Large-Scale B Cell Receptor Sequence Analysis Using Phylogenetics and Machine Learning

Amrit Dhar

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

Doctor of Philosophy

University of Washington

2019

Reading Committee:

Volodymyr Minin, Chair

Frederick Matsen, Chair

Noah Simon

Program Authorized to Offer Degree:
Statistics

University of Washington

Abstract

Large-Scale B Cell Receptor Sequence Analysis Using Phylogenetics and Machine Learning

Amrit Dhar

Co-Chairs of the Supervisory Committee:
Affiliate Professor Volodymyr Minin
Department of Statistics

Affiliate Professor Frederick Matsen
Department of Statistics

The adaptive immune system synthesizes antibodies, the soluble form of B cell receptors (BCRs), to bind to and neutralize pathogens that enter our body. B cells are able to generate a diverse set of high affinity antibodies through the affinity maturation process. During maturation, "naive" BCR sequences first accumulate mutations according to a neutral evolutionary process called somatic hypermutation (SHM), which may modify the associated binding affinities, and then are subject to natural selection by clonal expansion, which promotes the higher affinity antibodies. The set of mutated BCRs that result from a single naive BCR undergoing SHM can be referred to as a "clonal family". In my thesis, I study the mechanisms that govern the aforementioned evolutionary and selective processes of BCR sequences with the goal of better understanding how naive B cells diversify into mature B cells with high binding affinities.

It is frequently important to infer the full evolutionary paths from a given naive BCR sequence to the corresponding mature BCR sequences in the clonal family. Stochastic mapping, a missing data imputation technique, can be used to estimate the mutational trajectories mentioned above; it is a simulation-based method for probabilistically mapping substitution histories onto phylogenies according to continuous-time Markov models of evolution. Current

simulation-free algorithms can compute the mean but not any higher-order moments of the number of substitutions or of other stochastic mapping summaries; these algorithms scale linearly in the number of tips of the phylogenetic tree. I present the first simulation-free dynamic programming algorithm that calculates prior and posterior mapping variances and scales linearly in the number of phylogeny tips. This procedure suggests a general framework that can be used to efficiently compute higher-order moments of stochastic mapping summaries without simulations.

Before one can perform clonal lineage or ancestral sequence inference in a clonal family, one must first obtain an estimate of the clonal phylogenetic tree. Currently, standard phylogenetic inference techniques are used to model the SHM process; however, these methods do not account for all the complexities associated with this mutation process. I introduce a novel approach to inference that is based on a phylogenetic hidden Markov model (phylo-HMM). This technique is not only based on a more biologically realistic model of evolution but also designed to scale to the large datasets that result from high-throughput sequencing.

In the antibody engineering field, researchers would like to infer the most likely per-site substitutions that are allowed in a clonal family. Unfortunately, many clonal families are small in size and do not have enough observed sequence information to accurately answer the preceding question. Despite this, there are structural properties associated with BCR sequences that are common across clonal families. I propose a penalized regression model that leverages aggregated amino acid count data (also known as "substitution profiles") in large clonal families to predict the substitution profiles in smaller clonal families. I show that there is information, possibly embedded through structural and functional constraints, contained within these large clonal families that can be shared with the smaller ones to enhance their substitution profile predictions. It is important to note that this regularized model assumes independence across sites, which is not a realistic assumption, so I consider extensions to models that account for coevolving sites.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

While obtaining a PhD requires an extraordinary amount of work and effort on the part of the graduate student, there are always others who assist in this endeavor as well. I want to thank all my mathematics teachers in secondary and high school for helping to cultivate my raw ability. In particular, I'm deeply indebted to my high school geometry teacher, Mehebub Karmali, for introducing proof-based mathematics to me and showing me that mathematics was more than just memorizing formulas and "plugging and chugging". In college, I was interested in Business Administration as a major and might not have studied Statistics if it were not for the Student Learning Center (campus tutoring center). I want to thank Mike Leong and Mike Wong, the mathematics/statistics division leaders, for fostering a great learning environment for myself and others and showing me how to effectively teach mathematical and statistical principles to other students. These pedagogical skills have helped me when I need to give presentations to a wide variety of audiences. I am grateful to Jon McAuliffe for teaching an introductory course in machine learning that helped motivate me to want to go to graduate school. Jon and Xin Guo, my stochastic processes instructor, both believed I could be successful as a graduate researcher despite my lack of coursework in theoretical mathematics and I thank them for having confidence in my ability and writing recommendation letters for me. In addition, I am appreciative to Vladimir Minin, Galen Shorack, and Marina Meila for also providing recommendation letters when I applied to the Statistics PhD program here at the University of Washington. I also want to thank my PhD committee (Vladimir Minin, Erick Matsen, Noah Simon, and Leo Stamatatos) for reading my thesis and providing useful insight on my work.

One of the best parts about graduate school was being surrounded by amazing peers.

I am thankful to Arman Bilge and Andy Magee for being willing to discuss any and all phylogenetics-related and programming topics. Without the help of Duncan Ralph, the `linearham` inference program could not have been constructed so I'm extremely grateful for his assistance in that project. Kristian Davidsen was a joint first author with me on the work discussed in Chapter 5 so I want to thank him for his contributions and his pleasant attitude. Jean Feng and Tyler Starr both made important contributions to the `linearham` project and I am appreciative of their assistance. Throughout my time at the University of Washington, I collaborated with Laura Doepker, an immunologist at the Fred Hutchinson Cancer Research Center, and she helped me understand how to effectively communicate and deliver statistical insight to audiences not necessarily familiar with statistical methodology.

I've had amazing friends growing up who've supported me and made life more enjoyable than it should have been. My high-school and college friends (Zoheb Allam, David Pereira, Ronak Naik, Ravdeep Grewal, Paul Escobar, and Neha Agarwal) have stuck with me despite my being so busy with graduate work and I'm grateful for all the good times we've had together and hopefully have many more. I'm happy to have made a great friend in Wenxiao Gu during my time as a Masters student and am appreciative of all his support he has thrown my way. I'm thankful to my two Saudi Arabian ex-roommates, Mohammed Alhubail and Omar Adel AlSughayer, for all the fun we had living in Stevens Court. I'm grateful to Christopher Aicher, Corinne Jones, Eric Kernfeld, Wesley Lee, Kyle Lo, Amy Marsha, Hoiyi Ng, and Luca Weihs for being great friends and playing an excessive amount of board/card games with me, including "Dungeons & Dragons", "One Night Ultimate Werewolf", "Dead of Winter: A Crossroads Game", and the ever-popular "A Game of Thrones: The Board Game Second Edition". I am appreciative of "Marketing Intern Level III" Michael Logsdon, "Director of Brand Management" Ben Logsdon, and "Concerned Citizen" Zach Stednick for fun times at Big Time Brewery discussing sports, local politics, Kwame Brown, and things of that nature. I want to send a special shout-out to my man, Stephen A. Smith, for being

the hype man of my life. Of course, I'm also thankful for the love and support from my girlfriend, Tierney Burkhardt, and her two cats, Dinah and Oliver.

My family also had a huge impact in how I became who I am today. I'm grateful for being raised by wonderful parents, Sunil Dhar and Renuka Dhar, who encouraged me to pursue further studies. I'm also thankful for my brother, Ankur Dhar, for not only being supportive of my work but also understanding the struggles with being a PhD student firsthand. My cousins (Vineet Bhan, Nikhil Bhan, Aneesh Dhar, Anmol Dhar, and Himani Dhar) were a big part of my life during childhood and now so I'm grateful for their support in this venture of mine. I'm also lucky to have wonderful grandparents (Omkar Nath Dhar, Raj Dulari Dhar, Moti Lal Pandit, and Rita Pandit), who were always so concerned about the progress of my PhD. My cousin, Viresh Thusu, suggested that I get a Masters even when I was rejected from every PhD program I applied to the first go-around and if I hadn't heeded his sage advice, I might not be where I am today.

My two advisors, Vladimir Minin and Erick Matsen, were so incredible and instrumental to my success during my PhD. Both of them were extremely helpful when I was stuck on problems, easy to work with, and encouraged me to take my research in whatever direction I wanted. Vladimir had deep expertise in statistical methods development, but he always reminded me to think about the scientific problem of interest as the goal of developing methods should always be tied to solving a scientific question. Erick had a similar mindset about statistical methods development and through him, I was able to collaborate with immunologists who were interested in understanding B cell receptor properties. There was also a heavy computational focus in Erick's lab and my experience in his lab helped me become proficient in programming. Usually, it is a graduate student's dream to have a good advisor and in my case, I was lucky to have two of them. It's been an amazing experience working under both of them and I'm proud to call myself a student of theirs.

DEDICATION

To my late grandfather, Omkar Nath Dhar

Chapter 1

INTRODUCTION TO THE IMMUNE SYSTEM

1.1 Overview

The immune system is a complex host defense mechanism that neutralizes a wide variety of pathogens such as bacteria, viruses, and other parasites. The two main components of the immune system are the innate immune system and the adaptive immune system. The innate immune system exists in most animals and presents an immediate defense that is not specialized to any particular pathogen, while the adaptive immune system is found only in vertebrates and adapts to protect the host against specific invaders. Both subsystems work together to mount an effective immune response. For instance, the innate immune system activates and helps guide the activities of the adaptive immune system and the adaptive immune system aids the innate system in killing pathogens. To illustrate the interdependence between the two systems in more detail, we describe a hypothetical immune response to a microbial infection.

The first line of defense against invading microbes consists of the physical barriers such as skin surfaces and mucous membranes that line our digestive, respiratory, and reproductive tracts. If the pathogens are able to cross these physical barriers into the blood stream, the innate system then generates a quick response to try neutralizing them. Some of the innate system's defense mechanisms include macrophages, complement system proteins, and natural killer (NK) cells. Macrophages are a type of white blood cell that live in tissues waiting for microbes to enter the body; think of them as the immune system's "sentinels". They are naturally attracted to foreign microorganisms and are able to ingest and destroy them via the phagocytosis process. If the ratio of microbes to macrophages is too high at the site of infection, the macrophages secrete proteins called cytokines, which induce other immune

cells such as neutrophils and NK cells to help battle the infection. Neutrophils are short-lived white blood cells whose sole purpose is to kill pathogens via phagocytosis; NK cells perform a similar role, but focus on destroying compromised host cells such as tumor cells and virus-infected cells. If necessary, neutrophils and NK cells also release cytokines to attract more innate immune cells to the fight. Complement proteins circulate in the blood and tissues and interact with the other innate immune cells to contribute to the immune response. Some of these proteins combine together to form membrane attack complexes, which puncture the cell membranes of pathogens and kill them. They also are able to encourage more macrophages and neutrophils to join the immune effort and opsonize microbes (i.e. prepare them for phagocytosis). While we did not discuss all the connections between the agents of the innate immune system, one can clearly see that there is an immunological chain reaction associated with the innate system's response to a microbial infection. Often, the infecting pathogens are able to mutate fast enough to render the innate system's defenses useless; in this case, the innate immune system activates the adaptive immune system to assist in combating the rapidly evolving infection.

