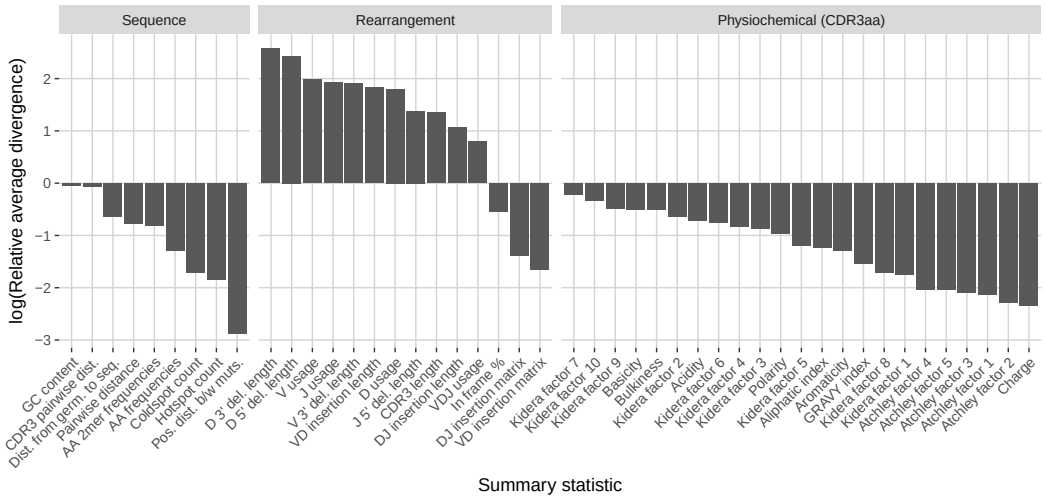
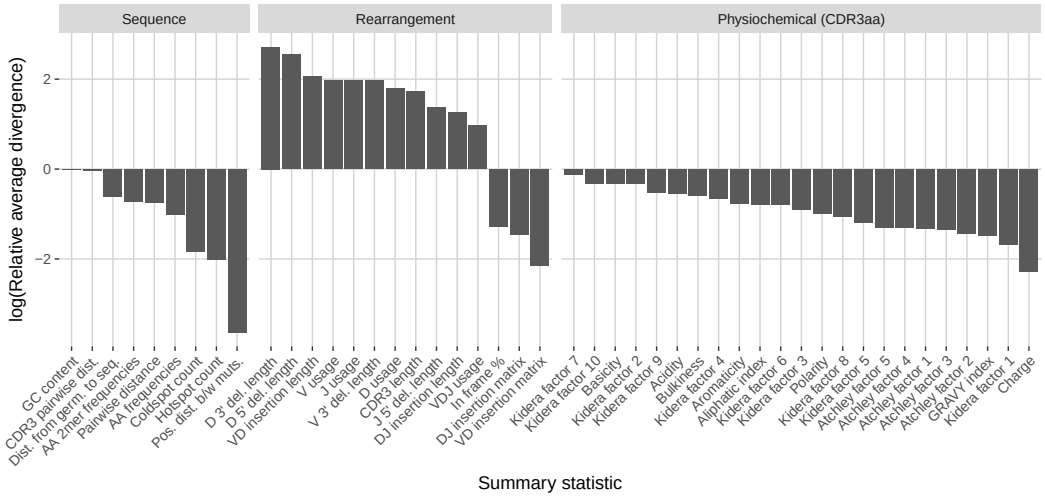


Figure 3.8: Frequency polygon plots of each univariate summary distribution for the `p_f1`, `p_f1_sim`, `p_g1`, and `p_g1_sim` datasets.



(a) LRAD-data values for each relevant IGH statistic available from `partis`.



(b) LRAD-sim values for each relevant IGH statistic available from `partis`.

Figure 3.9: Summary scores, denoted as "log(Relative average divergence)" or "LRAD," for each statistic in the `partis` model validation experiment. For both cases, a high score indicates a well-replicated statistic by the simulations with respect to their corresponding experimental repertoires of productive IGH sequences.

3.3 Methods

3.3.1 Divergence

We use the Jenson-Shannon (JS) divergence for comparing distributions of scalar quantities, which constitutes most summaries in `sumrep`. The Jenson-Shannon divergence of probability distributions P and Q with densities $p(\cdot)$ and $q(\cdot)$ is a symmetrized Kullbeck-Leiber divergence, defined as

$$\mathrm{JSD}\left(P \mid\mid Q\right) := \frac{\mathrm{KLD}\left(P||M\right) + \mathrm{KLD}\left(Q||M\right)}{2} \tag{3.1}$$

where $M := (P+Q)/2$ and $\mathrm{KLD}(P||M)$ is the usual KL-divergence,

$$\mathrm{KLD}\left(P_1 \mid\mid P_2\right) := \mathrm{E}_{\mathbf{X}\sim P_1}\left[\log\left(\frac{p_1(\mathbf{X})}{p_2(\mathbf{X})}\right)\right]. \tag{3.2}$$

In the case where P and Q are both discrete distributions, this becomes

$$\mathrm{KLD}\left(P_1 \mid\mid P_2\right) = \sum_{i\in\mathrm{supp}(P_1)} p_1(i)\log\left(\frac{p_1(i)}{p_2(i)}\right) \tag{3.3}$$

where $\mathrm{supp}(P)$ is the countable support of distribution P. Because the discrete formulation has computational benefits over the continuous one, we discretize continuous samples and treat them as discrete data. By default, we use $B = \max\left(\left\lceil\sqrt{\min(m,n)}\right\rceil, 2\right)$ bins of equal length, where $m = |\mathrm{supp}(P)|$ and $n = |\mathrm{supp}(Q)|$, which is designed to scale with the complexity of m and n simultaneously. We also discard bins which would lead to an infinite KL divergence for numerical stability.

For counts of categorical data, we instead appeal to the sum of absolute differences, or ℓ_1 divergence, for comparison:

$$d_{\ell_1}(R_1, R_2; c, \mathcal{S}) = \sum_{s\in\mathcal{S}} \left|c(s;R_1) - c(s;R_2)\right|. \tag{3.4}$$

In words, (3.4) iterates over each element s in some set $\mathcal{S}$, calculates the count c of s within repertoires R_1 and R_2 respectively, takes the absolute difference of counts, and appends this to a rolling sum. This metric is well suited for comparing marginal or joint V/D/J-gene

usage distributions. For example, if $\mathcal{V}$, $\mathcal{D}$, and $\mathcal{J}$ represent the germline sets of V, D, and J genes, respectively, define usage u of gene triple $(v, d, j) \in \mathcal{V} \times \mathcal{D} \times \mathcal{J}$ for repertoire R as

$$u(R; v, d, j) = \#\left\{s \in R : s_v = v, s_d = d, s_j = j\right\}, \tag{3.5}$$

where e.g. s_v = the V gene of s. Then an appropriate divergence for the joint VDJ gene usage for repertoires R_1 and R_2 is

$$d(R_1, R_2; u, \mathcal{V}, \mathcal{D}, \mathcal{J}) = \sum_{v \in \mathcal{V}} \sum_{d \in \mathcal{D}} \sum_{j \in \mathcal{J}} \left|u(v, d, j; R_1) - u(v, d, j; R_2)\right|. \tag{3.6}$$

The ℓ_1 divergence is also relevant for computing amino acid frequency and 2mer frequency distributions. Note that we can normalize the counts to become relative frequencies and apply (3.4) on the resultant scale which may be better suited to the application, especially when dataset sizes differ notably.

3.3.2 Approximating distributions via subsampling and averaging

Computing full summary distributions over large datasets can be intractable. However, we can compute a Monte Carlo distribution estimate by repeatedly subsampling from the full distribution P_{true} and aggregating our increasingly refined empirical distributions $\mathbb{P}_n^{(i)}$ until convergence. Let the ith iterate distribution have the form

$$\mathbb{P}_n^{(i)}(\cdot) = \frac{1}{n} \sum_{j=1}^{n} \mathbb{1}_{X_j}(\cdot) \tag{3.7}$$

where $X_1, \ldots, X_n$ are values that have already been sampled from P_{true}. In the $(i+1)$th iteration, we sample new points $X_1^*, \ldots, X_k^* \sim P_{\text{true}}$, and update using a rolling-average expression for empirical distributions:

$$\mathbb{P}_{n+k}^{(i)}(\cdot) = \frac{n\mathbb{P}_n^{(i)}(\cdot) + \sum_{j=1}^{k} \mathbb{1}_{X_j^*}(\cdot)}{n+k}. \tag{3.8}$$

We can consider our approximations to be converged when $\text{JSD}\left(\mathbb{P}_n^{(i)},\ \mathbb{P}_{n+k}^{(i+1)}\right) < \varepsilon$ for some specified tolerance ε.

Consider a summary statistic $\sigma(\cdot)$ which produces a set of values over a repertoire R, so that $\sigma(R) = \{x_1, \ldots, x_n\}$. We can specify a corresponding empirical distribution $P_{\text{true}}(\cdot) = \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}_{x_j}(\cdot)$. To obtain a sample $X_1^*, \ldots, X_k^* \sim P_{\text{true}}(\cdot)$ for most such summaries σ, we can simply get a random subset $S \subset R$, and then compute $\sigma(S) = \{X_1^*, \ldots, X_k^*\}$. Algorithm 1 formalizes this as it pertains to the distribution approximation discussed above, which comprises the general distribution approximation routine for `sumrep`.

Algorithm 1 Compute automatic approximate distribution

Input: repertoire R, summary s, batch size m, convergence tolerance ε

Output: subsampled approximation to the full distribution d of R

$R_0 \leftarrow \text{subsample}(R, m)$

$d_0 \leftarrow s(R_0)$

$n \leftarrow 1$

$\text{error} \leftarrow \infty$

while $\text{error} > \varepsilon$ **do**:

  $R_{\text{samp}} \leftarrow \text{subsample}(R, m)$

  $d_{\text{samp}} \leftarrow s(R_{\text{samp}})$

  $d_n \leftarrow \text{concatenate}(d_{n-1}, d_{\text{samp}})$

  $\text{error} \leftarrow \text{JSD}(d_{n-1}, d_n)$

  $n \leftarrow n + 1$

return d_n

An alternative would be to simply compute the distribution on one subsample of the data and use this as a proxy distribution. The main advantage of Algorithm 1 over such an approach is that it provides a sense of convergence to the full distribution via the tuning parameter ε, while automatically determining the size of the necessary subsample. The algorithm can also be tuned according to batch size m, which `sumrep` takes to be 30 by default. We conduct a performance analysis of Algorithm 1 in Appendix A and empirically demonstrate efficiency gains in a variety of realistic settings without sacrificing much

accuracy.

It turns out that some summaries induce distributions for which Algorithm 1 is inherently ill-suited. This occurs when a summary applied to a subset of a dataset does not follow the same distribution as the summary applied to the full dataset. For example, consider the nearest neighbor distance of a sequence s_i with respect to a multiset of sequences R (i.e. elements in R can have multiplicity ≥ 1),

$$d_{\text{NN}}(s_i, R) := \min_{s \in R \setminus \{s_i\}} d(s_i, s), \tag{3.9}$$

where $d(\cdot, \cdot)$ is a string metric (e.g. the Levenshtein distance). We can write the full nearest neighbor distribution as $\sigma_{\text{NN}}(R) := \{d_{\text{NN}}(s_i; R)\}_{s_i \in R}$. However, for any subset S of R, $d_{\text{NN}}(s_i, S) \geq d_{\text{NN}}(s_i, R)\ \forall i$ necessarily, since R will have the same sequences to iterate over, and possibly more sequences, which can only result in the same or a smaller minimum. This causes $\sigma(S)$ to be a heavily biased approximation of the full distribution $\sigma(R)$.

In this case, we can still obtain an unbiased approximate to the nearest neighbor distance distribution using the following modification of Algorithm 1. For each iteration, sample a small batch $B = (s_1, \ldots, s_b)$ of b sequences, and compute d_{NN} of each $s_i \in B$ to the full repertoire R, yielding $\{d_{\text{NN}}(s_i; R)\}_{s_i \in B}$. Since each batch B computes the exact nearest neighbor with respect to R, we get the true value of d_{NN} for each $s \in B$. The gain in efficiency stems from the fact that we only compute this true d_{NN} for a subsample of the sequences of the full repertoire R. Thus, appending batches to a running distribution until convergence as in Algorithm 1 will produce increasingly refined, unbiased approximations as the tolerance decreases. Algorithm 2 formalizes this procedure.

Algorithm 2 may yield a high runtime if R is large, the sequences in R are long, or the tolerance ε is small. Nonetheless, we empirically demonstrate in Appendix B that in the case of typical BCR sequence reads, even very small tolerances incur reasonable runtimes, and when R is large, the algorithm is orders of magnitude faster than computing the full distribution over R. We show that the efficiency and accuracy varies by summary statistic in Appendix B, and identify appropriate defaults accordingly. Specifically,

`sumrep` uses $\varepsilon = 0.001$ for arbitrary summary approximation routines and $\varepsilon = 10^{-4}$ for `getNearestNeighborDistribution`. Moreover, `sumrep` retrieves approximate distributions by default only for `getPairwiseDistanceDistribution`, `getNearestNeighborDistribution`, and `getCDR3PairwiseDistanceDistribution`.

Algorithm 2 Compute automatic approximate nearest neighbor distance distribution

Input: repertoire R, distance d, batch size m, convergence tolerance ε

Output: subsampled approximation to the full nearest neighbor distribution d_{NN} of R

$d_0 \leftarrow$ doBatchStep(R, m)
$n \leftarrow 1$
error $\leftarrow \infty$
while error $> \varepsilon$ **do**:
  $d_{\text{samp}} \leftarrow$ doBatchStep(R, m)
  $d_n \leftarrow$ concatenate$(d_{n-1}, d_{\text{samp}})$
  error $\leftarrow$ JSD(d_{n-1}, d_n)
  $n \leftarrow n + 1$
return d_n

function doBatchStep(R, m)
  for $i = 1, \ldots, m$ **do**:
    $s_i \leftarrow$ subsample$(R, 1)$
    $d_i \leftarrow d_{\text{NN}}(s_i; R)$
  return $(d_1, \ldots, d_m)$

3.3.3 Summary statistic informativeness ranking

To quantify the relative informativeness of various summary statistics in distinguishing between different datasets, we perform a multinomial lasso regression where covariates are sequence-level summaries and the response is dataset identity. Since ℓ_1 multinomial regression outputs a separate coefficient vector $\boldsymbol{\beta}$ for each response value, we aggregate by taking

medians of each dataset-specific lasso ordering for each summary to get the final score. This also yields a range of rankings to assess the variation in scores by summary and by inferential model (e.g. `partis`, `IGoR`). In the case of ties, we randomize rankings to avoid alphabetization biases or other similar artifacts. Detailed pseudocode is provided in Algorithm 3.

Algorithm 3 Rank summary statistics by informativeness

Input: annotations datasets $d_1, \ldots, d_D$, sequence-level summaries $\mathbf{s}(\cdot) = [s_1(\cdot), \ldots, s_S(\cdot)]$, lasso parameters $\lambda_1, \ldots, \lambda_\Lambda$

Output: A vector of ranks for the summaries

for $d = d_1, \ldots, d_D$ **do**:

$\mathbf{X}_d \leftarrow [\mathbf{s}(d_1), \ldots, \mathbf{s}(d_D)]$

$$\mathbf{X} \leftarrow \begin{bmatrix} \mathbf{X}_{d_1} \\ \vdots \\ \mathbf{X}_{d_D} \end{bmatrix}$$

$$\mathbf{y} \leftarrow \begin{bmatrix} \text{rep}(1, \text{rows}(d_1))^\intercal \\ \vdots \\ \text{rep}(D, \text{rows}(d_D))^\intercal \end{bmatrix}$$

▷ $\text{rows}(d_i)$ is the number of sequences in the ith dataset

for $\lambda = \lambda_1, \ldots, \lambda_\Lambda$ **do**:

$(\boldsymbol{\beta}^\lambda_{d_1}, \ldots, \boldsymbol{\beta}^\lambda_{d_D}) \leftarrow \text{MultinomialLasso}(\mathbf{X}, \mathbf{y}; \lambda)$

for $d = d_1, \ldots, d_D$ **do**:

for $s = s_1, \ldots, s_S$ **do**:

$t_{d,s} \leftarrow \min\left(\min\left\{\lambda_1 \le \lambda \le \lambda_\Lambda : \beta^\lambda_{d,s} > 0 \; \forall t > \lambda\right\}, \infty\right)$

$\mathbf{r}_d = \text{rank}(t_{d,s_1}, \ldots, t_{d,s_S})$

$\mathbf{R} = (\mathbf{r}_{d_1}, \ldots, \mathbf{r}_{d_D})$

$\text{scores} = \text{rank}\left(\text{median}_{s_1}(\mathbf{R}), \ldots, \text{median}_{s_S}(\mathbf{R})\right)$

return scores

This approach only works for sequence-level summaries $s \in \mathbb{R}^n$ for a dataset d of $n = \text{rows}(d)$ sequences in order to form a well-defined design matrix $\mathbf{X} \in \mathbb{R}^{\left(\sum_{i=1}^{D} \text{rows}(d_i)\right) \times S}$ over all datasets $d = d_1, \ldots, d_D$ under consideration. For example, it is unclear how to incorporate the pairwise distance distribution, which is not a sequence-level summary, as a covariate, since this summary in general yields a column of a larger length than the number of sequences. Still, as most summaries considered above can be applied at the sequence level, this method greatly reduces the number of summaries the user needs to examine.

3.3.4 Model validation of `IGoR`

We used the `-infer` subcommand of `IGoR` to fit custom, dataset-specific models for each experimental dataset. Since we were interested in many CDR3-based statistics and `IGoR` does not currently include inferred CDR3 sequences with rearrangement scenarios, we used `IgBLAST` to extract CDR3s for each sequence. For each sequence, we considered only the rearrangement scenario with the highest likelihood as determined by `IGoR`. When a list of more than one potential genes was given as the gene call, we considered only the first gene in the list. Several fields were renamed to match the AIRR specification when the definitions align without ambiguity. As described in Results, we trained on productive sequences and restricted the simulation to productive sequences.

We applied `IGoR` in this way to six datasets of TRB sequences from [12], which studied T cell repertoires from donors ranging from newborn children to centenarians.

3.3.5 Model validation of `partis`

We used `partis` to infer custom generative models for each experimental dataset. We ran the `partition` subcommand to incorporate underlying clonal family clustering among sequences during inference, and then downsampled each observed and simulated dataset so that each clonal family is represented by one sequence. Since `partis` returns a list of the top most likely annotations scenarios for each rearrangement event, we considered only the scenario with the highest model likelihood for each sequence. We denote the `indel_reversed_seqs`

field as `sequence_alignment` and `naive_seq` as `germline_alignment` as they satisfy these definitions from the AIRR Rearrangement schema. Several other fields are renamed to match the AIRR specification when the definitions align without ambiguity.

Before running summary comparisons, we randomly downsample to one receptor per clonal family to get a dataset consisting of unique clonotypes for both the observed and simulated datasets. We do this since `partis simulate` draws from distributions over clonal families for each rearrangement event as inferred from `partis partition`. While it is possible to simulate multiple leaves for each rearrangement, it is not obvious how to best synchronize this with the observed clonal family distributions. A more involved analysis would attempt to mimic the clone size distribution in data as closely as possible, potentially with correlations between clone size and other rearrangement parameters, and assess sequence-level statistics within each clonal family. Here we opt to subsample to unique clones and avoid abundance biases altogether.

We applied `partis` in this way to six datasets of IgH sequences from [29], which studied B cell repertoires from donors prior to and following an influenza vaccination.

3.3.6 Scoring summary statistic replication by model

We wish to measure how well a given statistic is replicated when a model performs simulations using parameters inferred from an observed repertoire dataset. One approach is to score the statistic s based on the average divergence of observations to their simulated counterparts when applying $s(\cdot)$, and the average divergence of observations to other observations when applying $s(\cdot)$. Suppose we have k experimental repertoires of immune receptor sequences, and let $R_{i,\text{obs}}$ and $R_{i,\text{sim}}$, $1 \leq i \leq k$, denote the ith observed and simulated repertoire, respectively. For a given statistic s, let $\mathcal{D}_s(R_1, R_2)$ be the divergence of repertoires R_1 and R_2 with respect to s. We can score a simulator's ability to recapitulate s from the observed

repertoire to the simulated via the following log relative average divergence (LRAD):

$$\text{LRAD-data}(s) := \log\left(\frac{\frac{1}{\frac{1}{2}k(k-1)}\sum_{i=1}^{k}\sum_{j\neq i}\mathcal{D}_s\left(R_{i,\text{obs}}, R_{j,\text{obs}}\right)}{\frac{1}{k}\sum_{i=1}^{k}\mathcal{D}_s\left(R_{i,\text{obs}}, R_{i,\text{sim}}\right)}\right). \tag{3.10}$$

For a given summary s, LRAD-data will be positive if the simulated repertoires tend to look more like their experimental counterparts in terms of this summary than experimental repertoires look like other experimental repertoires, and negative if experimental repertoires tend to look more like other experimental repertoires than they do their simulated counterparts. In other words, LRAD-data scores how well a simulator can differentiate s from an experimental repertoire among other repertoires, and recapitulate s into its simulation. Applying the log to the ratio allows for the magnitudes of scores to be directly comparable (so that a summary with score $a > 0$ performs as well as a summary with score $-a < 0$ performs poorly).

Another related score compares the average divergence of observations to their simulated counterparts, and the average divergence of simulations to other simulations. Formally, this becomes

$$\text{LRAD-sim}(s) := \log\left(\frac{\frac{1}{\frac{1}{2}k(k-1)}\sum_{i=1}^{k}\sum_{j\neq i}\mathcal{D}_s\left(R_{i,\text{sim}}, R_{j,\text{sim}}\right)}{\frac{1}{k}\sum_{i=1}^{k}\mathcal{D}_s\left(R_{i,\text{obs}}, R_{i,\text{sim}}\right)}\right) \tag{3.11}$$

where the difference from (3.10) is that the divergences in the numerator are applied to simulated-simulated dataset pairs rather than observed-observed dataset pairs. LRAD-sim for a given summary will be positive if simulated repertoires tend to look more like their experimental counterparts in terms of this summary than simulated repertoires look like other simulated repertoires, and negative if the simulated repertoires tend to look more alike.

These scores underlie the model validation analyses of `partis` and `IGoR` simulations in the Results section, and comprise the values displayed in Figures 3.7 and 3.9. However, this framework can be used to validate any immune receptor repertoire simulator which outputs the fields compatible with the summaries in Table 3.1, or more generally any set of summaries generated by a model-based simulator that is not supported directly by `sumrep`.

A feature of our methodology is that we use the same tool to produce simulations that we used to produce the annotations. To examine the sensitivity of this method, we performed a separate analysis by obtaining dataset annotations from standalone `IgBLAST` [84], and comparing these to simulations based on `partis` annotations using IMGT germline databases. This is discussed in detail in Appendix E; in particular, we find that scores differ to varying extents between the tools, and argue that while there are probably some biases when using a common tool for annotations and simulations, this is also driven by the differences in the nature of the tools' specifications. We did not perform a similar analysis for `IGoR` annotations since `IgBLAST` was used to infer CDR3s within the `IGoR` workflow.

3.3.7 Materials

The raw data for the TCR summary divergence MDS analysis comes from [61], which was postprocessed into a suitable format for analysis. For each donor-timepoint combination, a single blood draw was split in replicas at the level of cell mixture.

The raw data for the BCR summary divergence MDS analysis comes from [68]; `IgBLAST`-preprocessed data was downloaded from VDJServer in the AIRR format. For quality control, sequences with a run of 3 or more `N` bases in the raw sequence were discarded.