Unlike the innate immune system, the adaptive immune system can calibrate the immune response to the distinct pathogens in the body using B cells and T cells. These cells are derived from blood stem cells in the bone marrow and have receptors on their cell surfaces that are specific for a given pathogen. B cells secrete antibodies, the soluble form of B cell receptors (BCRs), that either opsonize their specific target microorganisms or neutralize them directly. More importantly, B cells undergo a Darwinian process of mutation and selection that improves their binding affinity for their target pathogen. Before B cells can begin to assist in the immune response, they must be activated by mature T cells.

In order for T cells to mature, they must be presented with antigen (i.e. a microbial surface protein) by an antigen-presenting cell. An antigen-presenting cell (APC) is a cell that displays antigen fragments on the major histocompatibility complex (MHC) molecules on its surface. Most cells in the body have class I MHC molecules on their surfaces; whereas only B cells, dendritic cells, and macrophages express class II MHC molecules on their

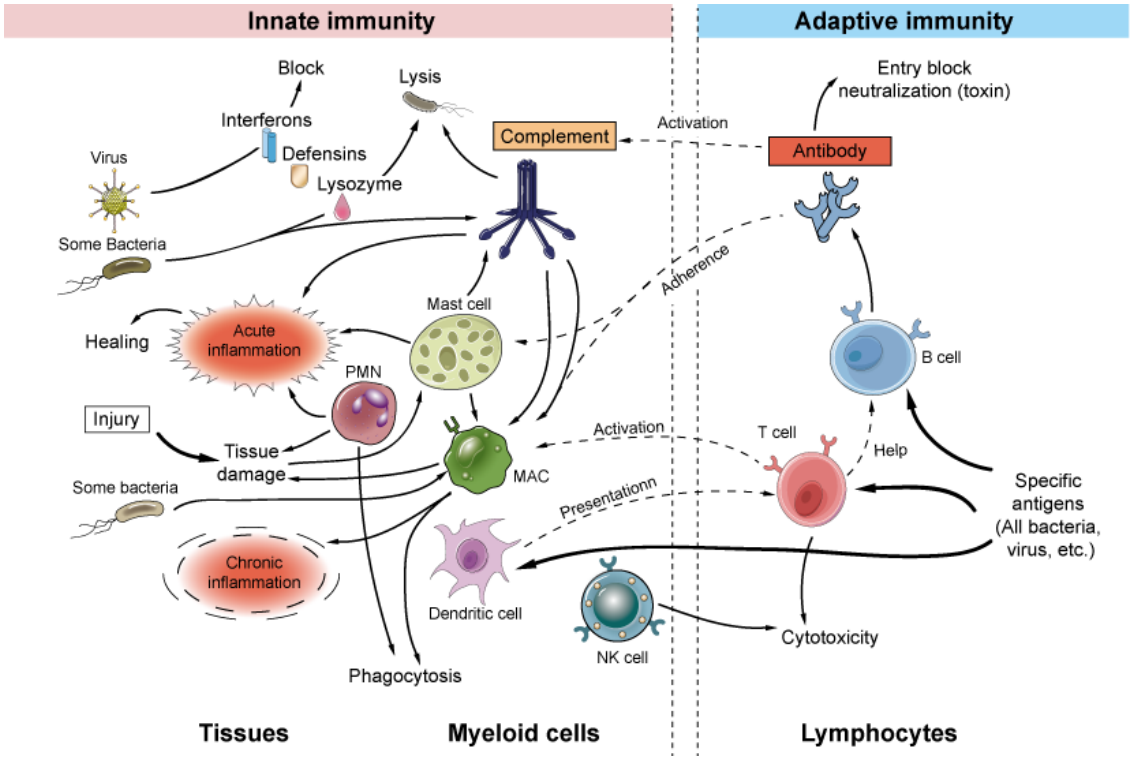


Figure 1.1: A diagram illustrating the connections between the innate and adaptive immune systems. Note that MAC and PMN stand for macrophage and polymorphonuclear leukocyte, respectively. This image was taken from `http://www.creative-diagnostics.com/innate-and-adaptive-immunity.htm`.

membranes. A killer T cell can recognize antigens on the class I MHC molecules of cells; if this T cell receptor (TCR) is specific for the antigen on these MHC molecules, then the bound cell is destroyed. In contrast, a helper T (Th) cell can only be activated if its TCR binds to antigen on class II MHC molecules. For example, if a macrophage has phagocytosed a microbe, it will display the antigen fragments on its class II MHC surface molecules. If the TCR on a Th cell binds to the antigen on the macrophage's class II MHC molecules, then the activated Th cell will release cytokines to help influence the activity of other immune cells and proliferate to build a collection of mature Th cells with the same antigen specificity. These mature Th cells can then be used to, among other things, activate the maturation process of B cells.

However, before a B cell can activate, its BCR must bind to its cognate antigen. Once this process occurs, the antigen is transferred inside the B cell and broken down into smaller peptide fragments; these fragments are then presented on the class II MHC molecules of the B cell. The mature Th cell that is specific to that antigen can now bind to the B cell and activate it. Upon activation, the B cell undergoes rapid proliferation and maturation. The adaptive immune system can not only respond to immediate pathogenic threats by generating fine-tuned antibodies but also initiate a faster immune response if reinfected by the same pathogens. While we have not described every single connection between the innate and adaptive immune systems (Figure 1.1), we hope it is now easy to see how the two subsystems cooperate to mount an effective immune response.

1.2 B Cell Receptors: Review

Before we can discuss the B cell maturation process in more detail, we first must describe the structure of a BCR and explain how the BCRs on "naive" (i.e. inactivated) B cells are created. BCRs are Y-shaped proteins located on the outer surface of B cells (Figure 1.2) and composed of heavy chain and light chain proteins. These heavy and light chain sequences are constructed from variable (V), diversity (D), and joining (J) germline gene segments (i.e. DNA sequences).

The body is able to generate a diverse set of naive BCR heavy chain proteins due to the VDJ rearrangement process (Figure 1.3). In this process, the B cell first randomly selects V, D, and J gene segments from the respective gene pools in the body. Before joining the gene segments together, the B cell randomly deletes nucleotides at both ends of the V-D and D-J junction regions (i.e. exonuclease deletions) and randomly inserts nucleotides in the same junction regions (i.e. non-templated insertions). The light chain sequence for a naive B cell is constructed using a similar rearrangement process that does not include D germline genes. The heavy and light chain sequences are then bound together to form the antigen-binding or variable (Fab) region of the BCR. Each BCR has two identical Fab regions because humans are diploid organisms and a BCR must be specific to only a single antigen. Although the

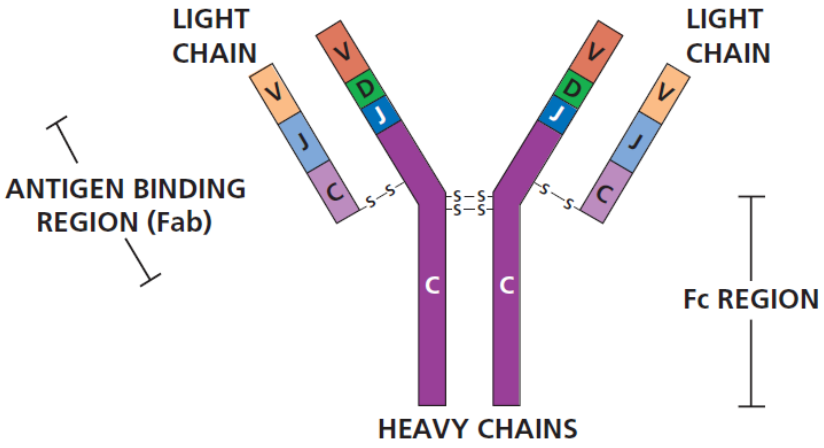


Figure 1.2: An illustration of a BCR. Each BCR consists of two identical pairs of heavy chain and light chain sequences; together, these proteins make up the antigen-binding or variable regions of the BCR. The BCR constant regions determine the antibody class of the B cell; by default, the constant regions for IgM or IgD are used to form the naive BCR. This image was taken from (Sompayrac, 2015).

VDJ rearrangement process samples germline genes from the same gene pools for all the naive BCR Fab regions in a given individual, different people may have different collections of germline genes. The constant (Fc) region of the BCR specifies the class of antibody that can be secreted from the B cell; naive B cells default to using IgM or IgD constant regions.

The heavy and light chain sequences can be partitioned into framework (FWK) and complementarity-determining (CDR) regions. The BCR binding affinity is largely determined by the sequence segments in the CDR regions. Among all the CDR regions, the CDR3 region contributes the most to antigen-binding specificity and has the highest amount of sequence variability. The latter is not too surprising because the CDR3 portion of the heavy chain sequence contains V, D, and J gene nucleotides as well as non-templated insertions (Figure 1.3). The FWK regions encode structural and possibly functional constraints of the BCR and thus can be strongly conserved sections of the heavy and light chains.

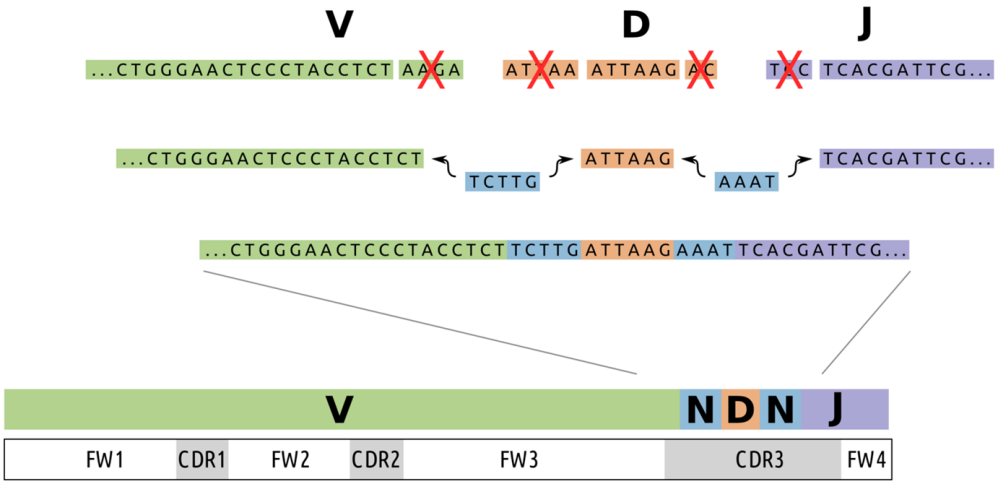


Figure 1.3: A pictorial representation of the heavy chain VDJ rearrangement process. First, V (green), D (orange), and J (purple) genes are randomly selected from the respective gene pools in the body. Then, nucleotides are randomly deleted (red X's) from both ends of the V-D and D-J junction regions and random bases (blue) are added to the same junction regions before the V, D, and J germline genes can be joined together. The BCR heavy and light chain sequences can be partitioned into framework (FWK) and complementarity-determining (CDR) regions. This image was taken from (Ralph and Matsen IV, 2016a).