For the TCR model validation analysis, we use six datasets from [12], corresponding to labels A4_i107, A4_i194, A5_S9, A5_S10, A5_S15, and A5_S22. For tractability purposes, we chose the six datasets with the fewest number of sequence reads; the number of reads from these six datasets used in the analysis ranged from 37,363 sequences to 243,903 sequences. These datasets consist of consensus RNA sequences assembled using UMIs. Most of these sequences are productive; as previously described, for this example application we are benchmarking `IGoR`'s ability to fit complete repertoires rather than only non-productive repertoires.

The data for the BCR model validation analyses originated from samples first sequenced and published in [40], although we used the Illumina MiSeq data published in [29] for our analyses. These datasets represent repertoires of three human donors from multiple time

points following an influenza vaccination. We use datasets from time points −1h and −8d for the FV, GMC, and IB donors for the summary informativeness and `partis` model validation analyses; the +1h, +7d, and +28d datasets for the FV, GMC, and IB donors for the summary informativeness validation; and the FV -1h dataset for the approximation routine performance analyses in appendices 1 and 2.

3.4 Conclusions

We have presented a general framework for efficiently summarizing, comparing, and visualizing AIRR-seq datasets, and applied it to several questions of scientific interest. One can imagine many further applications of `sumrep`, as well as promising avenues of research: contrasting repertoires in the context of antigen response or vaccination design and evaluation may shed some light on which summaries can distinguish between such covariates; and comparing the summary distributions of naive repertoires from multiple healthy individuals is likely to aid our understanding of the patterns of variability exhibited by "normal" repertoires, which in turn may aid the detection of repertoire abnormalities. `sumrep` could also be used to evaluate the extent to which artificial lymphocyte repertoires look like natural ones [24].

There are several other packages dedicated to detailed summaries and visualization of immune receptor repertoires. The `tcR` [52] and `bcRep` [6] packages for R include methods for retrieving and comparing gene usage summaries, computing clonotype diversity indices, and visualizing various repertoire summaries. `VDJtools` [72] is a command line tool which performs similar repertoire summarization, comparison, and visualization tasks for TCR data. Desktop GUI-based programs include `ImmunExplorer` [69] and `Vidjil` [20]. `Vidjil` is also available as a webserver, as is `ASAP` [2]. `Antigen Receptor Galaxy` [34] offers online access to many analysis tools. These tools have a subset of the summary statistics described here, and do not have the comparative analysis features of `sumrep`. The `IGoR` [43] software features an algorithm for summarizing statistics of the V(D)J rearrangement process; however, its main focus is on learning the basic model for non-productive T- and B-cell repertoire and

it does not provide any built-in methods for comparing inferred models between datasets.

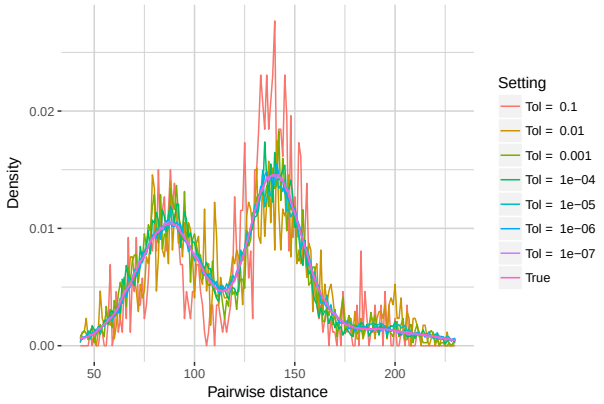
A natural extension of the model validation in this report would be to assess the performance of many competing repertoire analysis tools over a larger group of datasets. `sumrep` can be also used to detect systemic biases between different library preparation protocols and control for batch effects that can confound meta-analysis of AIRR-Seq data. Moreover, while many of the summaries are applied to the CDR3 region by default, it would be interesting to perform separate analyses restricted to different CDRs and framework regions, as physiochemical characteristics of these regions can differ greatly.

Finally, although `sumrep` already supports the AIRR rearrangement schema by default, we plan to thoroughly integrate `sumrep` as a downstream analysis tool for any AIRR-compliant software or workflow.

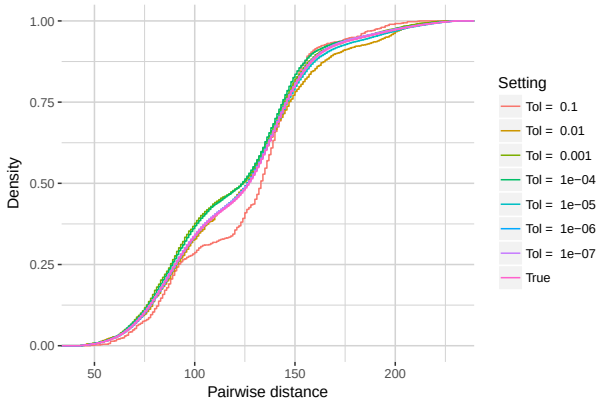
3.5 Appendix A: Performance analysis of Algorithm 1

Here, we run Algorithm 1 on the `partis`-annotated FV -1h dataset (henceforth referred to as `p_f1`), subsampled without replacement to 10,000 sequences for tractability. We compute the pairwise distance distribution of CDR3 sequences for the full subsampled dataset, and approximate distributions with tolerances $\varepsilon \in \{0.1, 0.001, \ldots, 10^{-7}\}$. We replicate this experiment for 10 trials so that the subsampled dataset remains the same, but a new instance of the subsampling algorithm is run each time. Figure 3.10a shows a frequency polygon of each distribution and figure 3.10b shows their empirical cumulative distribution functions. We see that the approximate distributions appear to converge to the full distribution as the tolerance gets smaller. Figure 3.10c displays the KL-divergence to the true distribution for each tolerance, again indicating convergence to the truth. Figure 3.10d displays the runtimes and log-runtimes for each tolerance as well as the true "population" runtime for the full dataset; while the runtime grows exponentially as $\varepsilon \to 0$, the approximation algorithm is still much faster than computing the full distribution for each considered value of ε.

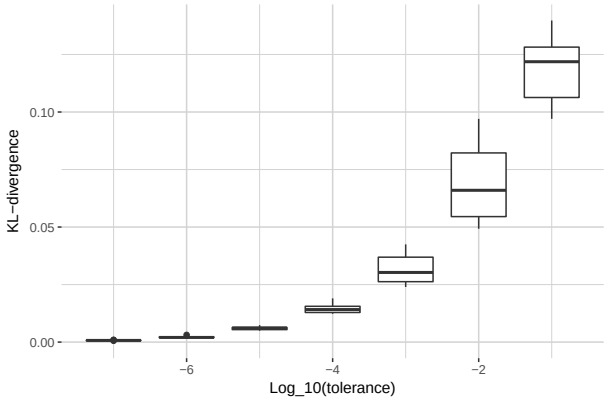
Next we investigate the effect of dataset size on the performance of Algorithm 1. For sample sizes $n \in \{\exp(6), \ldots, \exp(10)\}$, we subsample `p_f1` without replacement to n sequences



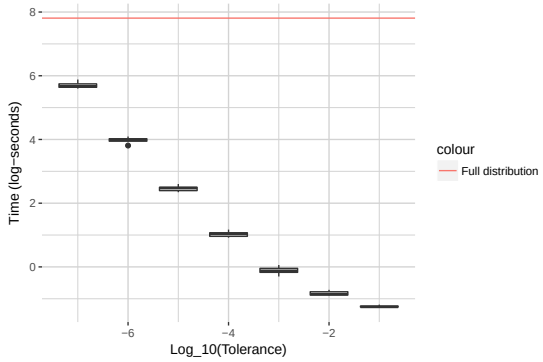
(a) Frequency polygons of true and subsampled pairwise distance distributions by tolerance.



(b) ECDF of true and subsampled pairwise distance distributions by tolerance.



(c) KL-divergence to true pairwise distance distribution by tolerance, taken over 10 trials of the algorithm.



(d) Runtime (in log-seconds) for Algorithm 1 by tolerance, taken over 10 trials.

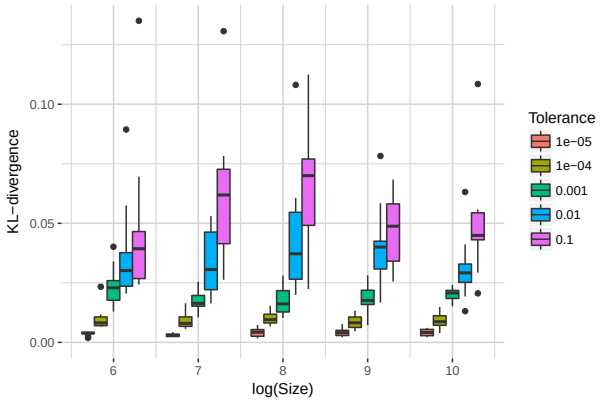
Figure 3.10: Performance of Algorithm 1 by tolerance applied to the pairwise distance distribution.

and compute the pairwise distance distribution of CDR3 sequences for the full subsampled dataset as well as those given by tolerances $\varepsilon \in \{0.1, 0.01, ..., 10^{-5}\}$. We perform this experiment 10 times for each n. Boxplots of the KL-divergence by $\log(n)$ and tolerance over all trials are displayed in Figure 3.11a. We see no obvious trend in the effect of dataset size on the KL-divergence for any choice of tolerance for the pairwise distribution. Boxplots of the runtime (in log-seconds) by log(size) and tolerance are shown in Figure 3.11b, showing that runtime increases with sample size for high tolerance, but tends towards a constant runtime by sample size as tolerance decreases. Boxplots of the log-efficiency by log(size) and tolerance are shown in Figure 3.11c, where

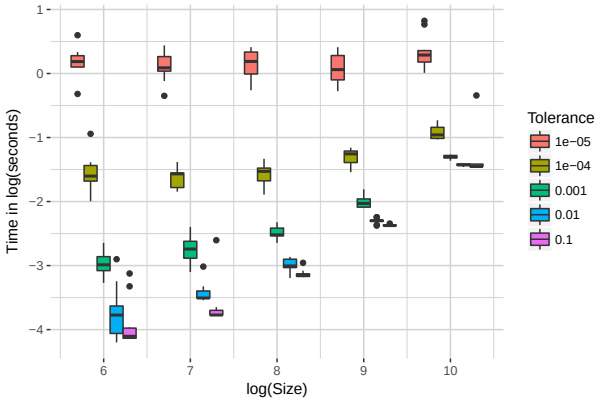
$$\text{Efficiency} := \frac{\text{time to compute full distribution}}{\text{time to compute approximate distribution}}. \tag{3.12}$$

Here we plot efficiency on a log scale, so that the line $y = 0$ corresponds to instances when the true and approximate routines have identical runtimes. Thus, the region $y > 0$ corresponds to instances when Algorithm 1 outperforms the computation of the full nearest neighbor distribution. For moderate to large datasets and reasonable choices of ε, the approximate routine is much more efficient than computing the full distribution. Efficiency also appears to increase exponentially with dataset size, although decreases at least exponentially as tolerance decreases. Nonetheless, the accuracy of Algorithm 1 applied to the pairwise distance distribution is scalable to large datasets while leading to large gains in runtime efficiency for reasonable choices of ε.

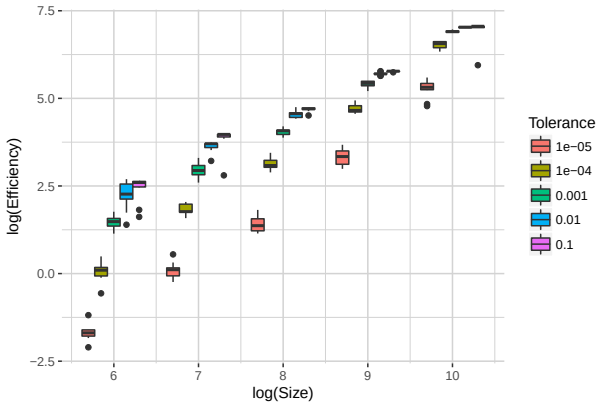
Finally, we investigate the effect of summary statistic on the performance of Algorithm 1. We run the algorithm for the pairwise distance, GC content, hotspot count, coldspot count, and distance from germline to sequence distributions on `p_f1` subsampled without replacement to 10,000 rows. For each summary, we run the algorithm for tolerances $\varepsilon \in \{0.1, \ldots, 10^{-5}\}$. We perform this experiment 10 times for each (summary, ε) combination. Figures 3.12a, 3.12b, and 3.12c show the KL-divergence to the full dataset distributions, runtimes, and efficiencies, respectively, by summary and tolerance over all trials. We see that the KL divergence, runtime, and efficiency of the approximation routine



(a) KL-divergence to true pairwise distance distribution by tolerance and log(size) of dataset, taken over 10 trials of the algorithm.

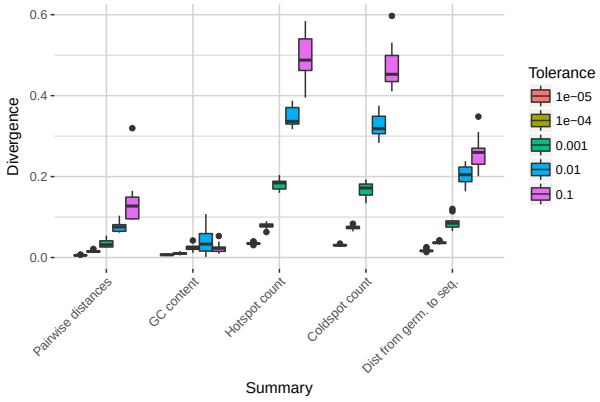


(b) Runtime by tolerance and log(size) of dataset, taken over 10 trials of the algorithm.

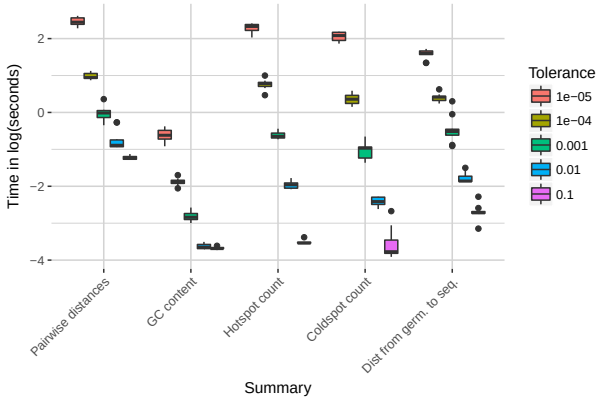


(c) Efficiency by tolerance and log(size) of dataset, taken over 10 trials of the algorithm.

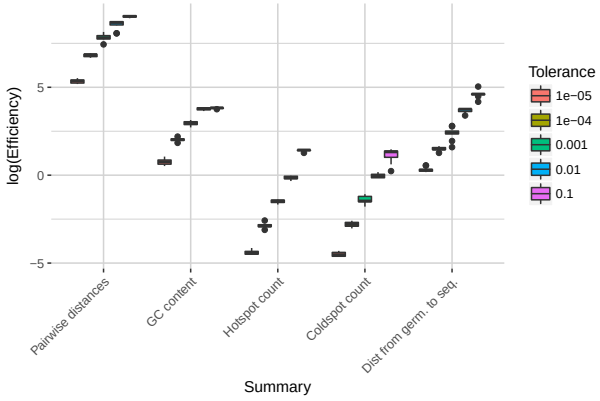
Figure 3.11: Performance of Algorithm 1 by sample size and tolerance applied to the pairwise distance distribution.



(a) KL-divergence to true summary distributions by tolerance, taken over 10 trials of the algorithm



(b) Runtime by summary distribution and tolerance, taken over 10 trials of the algorithm



(c) Efficiency by summary distribution and tolerance, taken over 10 trials of the algorithm

Figure 3.12: Performance of Algorithm 1 by summary statistic and tolerance applied to the pairwise distance distribution.

depends on the summary in question. In particular, the approximation routine for hotspot and coldspot count distributions does not yield as high of an efficiency for moderately low tolerance, and struggles to minimize the KL-divergence to the true distribution for higher tolerances. This is likely due to the fact that the full hotspot and coldspot count distributions is extremely fast to compute even for large datasets.

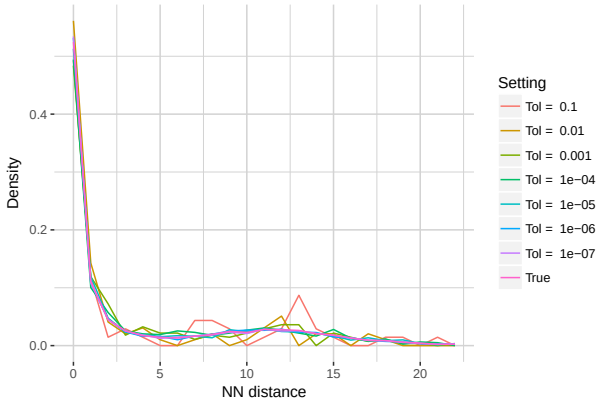
These results suggest that convergence and efficiency will vary by summary, and the user should be aware of this fact when choosing whether to run the approximation routine as well as an appropriate tolerance. By default, `sumrep` uses $\varepsilon = 0.001$ for arbitrary summary approximation routines, and retrieves approximate distributions by default only for `getPairwiseDistanceDistribution`, `getNearestNeighborDistribution`, and `getCDR3PairwiseDistanceDistribution`.

3.6 Appendix B: Performance analysis of Algorithm 2

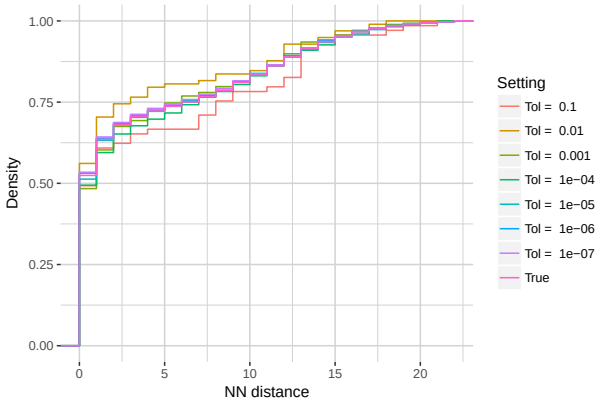
Here, we assess the modification of the distribution approximation routine for the nearest neighbor distribution. We run Algorithm 2 on `p_f1` subsampled without replacement to 10,000 sequences for tractability. We compute the nearest neighbor distribution of CDR3 nt sequences for the full subsampled dataset, and approximate distributions with tolerances $\varepsilon \in \{0.1, 0.001, \ldots, 10^{-7}\}$. We replicate this experiment for 10 trials in the same manner as detailed in Appendix A.

Figure 3.13a shows a frequency polygon of each distribution, and Figure 3.13b shows their empirical cumulative distribution functions. Figure 3.13c shows KL divergences of approximate distributions to the true distribution which decay as $\varepsilon \to 0$. Indeed, these three figures indicate that the approximate distributions converge to the full distribution as $\varepsilon \to 0$. Figure 3.13d displays boxplots of the runtime in log-seconds for Algorithm 2 as well as the runtime to compute the full distribution. In this case, we see that the Algorithm 2 becomes slower than computing the full distribution when $\varepsilon \lesssim 10^{-5}$.

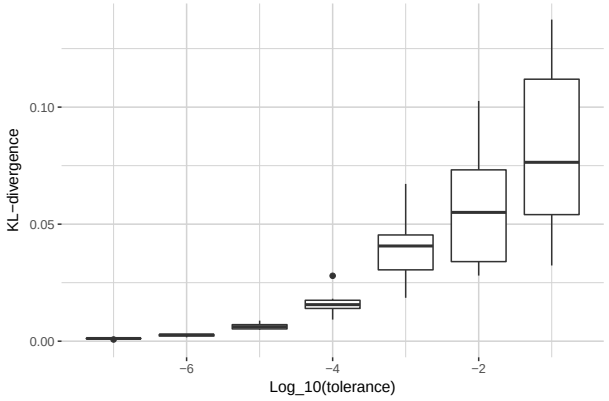
To assess the effect of sequence lengths on Algorithm 2, we perform the same experiment as above on pairwise aligned VDJ sequences (via the `sequence_alignment` column rather



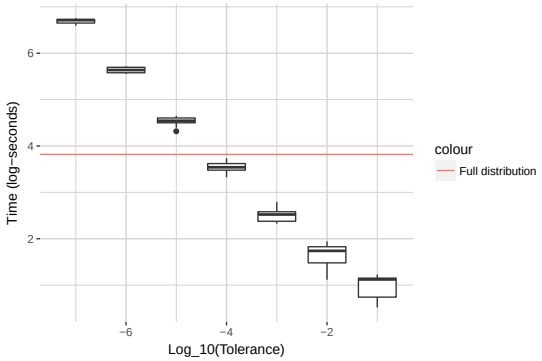
(a) Frequency polygons of true and subsampled nearest neighbor distance distributions by tolerance.



(b) ECDF of true and subsampled nearest neighbor distance distributions by tolerance.



(c) KL-divergence to true nearest neighbor distance distribution by tolerance, taken over 10 trials of the algorithm.



(d) Runtime (in seconds) and log-runtime (in log-seconds) for Algorithm 2 by tolerance, taken over 10 trials.

Figure 3.13: Performance of Algorithm 2 by tolerance applied to the nearest neighbor distribution of CDR3nt sequences.

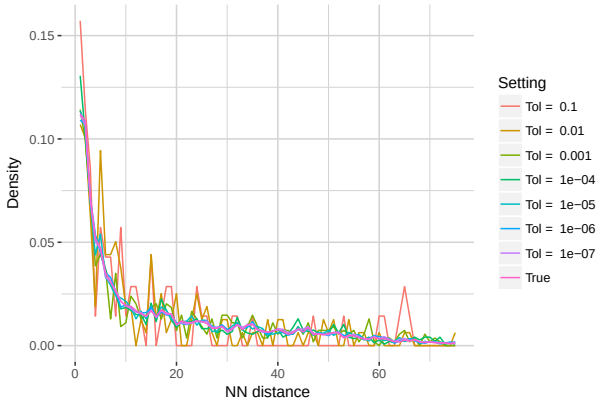
than inferred CDR3 sequences. These length distributions are different by about an order of magnitude. We note that the pairwise aligned VDJ sequences are the default for Algorithm 2 within `sumrep`, although we anticipate users to examine this distribution for CDR3s as well as full V(D)J sequences. We run Algorithm 2 on the same subsampled 10,000 sequences of `p_f1`.

Figure 3.14a shows a frequency polygon of the same distributions, and Figure 3.14b shows their empirical cumulative distribution functions. Moreover, Figures 3.14c and 3.14d show the KL-divergences to truth and runtimes, respectively. It seems that the KL divergence to the truth may converge more slowly for `sequence_alignment` sequences rather than CDR3s, although the approximate procedure seems to outperform the full distribution for a slightly larger range of ε values (i.e. until ε nears 10^{-6}).

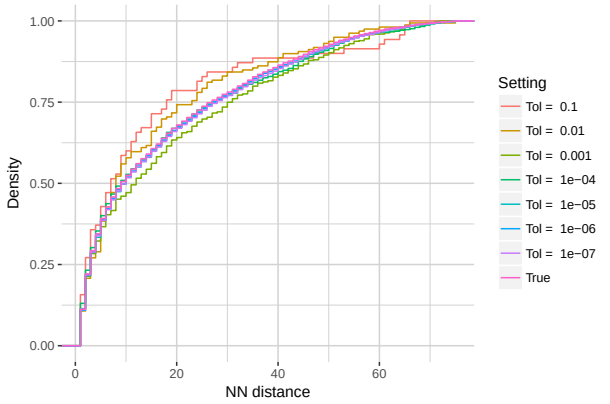
Next we investigate the effect of dataset size on the performance of Algorithm 2. For sample sizes $n \in \{\exp(6), \ldots, \exp(10)\}$, we subsample `p_f1` without replacement to n sequences and compute the pairwise distance distribution of CDR3 sequences for the full subsampled dataset as well as those given by tolerances $\varepsilon \in \{0.1, ..., 10^{-5}\}$. We perform this experiment 5 times for each n.