1.3 B Cell Maturation Process

Once a naive B cell is activated in a lymph node or the spleen, it forms a germinal center (GC), which is a specialized microenvironment where B cell maturation occurs. A GC can be divided into a dark zone and a light zone. In the dark zone of a GC, the naive BCR first accumulates point mutations according to a neutral evolutionary process called somatic hypermutation (SHM), which may modify the associated binding affinity, and then undergoes cellular replication. The SHM mutations are influenced by the local sequence context and concentrated in the CDR regions. After several rounds of mutation and cell division in the dark zone, the BCRs then travel to the light zone, where natural selection takes place. In the light zone, BCRs compete for limiting growth resources, which includes the availability of antigen. The BCRs with high binding affinity are favored for positive selection leading to their survival, while the low affinity BCRs are either destroyed or sent back to the dark

zone to go through more rounds of SHM and proliferation. This process of B cell mutation, expansion, and selection produces a collection of B cells with receptors that have improved binding specificities for their cognate antigen.

In addition to promoting high affinity BCRs, the GC also induces B cells to experience class switching. Class switching is a procedure that changes the antibody class of a B cell by altering the Fc region of the BCR. Thus, BCRs that originally released IgM or IgD antibodies can instead encode IgG, IgA, or IgE antibodies and maintain the same level of antigen-binding specificity. Before the mature B cells can exit the GC, each B cell must choose to become either a plasma B cell or a memory B cell. A plasma B cell serves as an antibody factory to fight immediate pathogenic threats and has a short lifespan, while a memory B cell initiates a faster immune response in case of a reinfection and lives longer than a plasma B cell. Given this complex maturation process, it should now be clear that B cell diversification is one of the most important functions of the adaptive immune system. For more information on the immune system, we refer readers to (Sompayrac, 2015) and (Murphy and Weaver, 2016).

In this thesis, we analyze the SHM and clonal selection processes that govern B cell maturation by using phylogenetic and machine learning tools. Phylogenetic tree models provide a realistic and mathematically convenient way to represent the B cell evolutionary dynamics in a GC and we describe these models in the context of maximum likelihood estimation in Chapter 2. In particular, we would like to infer the mutational pathways that result from SHM and discuss how stochastic mapping can be used to estimate these substitution histories in Chapter 3. We also describe a specialized phylogenetic model for SHM in Chapter 4 that more accurately portrays the B cell development process. Understanding how the SHM and clonal selection processes work together to form high affinity antibodies is also of the utmost importance and Chapter 5 details our efforts in achieving this goal. We conclude this thesis in Chapter 6 by discussing some future directions of this current work.

Chapter 2

MAXIMUM LIKELIHOOD INFERENCE

2.1 Introduction

Maximum likelihood estimation is an extremely popular statistical inference framework that is used to estimate the parameters in a probabilistic data generating model. This conceptually simple method provides parameter estimates that have good statistical properties. The maximum likelihood principle suggests that the best estimates for model parameters are obtained by picking the parameter values that make the probability of observing the data (i.e. likelihood) the highest. Under the assumption of independent and identically distributed data, these maximum likelihood estimates (MLEs) tend to the true, unknown parameter values as the number of observations tends to infinity — this is called consistency in statistics. Also, these estimators use available data in the most optimal way, again as one observes more and more data — this is called efficiency in statistics. Under fairly general and reasonable regularity conditions, consistency and efficiency hold for a large class of maximum likelihood estimators (van der Vaart, 1998). This easy-to-understand estimation principle along with the associated optimality properties for a wide class of likelihood models make maximum likelihood an attractive procedure for many parameter estimation problems, including the problem of estimating phylogenetic relationships from molecular sequence data.

2.2 Phylogenetic Likelihood

2.2.1 Description

Here we will examine the likelihood of molecular sequence data as a function of unknown phylogenetic model parameters, which consist of a tree topology, branch lengths, and continuous-time Markov chain (CTMC) substitution model parameters. Throughout this presentation,

we restrict attention to DNA sequence data for simplicity (although the methods we describe are compatible with other discrete character datasets as well). Suppose we observe m aligned sequences of DNA (potentially corresponding to m distinct species), where each sequence has nucleotide observations recorded at n distinct sites. Gaps in the alignment are usually treated as missing data, but more accurate treatment of insertions and deletions is possible (Redelings and Suchard, 2005; Liu et al., 2012).

Given a tree topology with branch lengths, we use a CTMC substitution model to calculate the probabilities of state changes along the branches of the tree. Specifically if t denotes a branch length on a tree, CTMC substitution models allow one to calculate $p_{ij}(t)$, which denotes the probability of going from state i to state j on a branch of length t, where $i, j \in \{A, G, C, T\}$. It is common to use a reversible CTMC substitution model on a tree (Felsenstein, 2004); a reversible substitution model is a Markov substitution model that, if started at stationarity, can be run backwards in time, with the resulting backward Markov model following the same probability law as the original forward model. Note that branch lengths are commonly measured in expected number of substitutions per site, not in clock time, because estimating substitution rates and branch lengths in units of clock time requires additional information about branching and/or sampling times in the phylogeny (Drummond et al., 2006).

Two assumptions are made that are crucial to the rest of the analysis (Felsenstein, 2004):

1. Evolution at different sites (on a given tree) is independent.

2. Conditional on the internal node states, evolution proceeds independently on different branches of the phylogeny.

Let $L(\tau, \mathbf{t}, \boldsymbol{\theta})$ be the likelihood corresponding to the $m \times n$ DNA sequence alignment matrix $\mathbf{y}$ for a given tree topology τ with branch length vector $\mathbf{t}$ and substitution model parameter

vector $\boldsymbol{\theta}$. We write the likelihood as:

$$L(\tau, \mathbf{t}, \boldsymbol{\theta}) = \Pr(\mathbf{y}; \tau, \mathbf{t}, \boldsymbol{\theta}) = \prod_{i=1}^{n} \Pr(\mathbf{y}_i; \tau, \mathbf{t}, \boldsymbol{\theta}) = \prod_{i=1}^{n} L_i(\tau, \mathbf{t}, \boldsymbol{\theta}), \tag{2.1}$$

where $\mathbf{y}_i$ is the $m \times 1$ vector of observed nucleotides at the ith site and $L_i(\tau, \mathbf{t}, \boldsymbol{\theta})$ is the site i likelihood. This factorization follows directly from the first independence assumption given above. Thus, we can find the likelihood of the whole sequence matrix by finding the likelihoods for each of the n sites. Suppose we observed the nucleotide vector (A, T, C, T) at a particular site (assuming there were only $m = 4$ aligned sequences). We will use the example tree τ_{ex} given in Figure 2.1 to help illustrate how to calculate the likelihood at this site.

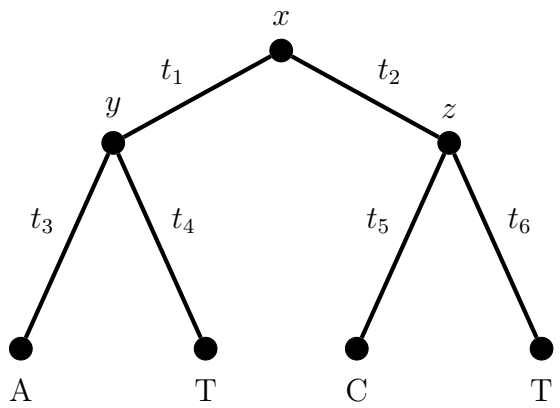


Figure 2.1: An example phylogenetic tree. Letters x, y, z represent the unobserved internal node states where x is associated with the root node, τ_{ex} specifies the tree topology, and $\mathbf{t}_{ex} = (t_1, t_2, ..., t_6)$ denotes the vector of branch lengths. Given the tree topology τ_{ex} and branch lengths $\mathbf{t}_{ex}$, we can calculate the likelihood of observing the nucleotide vector (A, T, C, T).

Using the tree τ_{ex}, we decompose the likelihood of this nucleotide vector in the following

manner:

$$\begin{aligned}\Pr(A,T,C,T;\tau_{ex},\mathbf{t}_{ex},\boldsymbol{\theta}) &= \sum_x \sum_y \sum_z \Pr(A,T,C,T,x,y,z;\tau_{ex},\mathbf{t}_{ex},\boldsymbol{\theta}) \\ &= \sum_x \sum_y \sum_z \pi_x p_{xy}(t_1) p_{xz}(t_2) p_{yA}(t_3) p_{yT}(t_4) p_{zC}(t_5) p_{zT}(t_6),\end{aligned} \tag{2.2}$$

where the summations are over the elements in $\{A, G, C, T\}$. We obtain equation (2.2) by conditioning on the internal node states and by invoking the assumption of independent evolution across branches. Note that, following common practice, we assumed that the initial distribution at the root of the phylogeny is $\boldsymbol{\pi} = (\pi_A, \pi_G, \pi_C, \pi_T)^T$, the stationary distribution of the substitution model (Page and Holmes, 2009). Looking at equation (2.2), it is clear that we will need to take a sum over $4^3 = 64$ probabilities. For only $m = 4$ terminal nodes, this computation is reasonable; as m grows, the computation becomes problematic because the sum will involve 4^{m-1} terms over $m - 1$ internal nodes. Felsenstein (1981) explains how to compute the phylogenetic likelihood in $O(m)$ time (see Chapter 3 for more details).

2.2.2 Root Invariance and the Pulley Principle

Before performing maximum likelihood phylogeny estimation, it is essential to understand the types of phylogenetic trees that will be estimated. Even though it may seem like we are estimating rooted phylogenies, it turns out that under the assumption of substitution model reversibility and without further assumptions or external information maximum likelihood methods can only estimate unrooted trees. This result is a direct consequence of the Pulley Principle, first discussed in (Felsenstein, 1981). Under the assumptions of a reversible substitution model, unconstrained branch lengths, and a root nucleotide distribution starting at stationarity, the Pulley Principle states that the root may be placed anywhere on the tree without affecting the likelihood. This implies that the root is unidentifiable using likelihood methods for phylogeny estimation because the likelihood is invariant to the placement of the root. In fact, we are not estimating a single rooted tree, but an equivalence class of

rooted trees that corresponds to a unique unrooted tree (Felsenstein, 1981). In Figure 2.2, we present two rooted trees that lie in the same equivalence class; the unrooted tree that corresponds to this equivalence class is shown below the two rooted trees. As we shift the root node associated with x, the tree topology and branch length parameters change, but the likelihood value remains the same.

2.3 Implementations

Maximum likelihood methods for phylogenetic inference search for the unrooted tree topology, branch lengths, and substitution model parameters that maximize the phylogenetic likelihood described above. Because the parameter space of this phylogenetic model has discrete and continuous components, likelihood optimization is more difficult here than in standard statistics applications. On the one hand, the set of possible topologies is extremely large (Felsenstein, 1981) so various heuristics are used to find the tree topology that has the highest likelihood, while on the other, the branch length and substitution model parameter optimization problems are non-convex, indicating that no existing optimization algorithm can guarantee to solve these problems. Despite these difficulties, there are many software packages that perform maximum likelihood phylogeny estimation.

Three of the more popular software packages for maximum likelihood phylogenetic inference are PHYLIP (Felsenstein, 1989), PhyML (Guindon et al., 2010), and RAxML (Stamatakis, 2014). PHYLIP is one of the oldest and most popular maximum likelihood software packages. The supported data types include DNA sequences, RNA sequences, protein sequences, discrete characters, and continuous characters (i.e. gene frequencies). This package also includes programs to carry out parsimony analysis and distance matrix methods. PhyML works with both nucleotide and protein sequence data, has an easy-to-use online interface, and can handle a wide variety of substitution models, rate heterogeneity across sites, and the phylogenetic bootstrap. RAxML uses parallel processing and hill climbing heuristics to find maximum likelihood phylogenies and also allows for parsimony reconstruction and bootstrapping. Even though likelihood maximization in the context of a phylogenetic model

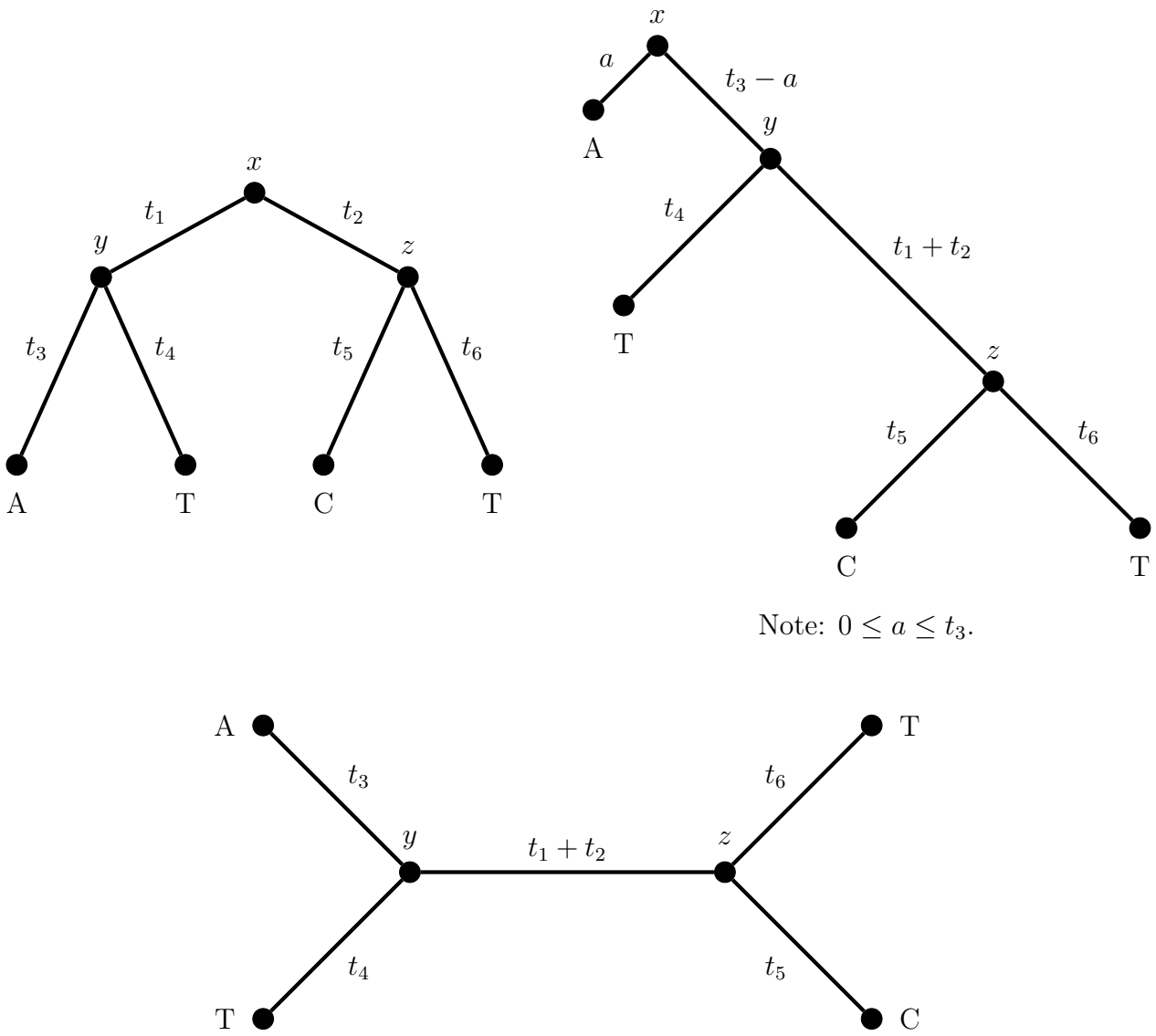


Figure 2.2: Three trees with equivalent likelihood values under the assumptions of the Pulley Principle. The Pulley Principle states that the root of a tree may be placed anywhere on the tree without affecting the likelihood value. The two rooted trees (top) are contained within a larger equivalence class of rooted trees that uniquely corresponds to the given unrooted tree (bottom).

is difficult, the plethora of existing maximum likelihood software tools allows researchers to obtain phylogeny estimates that have associated optimality properties with little effort.

Chapter 3

CALCULATING HIGHER-ORDER MOMENTS OF PHYLOGENETIC STOCHASTIC MAPPING SUMMARIES IN LINEAR TIME

3.1 Introduction

Given a multiple sequence alignment of DNA nucleotides, scientists are often interested in reconstructing a phylogenetic tree to help them learn more about the ancestral relationships between the sequences and the underlying evolutionary process (Yang, 2006). However, in some cases, phylogeny estimation by itself does not provide all the needed information about sequence evolution because we observe data only at the tips of the phylogeny. We do not have much insight into the evolution taking place on the different tree branches other than through the estimated branch lengths, which are usually specified in terms of the expected number of substitutions per site (Felsenstein, 2004, Chapter 13). However, researchers are often interested in making inferences about the evolutionary process on the phylogeny because these inferences could be used to answer important scientific questions. For instance, estimates of non-synonymous/synonymous substitution rate ratios on a phylogeny are commonly used to test for positive selection on protein-coding genes (Nielsen and Yang, 1998). Stochastic mapping can be used to accurately estimate these ratios and, more generally, can help us make reliable inferences about the latent evolutionary process on the phylogeny (Nielsen, 2002; Huelsenbeck et al., 2003; Dimmic et al., 2005; Zhai et al., 2007; Lemey et al., 2012). Stochastic mapping is a simulation-based technique used to probabilistically map substitution histories onto phylogenies according to CTMC models of evolution. This approach was motivated by the need for alternatives to parsimony mapping, which focuses attention on mappings requiring the fewest substitutions. Such minimal mappings obtained

from parsimony analysis need not be realistic.

Stochastic mapping was first introduced by Nielsen (2002), who described how to sample substitution mappings from the posterior probability distribution of mappings for a single trait or a site in a multiple sequence alignment. By using Nielsen's sampling procedure, one can compute Monte Carlo estimates for the posterior mean and/or variance, among other properties of the posterior distribution, of any mutational mapping summary random variable of interest. The two most popular stochastic mapping summaries are the number of particular types of substitutions (labeled substitution counts) and the time spent in a particular group of states (labeled dwelling times) on the tree. In most applications, substitution mappings are simulated only to estimate the mean and/or variance of the two mapping summaries discussed above (Minin and Suchard, 2008b). Recognizing this, Minin and Suchard (2008b) synthesized previous work of Hobolth and Jensen (2005), Dutheil et al. (2005), and Holmes and Rubin (2002), among others, and developed an efficient algorithm that analytically calculates the expectations of the aforementioned mapping summaries. The authors compute restricted expectations of CTMC labeled substitution counts (Ball and Milne, 2005; Minin and Suchard, 2008a) and labeled dwelling times (Neuts, 1995; Guindon et al., 2004; Minin and Suchard, 2008b) on each tree branch and propagate these expectations across the phylogeny using a generalized pruning algorithm (Felsenstein, 1981). Similar to Felsenstein's pruning algorithm, the algorithm of Minin and Suchard (2008b) scales linearly in the number of phylogeny tips. Minin and Suchard (2008b) drew inspiration from the work of Schadt et al. (1998), who formulated a similar algorithm that computes first derivatives of phylogenetic likelihood functions. Unfortunately, it is not straightforward to extend the algorithm of Minin and Suchard (2008b) to efficiently calculate the variances of the previously mentioned mapping summaries; as a result, these stochastic mapping variances can only be approximated using Monte Carlo simulations.

In this chapter, we present a simulation-free dynamic programming algorithm that calculates prior and posterior mapping variances and scales linearly in the number of phylogeny tips. We draw upon concepts introduced by Kenney and Gu (2012), who developed a recur-

sive procedure for calculating second derivatives of phylogenetic likelihood functions. Our procedure suggests a general framework that can be used to efficiently compute higher-order moments of stochastic mapping summaries without simulations. The structure of the rest of the chapter is as follows. Section 3.2 introduces notation that is used throughout the entire chapter and discusses our research problem more formally. In Section 3.3, we give a description of our algorithm for efficiently calculating these stochastic mapping variances. In Section 3.4, we demonstrate the usefulness of our algorithm by extending previously developed statistical tests for rate variation across sites and for detecting evolutionarily conserved regions in genomic sequences. Concluding remarks are presented in Section 3.5.

3.2 Notation and Problem Background

We use much of the notation provided by Minin and Suchard (2008b). Suppose we have a discrete evolutionary trait X (i.e. DNA base) that takes on m distinct states and that evolves according to a CTMC on a phylogeny. This evolutionary process, call it $\psi_{\boldsymbol{\theta}}$, depends on the parameter set $\boldsymbol{\theta} = \{\tau, \mathbf{t}, \mathbf{Q}, \boldsymbol{\pi}\}$, which consists of a rooted tree topology τ with n tips and $B_n = 2n - 2$ branches; branch lengths $\mathbf{t} = (t_1, ..., t_{B_n})$; a reversible CTMC rate matrix $\mathbf{Q} = \{q_{ij}\}$ for $i, j = 1, ..., m$; and a CTMC stationary distribution $\boldsymbol{\pi} = (\pi_1, ..., \pi_m)^T$. We assume that our evolutionary process starts at stationarity (i.e. we assume that the root distribution is equal to $\boldsymbol{\pi}$). While not necessary, this commonly used assumption ensures that the stochastic mapping moments will be invariant to the placement of the root (Minin and Suchard, 2008b). When this assumption is not used, as often is the case in analyses of morphological traits (Pagel, 1999), our methods still work without modification, but the root of the tree has to be specified by the user. Matrix $\mathbf{P}(t) = \{p_{ij}(t)\} = \exp(\mathbf{Q}t)$ represents the CTMC transition probability matrix for a branch of length t.