Figures 3.15a, 3.15b, and 3.15c display boxplots of the KL-divergence to truth, runtime, and time efficiency, respectively. There is not an obvious trend in KL divergence to truth for a given tolerance as sample size increases, although the variability is higher for high tolerances. As expected, runtime increases as tolerance decreases, and also increases with the size of the dataset. This is reasonable since each batch iteration of Algorithm 2 must compute the nearest neighbor distance from each sequence in batch B to the full repertoire R, which certainly increases in time complexity as R increases.

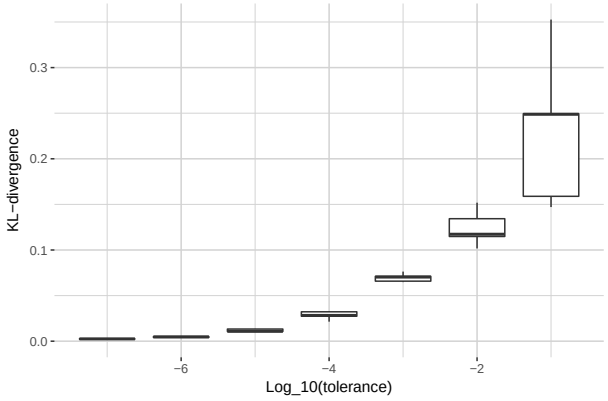
Next we look at the efficiency relative to computing the full distribution as defined in Equation 3.12. Examining the boxplots near $y = 0$ by log(size), we see that for a dataset of size $\exp(k)$, we would need a tolerance of at least $\frac{1}{10^{k-4}}$. For example, for $\log(\text{size}) = 6$, we see that tolerances higher than $0.01 = \frac{1}{100} = \frac{1}{10^{6-4}}$ would on average yield an efficiency greater than one. This suggests that, for a dataset with n CDR3 sequences, a sensible rule of thumb



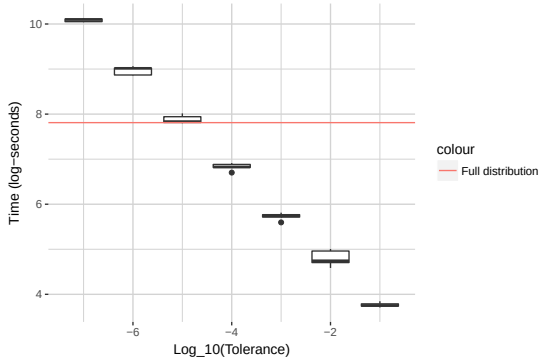
(a) Frequency polygons of true and subsampled nearest neighbor distance distributions by tolerance.



(b) ECDF of true and subsampled nearest neighbor distance distributions by tolerance.

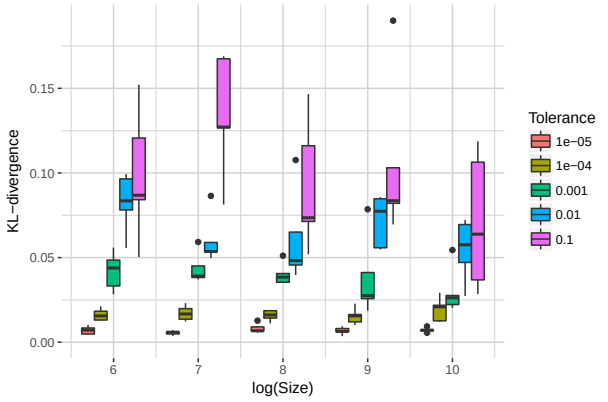


(c) KL-divergence to true nearest neighbor distance distribution by tolerance, taken over 10 trials of the algorithm.

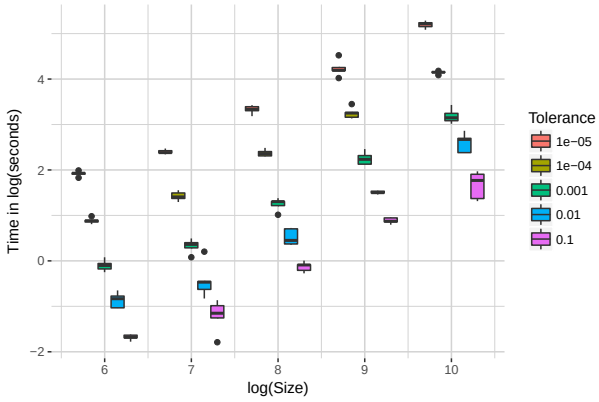


(d) Runtime (in seconds) and log-runtime (in log-seconds) for Algorithm 1 by tolerance, taken over 10 trials.

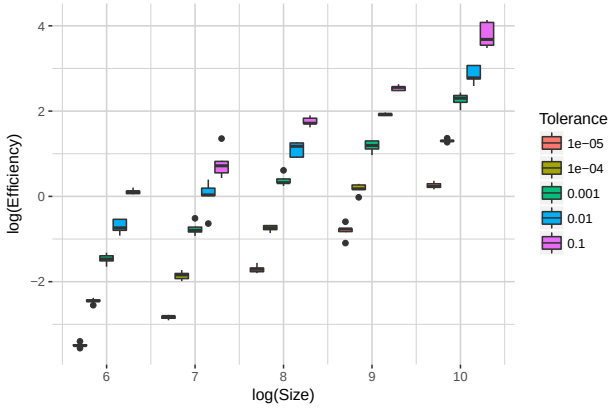
Figure 3.14: Performance of Algorithm 2 by tolerance applied to the nearest neighbor distribution of pairwise-aligned VDJ sequences.



(a) KL-divergence to true nearest neighbor distribution by tolerance and log(size) of dataset, taken over 10 trials of the algorithm.



(b) Time complexity of the approximate nearest neighbor distribution by tolerance and log(size) of dataset, taken over 10 trials of the algorithm.



(c) log(Efficiency) vs and log(size) of dataset by tolerance, taken over 10 trials of the algorithm.

Figure 3.15: Performance of Algorithm 2 by sample size and tolerance applied to the nearest neighbor distribution of CDR3nt sequences.

would be to choose $\varepsilon > \frac{1}{10^{k-4}} = \frac{1}{10^{\log(n)-4}}$. This will of course be more or less appropriate for a given dataset depending on the nature of the repertoire from which it was sampled.

Finally, we perfrom the same experiment but using `sequence_alignment` sequences for the nearest neighbor distance distribution. Figures 3.16a, 3.16b, and 3.16c display boxplots of the KL-divergence to truth, runtime, and time efficiency, respectively. There is evidence of a positive trend of the KL-divergence as sample size increases for $\varepsilon = 0.1$, although this trend seems to diminish for each other tolerance. Runtimes increase with given sample size and tolerance, and are generally higher than they are for CDR3 sequences as expected. It turns out that the efficiencies follow the same rule of thumb we derived for the CDR3 sequence situation. In particular, choosing $\varepsilon > \frac{1}{10^{k-4}} = \frac{1}{10^{\log(n)-4}}$ will on average lead to an increase in efficiency with respect to the full distribution for `sequence_alignment` sequences as well as CDR3 sequences. While this may depend on the dataset in question, we recommend this as a good point of reference for general use.

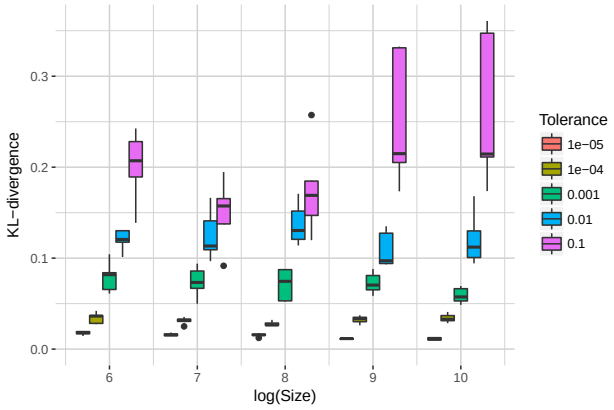
The user should use these results as well as problem-specific considerations when deciding whether or not to use Algorithm 2 instead of computing the full distribution, and if so, which tolerance to use. By default, `sumrep` retrieves the approximate rather than full nearest neighbor distribution, and uses $\varepsilon = 10^{-4}$ unless otherwise modified.

3.7 Appendix C: Multinomial lasso path plots

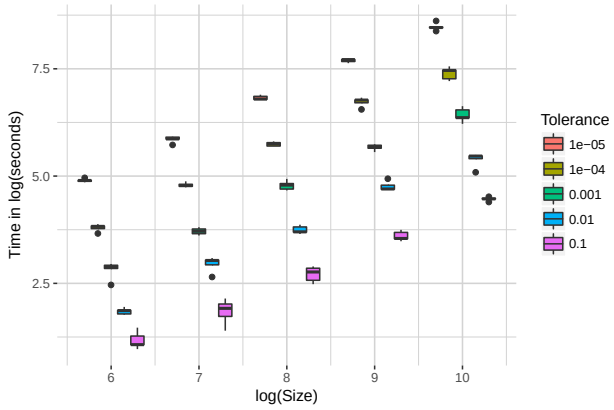
Figure 3.17 displays the lasso path plots which illustrate the coefficient values of each response vector utilized in Algorithm 3. Figure 3.17a shows paths for `IGoR` annotations of six TRB datasets from [12], and Figure 3.17b shows paths for `partis` annotations for six IGH datasets from [40].

3.8 Appendix D: Model validation analysis workflows

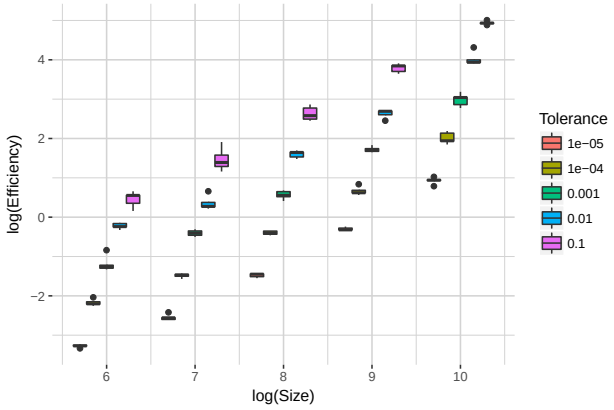
Figure 3.18a illustrates the `IGoR` model validation workflow. We employ `IgBLAST` to obtain CDR3 sequences for the observed sequences, which `IGoR` only outputs for generated



(a) KL-divergence to true nearest neighbor distribution by tolerance and log(size) of dataset, taken over 10 trials of the algorithm.

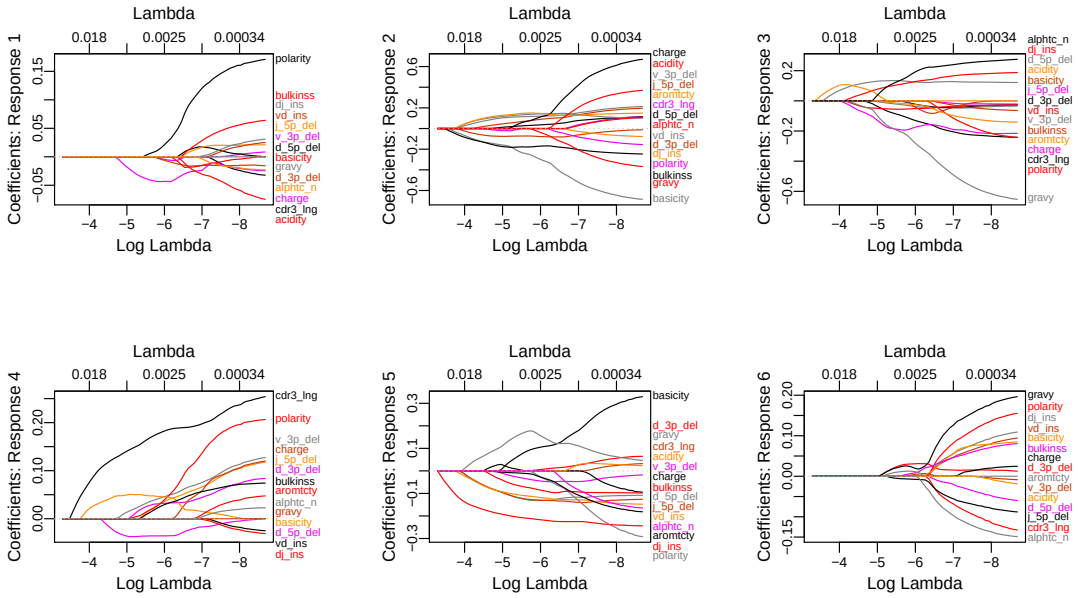


(b) Time complexity of the approximate nearest neighbor distribution by tolerance and log(size) of dataset, taken over 10 trials of the algorithm.

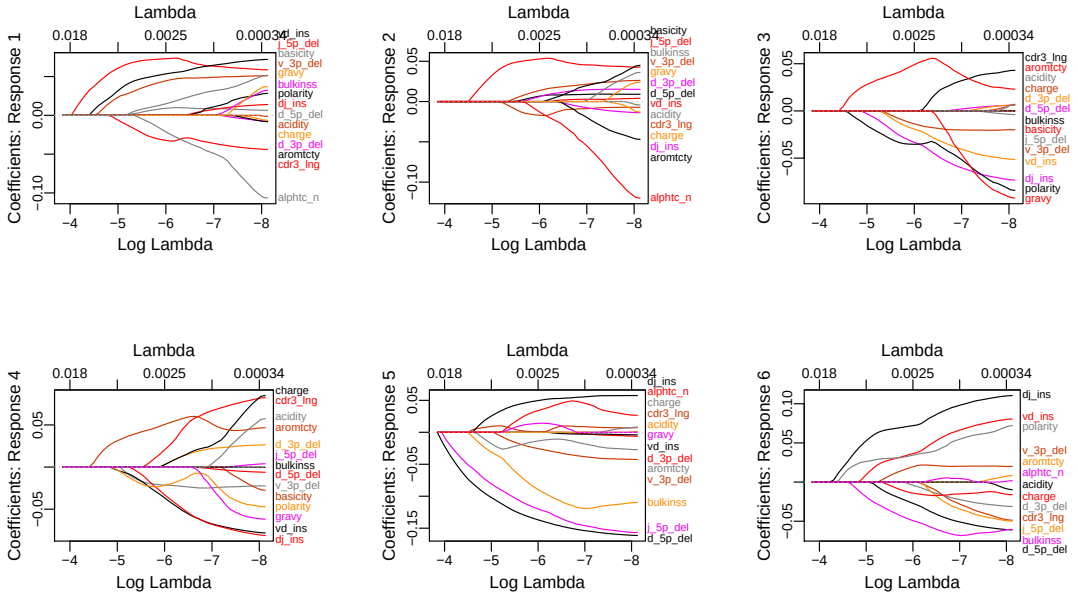


(c) log(Efficiency) vs and log(size) of dataset by tolerance, taken over 10 trials of the algorithm.

Figure 3.16: Performance of Algorithm 2 by sample size and tolerance applied to the nearest neighbor distribution of pairwise-aligned VDJ sequences.

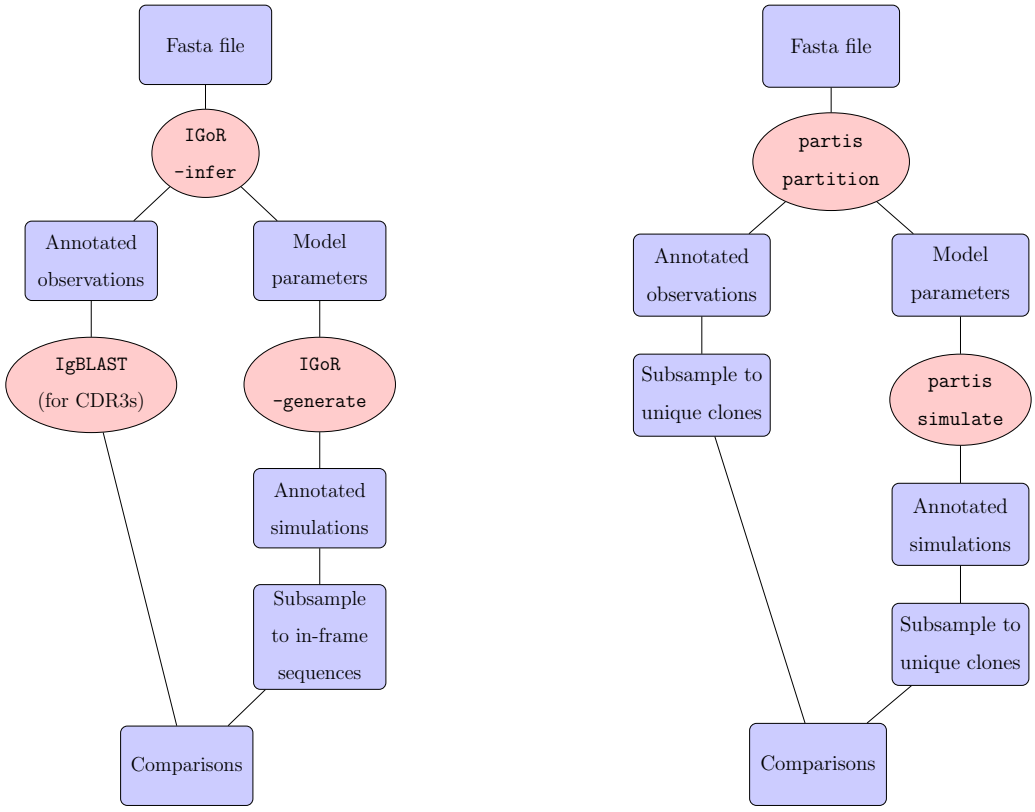


(a) Lasso paths for six `IGoR`-annotated Britanova datasets.



(b) Lasso paths for six `partis`-annotated Laserson datasets.

Figure 3.17: Multinomial lasso paths of summary coefficients by dataset identity.



(a) Workflow for comparing a given observed repertoire dataset to an example simulated dataset via `IGoR`.

(b) Workflow for comparing a given observed repertoire dataset to an example simulated dataset via `partis`.

Figure 3.18: Workflow diagrams for the `IGoR` and `partis` model validation analyses.

sequences. Moreover, because we fit `IGoR` models on predominantly productive TRB sequences, we consider only `IGoR`-generated sequences whose V and J segments are in-frame.

Figure 3.18b illustrates the `partis` model validation workflow as described in the Methods section. We first run `partis partition` on each fasta file of IGH sequence reads to obtain annotations for each sequence, as well as a directory containing model parameters for inference and simulation. We can then run `partis simulate` with these model parameters as input to generate a synthetic datset of IGH annotations. We subsample both the

experimental and simulated annotation datasets to unique clones. Then, we compare each IGH-relevant summary for the two resultant annotations datasets, yielding a divergence value for each summary.

3.9 Appendix E: Comparison of summary scores using `IgBLAST` *annotations*

Recall that for the standard `partis` model validation procedure, `partis` is used for both inference as well as simulation. Here we examine the influence of using the same tool for inference and simulation by using `IgBLAST` for inference, and comparing the annotations dataset output from `IgBLAST` to the corresponding simulations from `partis`. The workflow for this procedure is displayed in Figure 3.19, which is essentially the diagram in Figure 3.18b with an additional path describing the `IgBLAST`/Change-O pipeline. Change-O was used to parse the `IgBLAST` output, as well as partition the sequences into inferred clonal families [30].

Figure 3.20a shows the LRAD-data scores by summary when using `IgBLAST` for annotation and `partis` for simulation. Figure 3.20b shows the difference of each score in Figure 3.9a and each score in Figure 3.20a. Frequency polygons of summary distributions of three pairs of `IgBLAST`-annotated and `partis`-simulated datasets are shown in Figure 3.21. The plots show a high level of agreement for most summaries, with all but six of them differing by less than one units, and a strong majority of them close to zero. Where differences arise, this is likely the result of differences in how `partis` and `IgBLAST` perform annotations. For example, we see that the insertion length distributions highly disagree in scores. This is at least partially attributable to the star-tree assumption on which `partis` operates, which is prone to overestimate insertion lengths in an effort to better estimate the ultimate naive sequence. Indeed, examining the VD insertion length distribution shows that `IgBLAST` tends to assign a similar distribution to each dataset, whereas `partis` leads to more variable distributions with right skew due to the star-tree assumption. Moreover, if `IgBLAST` tends to assign a similar insertion length distribution to every dataset, then this will make it difficult for a simulator designed to match particular insertion lengths distributions to behave more like the `IgBLAST` distributions. Thus, inherent differences in annotation tools will certainly