We define $\Theta = \{1, ..., B_n\}$ to be the set of branch indices of τ. Let $\Theta_b = \{b^* \in \Theta \mid b^* \preceq b\}$ denote the set of branch indices in the subtree relating all descendants of branch b, including b, where $b^* \preceq b$ for $b, b^* \in \Theta$ if either b^* is a descendant of b or b^* is equal to b. Let $\mathcal{I} \subset \Theta$ represent the set of internal branches (i.e. branches that connect two internal

nodes) and $\mathcal{E} = \Theta \setminus \mathcal{I}$ represent the set of terminal branches (i.e. branches that connect an internal node to a tip node). Let $\mathbf{D} = (D_1, ..., D_n)$ denote the trait values observed at the n tips of τ, $\mathbf{D}_{1:L} = \{\mathbf{D}_1, ..., \mathbf{D}_L\}$ signify an alignment of length L, and $\mathbf{i} = (i_1, ..., i_{n-1})$ represent the unobserved internal node states of τ. In addition, we let $\mathbf{i}_b$ be the vector of internal node states for the subtree strictly beneath branch b. Note that the internal nodes of τ are labeled with integers $\{1, ..., n-1\}$ starting from the root of the tree; the corresponding labels of the branches and tips of τ are assigned arbitrarily. We also introduce $\mathbf{i}^* = (i_1^*, ..., i_{n-1}^*, i_n^*, ..., i_{2n-1}^*) = (i_1, ..., i_{n-1}, D_1, ..., D_n)$, which is the concatenation of $\mathbf{i}$ and $\mathbf{D}$. For each branch $b \in \Theta$, $p(b)$ and $c(b)$ represent the node labels (in $\mathbf{i}^*$) of the parent and child of branch b, respectively.

Most stochastic mapping applications infer properties about the evolutionary process on the phylogeny through the use of a summary measure H. We restrict attention to additive mapping summaries of the form:

$$H_\Omega \equiv H_\Omega(\mathbf{M}) = \sum_{b \in \Omega} h(\{X_{bt}\}), \tag{3.1}$$

where $\mathbf{M} = (\{X_{1t}\}, ..., \{X_{B_n t}\})$ denotes the collection of CTMC trajectories along the branches of τ, $\Omega \subseteq \Theta$ represents a predefined set of branch indices, and h signifies a summary measure applied to a single CTMC trajectory. Let $\mathcal{L} \subset \{1, ..., m\}^2$ be a set of state pairs that labels substitutions of trait X and $\mathbf{w} \subset \{0, 1\}^m$ be a set that labels states of trait X. For any given CTMC path $\{X_t\}$ in $[0, t)$, the two most popular choices of h are $h_1(\{X_t\})$, which counts the number of substitutions labeled by set $\mathcal{L}$, and $h_2(\{X_t\})$, which measures the dwelling time in states labeled by set $\mathbf{w}$ (Minin and Suchard, 2008b). In this chapter, we work exclusively with the summary function h_1 as this summary measure is used in both of our scientific applications. However, we do note that our algorithmic results hold true regardless of the specific summary measure used.

Minin and Suchard (2008b) were able to calculate the posterior mapping expectation $\mathrm{E}(H_\Omega|\mathbf{D})$ for both h_1 and h_2 in $O(n)$ time and with $O(n)$ storage but were unable to achieve

the same space-time complexity when calculating the posterior mapping variance $\mathrm{Var}(H_\Omega|\mathbf{D})$. Before we delve into the difficulties associated with computing $\mathrm{Var}(H_\Omega|\mathbf{D})$, we refresh our readers on two important quantities:

$$\mathrm{E}(H_\Omega^2 \mathbb{1}_{\mathbf{D}}) = \mathrm{E}(H_\Omega^2|\mathbf{D}) \times \mathrm{P}(\mathbf{D}), \tag{3.2}$$

$$e_{ij}^{[k]}(h,t) = \mathrm{E}\Big\{ h(\{X_t\})\big[h(\{X_t\})-1\big]...\big[h(\{X_t\})-k+1\big]\mathbb{1}_{\{X_t=j\}}\Big| X_0 = i\Big\}, \tag{3.3}$$

where $\mathbb{1}_{\{\cdot\}}$ represents the indicator function; $k = 1, 2, ...$; $i, j = 1, ..., m$; and $\mathrm{P}(\mathbf{D})$ denotes the phylogenetic likelihood defined as the probability of observing the tip sequence $\mathbf{D}$. Equation (3.2) connects the restricted mapping second moment $\mathrm{E}(H_\Omega^2 \mathbb{1}_{\mathbf{D}})$ to the posterior mapping second moment $\mathrm{E}(H_\Omega^2|\mathbf{D})$. As Minin and Suchard (2008b) state, the restricted expectation in equation (3.2) integrates over all evolutionary mappings consistent with $\mathbf{D}$ on the tips of τ. Since $\mathrm{P}(\mathbf{D})$ can be easily computed using the pruning algorithm (Felsenstein, 1981), we focus our attention on calculating $\mathrm{E}(H_\Omega^2 \mathbb{1}_{\mathbf{D}})$. Quantity $e_{ij}^{[k]}(h,t)$ denotes the kth restricted factorial moment of $h(\{X_t\})$ for a CTMC path $\{X_t\}$ in $[0,t)$ that starts in state i and ends in state j. We let $\mathbf{e}^{[k]}(h,t) = \{e_{ij}^{[k]}(h,t)\}$ represent the corresponding restricted factorial moment matrix. Minin and Suchard (2008a) derive a simple recurrence relation to calculate $\mathbf{e}^{[k]}(h_1,t)$ for $k = 1, 2, ...$; a similar relation exists for h_2 as well (Minin and Suchard, 2008b).

To help us illustrate the computational challenges inherent in calculating $\mathrm{Var}(H_\Omega|\mathbf{D})$, we express $\mathrm{E}(H_\Omega^2 \mathbb{1}_{\mathbf{D}})$ in the following manner (suppressing the fact that $b, b' \in \Omega$ for brevity):

$$\mathrm{E}(H_\Omega^2 \mathbb{1}_{\mathbf{D}}) = \mathrm{E}\left[\left(\sum_b h(\{X_{bt}\})\right)^2 \mathbb{1}_{\mathbf{D}}\right] \tag{3.4}$$

$$= \sum_b \mathrm{E}\big[h(\{X_{bt}\})^2 \mathbb{1}_{\mathbf{D}}\big] + \sum_{b\neq b'} \mathrm{E}\big[h(\{X_{bt}\})h(\{X_{b't}\})\mathbb{1}_{\mathbf{D}}\big] \tag{3.5}$$

$$= \sum_b \sum_{\mathbf{i}} \mathrm{E}\big[h(\{X_{bt}\})^2|\ \mathbf{i},\mathbf{D}\big]\mathrm{P}(\mathbf{i},\mathbf{D}) + \sum_{b\neq b'} \sum_{\mathbf{i}} \mathrm{E}\big[h(\{X_{bt}\})h(\{X_{b't}\})|\ \mathbf{i},\mathbf{D}\big]\mathrm{P}(\mathbf{i},\mathbf{D}) \tag{3.6}$$

$$
\begin{aligned}
&= \sum_{b} \sum_{\mathbf{i}} \mathrm{E}\big[h(\{X_{bt}\})^2 |\ i^*_{p(b)}, i^*_{c(b)}\big] \pi_{i^*_1} \prod_{b^* \in \Theta} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}) \\
&+ \sum_{b \neq b'} \sum_{\mathbf{i}} \mathrm{E}\big[h(\{X_{bt}\}) h(\{X_{b't}\}) |\ i^*_{p(b)}, i^*_{c(b)}, i^*_{p(b')}, i^*_{c(b')}\big] \pi_{i^*_1} \prod_{b^* \in \Theta} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*})
\end{aligned} \tag{3.7}
$$

$$
= \sum_{b} \sum_{\mathbf{i}} \Big[e^{[2]}_{i^*_{p(b)} i^*_{c(b)}}(h, t_b) + e^{[1]}_{i^*_{p(b)} i^*_{c(b)}}(h, t_b) \Big] \pi_{i^*_1} \prod_{b^* \in \Theta \setminus \{b\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}) \tag{3.8}
$$

$$
+ \sum_{b \neq b'} \sum_{\mathbf{i}} e^{[1]}_{i^*_{p(b)} i^*_{c(b)}}(h, t_b) e^{[1]}_{i^*_{p(b')} i^*_{c(b')}}(h, t_{b'}) \pi_{i^*_1} \prod_{b^* \in \Theta \setminus \{b, b'\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}). \tag{3.9}
$$

The single summation over b in (3.8) can be efficiently computed by utilizing a modified version of the generalized pruning algorithm presented in (Minin and Suchard, 2008b). However, the straightforward double summation over $b \neq b'$ in (3.9) requires at most $O(B_n^2) = O((2n-2)^2) = O(n^2)$ computations. In the next section, we describe how to overcome this computational roadblock via a post-order tree traversal algorithm that computes $\mathrm{E}(H_\Omega^2 \mathbb{1}_{\mathbf{D}})$ (and therefore $\mathrm{Var}(H_\Omega | \mathbf{D})$) in $O(n)$ time and with $O(n)$ storage.

3.3 Methods

3.3.1 Algorithm Setup

To help motivate our procedure and make it easier to understand, we utilize illustrations of various "colored" phylogenies. Figure 3.1 displays the example "uncolored" tree we use to create all these illustrations. Note that for a given $b \neq b'$ where $b, b' \in \Omega$, the corresponding summand in (3.9) is:

$$
\sum_{\mathbf{i}} e^{[1]}_{i^*_{p(b)} i^*_{c(b)}}(h, t_b) e^{[1]}_{i^*_{p(b')} i^*_{c(b')}}(h, t_{b'}) \pi_{i^*_1} \prod_{b^* \in \Theta \setminus \{b, b'\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}). \tag{3.10}
$$

If we replace the restricted first moments in (3.10) with the appropriate transition probabilities, then expression (3.10) would represent the phylogenetic likelihood. From equation (3.5), we know that expression (3.10) is equal to the restricted product moment $\mathrm{E}\big[h(\{X_{bt}\}) h(\{X_{b't}\}) \mathbb{1}_{\mathbf{D}}\big]$.

In Figure 3.1, we present two "colored" phylogenies to help visualize the calculation of $\mathrm{E}\big[h(\{X_{bt}\})h(\{X_{b't}\})\mathbb{1}_{\mathbf{D}}\big]$. Thus, we can visualize the double sum over $b \neq b'$ in (3.9) by imagining the red and blue colorings being permuted across all branches in Ω. The cached vectors used in our procedure are described in a similar fashion.