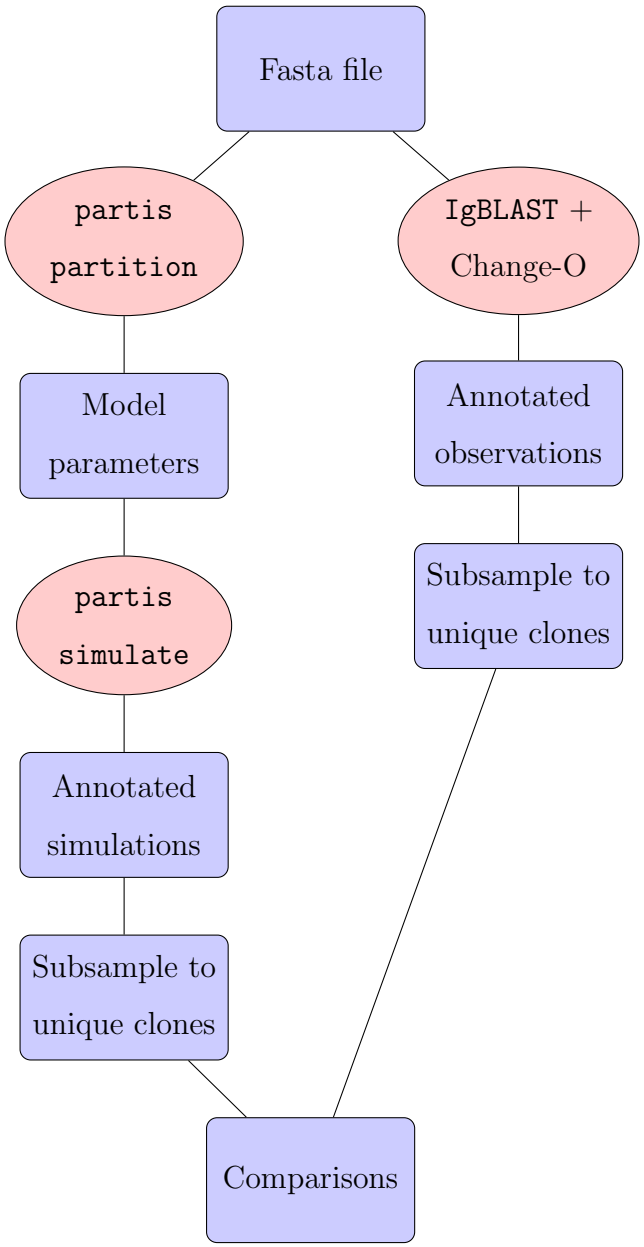
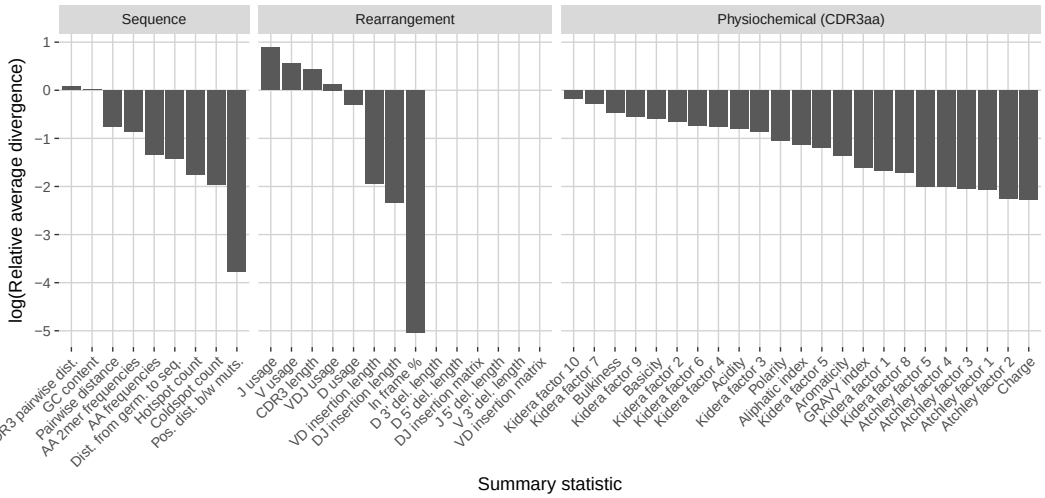
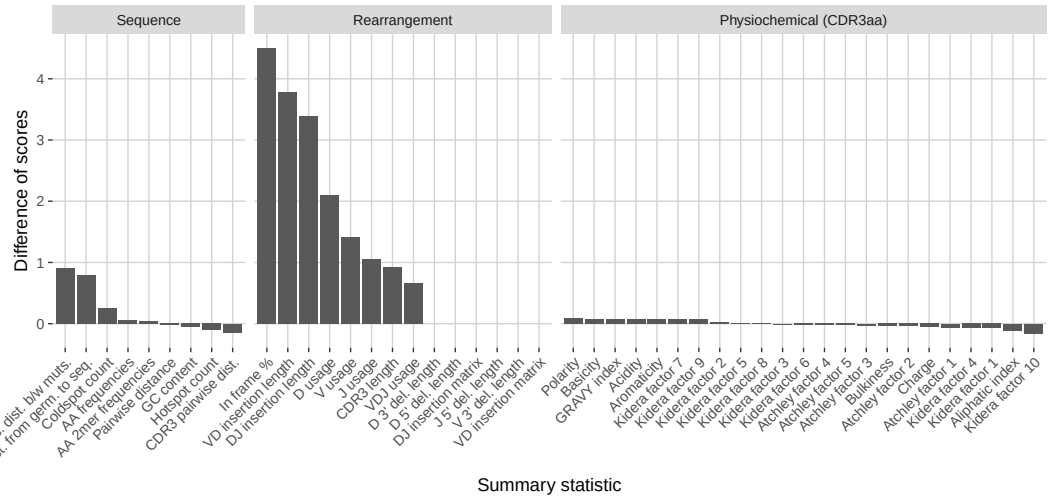


Figure 3.19: Workflow diagram for `partis` model validation when comparing `partis` and `IgBLAST` annotations to partis simulations



(a) Comparing divergences for `IgBLAST` annotations and `partis` simulations based on the same individual observed repertoires. We use the default germline databases in `IgBLAST` for consistency.



(b) Difference in LRAD values when using `IgBLAST` versus `partis` for annotations.

Figure 3.20: Summary scores for each statistic in the `partis` model validation experiment when comparing `partis` simulations to `IgBLAST` annotations. In both plots, a high score indicates a well-replicated statistic by the simulations. Summaries without a score are not readily available from AIRR-formatted `IgBLAST` output.

lead to differences in summary scores, regardless of how accurate either tool is. Hence, it is important to understand that a given annotations-based summary should be considered in the context of the tool which provided annotations, and not as a ground-truth summary of the actual gene usage/indel statistics.

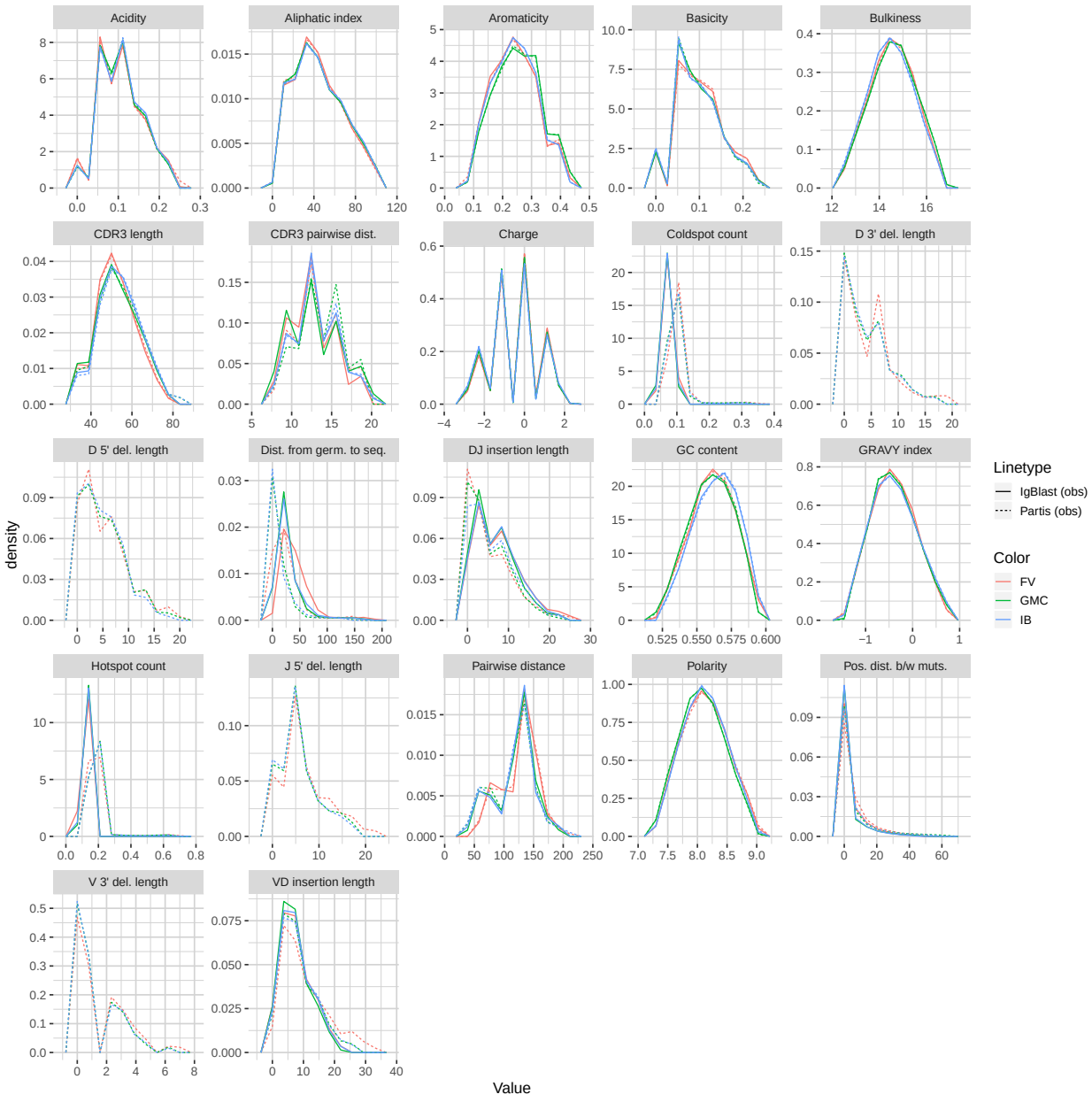


Figure 3.21: Summary distribution frequency polygons of `partis` versus `IgBLAST` annotations of experimental datasets from three donors at time point -1h.

Chapter 4

STATISTICAL COMPARISON OF T CELL RECEPTOR REPERTOIRES USING OPTIMAL TRANSPORT

4.1 Introduction

The arrival of high-throughput sequencing has given scientists the ability to sample TCR repertoires with unprecedented precision, paving the way for immense progress within the field of computational immunology. Often, this reduces to a situation wherein a researcher wishes to compare two TCR repertoire datasets and extract meaningful differences between them. For example, the pair of datasets could be samples of an individual's TCR repertoire before and after a vaccination, and the researcher might wish to determine the responding TCRs in the post-vaccination repertoire.

Most current methods of repertoire comparison involve reducing the TCR sequences into simpler summaries and then comparing these summaries, such as examining differences in gene usage frequencies and CDR3 sequences [33, 44, 15, 25, 11, 9, 54]. As comparing full CDR3 sequences can be highly involved, one approach is to simply compare CDR3 length distributions [47, 39]. These approaches fail to capture other interesting aspects of the germline-encoded regions such as gene similarity, as well as the relative importance of the CDRs and framework regions for TCR binding specificity. Alternatively, one can project a TCR repertoire onto a simpler space and compare values within the resultant embedding. For example, several studies have examined the distributions of $k$mer occurrences to classify TCR repertoires [75, 56, 14]. However, the space of $k$mer distributions is still very high-dimensional and discards important positional information within TCR sequences. Other authors instead look at t-SNE projections of repertoires [86], but this still incurs a loss of information and loses immunological meaning.

The more focused problem of inferring specificity from TCR sequences has been approached by tracking individual clones, comparing to a probabilistic model, or using epitope-specific machine-learning models. Tracking distributions of clonotypes over multiple time-points [61] typically requires at least three longitudinal datasets per individual. A recent technique addresses this issue by detecting regions within a single repertoire that are significantly enriched according to some baseline generative model [60]. While this is a substantial advance, the method is only as good as the underlying generative model of functional sequences. Other machine learning techniques build predictive models using labeled training data [27, 36, 35], although these techniques often require a specified antigen epitope, can be limited by the amount of publicly-available data, and rely on models that can be difficult to interpret.

We wish to overcome these drawbacks with a procedure that performs comparisons between two empirical repertoires in a fast, interpretable, and precise manner. Thinking of a sample TCR repertoire as an empirical distribution of observed sequences, the problem reduces to comparing two discrete probability distributions using some measure of statistical divergence. There are many commonplace methods for comparing discrete distributions, but these methods are hardly appropriate for TCR datasets, which comprise a very sparse sample from the very vast space of possible TCRs. One way to assuage this sparsity is to equip the sample space of TCR sequences with some metric which provides distributional comparison. While several such metrics on probability distributions have been established, we focus on a particular class of methods known as optimal transport, which boasts favorable theoretical and computational properties along with an intuitive interpretation. Moreover, while classical optimal transport methods are computationally intensive and often scale at least cubically with the number of statistical parameters, a recent extension uses Sinkhorn distances to constrain the underlying optimization problem to get tractable approximations with high accuracy [16].

In this report we apply the Sinkhorn approach along with TCRdist, a recently-created distance between TCRs [18], to formulate a nonparametric approach to the comparison of

TCR repertoires. We motivate our methods using the intuition underlying optimal transport, and demonstrate that our methods are able to identify clusters of TCR sequences that constitute biologically meaningful differences between repertoires using multiple case studies. We also describe and validate a randomization test to assess the whether our identified TCRs are significantly enriched in a target repertoire with reference to a source repertoire.

4.2 Materials and methods

4.2.1 An optimal transport formulation of TCR repertoire comparison

Optimal transport compares two probability distributions in terms of the total amount of "work" required to transform one probability distribution into the other. In this context, work is defined as the product of the probability mass (i.e., the normalized frequency of a value's occurrence) between objects in the joint sample space and the distance between them (according to some specified distance function). To illustrate, one might think of these distributions as soldiers on a battlefield: one can compare two distributions of soldiers by the minimal amount of overall work (total amount of marching among all soldiers) that is required to move them from one configuration to another. The strength of the optimal transport approach for this application is that it quantifies not only the minimal total amount of transport needed given a particular mapping of mass in one distribution to the other, it also returns a description of how the transport is performed. In our soldier example, this would be the particular marching orders for each soldier concerning how they should be dispatched into the second configuration.

Returning now to TCRs, we can consider each TCR to be a soldier. The "marching distance" is defined by TCRdist [18], so that the result of an optimal transport analysis of two repertoires is a mapping of TCRs in one repertoire to another in which similar TCRs are matched to one another. (Note that we can match part of one TCR to another TCR by assigning a fraction of the probability mass between them, so there is no difficulty in having repertoires of different sizes.) TCRs that have no close relative in another repertoire must

travel a long distance, which can be easily identified from the optimal mapping. That is, large values of the optimal transport matrix hint at some discrepancy between the underlying TCR distributions. This will in turn allow us to identify regions of significant difference between the two repertoires (Figure 4.1). This concept will underly our methodology to detect notable regional differences between two TCR repertoires.

In the remainder of this section, we briefly review discrete optimal transport and how it can be leveraged to compare TCR repertoires. We then derive a score to detect which individual TCRs appear to be enriched in a target repertoire with respect to a source repertoire using the optimal transport matrix. Using these scores, we develop a clustering procedure to extract local regions of high-scoring TCRs, as well as a procedure to infer sequence motifs that characterize these clusters. We also describe a randomization test to obtain statistical significance estimates for these scores. We conclude this section with a discussion of the repertoire datasets that are analyzed in the Results section.

4.2.2 Discrete optimal transport

Suppose we have two discrete probability distributions described by vectors $\mathbf{r} = (r_1, \ldots, r_n)^\top$, the probability masses assigned to objects $x_1, \ldots, x_n$, respectively, and $\mathbf{c} = (c_1, \ldots, c_m)^\top$, the probability masses assigned to objects $y_1, \ldots, y_n$, respectively, so that $\mathbf{r}$ and $\mathbf{c}$ contain non-negative entries, and both sum to one. We can consider the set of admissible couplings [42], encoded as joint probability matrices whose row-sums correspond to $\mathbf{r}$ and whose column-sums correspond to $\mathbf{c}$:

$$\mathbf{U}(\mathbf{r}, \mathbf{c}) := \left\{ \mathbf{P} \in \mathbb{R}_+^{n \times m} : \mathbf{P}\mathbf{1}_m = \mathbf{r} \;\&\; \mathbf{P}^\mathsf{T}\mathbf{1}_n = \mathbf{c} \right\}, \tag{4.1}$$

where $\mathbf{1}_k = (1, \ldots, 1)^\mathsf{T} \in \mathbb{R}^k$. In other words, we are considering all joint probability distributions whose marginal distributions correspond to $\mathbf{r}$ and $\mathbf{c}$. For a given matrix $\mathbf{P}$, we can interpret the entry p_{ij} as the amount of mass "assigned" or "transported" between the object x_i (which has r_i total mass) and object y_j (which has y_j total mass).

We formalize this in the language of measure theory which will provide us with a rigorous

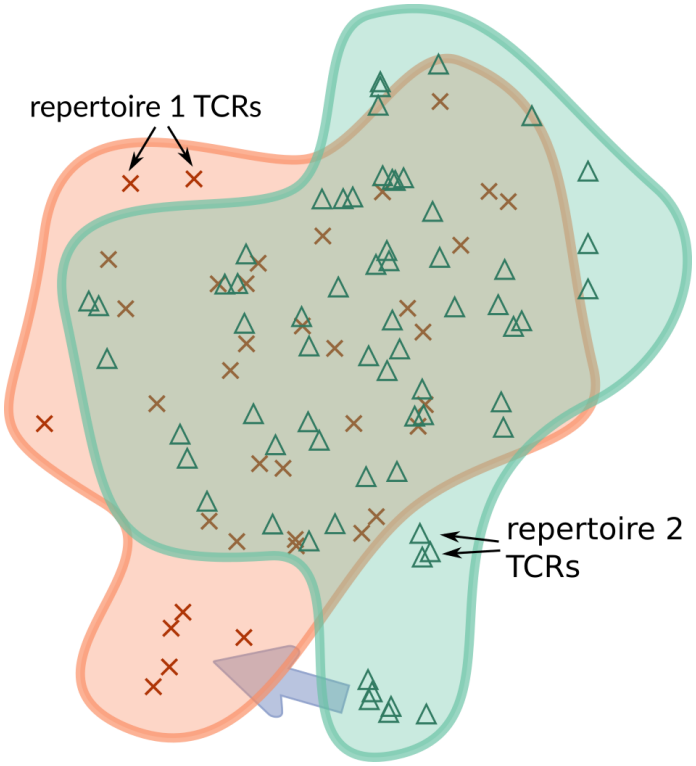
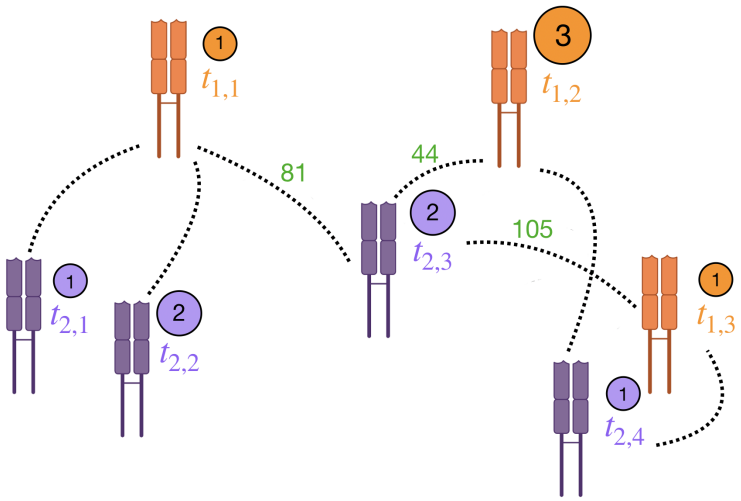


Figure 4.1: A schematic of TCR distribution comparison. Each symbol represents a TCR in an abstract space in which distance is defined via TCRdist [18], and the two regions represent two population repertoires of interest. Each repertoire is given its own color (here orange and green). The purple arrow shows that there are regions of these TCR distributions for the green repertoire that do not have a close equivalent in the orange repertoire, which will be identified by our optimal transport methods.

specification of our methods below. Let $\Sigma_k = \left\{\mathbf{a} \in [0,1]^k : \sum_{i=1}^k a_i = 1\right\}$ be the standard k-simplex. Let $\delta_x(\cdot)$ denote the Dirac delta measure centered on a fixed point x, which evaluates to 1 if the input is x and 0 otherwise [4]. Consider discrete probability measures $\mu(\cdot) = \sum_{i=1}^n r_i \delta_{x_i}(\cdot)$ and $\nu(\cdot) = \sum_{j=1}^m c_i \delta_{y_i}(\cdot)$ on respective sample spaces $\mathcal{X}$ and $\mathcal{Y}$, with $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, $\{y_1, \ldots, y_m\} \subset \mathcal{Y}$, where $\mathbf{r} = (r_1, \ldots, r_n)^\top \in \Sigma_n$ and $\mathbf{c} = (c_1, \ldots, c_m)^\top \in \Sigma_m$ denote the same vectors as above.



(a) A schematic of two TCR repertoires $R_1 = \{t_{1,1}, t_{1,2}, t_{1,3}\}$ and $R_2 = \{t_{2,1}, t_{2,2}, t_{2,3}, t_{2,4}\}$ residing in an abstract space defined by TCRdist. The circle adjacent to each TCR displays its clonotype abundance. TCRdist values are shown (in green) from $t_{2,3}$ to each of the TCRs in R_1, although a TCRdist value is defined between each pair.

$$\mathbf{D} = \begin{array}{c} \\ t_{1,1} \\ t_{1,2} \\ t_{1,3} \end{array} \begin{array}{c} \begin{array}{cccc} t_{2,1} & t_{2,2} & t_{2,3} & t_{2,4} \end{array} \\ \begin{pmatrix} 71 & 76 & 81 & 182 \\ 190 & 179 & 44 & 93 \\ 205 & 193 & 105 & 24 \end{pmatrix} \end{array} \quad \mathbf{r} = \begin{pmatrix} 1/5 \\ 3/5 \\ 1/5 \end{pmatrix} \quad \mathbf{c} = \begin{pmatrix} 1/6 \\ 2/6 \\ 2/6 \\ 1/6 \end{pmatrix}$$

$$\mathbf{P}^* = \begin{array}{c} \\ t_{1,1} \\ t_{1,2} \\ t_{1,3} \end{array} \begin{array}{c} \begin{array}{cccc} t_{2,1} & t_{2,2} & t_{2,3} & t_{2,4} \end{array} \\ \begin{pmatrix} 1/6 & 1/30 & 0 & 0 \\ 0 & 4/15 & 1/3 & 0 \\ 0 & 1/30 & 0 & 1/6 \end{pmatrix} \end{array}$$

(b) The mathematical objects that describe the setup illustrated in (a). Here, $\mathbf{D}$ is the matrix of pairwise TCRdist values, $\mathbf{r}$ is a vector of distribution mass values for each TCR in R_1, $\mathbf{c}$ is a vector of distribution mass values for each TCR in R_2, and $\mathbf{P}^*$ is the optimal transport matrix.

Figure 4.2: An illustration of our optimal transport formulation of TCR repertoire comparison.

For a given distance matrix $\mathbf{D} \in \mathbb{R}^{n \times m}$, the classical "Kantorovich" optimal transport problem seeks the solution of

$$L_{\mathbf{D}}(\mathbf{r}, \mathbf{c}) := \min_{\mathbf{P} \in \mathbf{U}(\mathbf{r}, \mathbf{c})} \langle \mathbf{D}, \mathbf{P} \rangle \tag{4.2}$$

where $\langle \mathbf{A}, \mathbf{B} \rangle := \sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij} b_{ij}$ for $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times m}$. That is, we compute the optimal transport matrix $\mathbf{P}$ which minimizes the sum of entrywise products of distance and probability mass. We interpret this as the total amount of "work" to move the mass of one distribution to another. Hence, this distance between probability distributions is often referred to as the Earth-mover's distance (EMD), and is also known as the Wasserstein metric. It is important to note that we are working with two notions of distance: the distance defined on the sample space between two objects and represented by the matrix $\mathbf{D}$, and the overall distance between the two full probability distributions defined by the EMD.

Unfortunately, computing the EMD of two discrete distributions scales as $\mathcal{O}(k^3 \log(k))$, where $k = \max(m, n)$, when no restrictions are placed on the metric d that parametrizes the distance matrix $\mathbf{D}$. Cuturi (2013) overcomes this by regularizing the entropy of the couplings $\mathbf{P}$ which drive the minimization [16]. In particular, they introduce the Sinkhorn distance

$$d_{\mathbf{D}}^{\lambda}(\mathbf{r}, \mathbf{c}) := \left\langle \mathbf{D}, \mathbf{P}^{\lambda} \right\rangle \tag{4.3}$$

where

$$\mathbf{P}^{\lambda} = \underset{\mathbf{P} \in \mathbf{U}(\mathbf{r}, \mathbf{c})}{\arg\min} \left\{ \langle \mathbf{D}, \mathbf{P} \rangle - \frac{1}{\lambda} h(\mathbf{P}) \right\} \tag{4.4}$$

and $h(\mathbf{P}) := -\sum_{i=1}^{d} \sum_{j=1}^{d} p_{i,j} \log(p_{i,j})$ is the Shannon entropy of $\mathbf{P}$. This regularization serves two main purposes. First, we can interpret the regularization term as an invocation of the principle of maximum entropy, which encodes the intuition that we should choose a distribution with the fewest assumptions (i.e., the most entropy) when considering a set of viable candidate distributions. In addition, the regularization introduces smoothing into the transport plan between $\mathbf{r}$ and $\mathbf{c}$ which leads to an approximate but much faster solution (the tuning parameter λ controls this speed-accuracy tradeoff). Cuturi (2013) shows that the

regularization term constrains the optimization region of admissible couplings $\mathbf{U}$ to a new region $\mathbf{U}_\alpha$ such that

$$\mathbf{U}_\alpha(\mathbf{r}, \mathbf{c}) = \left\{\mathbf{P} \in \mathbf{U}(\mathbf{r}, \mathbf{c}) : \mathrm{KL}\left(\mathbf{P} || \mathbf{r}\mathbf{c}^\mathsf{T}\right) \leq \alpha\right\}; \tag{4.5}$$

we recall this derivation in Appendix A. Thus, we can interpret the Sinkhorn distance as the result of minimizing the work to move one distribution to another while maintaining a relatively simple coupling, in the sense that its KL-divergence to the independent joint distribution (whose coupling is exactly $\mathbf{a}\mathbf{b}^\mathsf{T}$) is small.