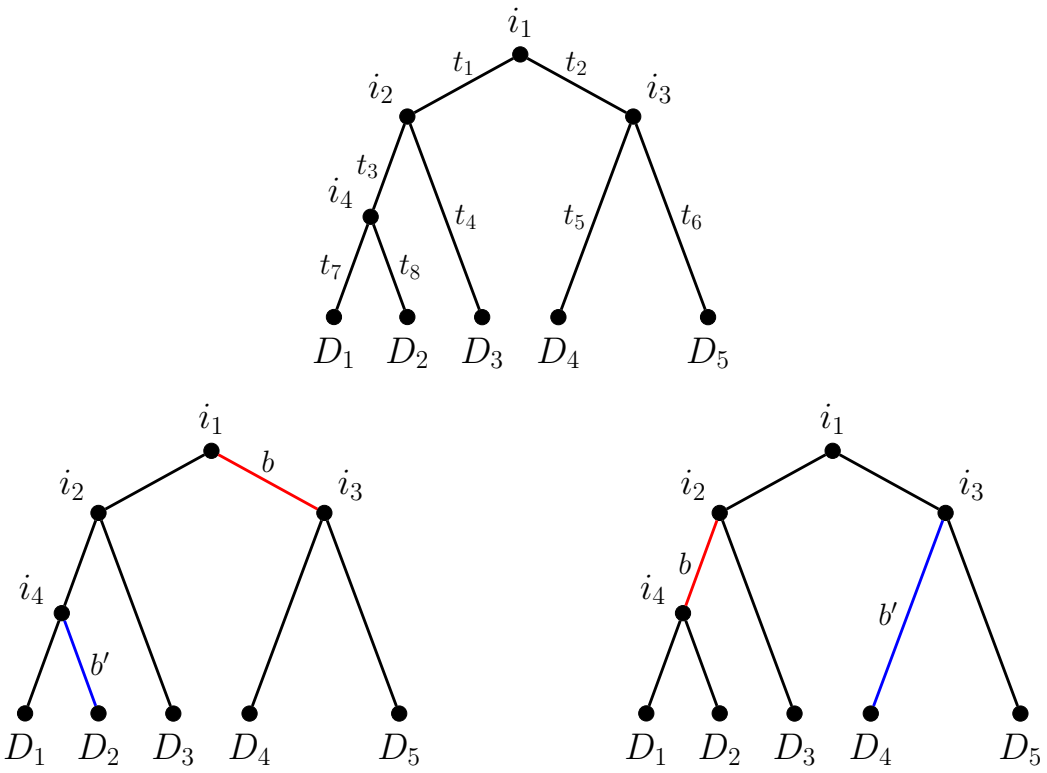


Figure 3.1: The example phylogenies we use to help motivate our algorithmic procedure. (Top) The "uncolored" tree used to create all the "colored" phylogenies. This tree has $n = 5$ tips, $B_n = 8$ branches with branch lengths $\mathbf{t} = (t_1, ..., t_8)$, internal node states $\mathbf{i} = (i_1, ..., i_4)$, and tip states $\mathbf{D} = (D_1, ..., D_5)$. In addition, $\Theta = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $\mathcal{I} = \{1, 2, 3\}$, and $\mathcal{E} = \{4, 5, 6, 7, 8\}$. (Bottom) Two "colored" phylogenies that illustrate the calculation of $\mathrm{E}\big[h(\{X_{bt}\})h(\{X_{b't}\})\mathbb{1}_{\mathbf{D}}\big]$. The colored branches specify the locations of the restricted first moments, while the uncolored branches determine the locations of the transition probabilities. The first tree (left) and second tree (right) visualize the calculations of $\mathrm{E}\big[h(\{X_{2t}\})h(\{X_{8t}\})\mathbb{1}_{\mathbf{D}}\big]$ and $\mathrm{E}\big[h(\{X_{3t}\})h(\{X_{5t}\})\mathbb{1}_{\mathbf{D}}\big]$, respectively, for the "uncolored" tree shown above.

Let $\mathbf{F}_u = (F_{u1}, ..., F_{um})^T$ be the vector of partial likelihoods at node u, where F_{ui} denotes

the probability of the observed data at only the tips that descend from node u, given that the state of node u is i. We let $\mathbf{S}_b = (S_{b1}, ..., S_{bm})^T$ denote the vector of directional likelihoods at branch b, where S_{bi} represents the likelihood of the observed data at only the tips that descend from branch b, given that the state of parent node $p(b)$ is i. Minin and Suchard (2008b) utilize these $\mathbf{F}_u$ and $\mathbf{S}_b$ vectors within their algorithm for computing the posterior mapping expectation $\mathrm{E}(H_\Omega|\mathbf{D})$. We also define vectors $\mathbf{V}_b^{[1]} = \left(V_{b1}^{[1]}, ..., V_{bm}^{[1]}\right)^T$, $\mathbf{V}_b^{[2]} = \left(V_{b1}^{[2]}, ..., V_{bm}^{[2]}\right)^T$, and $\mathbf{W}_b = (W_{b1}, ..., W_{bm})^T$. Elements of vector $\mathbf{V}_b^{[1]}$ are defined as follows:

$$V_{bi}^{[1]} = \sum_{b^\dagger} \sum_{\mathbf{i}_b} e^{[1]}_{i^*_{p(b^\dagger)} i^*_{c(b^\dagger)}}(h, t_{b^\dagger}) \prod_{b^* \in \Theta_b \setminus \{b^\dagger\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}), \tag{3.11}$$

where the state of parent node $p(b)$ is i and $b^\dagger \in \Omega_b$ for the set of branches of interest in the subtree that is "below" branch b (including b), $\Omega_b = \Omega \cap \Theta_b$. An illustration of $\mathbf{V}_b^{[1]}$ is given in Figure 3.2. We can interpret $\boldsymbol{\pi}^T \mathbf{V}_b^{[1]}$ as the restricted mapping first moment $\mathrm{E}(H_{\Omega_b} \mathbb{1}_{\mathbf{D}})$ for the subtree that is "below" branch b (including b). Kenney and Gu (2012) cache a vector similar to $\mathbf{V}_b^{[1]}$ within their algorithm for computing second derivatives of phylogenetic likelihood functions. If we replace the restricted first moment in expression (3.11) with the appropriate transition probability derivative, then we would recover this cached vector used in (Kenney and Gu, 2012). Similarly, $V_{bi}^{[2]}$ is defined the same as $V_{bi}^{[1]}$, except in the case of $V_{bi}^{[2]}$, we replace the restricted first moment in expression (3.11) with the corresponding second restricted factorial moment $e^{[2]}_{i^*_{p(b^\dagger)} i^*_{c(b^\dagger)}}(h, t_{b^\dagger})$. The visual depiction of $\mathbf{V}_b^{[2]}$ is analogous to that of $\mathbf{V}_b^{[1]}$ in Figure 3.2. Elements of vector $\mathbf{W}_b$ are defined as follows:

$$W_{bi} = \sum_{b^\dagger \neq b^{\dagger\dagger}} \sum_{\mathbf{i}_b} e^{[1]}_{i^*_{p(b^\dagger)} i^*_{c(b^\dagger)}}(h, t_{b^\dagger}) e^{[1]}_{i^*_{p(b^{\dagger\dagger})} i^*_{c(b^{\dagger\dagger})}}(h, t_{b^{\dagger\dagger}}) \prod_{b^* \in \Theta_b \setminus \{b^\dagger, b^{\dagger\dagger}\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}), \tag{3.12}$$

where the state of parent node $p(b)$ is i and $b^\dagger, b^{\dagger\dagger} \in \Omega_b$ for Ω_b defined as above. A pictorial representation of $\mathbf{W}_b$ is provided in Figure 3.2. $\boldsymbol{\pi}^T \mathbf{W}_b$ can be viewed as the sum of restricted product moments $\mathrm{E}\left[h(\{X_{b^\dagger t}\}) h(\{X_{b^{\dagger\dagger} t}\}) \mathbb{1}_{\mathbf{D}}\right]$ over $b^\dagger \neq b^{\dagger\dagger}$ $(b^\dagger, b^{\dagger\dagger} \in \Omega_b)$ for the subtree that is "below" branch b (including b). In the next subsection, we describe our algorithm in full

detail and provide some intuition behind the recursive formulas used in our procedure.

3.3.2 Algorithm Recursion

Our post-order tree traversal algorithm recursively computes $\mathbf{F}_u$, $\mathbf{S}_b$, $\mathbf{V}_b^{[1]}$, $\mathbf{V}_b^{[2]}$, and $\mathbf{W}_b$ at all nodes $u \in \{1, ..., n-1, n, ..., 2n-1\}$ and branches $b \in \Theta$. Like the pruning algorithm, this procedure starts at the tips of the tree and continues through all ancestral nodes until it arrives at the root of the tree. We start by describing how these vectors are initialized at the terminal nodes/branches of τ and then specify the recursive formulas used to calculate $\mathbf{F}_u$, $\mathbf{S}_b$, $\mathbf{V}_b^{[1]}$, $\mathbf{V}_b^{[2]}$, and $\mathbf{W}_b$ at the internal nodes/branches of τ.

First, we follow standard practice and set $F_{ui} = \mathbb{1}_{\{i=i_u^*\}}$ for all terminal nodes $u \in \{n, ..., 2n-1\}$ and $i = 1, ..., m$. Oftentimes, we have partially observed and/or missing data at the tips of τ and our initialization of F_{ui} can be adjusted to reflect this information (Felsenstein, 1981). For terminal branches $b \in \mathcal{E}$, we set:

$$S_{bi} = \sum_{j=1}^{m} p_{ij}(t_b) F_{c(b)j}, \tag{3.13}$$

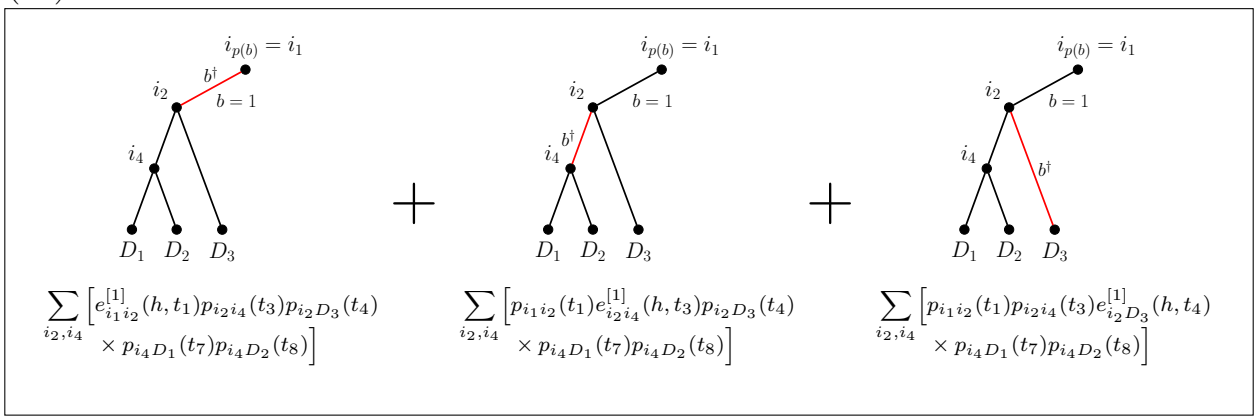
for $i = 1, ..., m$. Using matrix notation, we express the equation in (3.13) as $\mathbf{S}_b = \mathbf{P}(t_b)\mathbf{F}_{c(b)}$. The initializations of $\mathbf{V}_b^{[1]}$ and $\mathbf{V}_b^{[2]}$ depend on whether or not $b \in \Omega$. For all terminal branches $b \in \mathcal{E}$, we define:

$$V_{bi}^{[1]} = \sum_{j=1}^{m} e_{ij}^{[1]}(h, t_b) F_{c(b)j} \mathbb{1}_{\{b \in \Omega\}}, \tag{3.14}$$

$$V_{bi}^{[2]} = \sum_{j=1}^{m} e_{ij}^{[2]}(h, t_b) F_{c(b)j} \mathbb{1}_{\{b \in \Omega\}}, \tag{3.15}$$

for $i = 1, ..., m$. The vectorized representations of equations (3.14) and (3.15) are $\mathbf{V}_b^{[1]} = \mathbf{e}^{[1]}(h, t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b \in \Omega\}}$ and $\mathbf{V}_b^{[2]} = \mathbf{e}^{[2]}(h, t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b \in \Omega\}}$, respectively. Finally, $W_{bi} = 0$ for all terminal branches $b \in \mathcal{E}$ and $i = 1, ..., m$. Note that the definitions of $\mathbf{V}_b^{[1]}$, $\mathbf{V}_b^{[2]}$, and $\mathbf{W}_b$ for