4.2.3 Applying optimal transport to TCR repertoire comparison

For our purposes, the sample spaces $\mathcal{X}$ and $\mathcal{Y}$ discussed above will denote the same set of possible TCRβ sequences we can observe in a sample TCRβ repertoire. We define this set $\mathcal{X}$ to be all valid pairs t of TRBV genes and CDR3 amino acid sequences, e.g., $t =$ (TRBV27*01, `CASSLGTGQYEQYF`). We use "empirical repertoire", or simply "repertoire", to mean a repertoire sample $R = (t_1, \ldots, t_n)$ containing n (TRBV, CDR3aa) pairs along with corresponding abundances $(a_1, \ldots, a_n) \in (\mathbb{Z}^+)^n$. Relating this to the notation of the previous section, we have $x_i = t_i$ as the sample points, and $c_i = a_i / \sum_k a_k$ as the corresponding mass coefficients; analogous quantities are used to define y_i and r_i for a second repertoire.

For our distance function d, we use a version of TCRdist, a similarity-weighted mismatch distance between potential pMHC-contacting loops of two given TCRs [18]. The version we use involves only the TRB chain as single-cell data detailing the joint TRA and TRB chains per cell is of limited availability. Moreover, our methods omit the TRBJ gene beyond the segment specified in the CDR3, as we believe incorporating the non-CDR3 J sequence would yield negligible inferential improvements at the expense of increased runtime and complexity. Specifically, if $\mathbf{a}_1^c$ and $\mathbf{a}_2^c$ are the amino acid sequences of CDR c for TCRs 1 and 2 respectively, then

$$\mathrm{TCRdist}(t_1, t_2) := \sum_{c \in \mathrm{CDRs}} \sum_{i \in p} w(c)\, \mathrm{AAdist}((a_1^c)_i, (a_2^c)_i; c) \tag{4.6}$$

where:

- $\text{CDRs} := \{\text{CDR1}\beta, \text{CDR2}\beta, \text{CDR2.5}\beta, \text{CDR3}\beta\}$
- $w(c) := \begin{cases} 3, & c = \text{CDR3}\beta \\ 1, & \text{else} \end{cases}$
- $\text{AAdist}(a_1, a_2; c) :=$
$$\begin{cases} 0, & a_1 = a_2 \\ 8, & \text{exactly one of } a_1 \text{ or } a_2 \text{ is `-'}, \quad c = \text{CDR3}\beta \\ 4, & \text{exactly one of } a_1 \text{ or } a_2 \text{ is `-'}, \quad c \neq \text{CDR3}\beta \\ \min(4, 4 - \text{BLOSUM62}(a_1, a_2)), & \text{else} \end{cases}$$
- BLOSUM62 is a widely-used substitution matrix for amino acids that was estimated using log odds scoring of frequencies from a large and trusted alignment database (called BLOCKS) [32].

Figure 4.2 illustrates a simple example of this setup with two TCR repertoires spread out in an abstract space, where the distance between TCRs is defined by TCRdist. The three orange TCRs spanning in the upper-right of the image belong to some repertoire R_1, and the four purple TCRs spanning the bottom of the image belong to some other repertoire R_2. Their respective abundances are displayed in the adjacent circles. The dotted lines between TCRs represent the TCRdist values between them, with a few distances are shown in green for $t_{2,3}$ (the rest are omitted for brevity).

4.2.4 Effort and loneliness

Let $\widehat{\mathbf{P}}$ be an estimate of the optimal transport matrix (such as the Sinkhorn approximation $\mathbf{P}^{\lambda}$) between repertoires R_1 and R_2, with corresponding distance matrix $\mathbf{D}$. Define the "effort" matrix as the Hadamard product of $\widehat{\mathbf{P}}$ and $\mathbf{D}$,

$$\mathbf{E} := \widehat{\mathbf{P}} \odot \mathbf{D} = \left(\widehat{\mathbf{P}}_{ij}\mathbf{D}_{ij}\right)_{1 \leq i \leq n, 1 \leq j \leq m} \in \mathbb{R}^{n \times m}. \tag{4.7}$$

For any $t_i \in R_1$, $t_j \in R_2$, define

$$\text{PairedEffort}(t_i, t_j) := \mathbf{E}_{ij} \equiv \widehat{\mathbf{P}}_{ij} \, \text{TCRdist}(t_i, t_j) \tag{4.8}$$

which can be interpreted as the entrywise amount of "effort" or "work" used in the optimal transport matrix to move the mass at TCR t_i to TCR t_j.

We wish now to define a score that quantifies the isolation of a given TCR in one repertoire relative to some reference repertoire, where a high score indicates that the TCR is characteristic of its own repertoire but unusual with respect to the reference repertoire. A naive score for a given TCR $t_2 \in R_2$ with respect to all the TCRs in R_1 is

$$\text{TotalLoneliness}(t_2 \mid R_1) = \sum_{t_1 \in R_1} \text{PairedEffort}(t_1, t_2) \tag{4.9}$$

which reduces to a sum of the column of $\mathbf{E}$ that indexes t. A drawback of Equation (4.9) is that there might be outlier TCRs in R_1 that also look lonely to R_2 as a result, and would yield a high loneliness value. We are interested in a more differential version of loneliness: a TCR that is lonely with respect to a different repertoire R_2 but not very lonely with respect to its own repertoire R_1 (as illustrated in Figure 4.1). This would suggest that t is indicative of some feature of R_1 not present in R_2 (e.g. a vaccination).

Instead, we consider the cumulative total loneliness around a neighborhood of size δ around each t:

$$\text{RelativeLoneliness}(t_2 \mid R_1; \delta) := \sum_{t' \in \mathcal{B}_\delta(t_2)} \text{TotalLoneliness}(t' \mid R_1) \tag{4.10}$$

where $\mathcal{B}_\delta(t_2) = \{t' : \text{TCRdist}(t_2, t') < \delta\}$. This reduces to a sum of all columns indexing some TCR in $\mathcal{B}_d(t_2)$. The *relative loneliness* (4.10) will be small when t is an outlier in both repertoires, since there won't be many neighboring TCRs t' in the ball. Further, (4.10) will be large for a TCR with many neighbors in R_1 but few in R_2, since there will be many TCRs all with relatively high transport values. Because of these properties, we use (4.10) as the core scoring mechanism for our methods and analyses presented here, and will simply call this value *loneliness*.

The expression in (4.10) relies on a neighborhood radius parameter δ which requires tuning: setting δ too small will lead to unstable results as there will rarely be neighbors for a given t, and setting δ too large will assign too many neighbors to each TCR and grossly inflate the scores. However, we show that (4.10) consistently behaves better than (4.9) as an indicator of lonely groups of TCRs for each sensible radius δ.

4.2.5 Clustering

We wish to identify regions of similar TCRs that appear to be enriched in their own repertoire relative to a reference repertoire using our loneliness scores defined in (4.10). Suppose we have computed loneliness values for TCRs $t_1, \ldots, t_m \in R_2$ with respect to a source repertoire R_1. We first describe a procedure to identify the "loneliest" cluster in R_2, and then show how iterating this scheme will allow us to compute any remaining lonely clusters.

Start with the loneliest TCR $t_{\max}$ and some step size s (by default, we choose $s = 5$). In each iteration i, step out s units of TCRdist from previous radius r_{i-1}, and compute the mean loneliness of all TCRs within r_{i-1} and $r_{i-1} + s$ units of $t_{\max}$:

$$S_i := \{\tau : r_{i-1} \leq \text{TCRdist}(t_{\max}, \tau) < r_{i-1} + s\} \tag{4.11}$$

$$m_i := \frac{1}{|S_i|} \sum_{t \in S_i} \text{RelativeLoneliness}(t \mid R_1), \tag{4.12}$$

In the first iteration, we are just looking at the mean loneliness in the TCRdist ball of s units around $t_{\max}$. After that, each iteration looks at the mean loneliness of the semi-closed annulus of width s surrounding the previous region. Once we have have computed values of (4.12) for our full set of radii (e.g., $r = 0, 5, 10, \ldots, 200$), we can examine the relationship of mean loneliness vs radius to see if there is a breakpoint $r_{\text{breakpoint}}$ at which loneliness is no longer high. If a breakpoint is detected, we simply define our cluster as those TCRs which fall within $r_{\text{breakpoint}}$ units of TCRdist to $t_{\max}$. This procedure is illustrated in Figure 4.3.

The above procedure yields a cluster of the "loneliest" TCRs of our full repertoire. To identify further lonely clusters, we simply identify the loneliest TCR that has not yet been clustered, and apply the same procedure. We can iterate until a sensible stopping point, such

as when a breakpoint is unable to be estimated (discussed further in the next paragraph). The complete algorithm is formalized in Algorithm 4.

To estimate breakpoints in the above procedure, we perform a segmented regression, also known as a piecewise regression, of the mean annulus loneliness values m_i on the set of radii r_i. Univariate segmented regression assumes that the relationship between the response and predictor is described by a pair of differing line segments across two separate intervals separated by a breakpoint ρ, with the line segments coinciding at ρ. For our response M, the mean annulus loneliness, with a fixed cluster radius r as our predictor, our model becomes

$$\mathsf{E}\left[M \mid r\right] = \beta_0 + \beta_1 r + \beta_2 (r - \rho)_+ \tag{4.13}$$

$$= \begin{cases} \beta_0 + \beta_1 r, & r \leq \rho \\ (\beta_0 - \beta_2 \rho) + (\beta_1 + \beta_2) r, & r > \rho \end{cases}. \tag{4.14}$$

Here, $x_+ = x$ if $x > 0$, and 0 otherwise. The least squares method yields estimates $\widehat{\beta}_0, \widehat{\beta}_1, \widehat{\beta}_2$, and $\widehat{\rho}$ of the corresponding model parameters $\beta_0, \beta_1, \beta_2$, and ρ. We can then use $\widehat{\rho}$ as our estimate of the breakpoint radius $r_{\text{breakpoint}}$, and define our cluster as $\{t : \text{TCRdist}(t, t_{\max}) \leq \widehat{\rho}\}$. We use the `segmented` R package to estimate the coefficients of these models for our analyses [49]. Note that a breakpoint may be unable to estimated if the regression assumptions are not met well (there is no strong evidence of an "elbow" from the data), the initial breakpoint is not close enough to the "true" breakpoint, or there are not enough data to estimate the model parameters. Nonetheless, we verify the desired behavior of this regression approach to estimate cluster radii using a set of typical TRB datasets in Appendix A.

4.2.6 Motif inference

Given a cluster of TCRs, we would like to infer a motif describing their sequence homology. This is exacerbated my the fact that CDR3 lengths can vary by TCR. One solution is to generate a regular expression that describes the sequences in the cluster, either manually or through some algorithm. However, this can be messy and difficult, and raw regexes are not always easily interpretable by eye.

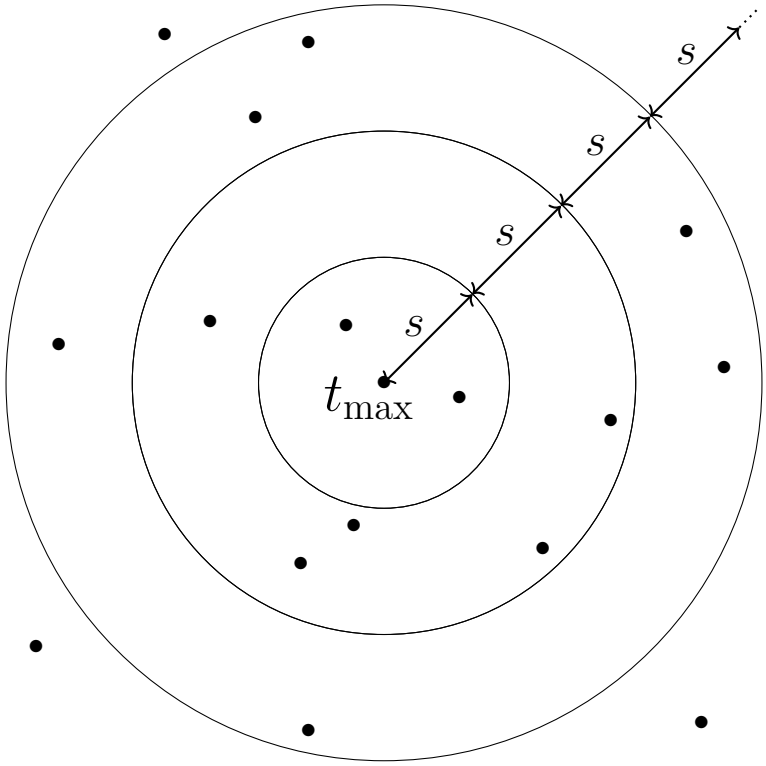


Figure 4.3: A schematic of our clustering procedure in Algorithm 4. Each point is a TCR portrayed in an abstract 2-D space, where the distance between points is determined by TCRdist. Our procedure starts by identifying the maximally lonely TCR $t_{\max}$ according to Equation (4.10). In each iteration, we step out s units of TCRdist, and compute the mean loneliness of all TCRs within the annulus defined by the current and previous radii (or ball in the first step). By construction of Equation (4.10), we expect the loneliness values to steadily decrease as we move away from $t_{\max}$, until we arrive at a radius where the loneliness values have stabilized. This "breakpoint radius" thus defines the radius of our cluster.

Algorithm 4 Computing clusters of lonely TCRs

Input: Repertoires R_1 and R_2, radius step size $s > 0$, maximum cluster count $C \geq 1$

Output: Vector of TCR clusters $\mathbf{c} = [c_1, \ldots, c_C]$ of R_2

$\mathbf{c} = []$
`keep_clustering` $\leftarrow$ True
while `keep_clustering` **do**
    $R_2^{\text{sub}} \leftarrow \{t_2 \in R_2 : t_2 \text{ is not clustered}\}$ ▷ get all un-clustered TCRs
    $t_{\max} \leftarrow \max_{t_2 \in R_2^{\text{sub}}} \text{RelativeLoneliness}(t_2; R_1)$ ▷ find lonelinest un-clustered TCR
    $r_{\text{prev}} \leftarrow 0$
    $r_{\text{current}} \leftarrow s$
    while $r \leq r_{\max}$ **do**
        $S \leftarrow \{t : r_{\text{prev}} < \text{TCRdist}(t, t_{\max}) \leq r_{\text{current}}\}$ ▷ define annulus
        $\ell_r \leftarrow \text{mean}_{t \in S} \text{RelativeLoneliness}(t; R_1)$ ▷ compute mean loneliness over all TCRs in annulus
        $r_{\text{prev}} \leftarrow r_{\text{current}}$
        $r_{\text{current}} \leftarrow r_{\text{current}} + s$ ▷ update annulus radii
    estimate breakpoint $r_{\text{breakpoint}}$ of ℓ_r vs. r
    if $r_{\text{breakpoint}} =$ `NULL` **then** ▷ we were unable to estimate a cluster radius, so terminate
        `keep_clustering` $\leftarrow$ False
    else ▷ we succeeded in detecting a cluster radius
        append $\{t : \text{TCRdist}(t, t_{\max}) \leq r_{\text{breakpoint}}\}$ to $\mathbf{c}$ ▷ append our cluster to the running vector
        if $|\mathbf{c}| = C$ **then** ▷ we have detected the maximum # of clusters
            `keep_clustering` $\leftarrow$ False
return c

Instead, we appeal to profile-HMMs, which describe the emission probabilities of amino acids at each position along a sequence while explicilty modeling position-specific insertion and deletion probabilities [21]. Profile-HMMs are readily implemented in the HMMer package (`http://hmmer.org/`). Estimating a profile-HMM π allows for several benefits:

- We can model an arbitrary cluster of TCR sequences with π without worrying about CDR3 length differences

- We can query other sequences against this profile to assess their homology to the cluster in a statistically rigorous manner. In particular, for any animo acid sequence σ, we can first compute a log-odds ratio "bit score" comparing the likelihood of observing σ from π to the "null" likelihood of observing σ from an independent, identically distributed random sequence model π_0. Then, we can compute an E-value which is based on the number of hits expected to achieve this bit score or greater by chance, i.e. if the search had instead been done using π_0. We further use these E-values to define "hard" motif memberships via the indicator $\mathbb{1}(e < e_{\text{crit}})$, for some specified critical threshold e_{crit}.

- We can readily visualize these profiles via enhanced sequence logos that display indel characteristics [71, 80]

As HMMer requires aligned sequences in order to build HMMs, we use MAFFT, a fast and popular tool that constructs a multiple-sequence alignment of a given set of query sequences [37], whenever we need such alignments for HMMer.

4.2.7 Significance estimates

We wish to attach significance estimates to our loneliness scores to determine whether high observed scores are improbable due to chance alone. For this, we perform the following randomization test. For trial $j \in \{1, \ldots, J\}$, randomly re-label the TCRs in R_1 and R_2 to get trial repertoires $\widetilde{R}_1^{(j)}$ and $\widetilde{R}_2^{(j)}$. Under the null hypothesis that R_1 and R_2 are samples

from the same (abstract) population repertoire of TCRs, each of these trial repertoires $\widetilde{R}_1^{(j)}$ and $\widetilde{R}_2^{(j)}$ will have the same sampling distribution as R_1 and R_2. We then compute (4.10) for each TCR in $\widetilde{R}_2$ with respect to $\widetilde{R}_1$. After J trials, we have obtained null distributions of loneliness scores for each $t \in R_2$. We can now compare the observed loneliness ℓ_{obs} of a given TCR t to its null distribution, rejecting the null that t could have been sampled from either R_1 or R_2 if ℓ_{obs} is sufficiently high (e.g., higher than the $1-\alpha$ quantile for a specified level α.

There are a couple of caveats to this approach. First, when we relabel TCRs during a given trial j, only some of the TCRs from R_2 will be present in $\widetilde{R}_2^{(j)}$. We handle this by maintaining score distributions only for the TCRs originally present in R_2, and appending trial scores for only those TCRs to their running distributions (thus ignoring the scores of TCRs in $\widetilde{R}_2^{(j)}$ that originally belonged to R_1). After J trials, we downsample these score distributions to the size of the smallest distribution. If there is a particular minimal sample size we desire for all of the score distributions, we could simply add a check in our routine to stop only when this minimal sample size has been attained (although this would lead to an increased runtime). Further, there may be substantial correlation between loneliness scores or some other intrinsic dependence between our TCR-specific hypothesis tests which could influence p-values derived from our randomization test, although this could be addressed with a controlling procedure.

4.2.8 Data

The following TCRβ repertoire datasets are used in the above analyses.

The majority of our analyses involve TCRβ repertoires collected from 23 genetically identical mouse lab strains [70]. For each mouse subject, three repertoires were sampled, corresponding to their CD4$^+$, CD8$^+$, and double negative (DN) intraepithelial lymphocyte (IEL) repertoires. Thus, there are $23 \times 3 = 69$ total IEL mouse datasets, to which we collectively refer as the *IEL data*. We will typically abbreviate CD4$^+$ as "CD4", and CD8$^+$ as "CD8". For brevity, we will define the collection of datasets for a given IEL type as a

subscripted $\mathcal{R}$. For example, $\mathcal{R}_{\text{CD4}}$ denotes the collection of 23 CD4$^+$ repertoires. Each repertoire consists of TCR sequences described by a V gene and CDR3aa pair (i.e., J genes are excluded from the analysis).

Our final analysis examines TCRβ repertoires collected from six human donors before and after an immunization with live yellow fever virus (YFV) vaccine [61]. Samples were taken from each donor at multiple timepoints: 7 days prior to vaccination ($-$7d), the day of vaccination (0d), 15 days following vaccination (+15d), and 45 days (+45d) following vaccination. This yields $6 \times 4 = 24$ human YFV datasets, to which we will collectively refer as the *YFV data*. Each repertoire is filtered to the 1,000 most abundant clones. As for the IEL data, each repertoire consists of TCR sequences described by a V gene and CDR3aa pair.

4.3 Results

We have defined a "loneliness" measure which captures TCRs that are characteristic of their own repertoire but unusual with respect to a reference repertoire. We have also presented a procedure to obtain clusters of sequences without equivalents in the reference repertoire based on these loneliness scores. In this section we apply our loneliness and clustering methodology to the IEL data and the YFV data, and show that our clusters capture meaningful differences between repertoires that we know are sampled from distinct populations.

4.3.1 Consistent loneliness dynamics across biological replicates of IEL mice

In this section we examine the behavior of our loneliness scores defined by (4.10) in the context the IEL data described above. The IEL data contain TCR repertoires of three distinct cell types, referred to as CD4, CD8 and DN (double negative) cells, from 23 genetically identical mice. These cell types differ in their expression of certain receptor proteins and their interactions with other cells, which impacts the binding properties of their corresponding TCR repertoires. Thus, we expect there to be meaningful differences in their respective TCR sequence distributions. In our analysis, we will focus on identifying regions of the DN

repertoire that are characteristic of the DN repertoire, but are unusual with respect to to the CD4 repertoire. This will allow us to use the CD8 repertoire as a useful comparison set since it will not influence the loneliness scores. Nonetheless, analogous analyses could be performed between any two cell types, with the third cell type available for comparison.

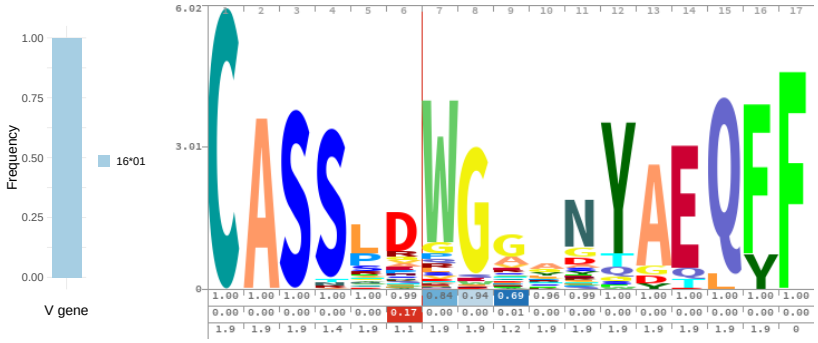
This data set has 23 sampled biological replicates for for each cell type, which allows us to account for the inherent biological variability of observing a given TCR in a sample. This provides us with a robust representation of each of the CD4, CD8, and DN population repertoires for our comparisons. In particular, we can get a sense of the overall differences between the DN and CD4 TCRβ sequences by combining each of the respective sets of repertoires into two large, representative datasets. Specifically, we concatenate $R_{\text{DN-1}}, \ldots, R_{\text{DN-23}}$ to obtain a combined DN repertoire $R_{\text{combined-DN}}$, and we concatenate $R_{\text{CD4-1}}, \ldots, R_{\text{CD4-23}}$ to obtain a combined CD4 repertoire $R_{\text{combined-CD4}}$.

Next, we compute RelativeLoneliness$(t; R_{\text{combined-CD4}})$ for each $t \in R_{\text{combined-DN}}$ to score each DN TCR t given the landscape of CD4 TCRs represented by $R_{\text{combined-CD4}}$. We also apply Algorithm 4 to $R_{\text{combined-DN}}$ with these loneliness scores to compute the top several lonely clusters. These clusters are constructed to be centered around the most lonely TCRs, and to encompass the surrounding regions of similarly lonely TCRs within an estimated TCRdist cutoff. We expect these top clusters to represent regions of significant difference between the CD4 and DN repertoires, since they should by construction contain the lonelinest TCRs that reside in sufficiently dense regions of the landscape obtained from the combined DN repertoires. We will refer to these top three loneliest clusters as the OT-Tremont, OT-Revere, and OT-Ida clusters, respectively. The OT-Tremont and OT-Revere clusters are so named because of their high similarities to the Tremont and Revere clusters described in Figure 5M of [70]. While the authors of [70] present detailed motif specifications, the Tremont cluster is dominantly characterized by the GT[VI]SNERLFF CDR3βaa motif, and the Revere cluster consists of a TRBV16 gene paired with a dominant DWG CDR3βaa motif. The OT-Ida cluster represents a novel TCR motif specification to the best of the authors' knowledge.

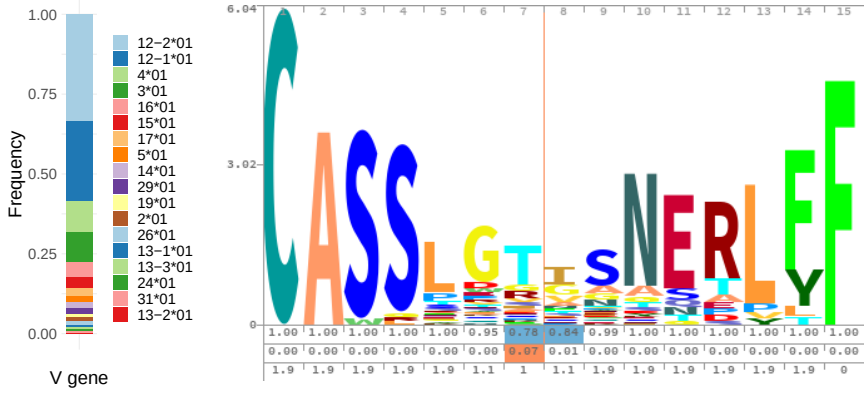
V gene usage and CDR3aa motifs for our three clusters are visualized in Figure 4.4. Each CDR3aa motif is visualized with a profile-HMM sequence logo obtained from Skylign [80]. The height of each stack is proportional to the level of conservation at that position, and the height of each amino acid within a stack is proportional to the probability of observing that amino acid at that position. The first row of numbers below the sequence logo displays each position's occupancy, or the probability of observing a non-gap character at that position (so that $(1 - \text{occupancy})$ gives the position's deletion probability). The second row displays the insertion probabilities at the respective positions, so that the kth value represents the probability of an insertion between positions k and $k+1$. The third row displays the expected insertion lengths of an insertion following position k, if an insertion exists.

Each cluster has distinctive features which suggest conservation of particular V genes and/or CDR3 amino acid motifs. The OT-Tremont cluster has the strictest V gene profile, containing only the TRBV16*01 gene. It also seems to include a conserved subsequence roughly spanning positions 5-8 in the sequence logo, as well as positions 11-17 which likely correspond to a stringent J gene specification. The OT-Revere cluster has a relatively loose V gene profile, containing 18 TRBV genes total, though the TRBV12-1*01 and TRBV12-2*01 genes comprise the majority of TRBV genes in this cluster. However, this cluster seems to have notable levels of conservation across most or all of the CDR3 sequence, with position 8 being the only one without a clearly dominant amino acid. The OT-Ida cluster has a strict V gene profile, with over 90% of the sequences consisting of the TRBV12-1*01 or TRBV12-2*01 genes. There is also evidence of varying levels of conservation throughout the CDR3, with only a few positions (6, 9, 10) lacking a dominant amino acid.

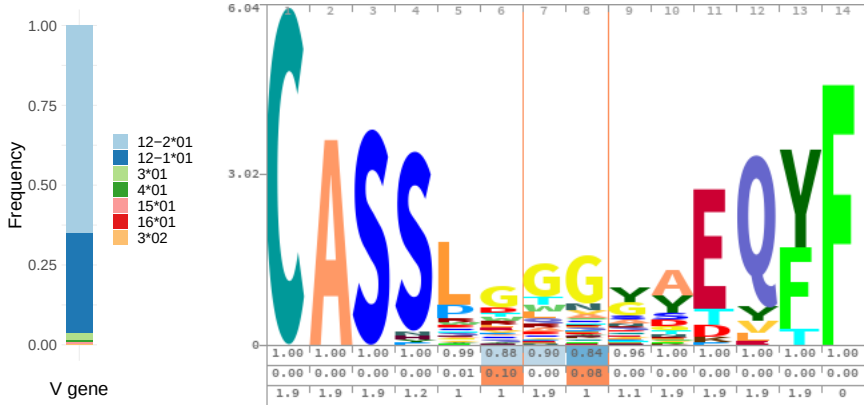
These automatically-generated clusters are able to capture regions of the DN repertoires that are distinguished from the CD4 repertoires. Figure 4.5 presents several plots that examine statistics of these clusters across the set of DN repertoires. To see how often each cluster is observed among the different individual DN, CD4, and CD8 repertoires, we can plot frequency polygons of each cluster prevalence empirical distribution (Figure 4.5a), where



(a) Visualization of the OT-Tremont cluster.



(b) Visualization of the OT-Revere cluster



(c) Visualization of the OT-Ida cluster

Figure 4.4: Visualizations of TRBV gene frequency statistics and CDR3aa sequence logos for the top three lonely clusters of the combined repertoire analysis.

the empirical prevalence of a cluster m within a repertoire R is defined as

$$\widehat{\text{Prevalence}}(c) := \widehat{\Pr}_{T\sim R}(T \in c) \tag{4.15}$$

$$= \frac{1}{|R|}\sum_{t\in R} \mathbb{1}(t \in c). \tag{4.16}$$

We see that each cluster tends to have higher prevalences in DN distributions compared to CD4 distributions, which indicates that these clusters are enriched in the DN population with respect to the CD4 population. For the OT-Tremont cluster, the prevalence is almost always zero for CD4 repertoires yet nonzero for each DN repertoire. For the OT-Revere and OT-Ida clusters, there are more nonzero prevalences in the CD4 repertoires, but consistently higher prevalences in the DN repertoires. Interestingly, the OT-Tremont and OT-Revere clusters tend to have similar prevalences among the CD4 and CD8 repertoires, whereas the OT-Ida cluster tends to have similar prevalences among the CD8 and DN repertoires. This matches the intuition behind our scores defined by (4.10), which seeks to identify regions of enrichment of DN repertoires with respect to CD4 repertoires, and not with respect to any arbitrary null distribution.

To get a sense of the loneliness dynamics of these clusters that does not rely on the scores obtained from the combined repertoires above, we performed the following experiment. For a given DN repertoire, we define the background set as all other DN repertoires, and the foreground set as all CD4 repertoires. The idea is that there will be intrinsic variability, or "background noise", that can be observed between repertoires of a common cell type, whereas two different cell types will also possess "foreground" variability that corresponds to biological differences between the repertoires. For each DN repertoire R_{DN}, and each $t \in R_{\text{DN}}$, compute the following background and foreground scores:

$$\text{bg-score}(t) := \frac{1}{|\mathcal{R}_{\text{DN}}| - 1}\sum_{R\in\mathcal{R}_{\text{DN}}\setminus R_{\text{DN}}} \text{RelativeLoneliness}(t \mid R) \tag{4.17}$$

$$\text{fg-score}(t) := \frac{1}{|\mathcal{R}_{\text{CD4}}|}\sum_{R\in\mathcal{R}_{\text{CD4}}} \text{RelativeLoneliness}(t \mid R) \tag{4.18}$$

We can interpret (4.17) as the average relative loneliness of a given DN TCR with respect to the background set of all other DN repertoires, and (4.17) analogously but with respect to the foreground set of all CD4 repertoires. We expect these averages to give stable estimates of how lonely this TCR looks compared to either of these two populations.

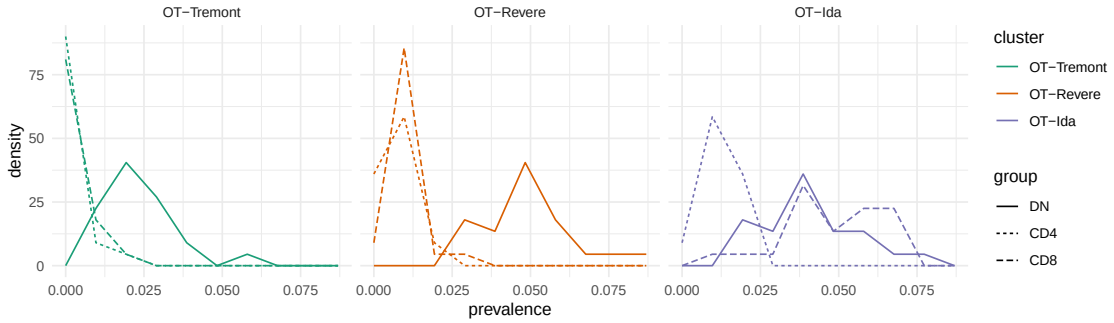
Distributions of scores obtained from (4.18) and (4.17), stratified by cluster, are shown in Figure 4.5b, in the top and bottom panels respectively. We see that these scores of each specified cluster tend to be higher than TCRs without a specified cluster for the background set, and these scores become amplified in the foreground set. This indicates that TCRs belonging to these clusters consistently have higher loneliness values compared to CD4 repertoires versus DN repertoires, as desired.

In addition to looking at the absolute loneliness values above, we can also examine the relative strength of each cluster's loneliness values within a given repertoire. One way to do this is to look at the empirical cumulative distribution function (ECDF) values of these scores by cluster, where a given fg-score ECDF, for example, is defined as

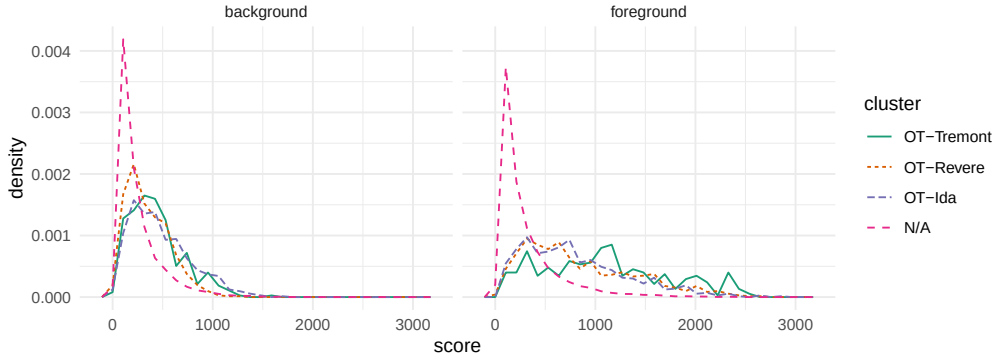
$$\begin{aligned} \text{ECDF}(s \mid R) &= \widehat{\Pr}_{T \sim R} \left(\text{fg-score}(T) \leq s \right) && (4.19) \\ &= \frac{1}{|R|} \sum_{t \in R} \mathbb{1} \left(\text{fg-score}(t) \leq s \right). && (4.20) \end{aligned}$$

A higher ECDF implies that the score is higher than most other observed scores; stratifying this by cluster will illustrate which clusters tend to have TCRs with higher loneliness values than the average TCR. We find that while TCRs in these three clusters tend to have higher ECDF scores than other TCRs in the background set, the ECDFs for these clusters become notably higher in the foreground set (Figure 4.5c).

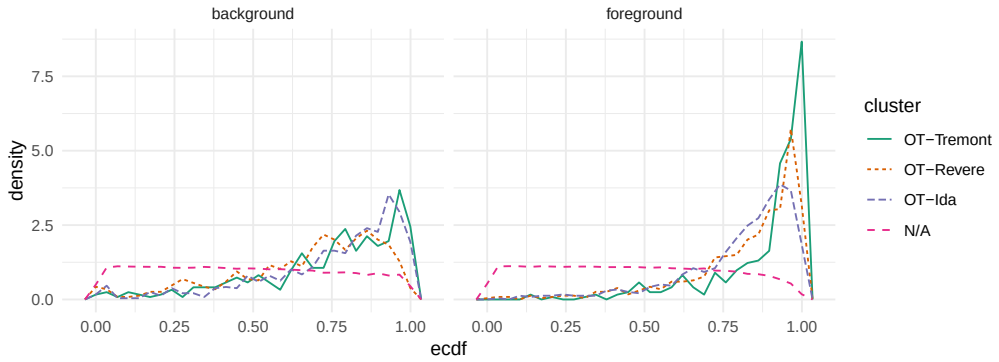
The results above demonstrate that the three clusters identified by our algorithm applied to the combined repertoires $\mathcal{R}_{DN}$ and $\mathcal{R}_{CD4}$ have amplified prevalences in the individual DN repertoires with respect to the CD4 repertoires, and yield consistently high loneliness scores across the individual replicate repertoires. This indicates that our algorithm is able to detect clusters of TCRs which seem to be differentially enriched between the subpopulations in question.



(a) Distributions of cluster prevalence across repertoires, stratified by cluster and cell type.



(b) Distributions of relative loneliness scores across repertoires, stratified by cell type group (background/foreground) and cluster.



(c) Distributions of relative loneliness ECDF values across repertoires, stratified by cell type group (background/foreground) and cluster.

Figure 4.5: Plots of several statistics that describe the across-repertoire cluster dynamics.

4.3.2 Validating randomization test scores with biological replicates

The randomization test framework presented earlier aims to determine how statistically significant an observed loneliness score is compared to what we would expect under a null model of having no significant differences between repertoires. The efficacy of this test depends on how accurately the randomization distributions replicate the dynamics of two repertoires that are truly sampled from the same population. We can benchmark this using the IEL biological replicates, since these replicates are samples from a common population of genetically independent mice; this allows us to quantify the statistical characteristics of the resultant TCR distributions, and consequently, the loneliness score distributions. If the statistical characteristics of these "replicate" loneliness distributions approximately match the statistical characteristics of the "randomization" loneliness distributions, this gives us a high degree of confidence in our testing procedure and significance estimates.

We apply the randomization test framework described in the methods section below to the IEL replicates as follows. First, we identify the largest DN TCRβ repertoire R_{DN} (subject #15; 1,737 sequences) and largest CD4 TCRβ repertoire R_{CD4} (dataset # 17; 864 sequences). We chose the largest dataset in hopes of obtaining the most stable parameter estimates, although relative and absolute sample sizes did not seem to majorly contribute to the behavior across various combinations of repertoires. For each $t \in R_{\text{DN}}$, we compute the observed score $s_{\text{obs}} = \text{RelativeLoneliness}(t; R_{\text{CD4}})$ using (4.10). Then, we compute the distribution of score values for t across the "background distribution" of all other DN repertoires,

$$S(t) := \{\text{RelativeLoneliness}(t; R') : R' \in \mathcal{R}_{\text{CD4}} \setminus R_{\text{CD4}}\}. \tag{4.21}$$

Since the set of CD4 repertoires are biological replicates, we can use $S(t)$ as a proxy for the true sampling distribution of s_{obs}. In particular, we can compute a "replicate" z-score

$$z(t) = \frac{s_{\text{obs}} - \text{mean}\,(S(t))}{\text{stddev}(S(t))} \tag{4.22}$$

to quantify how surprising the observed s_{obs} is with respect to the replicate null distribution (this also allows for the computation of p-values).

Next, we apply our randomization procedure to the same $(R_{\text{DN}}, R_{\text{CD4}})$ pair, to get a randomization distribution of score values

$$S^\star(t) := \left\{ \text{RelativeLoneliness}(t; R^\star) : R^\star \in \widehat{\mathcal{R}}_{\text{CD4}} \right\} \tag{4.23}$$

over our set of randomized CD4 datasets $\widehat{\mathcal{R}}_{\text{CD4}}$.

We can compute a corresponding "randomization" z-score

$$z^\star(t) = \frac{s_{\text{obs}} - \text{mean}\left(S^\star(t)\right)}{\text{stddev}(S^\star(t))}. \tag{4.24}$$

If our randomization procedure produces a reliable approximate null distribution of z-scores, and if our set of biological replicates approximate the sampling distribution of DN TCRβ sequences well, then we would expect there to be a notable relationship between $z(t)$ and $z^\star(t)$.

We see that the randomization z-scores $z^\star$ and replicate z-scores z exhibit a strong linear relationship (Figure 4.6a), with a correlation coefficient of $\rho \approx 0.877$. Further, we can assess how our randomization null distribution behaves as a proxy to the biological replicate null distribution by performing a standard linear regression of z on $z^\star$. Because the scatterplot reveals clear heteroskedasticity, we use sandwich estimation to obtain robust standard error estimates. This yields a significantly positive slope coefficient of $\beta \approx 0.656259$ ($p < 2{\times}10^{-16}$), with relatively high predictiveness (adjusted R^2: 0.769, $p < 2.2 \times 10^{-16}$). We note that there is some visual evidence that the relationship might exhibit nonlinearity in the right tail, particularly due to the OT-Tremont cluster which seems to mostly reside above the regression line. Nonetheless, we believe this model is still useful to understand the strength and general behavior of the relationship.

We also examine marginal density estimates of these z-scores stratified by cluster. In the "N/A" group, we find z-score densities with apparent bi-modal behavior. This is consistent with the assumption that there are approximately two sub-populations present in the DN repertoire: a sub-population of TCRs also common to the CD4 population, and a sub-population of TCRs specific to the DN repertoire. Indeed, for both distributions, the left

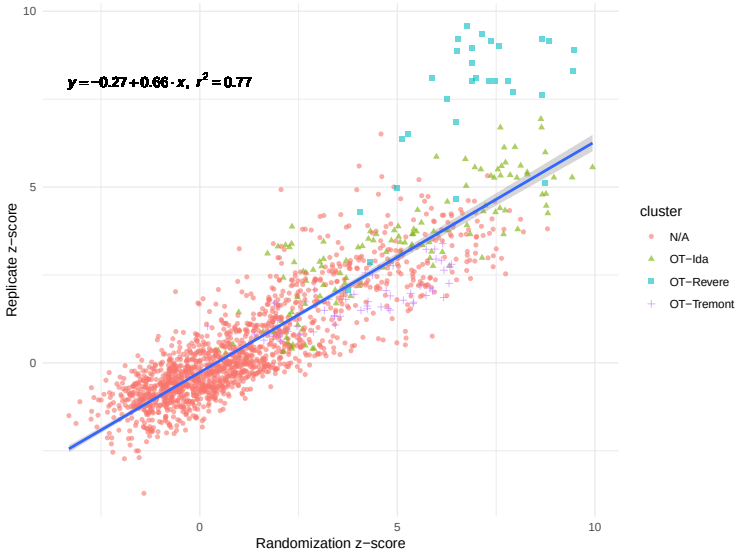
tail of this density stretches to slightly past $z = -2$, exhibiting behavior to a standard normal that would describe the null hypothesis of a TCR belonging to both populations, whereas the right tail stretches well past $z = 2$, providing evidence of DN-specific TCRs not categorized in our three clusters. For the three named clusters, the densities appear to have means notably higher than zero.

In summary, the approximate null distribution of loneliness scores from our randomization test appears to accurately represent the loneliness score distribution under the null hypothesis of two repertoires representing the same underlying population. Furthermore, we see that both the randomization null distribution and the replicate null distribution both lead to significantly high loneliness scores for the top three lonely clusters identified above. This indicates that we can confidently obtain significance estimates for loneliness scores when comparing two TCR repertoires.

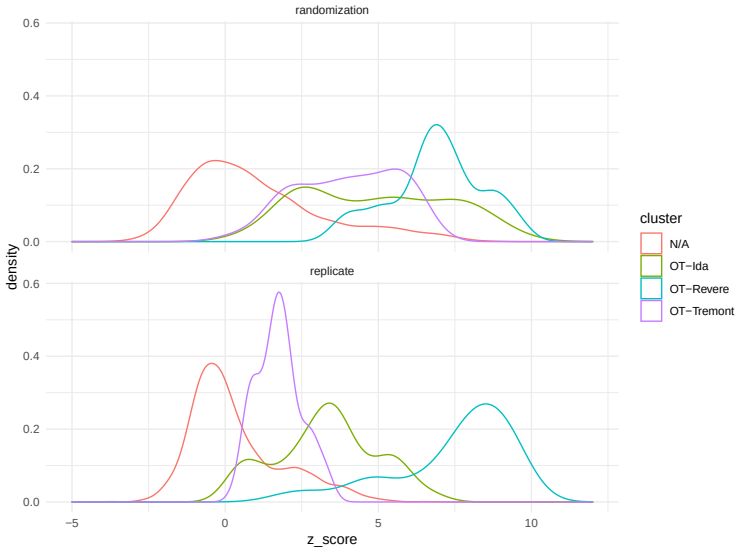
4.3.3 Identifying responsive TCRs to a yellow fever vaccination

Next, we benchmark the ability of our methods to detect meaningfully different regions between longitudinal repertoires using the YFV data discussed in the Materials and methods section. In particular, for each of the six human donors, we perform three comparisons: −7d vs 0d, 0d vs +15d, and 0d vs +45d. For each comparison, we compute the top 10 loneliest clusters using Algorithm 4. Since the immune response was estimated to peak at day 15 for all subjects and had contracted by day 45, we expect there to be many responsive TCRs in the +15d repertoire vs 0d, and we expect the number of false positives to increase for lower-ranked (i.e., less lonely) clusters. We expect some residual responsive TCRs in the +45d repertoire but with lower levels than the +15d repertoire. Finally, we expect little to no responsive TCRs in the −7d vs 0d comparison as both datasets were collected before the vaccination, and so this comparison serves as a control for the other two.

We compare our predictions to those made by Pogorelyy et al., the authors of the original study, for the same six donors [61]. Pogorelyy et al. applied a Bayesian statistical framework to the longitudinal sequence of repertoire snapshots to detect the TCR clones which



(a) Scatterplot of background z-scores versus randomization z-scores.



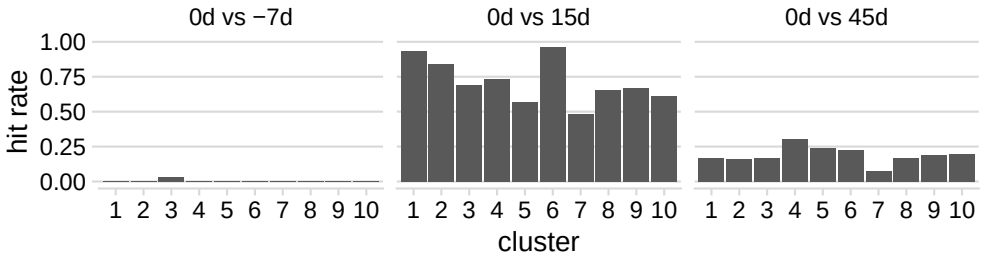
(b) Marginal density estimates of background z-scores and randomization z-scores.

Figure 4.6: Visualizations of the relationship between background and randomization z-scores.