(A)



(B)

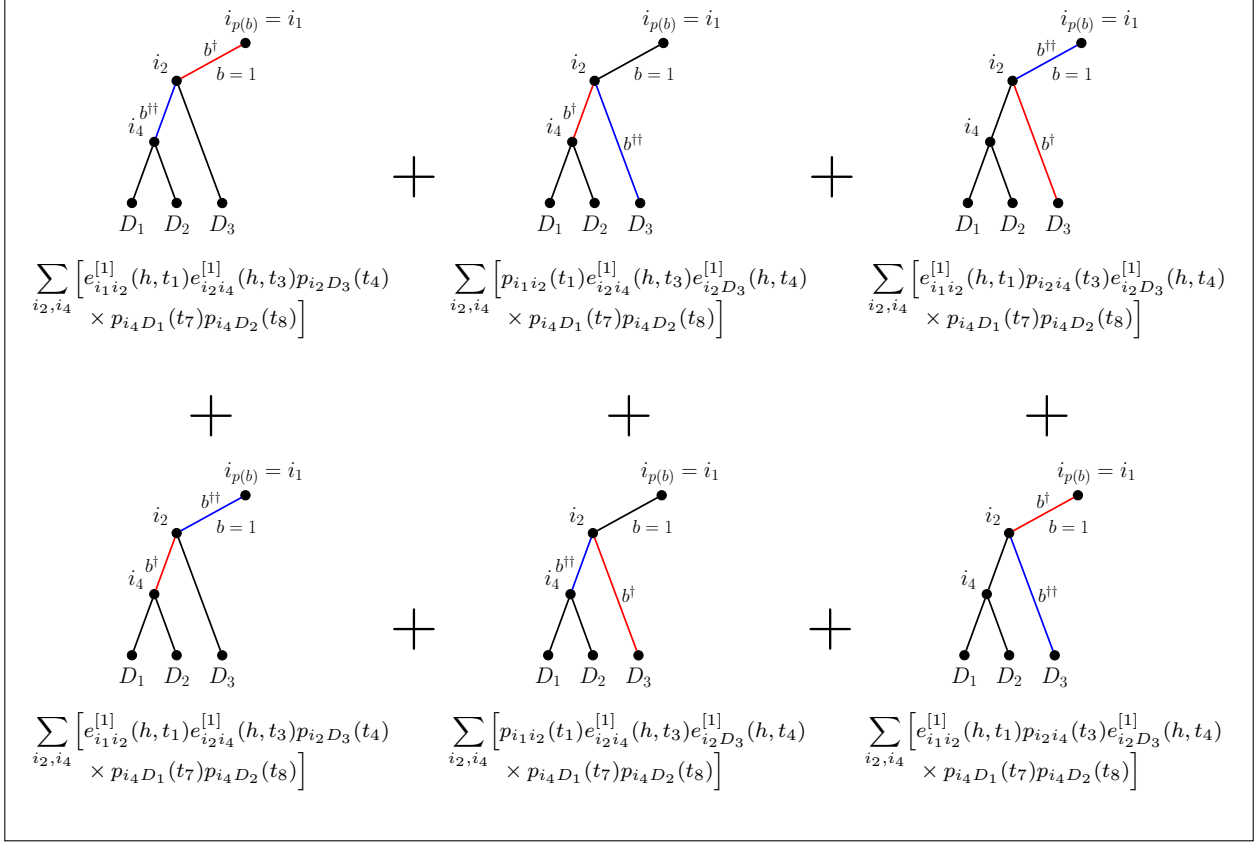


Figure 3.2: Visual depictions of the $\mathbf{V}_b^{[1]}$ and $\mathbf{W}_b$ vectors. (A) An illustration of the $\mathbf{V}_b^{[1]}$ vector. $V_{bi}^{[1]}$ can be interpreted as the sum over all "single-colored" phylogenies for the subtree defined by Θ_b and the predefined set of "colored" branches $\Omega_b = \Omega \cap \Theta_b$, conditional on the state of parent node $p(b)$ being i. We illustrate $\mathbf{V}_1^{[1]}$ for the "uncolored" phylogeny given in Figure 3.1, where $\Omega_1 = \{1, 3, 4\}$. (B) An illustration of the $\mathbf{W}_b$ vector. W_{bi} can be interpreted as the sum over all "double-colored" phylogenies for the same subtree and set of "colored" branches as described in (A), conditional on the state of parent node $p(b)$ being i. We illustrate $\mathbf{W}_1$ for the "uncolored" tree displayed in Figure 3.1, where $\Omega_1 = \{1, 3, 4\}$.

$b \in \mathcal{E}$ are consistent with the illustrations provided in Figure 3.2.

Now, we present the recursive formulas that compute $\mathbf{F}_u$, $\mathbf{S}_b$, $\mathbf{V}_b^{[1]}$, $\mathbf{V}_b^{[2]}$, and $\mathbf{W}_b$ for internal nodes $u \in \{1, ..., n-1\}$ and internal branches $b \in \mathcal{I}$. The recursion for the partial likelihood F_{ui} is the centerpiece of Felsenstein's pruning algorithm (Felsenstein, 1981):

$$F_{ui} = \underbrace{\left[\sum_{j=1}^{m} p_{ij}(t_{b_1}) F_{c(b_1)j}\right]}_{S_{b_1 i}} \times \underbrace{\left[\sum_{j=1}^{m} p_{ij}(t_{b_2}) F_{c(b_2)j}\right]}_{S_{b_2 i}}, \tag{3.16}$$

where b_1 and b_2 represent the two branches connecting node u to its two child nodes and $i = 1, ..., m$. In addition, as shown in the brackets above, equation (3.13) also denotes the recursion for S_{bi} at internal branches $b \in \mathcal{I}$ (Felsenstein, 1981). Thus, the recursive formula for F_{ui} in (3.16) can be compactly expressed as $\mathbf{F}_u = \mathbf{S}_{b_1} \circ \mathbf{S}_{b_2}$, where $\circ$ symbolizes element-wise multiplication between two vectors. The recursive equations used to calculate $V_{bi}^{[1]}$, $V_{bi}^{[2]}$, and W_{bi} at internal branches $b \in \mathcal{I}$ are:

$$V_{bi}^{[1]} = \sum_{j=1}^{m} \left[e_{ij}^{[1]}(h, t_b) F_{c(b)j} \mathbb{1}_{\{b \in \Omega\}} + p_{ij}(t_b) \Big(V_{b_1 j}^{[1]} S_{b_2 j} + V_{b_2 j}^{[1]} S_{b_1 j} \Big) \right], \tag{3.17}$$

$$V_{bi}^{[2]} = \sum_{j=1}^{m} \left[e_{ij}^{[2]}(h, t_b) F_{c(b)j} \mathbb{1}_{\{b \in \Omega\}} + p_{ij}(t_b) \Big(V_{b_1 j}^{[2]} S_{b_2 j} + V_{b_2 j}^{[2]} S_{b_1 j} \Big) \right], \tag{3.18}$$

$$\begin{aligned} W_{bi} = \sum_{j=1}^{m} \Bigg[& 2 \times e_{ij}^{[1]}(h, t_b) \Big(V_{b_1 j}^{[1]} S_{b_2 j} + V_{b_2 j}^{[1]} S_{b_1 j} \Big) \mathbb{1}_{\{b \in \Omega\}} \\ & + p_{ij}(t_b) \Big(2 \times V_{b_1 j}^{[1]} V_{b_2 j}^{[1]} + W_{b_1 j} S_{b_2 j} + W_{b_2 j} S_{b_1 j} \Big) \Bigg], \end{aligned} \tag{3.19}$$

respectively, where b_1 and b_2 represent the two branches that are "below" branch b and $i = 1, ..., m$. The recursive formulas presented in equations (3.17)-(3.19) can also be expressed as:

$$\mathbf{V}_b^{[1]} = \mathbf{e}^{[1]}(h, t_b) \mathbf{F}_{c(b)} \mathbb{1}_{\{b \in \Omega\}} + \mathbf{P}(t_b) \Big(\mathbf{V}_{b_1}^{[1]} \circ \mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{[1]} \circ \mathbf{S}_{b_1} \Big), \tag{3.20}$$

$$\mathbf{V}_b^{[2]} = \mathbf{e}^{[2]}(h, t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega\}} + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{[2]} \circ \mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{[2]} \circ \mathbf{S}_{b_1}\Big), \tag{3.21}$$

$$\begin{aligned}\mathbf{W}_b = {} & 2 \times \mathbf{e}^{[1]}(h, t_b)\Big(\mathbf{V}_{b_1}^{[1]} \circ \mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{[1]} \circ \mathbf{S}_{b_1}\Big)\mathbb{1}_{\{b\in\Omega\}} \\ & + \mathbf{P}(t_b)\Big(2 \times \mathbf{V}_{b_1}^{[1]} \circ \mathbf{V}_{b_2}^{[1]} + \mathbf{W}_{b_1} \circ \mathbf{S}_{b_2} + \mathbf{W}_{b_2} \circ \mathbf{S}_{b_1}\Big),\end{aligned} \tag{3.22}$$

respectively. From Figure 3.2, we know that elements of $\mathbf{V}_b^{[1]}$ can be interpreted as the sum over all "single-colored" phylogenies for the subtree defined by Θ_b and for the predefined set of "colored" branches $\Omega_b = \Omega \cap \Theta_b$, conditional on the state of parent node $p(b)$. Equation (3.20) partitions this sum into three distinct pieces: 1) $\mathbf{e}^{[1]}(h, t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega\}}$, 2) $\mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{[1]} \circ \mathbf{S}_{b_2}\Big)$, and 3) $\mathbf{P}(t_b)\Big(\mathbf{V}_{b_2}^{[1]} \circ \mathbf{S}_{b_1}\Big)$. The first piece represents the "single-colored" tree with "colored" branch b (only if $b \in \Omega$); the second piece represents a sum over "single-colored" phylogenies, where the "colored" branch is permuted across all branches in Ω_{b_1}; and the third piece represents another sum over "single-colored" phylogenies, where the "colored" branch is permuted across all branches in Ω_{b_2}. This partitioning allows us to compute $\mathbf{V}_b^{[1]}$ as a function of previously cached vectors $\mathbf{F}_{c(b)}$, $\mathbf{S}_{b_1}$, $\mathbf{S}_{b_2}$, $\mathbf{V}_{b_1}^{[1]}$, and $\mathbf{V}_{b_2}^{[1]}$. The recursions for $\mathbf{V}_b^{[2]}$ and $\mathbf{W}_b$ have analogous interpretations. Note that our algorithm requires $O(n)$ storage because we cache a constant (with respect to n) number of vectors at all nodes $u \in \{1, ..., n-1, n, ..., 2n-1\}$ and branches $b \in \Theta$. In addition, our procedure utilizes $O(n)$ computations because there are $O(B_n) = O(n)$ iterations in the algorithm and each iteration involves a constant (with respect to n) number of operations.