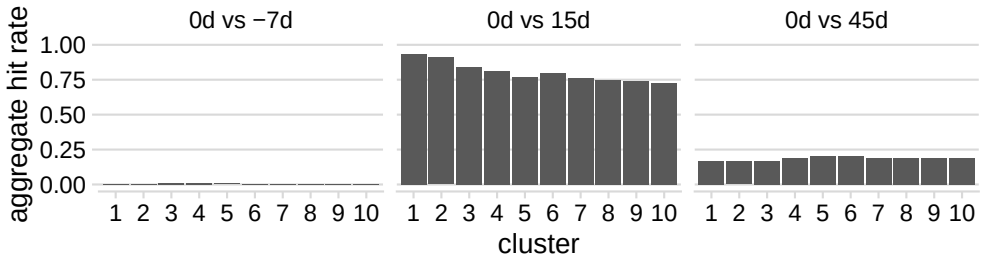
experienced significant proliferation and contraction, using biological replicates from day 0 to inform a null model of expected proliferation by chance. While their predictions do not constitute the ground truth of actual responsive TCR clones to the YFV vaccination, they can still serve as a useful performance benchmark. In particular, we can calculate the empirical probability that a clone our procedure detects as responsive was also detected by the original authors as responsive, and assess how this "hit rate" varies by timepoint, cluster rank, and donor.

We see that our hit rates behave according to our prior expectations, with larger hit rates for the +15d comparison, lower but non-neglible hit rates for the +45d comparison, and virtually no hits for the −7d comparison (Figure 4.7a). Moreover, the hit rates appear to be generally highest for the top-ranked clusters (i.e., the clusters with the highest loneliness), and decrease to more moderate values as for the lower-ranked clusters (with rank-6 clusters happening to have unusually high rates). We can obtain a smoothed version of these rates by calculating aggregate hit ranks for all clusters up to the given cluster number. For example, when the cluster number is 3, the hit rate is computed over all rank-1, rank-2, and rank-3 clusters. We observe a similar pattern, with a steady downward trend for the 15d comparison, and no apparent trend in the other two groups (Figure 4.7b).

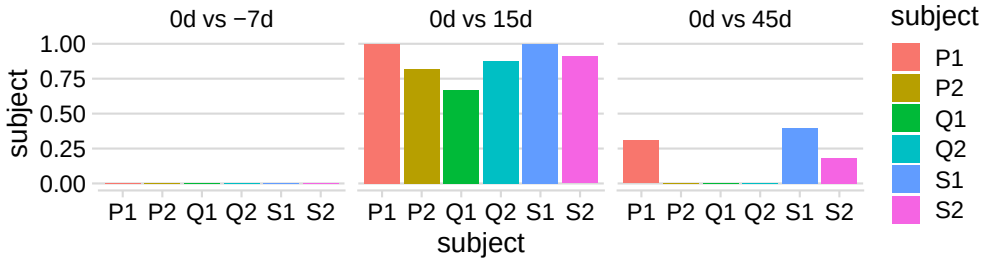
We also see that aggregate hit rates are fairly consistent across subjects, with rates for cluster rank ≤ 2 (i.e., the top two loneliest clusters for each subject) consistently high for the +15d comparison (Figure 4.7c), and mostly moderate to high rates for cluster rank ≤ 10 (i.e., the top ten loneliest clusters for each subject) (Figure 4.7d). Subject Q1 exhibits mildly exceptional behavior, with notably lower hit rates than the other donors in both cases, although the original authors also noted some abnormalities for subject Q1 in their analyses, such as comparatively low levels of responsive TCRs on +15d and +45d. Moreover, it appears that only subjects P1, S1, and S2 have nontrivial hit rates for both cluster rank ≤ 2 and cluster rank ≤ 10. These three subjects had the highest +15d hit rates in general, which suggests that the responsive clusters we found for day 15 were able to persist until day 45, or perhaps suggests a correlation between the strength of the immune response for



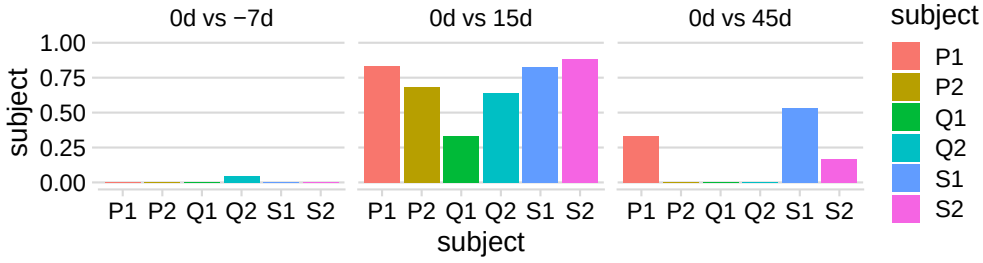
(a) Hit rates of inferred responsive TCRs by reference timepoint and cluster rank.



(b) Aggregate hit rates of inferred responsive TCRs by reference timepoint and cluster rank.



(c) Hit rates of inferred responsive TCRs by reference timepoint and donor, for cluster rank ≤ 2.



(d) Hit rates of our responsive TCR inferences by reference timepoint and donor, for cluster rank ≤ 10.

Figure 4.7: Various hit rate statistics for the YFV benchmark analysis.

these two timepoints.

To further assess whether our method is able to detect responsive clusters, we follow the validation of Pogorelyy et al. and examine an independent dataset of public TCRs obtained from VDJdb [73]. This dataset contains 264 sequences of TCRs previously shown to be responsive to a particular YFV epitope, as well as a control set of 370 sequences of TCRs responsive to an unrelated cytomegalovirus (CMV) epitope; call these the YFV validation set and the CMV validation set, respectively. Define our candidate set of responsive TCRs as those TCRs with a sequence present in any top-10 cluster for any of the six +15d comparisons. We compute two quantities for the YFV and CMV validation sets: the number of exact sequence matches found in our candidate set, as well as the number of sequences which belong to any of the top-10 clusters underlying our candidate set. In comparison, Pogorelyy et al. reported the number of exact sequence matches present in their candidate set, the number of sequences with no more than 1 CDR3aa mismatch from some TCR in their set, and the number of sequences with no more than 2 CDR3aa mismatches from a TCR in their set. These two respective comparative methods reflect the way their corresponding inferential methods identify responsive sequences.

We find that our methods are able to identify YFV sequences in the validation set while avoiding CMV sequences in the control set at comparable rates to the methods of Pogorelyy et al. Table 4.1 shows the results of the above experiment, as well as the results from Pogorelyy et al. (obtained from Table S2 of [61]). Our method detects 3 exact sequence matches to the YFV validation set, and 1 exact sequence match to the CMV validation set. Further, we detect 93 YFV validation sequences and 28 CMV sequences present in our candidate clusters, leading to a true positive/false positive ratio of $93/28 \approx 3.3$. In contrast, Pogorelyy detects a total of 18 exact sequence matches to the YFV set and zero exact matches to the CMV set. When allowing up to 2 CDR3aa mismatches, they detect 153 YFV sequences and 30 CMV sequences, leading to a true positive/false positive ratio of $153/30 = 5.1$. While we expect their approach, which uses the full trajectory of datasets across five timepoints for each subject, to perform better in this regard, our approach achieves competitive performance

Method	Antigen	S1	S2	P1	P2	Q1	Q2	Total
Ours (exact match)	CMV	0	0	0	0	0	1	1
Ours (is in a top-10 cluster)	CMV	2	2	4	4	4	12	28
Ours (exact match)	YFV	1	1	1	0	0	0	3
Ours (is in a top-10 cluster)	YFV	28	20	18	11	2	14	93
Pogorelyy (exact match)	CMV	0	0	0	0	0	0	0
Pogorelyy (1 CDR3aa mismatch)	CMV	0	0	0	1	0	2	3
Pogorelyy (2 CDR3aa mismatch)	CMV	5	5	5	3	2	10	30
Pogorelyy (exact match)	YFV	3	5	2	1	3	4	18
Pogorelyy (1 CDR3aa mismatch)	YFV	24	10	12	9	5	21	81
Pogorelyy (2 CDR3aa mismatch)	YFV	27	30	24	11	40	21	153

Table 4.1: Counts of matches between our inferred responsive yellow fever (YFV) sequences and either (YFV) or cytomegalovirus (CMV) sequences obtained from VDJdb, where the CMV sequences are used as a control. Also provided are analogous counts for responsive sequences inferred by Pogorelyy et al. [61]. Columns S1 - Q2 correspond to the six subjects discussed in [61], also discussed in the Materials and Methods section.

while only using two timepoints for each subject. Overall, this provides further evidence that our lonely clusters are able to extract YFV-responsive TCR clusters consistently across subjects, and that these clusters generalize beyond the training datasets.

There are a couple of explanations for the discrepancies that do arise between the inferred hits. First, as already mentioned, the set inferred by Pogorelyy et al. is not actually the ground truth of YFV-responsive TCRs, and both methods likely contain false positives as well as true negatives, both of which will impact the hit rates in Figure 4.7. Further, as mentioned, the approach of Pogorelyy used the full trajectory of repertoire snapshots to infer their set of responsive TCRs, whereas our method only looks at two snapshots at a time. In particular, our inferred positives corresponding to the +15d clusters make use of a fraction of the data used by Pogorelyy, yet we still identify a notable amount of their inferred positives, while avoiding a problematic false positive rate.

4.4 Discussion

We have described a nonparametric approach to TCR repertoire comparison driven by optimal transport and TCRdist, including a novel clustering algorithm that determines the regions of highest differential enrichment between two repertoires. We demonstrated that our framework can successfully extract biologically meaningful regions between distinct TCR populations through several analyses. Our methods were able to identify several clusters that are consistently enriched in the double negative T cell repertoire with respect to the $CD4^+$ T cell repertoire across biological replicates, and characterize their V gene and CDR3aa profiles. These clusters have significant overlap with clusters that were hand-identified by independent and close examination of a TCR data set. We also presented a randomization test to obtain significance estimates of our TCR scores, and validated them against a proxy null distribution comprised of the double negative biological replicates. Finally, our methods were able to detect responsive TCR clusters to a yellow fever virus immunization across multiple donors using only one post-vaccination repertoire snapshot per donor.

Our framework can be viewed as a nonparametric approach to detecting enriched TCR

regions in a target repertoire compared to a specified null distribution, which is manifest as a source or reference repertoire. Thus, the inferences will be valid insofar as our source repertoire is a representative sample from the underlying population of interest. This provides flexibility in which baseline distribution to compare against if we do have a reference repertoire that we are confident represents the population of interest, rather than relying on a model that might be biased towards a different or more general population. If we do not have a representative sample repertoire from the population of interest, an established model might yield more robust results. Thus, one must be decide which reference population should be used for the particular application, and how this reference population can be best represented, in order to choose the appropriate approach.

Moreover, a major drawback to our clustering mechanism (Algorithm 4) is that it cannot automatically estimate the number of clusters to return. Both of these considerations are in contrast to parametric approaches like ALICE [60], which rely on a parametric P_{gen} model for the null distribution but can be directly applied to a single repertoire and automatically return any significant cluster. Future work will investigate semi-parametric approaches which calibrate empirical reference repertoires against a global parametric null model in hopes of mitigating any overfitting to peculiarities in the empirical repertoire as well as the issue of determining the number of clusters.

Another future direction involves trying other distance functions between immune receptors, and seeing how other metrics impact the results. This could also lead to a generalization of our methods to B cell receptor (BCR) repertoire data, as there is no current equivalent to TCRdist for BCRs. One possible direction would be to examine the efficacy of BCR and TCR sequence embeddings within our optimal transport framework, such as the embeddings underlying recent variational autoencoders for TCR sequences [19].

One might also try another distance between probability distributions that also incorporates a metric function on the individual objects in the sample space. Perhaps the two most popular alternative distances between two probability distributions are known as the discrepancy metric and the Prokhorov metric. The discrepancy metric between probability

measures μ and ν is defined as

$$d_{\text{Discrepancy}}(\mu, \nu) := \sup_{\text{closed balls B}} |\mu(B) - \nu(B)|. \tag{4.25}$$

In other words, this metric looks at every possible ball, calculates the absolute difference in probability measures of the ball, and gives the largest such difference. Such a metric is oblivious to other differences occurring in the region space, and thus, seems less appropriate for distributions over TCRs where there could be many subregions with interesting behavior. The Prokhorov metric between μ and ν is defined as

$$d_{\text{Prokhorov}}(\mu, \nu) := \inf\{\varepsilon > 0 : \mu(B) \leq \nu(B^{\varepsilon}) \; \forall \text{ closed balls } B\}, \tag{4.26}$$

where $B^{\varepsilon} = \{x : \inf_{y \in B} d(x, y) \leq \varepsilon\}$. Similarly to the discrepancy metric, this metric fixates on an infimum over the region and fails to account for more subtle differences between distributions. Thus we believe that the optimal transport metric is the most appropriate in the TCR setting.

Appendix A: Visual examination of the efficacy of breakpoint estimation via segmented regression

Here, we perform a visual check of the assumptions used in our clustering procedure by applying it to the DN repertoires. In particular, we verify that the assumptions of the segmented regression specified by (4.14) hold in the relationship of mean annulus loneliness versus TCRdist from the centroid, and that this relationship has identifiable breakpoints for a typical repertoire. We perform Algorithm 4 on each full repertoire, which yields the "loneliest" cluster of each repertoire. Figure S1 displays scatterplots of the mean annulus loneliness vs radius for each subject, as well as lines depicting the segmented regression estimates (dashed lines are used for repertoires of fewer than 200 sequences). We see that we are able to successfully estimate a breakpoint $r_{\text{breakpoint}}$ for each subject, with $r_{\text{breakpoint}} \in (50, 100)$ for almost all subjects. When the repertoire contains fewer than 200 TCRs, the relationships can weaken (e.g. Subject 1), though Algorithm 4 still provides sensible regression estimates.

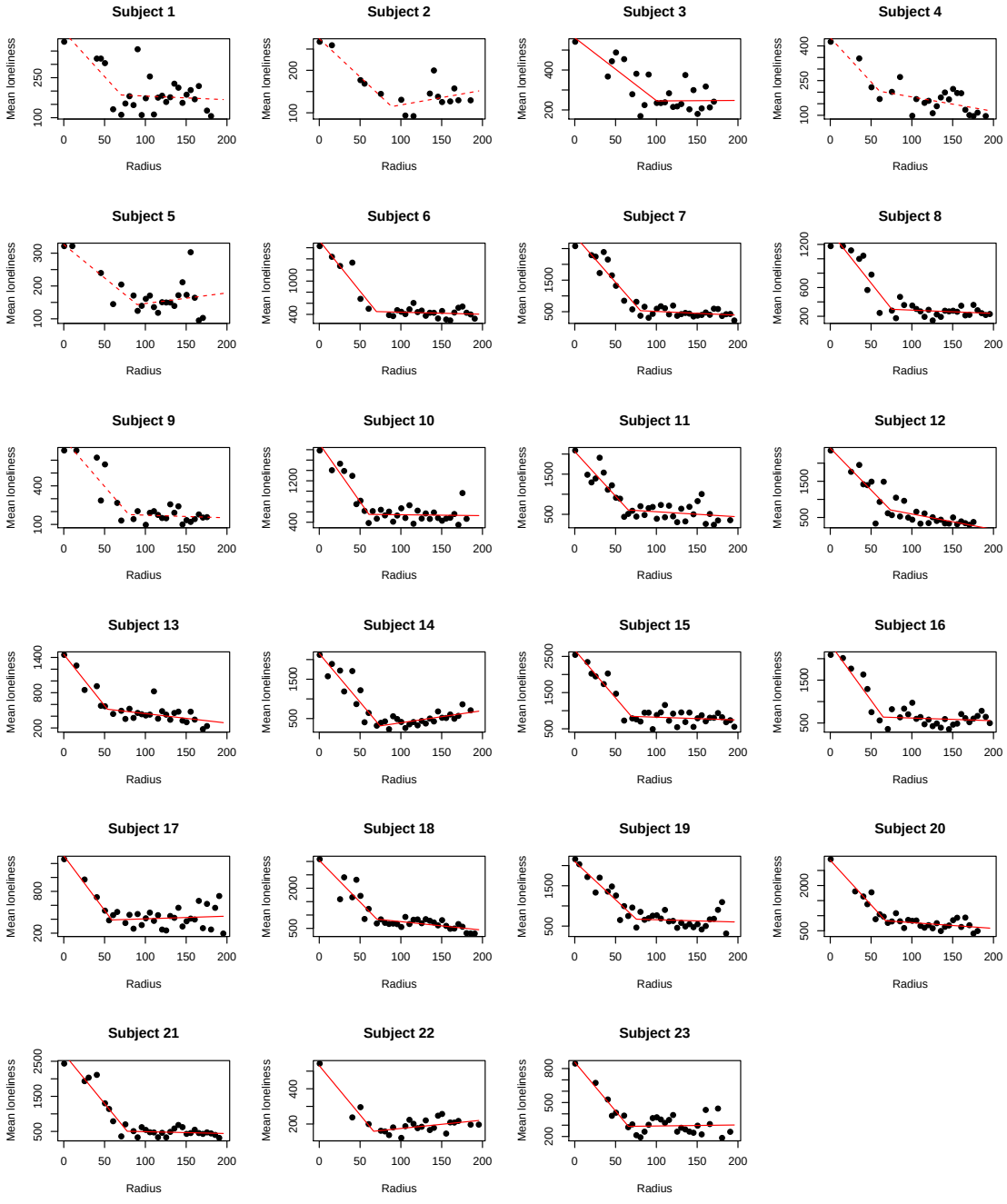


Figure S1: Scatterplots of mean annulus loneliness vs TCRdist radius for each of the DN repertoires, along with estimated segmented regression fits. Repertories with fewer than 200 TCRs have a dashed regression line.

When the repertoire contains at least 200 TCRs, we see consistent elbow behavior and convincing breakpoint estimates. Furthermore, violations of the least squares assumptions do not appear to be a concern.

Appendix B: Derivation of (4.5)

Fix $(\mathbf{r}, \mathbf{c}) \in \Sigma^2$, $\alpha > 0$, and write

$$\mathbf{P}^\star \equiv \underset{\mathbf{P}\in\mathbf{U}_\alpha(\mathbf{r},\mathbf{c})}{\arg\min} \langle \mathbf{P}, \mathbf{C}\rangle \tag{4.27}$$

$$= \underset{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})}{\arg\min} \langle \mathbf{P}, \mathbf{C}\rangle \text{ subject to } \mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right) \leq \alpha \tag{4.28}$$

$$= \underset{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})}{\arg\min} f(\mathbf{P}) \text{ subject to } g_\alpha(\mathbf{P}) \leq 0 \tag{4.29}$$

where $f(\mathbf{P}) := \langle \mathbf{P}, \mathbf{C}\rangle$ and $g_\alpha(\mathbf{P}) := \mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right) - \alpha$. We know that the inner product is linear in $\mathbf{P}$ since $\mathbf{C}$ is fixed; hence, it is convex. As the argmin operates over the convex set $\mathbf{U}(\mathbf{r}, \mathbf{c})$, we can appeal to the Lagrangian dual:

$$\underset{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})}{\text{minimize}}\, f(\mathbf{P}) \text{ subject to } g_\alpha(\mathbf{P}) \leq 0 \tag{4.30}$$

$$\implies \underset{u}{\text{maximize}} \inf_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \{f(\mathbf{P}) + u g_\alpha(\mathbf{P})\} \text{ subject to } u \geq 0 \tag{4.31}$$

$$\implies \underset{u\geq 0}{\text{maximize}} \inf_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + u\left[\mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right) - \alpha\right]\right\} \tag{4.32}$$

$$\implies \underset{u\geq 0}{\text{maximize}} \inf_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + u\mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right)\right\} \tag{4.33}$$

$$\tag{4.34}$$

Thus, for some maximizing $\gamma := \arg\max_{u\geq 0} \inf_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + u\mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right)\right\}$, we have that

$$\min_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} f(\mathbf{P}) \text{ subject to } g_\alpha(\mathbf{P}) \leq 0 = \inf_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + \gamma\mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right)\right\} \tag{4.35}$$

or equivalently,

$$\arg\min_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} f(\mathbf{P}) \text{ subject to } g_\alpha(\mathbf{P}) \leq 0 = \arg\inf_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + \gamma \mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right)\right\} \tag{4.36}$$

$$= \arg\min_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + \gamma \mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right)\right\} \tag{4.37}$$

where the last line follows since $\mathbf{U}(\mathbf{r}, \mathbf{c})$ is a closed polytope. Next, note that

$$\mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right) \equiv \sum_i \sum_j p_{ij} \log\left(\frac{p_{ij}}{r_i c_j}\right) \tag{4.38}$$

$$= \sum_i \sum_j p_{ij} \log(p_{ij}) - \sum_i \sum_j p_{ij} \log(r_i) - \sum_i \sum_j p_{ij} \log(c_j) \tag{4.39}$$

$$= -h(\mathbf{P}) - \sum_i \log(r_i) \sum_j p_{ij} - \sum_j \log(c_j) \sum_i p_{ij} \tag{4.40}$$

$$= -h(\mathbf{P}) - \sum_i \log(r_i) r_i - \sum_j \log(c_j) c_j \tag{4.41}$$

$$= -h(\mathbf{P}) + h(\mathbf{r}) + h(\mathbf{c}). \tag{4.42}$$

Thus, defining $\lambda := 1/\gamma$, we have

$$\mathbf{P}^\star \equiv \arg\min_{\mathbf{P}\in\mathbf{U}_\alpha(\mathbf{r},\mathbf{c})} \langle \mathbf{P}, \mathbf{C}\rangle \tag{4.43}$$

$$= \arg\min_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + \gamma \mathrm{KL}\left(\mathbf{P}||\mathbf{r}\mathbf{c}^\mathsf{T}\right)\right\} \text{ (by (4.36))} \tag{4.44}$$

$$= \arg\min_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle + \gamma \left[h(\mathbf{r}) + h(\mathbf{c}) - h(\mathbf{P})\right]\right\} \text{ (by (4.42))} \tag{4.45}$$

$$= \arg\min_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle - \gamma h(\mathbf{P})\right\} \tag{4.46}$$

$$= \arg\min_{\mathbf{P}\in\mathbf{U}(\mathbf{r},\mathbf{c})} \left\{\langle \mathbf{P}, \mathbf{C}\rangle - \frac{1}{\lambda} h(\mathbf{P})\right\} \tag{4.47}$$

$$\equiv \mathbf{P}^\lambda \quad \square \tag{4.48}$$

Chapter 5
CONCLUSION

We mentioned in the first chapter that the adaptive immune system is paradoxically well-understood in some ways and confounding in others. Chapter two immediately illustrated this by evaluating a recent interrogation into the mechanisms that underly SHM, which have otherwise been considered established for decades. Chapter three applied unprecedented rigor to the surprisingly nuanced problem of describing and comparing repertoires as well as validating probabilistic models of immune receptor repertoires. Chapter four went a bit further, presenting a novel inferential approach to comparing TCR distributions, and making strides towards the elusive goal of inferring specificity from TCR sequence.

We also mentioned in the first chapter that the uniquely stochastic nature of adaptive immune repertoires provides the core challenge associated with AIRR-seq analysis. Indeed, it is a formidable task to make statements about a process involving an astronomical number of possible receptors and antigen peptides, coupled with sophisticated evolutionary constraints. However, modern computation, advanced statistical methods, and an abundance of data give us the tools make real progress. For example, the Sinkhorn method used in Chapter four is a very efficient and accurate approximation to a classically intractable problem, and was derived less than a decade ago. By applying this method to nearly 100 repertoire datasets, we can glean meaningful scientific insights, and this would not have been possible ten years ago.

In a world where a pandemic can cause a global public health emergency as well as challenge our social, political, and economic values, studying the immune system seems as important as ever. By harnessing the power of modern statistics, we can continue to make robust discoveries to aid our understanding and decision making.

BIBLIOGRAPHY

[1] William R. Atchley, Jieping Zhao, Andrew D. Fernandes, and Tanja Drüke. Solving the protein sequence metric problem. *Proceedings of the National Academy of Sciences*, 102(18):6395–6400, 2005.

[2] Oren Avram, Anna Vaisman-Mentesh, Dror Yehezkel, Haim Ashkenazy, Tal Pupko, and Yariv Wine. ASAP - a webserver for immunoglobulin-sequencing analysis pipeline. *Front. Immunol.*, 9:1686, July 2018.

[3] Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48, 2015.

[4] John Benedetto. *Harmonic analysis and applications*. CRC Press, 1 edition, 1997.

[5] Jennifer Benichou, Rotem Ben-Hamo, Yoram Louzoun, and Sol Efroni. Rep-Seq: uncovering the immunological repertoire through next-generation sequencing. *Immunology*, 135(3):183–191, Mar 2012.

[6] Julia Bischof and Saleh M. Ibrahim. bcRep: R package for comprehensive analysis of B cell receptor repertoire data. *PLoS ONE*, 11(8):e0161569, August 2016.

[7] Carl Boettiger. An introduction to Docker for reproducible research. *Oper. Syst. Rev.*, 49(1):71–79, January 2015.

[8] Carl Boettiger. An introduction to docker for reproducible research. *SIGOPS Oper. Syst. Rev.*, 49(1):71–79, January 2015.

[9] CR Bolen, F Rubelt, JA Vander Heiden, and MM Davis. The repertoire dissimilarity index as a method to compare lymphocyte receptor repertoires. *BMC Bioinformatics*, 18(1):155, March 2017.

[10] Zachary A Bornholdt, Hannah L Turner, Charles D Murin, Wen Li, Devin Sok, Colby A Souders, Ashley E Piper, Arthur Goff, Joshua D Shamblin, Suzanne E Wollen, Thomas R Sprague, Marnie L Fusco, Kathleen B J Pommert, Lisa A Cavacini, Heidi L Smith, Mark Klempner, Keith A Reimann, Eric Krauland, Tillman U Gerngross, Dane K Wittrup, Erica Ollmann Saphire, Dennis R Burton, Pamela J Glass, Andrew B Ward, and Laura M Walker. Isolation of potent neutralizing antibodies from a survivor of the 2014 Ebola virus outbreak. *Science*, 351:1078–1083, February 2016.

[11] SD Boyd et al. Individual variation in the germline Ig gene repertoire inferred from variable region gene rearrangements. *J Immunol*, 184(12):6986–92, June 2010.

[12] O V Britanova et al. Dynamics of individual T cell repertoires: from cord blood to centenarians. *J. Immunol.*, 196(12):5005–5013, June 2016.

[13] Diego Chowell, Sri Krishna, Pablo D Becker, Clement Cocita, Jack Shu, Xuefang Tan, Philip D Greenberg, Linda S Klavinskis, Joseph N Blattman, and Karen S Anderson. TCR contact residue hydrophobicity is a hallmark of immunogenic CD8+ T cell epitopes. *Proc Natl Acad Sci U S A*, 112(14):E1754–62, Apr 2015.

[14] Mattia Cinelli, Yuxin Sun, Katharine Best, James M Heather, Shlomit Reich-Zeliger, Eric Shifrut, Nir Friedman, John Shawe-Taylor, and Benny Chain. Feature selection using a one dimensional naïve Bayes' classifier increases the accuracy of support vector machine classification of CDR3 repertoires. *Bioinformatics*, 33(7):951–955, 01 2017.

[15] MM Corcoran, GE Phad, Bernat N Vázquez, C Stahl-Henning, N Sumida, MA Persson, M Martin, and Hedestam GB Karlsson. Production of individualized V gene databases reveals high levels of immunoglobulin genetic diversity. *Nat Commun*, 7:13642, December 2016.

[16] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013.

[17] Gordon A Dale, Daniel J Wilkins, Caitlin D Bohannon, Dario Dilernia, Eric Hunter, Trevor Bedford, Rustom Antia, Ignacio Sanz, and Joshy Jacob. Clustered mutations at the murine and human IgH locus exhibit significant linkage consistent with templated mutagenesis. *J. Immunol.*, 203(5):1252–1264, September 2019.

[18] Pradyot Dash, Andrew J. Fiore-Gartland, Tomer Hertz, George C. Wang, Shalini Sharma, Aisha Souquette, Jeremy Chase Crawford, E. Bridie Clemens, Thi H. O. Nguyen, Katherine Kedzierska, Nicole L. La Gruta, Philip Bradley, and Paul G. Thomas. Quantifiable predictive features define epitope specific T cell receptor repertoires. *Nature*, 547(7661):89–93, 2017.

[19] Kristian Davidsen, Branden J. Olson, William S. DeWitt III, Jean Feng, Elias Harkins, Philip Bradley, and Frederick A. Matsen IV. Deep generative models for T cell receptor protein sequences. *eLife*, 8:e46935, 2019.

[20] Marc Duez, Mathieu Giraud, Ryan Herbert, Tatiana Rocher, Mikael Salson, and Florian Thonier. Vidjil: A web platform for analysis of high-throughput repertoire sequencing. *PLoS One*, 11:e0166126, November 2016.

[21] S R Eddy. Profile hidden Markov models. *Bioinformatics*, 14(9):755–763, 10 1998.

[22] Robert C Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, 32(5):1792–1797, March 2004.

[23] Yuval Elhanati, Anand Murugan, Curtis G Callan, Jr, Thierry Mora, and Aleksandra M Walczak. Quantifying selection in immune receptor repertoires. *Proc. Natl. Acad. Sci. U. S. A.*, June 2014.

[24] William J. J. Finlay and Juan C. Almagro. Natural and man-made V-gene repertoires for antibody discovery. *Front. Immunol.*, 3:342, 2012.

[25] D Gadala-Maria, G Yaari, M Uduman, and SH Kleinstein. Automated analysis of high-throughput B-cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. *Proc Natl Acad Sci U S A*, 112(8):E862–70, February 2015.

[26] George Georgiou, Gregory C Ippolito, John Beausang, Christian E Busse, Hedda Wardemann, and Stephen R Quake. The promise and challenge of high-throughput sequencing of the antibody repertoire. *Nature Biotechnology*, 32(2):158–168, 2014.

[27] Sofie Gielis, Pieter Moris, Wout Bittremieux, Nicolas De Neuter, Benson Ogunjimi, Kris Laukens, and Pieter Meysman. Detection of enriched T cell epitope specificity in full T cell receptor sequence repertoires. *Frontiers in Immunology*, 10:2820, 2019.

[28] N Goldman and Z Yang. A codon-based model of nucleotide substitution for protein-coding DNA sequences. *Mol. Biol. Evol.*, 11(5):725–736, September 1994.

[29] Namita T. Gupta, Kristofor D. Adams, Adrian W. Briggs, Sonia C. Timberlake, Francois Vigneault, and Steven H. Kleinstein. Hierarchical clustering can identify B cell clones with high confidence in Ig repertoire sequencing data. *The Journal of Immunology*, 198(6):2489–2499, 2017.

[30] Namita T. Gupta, Jason A. Vander Heiden, Mohamed Uduman, Daniel Gadala-Maria, Gur Yaari, and Steven H. Kleinstein. Change-O: a toolkit for analyzing large-scale B cell immunoglobulin repertoire sequencing data. *Bioinformatics*, 31(20):3356–3358, October 2015.

[31] James M Heather, Mattia Cinelli, Benny Chain, Katharine Best, Yuxin Sun, John Shawe-Taylor, Eric Shifrut, Shlomit Reich-Zeliger, and Nir Friedman. Feature selection using a one dimensional naive Bayes classifier increases the accuracy of support vector machine classification of CDR3 repertoires. *Bioinformatics*, 33(7):951–955, 01 2017.

[32] S Henikoff and J G Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

[33] D Hou et al. Immune repertoire diversity correlated with mortality in avian influenza A (H7N9) virus infected patients. *Sci Rep*, 6:33843, September 2016.

[34] Hanna IJspeert, Pauline A van Schouwenburg, David van Zessen, Ingrid Pico-Knijnenburg, Andrew P Stubbs, and Mirjam van der Burg. Antigen Receptor Galaxy: A user-friendly, web-based tool for analysis and visualization of T and B cell receptor repertoire data. *J. Immunol.*, 198:4156–4165, May 2017.

[35] Emmi Jokinen, Jani Huuhtanen, Satu Mustjoki, Markus Heinonen, and Harri Lähdesmäki. Determining epitope specificity of T cell receptors with TCRGP. *bioRxiv*, 2019.

[36] Vanessa Isabell Jurtz, Leon Eyrich Jessen, Amalie Kai Bentzen, Martin Closter Jespersen, Swapnil Mahajan, Randi Vita, Kamilla Kjærgaard Jensen, Paolo Marcatili, Sine Reker Hadrup, Bjoern Peters, and Morten Nielsen. NetTCR: sequence-based prediction of TCR binding to peptide-MHC complexes using convolutional neural networks. *bioRxiv*, 2018.

[37] Kazutaka Katoh, Kazuharu Misawa, Kei-ichi Kuma, and Takashi Miyata. MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Res.*, 30(14):3059–3066, 2002.

[38] Akinori Kidera, Yasuo Konishi, Masahito Oka, Tatsuo Ooi, and Harold A. Scheraga. Statistical analysis of the physical properties of the 20 naturally occurring amino acids. *Journal of Protein Chemistry*, 4:23–55, 02 1985.

[39] Kevin Larimore, Michael W McCormick, Harlan S Robins, and Philip D Greenberg. Shaping of human germline IgH repertoires revealed by deep sequencing. *J. Immunol.*, 189(6):3221–3230, August 2012.

[40] Uri Laserson, Francois Vigneault, Daniel Gadala-Maria, Gur Yaari, Mohamed Uduman, Jason A. Vander Heiden, William Kelton, Sang Taek Jung, Yi Liu, Jonathan Laserson, Raj Chari, Je-Hyuk Lee, Ido Bachelet, Brendan Hickey, Erez Lieberman-Aiden, Bozena Hanczaruk, Birgitte B. Simen, Michael Egholm, Daphne Koller, George Georgiou, Steven H. Kleinstein, and George M. Church. High-resolution antibody dynamics of vaccine-induced immune responses. *PNAS*, 111(13):4928–4933, April 2014.

[41] Marie-Paule Lefranc, Veronique Giudicelli, Chantal Ginestoux, Joumana Jabado-Michaloud, Geraldine Folch, Fatena Bellahcene, Yan Wu, Elodie Gemrot, Xavier Brochet, Jerome Lane, Laetitia Regnier, Francois Ehrenmann, Gerard Lefranc, and Patrice Duroux. IMGT, the international ImMunoGeneTics information system. *Nucleic Acids Research*, 37(suppl_1):D1006–D1012, 10 2008.

[42] Torgny Lindvall. *Lectures on the Coupling Method*. Wiley, 1992.

[43] Quentin Marcou, Thierry Mora, and Aleksandra M. Walczak. High-throughput immune repertoire analysis with IGoR. *Nat. Commun*, 9(561), 2018.

[44] V Martin, YC Bryan Wu, D Kipling, and D Dunn-Walters. Ageing of the B-cell repertoire. *Philos Trans R Soc Lond B Biol Sci*, 370(1676), September 2015.

[45] Lisa McFerrin. *HDMD: Statistical Analysis Tools for High Dimension Molecular Data DMD*, 2013. R package version 1.2.

[46] S P Methot and J M Di Noia. Chapter two - molecular mechanisms of somatic hypermutation and class switch recombination. In Frederick W. Alt, editor, *Advances in Immunology*, volume 133, pages 37–87. Academic Press, 2017.

[47] P Miqueu, M Guillet, N Degauque, JC Doré, JP Soulillou, and S Brouard. Statistical analysis of CDR3 length distributions for the assessment of T and B cell repertoire biases. *Mol Immunol*, 44(6):1057–1064, February 2007.

[48] Arnau Mir, Francesc Rossello, and Lucia Rotger. *CollessLike: Distribution and Percentile of Sackin, Cophenetic and Colless-Like Balance Indices of Phylogenetic Trees*, 2018. R package version 1.0.

[49] Vito M.R. Muggeo. segmented: an r package to fit regression models with broken-line relationships. *R News*, 8(1):20–25, 2008.

[50] Kenneth Murphy. *Janeway's Immunobiology*. Garland Science, Taylor & Francis Group, LLC, 8 edition, 2012.

[51] Anand Murugan, Thierry Mora, Aleksandra M. Walczak, and Curtis 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.

[52] Vadim I Nazarov, Mikhail V Pogorelyy, Ekaterina A Komech, Ivan V Zvyagin, Dmitry A Bolotin, Mikhail Shugay, Dmitry M Chudakov, Yury B Lebedev, and Ilgar Z Mamedov. tcr: an R package for T cell receptor repertoire advanced data analysis. *BMC Bioinformatics*, 16(1):175, May 2015.

[53] Branden J. Olson and Frederick A. Matsen IV. The Bayesian optimist's guide to adaptive immune receptor repertoire analysis. *Immunol. Rev.*, 284:148–166, 2018.

[54] Branden J. Olson, Pejvak Moghimi, Chaim A. Schramm, Anna Obraztsova, Duncan Ralph, Jason A. Vander Heiden, Mikhail Shugay, Adrian J. Shepherd, William Lees, and Frederick A. Matsen IV. sumrep: A summary statistic framework for immune receptor repertoire comparison and model validation. *Frontiers in Immunology*, 10:2533, 2019.

[55] Jared Ostmeyer, Scott Christley, William H. Rounds, Inimary Toby, Benjamin M. Greenberg, Nancy L. Monson, and Lindsay G. Cowell. Statistical classifiers for diagnosing disease from immune repertoires: a case study using multiple sclerosis. *BMC Bioinformatics*, 18(1):401, Sep 2017.

[56] Jared Ostmeyer, Scott Christley, William H. Rounds, Inimary Toby, Benjamin M. Greenberg, Nancy L. Monson, and Lindsay G. Cowell. Statistical classifiers for diagnosing disease from immune repertoires: a case study using multiple sclerosis. *BMC Bioinformatics*, 18(1):401, Sep 2017.

[57] Jared Ostmeyer, Scott Christley, Inimary T Toby, and Lindsay G Cowell. Biophysicochemical motifs in T-cell receptor sequences distinguish repertoires from tumor-infiltrating lymphocyte and adjacent healthy tissue. *Cancer Res*, 79(7):1671–1680, Apr 2019.

[58] H. Pagás, P. Aboyoun, R. Gentleman, and S. DebRoy. *Biostrings: String objects representing biological sequences, and matching algorithms*, 2017. R package version 2.44.2.

[59] E. Paradis, J. Claude, and K. Strimmer. APE: analyses of phylogenetics and evolution in R lanugage. *Bioinformatics*, 20(2):289–290, January 2004.

[60] Mikhail V. Pogorelyy, Anastasia A. Minervina, Mikhail Shugay, Dmitriy M. Chudakov, Yuri B. Lebedev, Thierry Mora, and Aleksandra M. Walczak. Detecting T-cell receptors involved in immune responses from single repertoire snapshots. *bioRxiv*, 2018.

[61] Mikhail V. Pogorelyy, Anastasia A. Minervina, Maximilian Puelma Touzel, Anastasiia L. Sycheva, Ekaterina A. Komech, Elena I. Kovalenko, Galina G. Karganova, Evgeniy S. Egorov, Alexander Yu. Komkov, Dmitriy M. Chudakov, Ilgar Z. Mamedov, Thierry Mora, Aleksandra M. Walczak, and Yuri B. Lebedev. Precise tracking of vaccine-responding T cell clones reveals convergent and personalized response in identical twins. *Proceedings of the National Academy of Sciences*, 115(50):12704–12709, 2018.

[62] Morgan N Price, Paramvir S Dehal, and Adam P Arkin. FastTree 2–approximately maximum-likelihood trees for large alignments. *PLoS One*, 5(3):e9490, March 2010.

[63] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.

[64] Duncan K Ralph and Frederick A Matsen, 4th. Consistency of VDJ rearrangement and substitution parameters enables accurate B cell receptor sequence annotation. *PLoS Comput. Biol.*, 12(1):e1004409, January 2016.

[65] Duncan K. Ralph and Frederick A. Matsen IV. Likelihood-based inference of B cell clonal families. *PLOS Comput. Biol.*, 12(10), October 2016.

[66] Ida Retter, Christophe Chevillard, Maren Scharfe, Ansgar Conrad, Martin Hafner, Tschong-Hun Im, Monika Ludewig, Gabriele Nordsiek, Simone Severitt, Stephanie Thies, America Mauhar, Helmut Blöcker, Werner Müller, and Roy Riblet. Sequence and characterization of the ig heavy chain constant and partial variable region of the mouse strain 129S1. *J. Immunol.*, 179(4):2419–2427, August 2007.

[67] I B Rogozin, Y I Pavlov, K Bebenek, T Matsuda, and T A Kunkel. Somatic mutation hotspots correlate with DNA polymerase eta error spectrum. *Nat. Immunol.*, 2(6):530–536, June 2001.

[68] Florian Rubelt, Christopher R. Bolen, Helen M. McGuire, Jason A. Vander Heiden, Daniel Gadala-Maria, Mikhail Levin, Ghia M. Euskirchen, Murad R. Mamedov, Gary E. Swan, Cornelia L. Dekker, Lindsay G. Cowell, Steven H. Kleinstein, and Mark M. Davis. Individual heritable differences result in unique cell lymphocyte receptor repertoires of naïve and antigen-experienced cells. *Nature Communications*, 7:11112 EP –, 03 2016.

[69] Susanne Schaller, Johannes Weinberger, Raul Jimenez-Heredia, Martin Danzer, Rainer Oberbauer, Christian Gabriel, and Stephan M Winkler. Immunexplorer (imex): a software framework for diversity and clonality analyses of immunoglobulins and T cell receptors on the basis of imgt/highv-quest preprocessed ngs data. *PLoS One*, 16:252, August 2015.

[70] Stefan A. Schattgen, Jeremy C. Crawford, Lee-Ann Van de Velde, Hiutung Chu, Sarkis K. Mazmanian, Phil Bradley, and Paul G. Thomas. Intestinal intraepithelial lymphocyte repertoires are imprinted clonal structures selected for MHC reactivity. *Sneak Peek*, October 2019.

[71] B. Schuster-Böckler, J. Schultz, and S. Rahmann. Hmm logos for visualization of protein families. *BMC Bioinformatics*, 5(7), 2004.

[72] Mikhail Shugay, Dmitriy V Bagaev, Maria A Turchaninova, Dmitriy A Bolotin, Olga V Britanova, Ekaterina V Putintseva, Mikhail V Pogorelyy, Vadim I Nazarov, Ivan V Zvyagin, Vitalina I Kirgizova, Kirill I Kirgizov, Elena V Skorobogatova, and Dmitriy M Chudakov. VDJtools: Unifying post-analysis of T cell receptor repertoires. *PLoS Comput. Biol.*, 11(11):e1004503, November 2015.

[73] Mikhail Shugay, Dmitriy V Bagaev, Ivan V Zvyagin, Renske M Vroomans, Jeremy Chase Crawford, Garry Dolton, Ekaterina A Komech, Anastasiya L Sycheva, Anna E Koneva, Evgeniy S Egorov, Alexey V Eliseev, Ewald Van Dyk, Pradyot Dash, Meriem Attaf, Cristina Rius, Kristin Ladell, James E McLaren, Katherine K Matthews, E Bridie Clemens, Daniel C Douek, Fabio Luciani, Debbie van Baarle, Katherine Kedzierska, Can Kesmir, Paul G Thomas, David A Price, Andrew K Sewell, and Dmitriy M Chudakov. VDJdb: a curated database of T-cell receptor sequences with known antigen specificity. *Nucleic Acids Res*, 46(D1):D419–D427, Jan 2018.

[74] Stephanie J Spielman and Claus O Wilke. Pyvolve: A flexible python module for simulating sequences along phylogenies. *PLoS One*, 10(9):e0139047, September 2015.

[75] Niclas Thomas, Katharine Best, Mattia Cinelli, Shlomit Reich-Zeliger, Hilah Gal, Eric Shifrut, Asaf Madi, Nir Friedman, John Shawe-Taylor, and Benny 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, 08 2014.

[76] Mark P.J. van der Loo. The stringdist package for approximate string matching. *The R Journal*, 6(1):111–122, June 2014.

[77] Jason A Vander Heiden, Gur Yaari, Mohamed Uduman, Joel N H Stern, Kevin C O'Connor, David A Hafler, Francois Vigneault, and Steven H Kleinstein. pRESTO: a toolkit for processing high-throughput sequencing raw reads of lymphocyte receptor repertoires. *Bioinformatics*, 30(13):1930–1932, July 2014.

[78] Jason Anthony Vander Heiden, Susanna Marquez, Nishanth Marthandan, Syed Ahmad Chan Bukhari, Christian E Busse, Brian Corrie, Uri Hershberg, Steven H Kleinstein, Frederick A Matsen, Iv, Duncan K Ralph, Aaron M Rosenfeld, Chaim A Schramm, AIRR Community, Scott Christley, and Uri Laserson. AIRR community standardized representations for annotated immune repertoires. *Front. Immunol.*, 9:2206, September 2018.

[79] Yan Wang, Katherine J L Jackson, William A Sewell, and Andrew M Collins. Many human immunoglobulin heavy-chain IGHV gene polymorphisms have been reported in error. *Immunol. Cell Biol.*, 86(2):111–115, February 2008.

[80] Travis J. Wheeler, Jody Clements, and Robert D. Finn. Skylign: a tool for creating informative, interactive logos representing sequence alignments and profile hidden markov models. *BMC Bioinformatics*, 15(1):7, 2014.

[81] Yu-Chang Wu, David Kipling, Hui Sun Leong, Victoria Martin, Alexander A Ademokun, and Deborah K Dunn-Walters. High-throughput immunoglobulin repertoire analysis distinguishes between human IgM memory and switched memory B-cell populations. *Blood*, 116(7):1070–1078, Aug 2010.

[82] Yu-Chang Bryan Wu, David Kipling, and Deborah K Dunn-Walters. The relationship between CD27 negative and positive B cell populations in human peripheral blood. *Front Immunol*, 2:81, 2011.

[83] Gur Yaari, Mohamed Uduman, and Steven H Kleinstein. Quantifying selection in high-throughput immunoglobulin sequencing data sets. *Nucleic Acids Res.*, 40(17):e134, May 2012.

[84] Jian Ye, Ning Ma, Thomas L. Madden, and James M. Ostell. IgBLAST: an immunoglobulin variable domain sequence analysis tool. *Nucleic Acids Res.*, 41(W1):W34–W40, July 2013.

[85] Leng-Siew Yeap, Joyce K Hwang, Zhou Du, Robin M Meyers, Fei-Long Meng, Agne Jakubauskaite, Mengyuan Liu, Vinidhra Mani, Donna Neuberg, Thomas B Kepler, Jing H Wang, and Frederick W Alt. Sequence-Intrinsic mechanisms that target AID mutational outcomes on antibody genes. *Cell*, 163:1124–1137, 2015.

[86] Ryo Yokota, Yuki Kaminaga, and Tetsuya J Kobayashi. Quantification of inter-sample differences in T-cell receptor repertoires using sequence-based information. *Front Immunol*, 8:1500, 2017.