3.3.3 Posterior Mapping Variance Computation

Our tips-to-root tree traversal procedure terminates after computing the vectors $\mathbf{F}_{root}$, $\mathbf{S}_{root_1}$, $\mathbf{S}_{root_2}$, $\mathbf{V}_{root_1}^{[1]}$, $\mathbf{V}_{root_2}^{[1]}$, $\mathbf{V}_{root_1}^{[2]}$, $\mathbf{V}_{root_2}^{[2]}$, $\mathbf{W}_{root_1}$, and $\mathbf{W}_{root_2}$, where $root$ denotes the root node label and $root_1$ and $root_2$ represent the two branches connecting the root node to its children. We use these cached vectors to efficiently calculate the posterior mapping variance $\mathrm{Var}(H_\Omega|\mathbf{D})$.

We first describe how to compute the restricted mapping second moment $\mathrm{E}(H_\Omega^2\mathbb{1}_\mathbf{D})$. We

remind our readers that the double sum over $b \neq b'$ $(b, b' \in \Omega)$ in (3.9) can be visualized as a sum over all "double-colored" phylogenies, where Ω denotes the predefined set of "colored" branches. There are four distinct cases of $b \neq b'$ that we consider in (3.9): 1) $b \preceq root_1, b' \preceq root_1$; 2) $b \preceq root_1, b' \preceq root_2$; 3) $b \preceq root_2, b' \preceq root_1$; and 4) $b \preceq root_2, b' \preceq root_2$. Cases 1) and 4) represent the "double-colored" phylogenies that have both "colored" branches on the same side of the root, while Cases 2) and 3) denote the "double-colored" phylogenies that have one "colored" branch on each side of the root. This decomposition of the double sum in (3.9) was suggested by Kenney and Gu (2012) in the context of computing second derivatives of phylogenetic likelihood functions. The sums over all "double-colored" phylogenies in Cases 1), 2), 3), and 4) are mathematically represented as $\boldsymbol{\pi}^T\Big(\mathbf{W}_{root_1} \circ \mathbf{S}_{root_2}\Big)$, $\boldsymbol{\pi}^T\Big(\mathbf{V}^{[1]}_{root_1} \circ \mathbf{V}^{[1]}_{root_2}\Big)$, $\boldsymbol{\pi}^T\Big(\mathbf{V}^{[1]}_{root_1} \circ \mathbf{V}^{[1]}_{root_2}\Big)$, and $\boldsymbol{\pi}^T\Big(\mathbf{W}_{root_2} \circ \mathbf{S}_{root_1}\Big)$, respectively. Thus, the double sum in (3.9) can be efficiently computed as:

$$\boldsymbol{\pi}^T\Big(2 \times \mathbf{V}^{[1]}_{root_1} \circ \mathbf{V}^{[1]}_{root_2} + \mathbf{W}_{root_1} \circ \mathbf{S}_{root_2} + \mathbf{W}_{root_2} \circ \mathbf{S}_{root_1}\Big). \tag{3.23}$$

The sum over $b \in \Omega$ in (3.8) can also be calculated using the cached vectors mentioned above. We can visualize the sum in (3.8) as a sum over all "single-colored" phylogenies, where Ω denotes the predefined set of "colored" branches. There are two cases of $b \in \Omega$ that we consider in (3.8): 1) $b \preceq root_1$ and 2) $b \preceq root_2$. Using logic similar to that described above, we can calculate the sum in (3.8) as:

$$\boldsymbol{\pi}^T\Big[\big(\mathbf{V}^{[1]}_{root_1} + \mathbf{V}^{[2]}_{root_1}\big) \circ \mathbf{S}_{root_2} + \big(\mathbf{V}^{[1]}_{root_2} + \mathbf{V}^{[2]}_{root_2}\big) \circ \mathbf{S}_{root_1}\Big]. \tag{3.24}$$

We obtain a simple formula for computing the restricted mapping second moment $\mathrm{E}(H^2_\Omega \mathbb{1}_{\mathbf{D}})$ by adding together the expressions in (3.23) and (3.24):

$$\begin{aligned}\mathrm{E}(H^2_\Omega \mathbb{1}_{\mathbf{D}}) = \boldsymbol{\pi}^T\Big[&2 \times \mathbf{V}^{[1]}_{root_1} \circ \mathbf{V}^{[1]}_{root_2} + \mathbf{W}_{root_1} \circ \mathbf{S}_{root_2} + \mathbf{W}_{root_2} \circ \mathbf{S}_{root_1} \\ &+ \big(\mathbf{V}^{[1]}_{root_1} + \mathbf{V}^{[2]}_{root_1}\big) \circ \mathbf{S}_{root_2} + \big(\mathbf{V}^{[1]}_{root_2} + \mathbf{V}^{[2]}_{root_2}\big) \circ \mathbf{S}_{root_1}\Big].\end{aligned} \tag{3.25}$$

Other quantities involved in the calculation of $\mathrm{Var}(H_\Omega|\mathbf{D})$ include the phylogenetic likelihood $\mathrm{P}(\mathbf{D})$ and the restricted mapping first moment $\mathrm{E}(H_\Omega \mathbb{1}_{\mathbf{D}})$. From (Felsenstein, 1981), we know that $\mathrm{P}(\mathbf{D}) = \boldsymbol{\pi}^T \mathbf{F}_{root}$. Minin and Suchard (2008b) express $\mathrm{E}(H_\Omega \mathbb{1}_{\mathbf{D}})$ in the following manner:

$$\mathrm{E}(H_\Omega \mathbb{1}_{\mathbf{D}}) = \sum_b \sum_{\mathbf{i}} e^{[1]}_{i^*_{p(b)} i^*_{c(b)}}(h, t_b) \pi_{i^*_1} \prod_{b^* \in \Theta \setminus \{b\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}), \tag{3.26}$$

where $b \in \Omega$. Notice that the right-hand side of equation (3.26) is virtually identical to the sum over $b \in \Omega$ in (3.8). Given our interpretations of the sums in (3.8) and (3.9), it is easy to see that:

$$\mathrm{E}(H_\Omega \mathbb{1}_{\mathbf{D}}) = \boldsymbol{\pi}^T \Big(\mathbf{V}^{[1]}_{root_1} \circ \mathbf{S}_{root_2} + \mathbf{V}^{[1]}_{root_2} \circ \mathbf{S}_{root_1} \Big). \tag{3.27}$$

Finally, the posterior mapping variance $\mathrm{Var}(H_\Omega|\mathbf{D})$ can be expressed as follows:

$$\mathrm{Var}(H_\Omega|\mathbf{D}) = \mathrm{E}(H^2_\Omega|\mathbf{D}) - \mathrm{E}(H_\Omega|\mathbf{D})^2 \tag{3.28}$$

$$= \frac{\mathrm{E}(H^2_\Omega \mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})} - \left[\frac{\mathrm{E}(H_\Omega \mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})} \right]^2. \tag{3.29}$$

We can compute $\mathrm{Var}(H_\Omega|\mathbf{D})$ by first calculating $\mathrm{P}(\mathbf{D})$, $\mathrm{E}(H_\Omega \mathbb{1}_{\mathbf{D}})$, and $\mathrm{E}(H^2_\Omega \mathbb{1}_{\mathbf{D}})$ using our post-order tree traversal algorithm and then substituting these quantities into equation (3.29).

3.3.4 Prior Mapping Variance Computation

Given our efficient calculation of the posterior mapping variance $\mathrm{Var}(H_\Omega|\mathbf{D})$, it is natural to ask whether it is possible to extend the tree traversal algorithm described above to the computation of the prior mapping variance $\mathrm{Var}(H_\Omega)$. There is a plethora of literature in evolutionary biology that discusses how to calculate the prior mapping variance for a single tree branch (Zheng, 2001; Bloom et al., 2007; Minin and Suchard, 2008a,b). Siepel et al. (2006) describe a dynamic programming procedure that approximates the probability distribution of H_Θ and use it to estimate $\mathrm{Var}(H_\Theta)$.

It turns out that we can exactly compute prior variances using a modified version of our tree traversal algorithm. The only changes that need to be made are to the initializations of $\mathbf{F}_u$ at all terminal nodes u. If we set $F_{ui} = 1$ for all terminal nodes u and $i = 1, ..., m$, then our recursive procedure will be able to compute the prior variance $\text{Var}(H_\Omega)$. We also note that all elements of the vectors $\mathbf{F}_u$ and $\mathbf{S}_b$ for $u \in \{1, ..., n-1, n, ..., 2n-1\}$ and $b \in \Theta$ are equal to 1 as a result of these modified initializations. Remember that the initializations of $\mathbf{F}_u$ can account for partially observed and/or missing data at the tips of τ. These modified $\mathbf{F}_u$ initializations are intuitive because prior mapping moments average over all possible observed trait values at the tips of the phylogeny and thus every combination of observed tip states needs to be accounted for in our $\mathbf{F}_u$ initializations. As we will see, the prior mapping variance $\text{Var}(H_\Omega)$ is essential to one of the applications we present in this chapter.

3.3.5 Prior and Posterior Mapping Covariances

Two other quantities of interest associated with stochastic mapping summaries are the prior mapping covariance $\text{Cov}(H_{\Omega_1}, H_{\Omega_2})$ and the posterior mapping covariance $\text{Cov}(H_{\Omega_1}, H_{\Omega_2}|\mathbf{D})$, where $\Omega_1, \Omega_2 \subseteq \Theta$ are predefined sets of branch indices. Efficient computations of the prior and posterior mapping covariances are interesting in their own right and utilized in one of the applications we present in this chapter. In the Appendix, we describe another tree traversal algorithm for computing these covariances. This new algorithm is a generalization of the recursive procedure described above and much of the intuition provided for our original procedure carries over to this new algorithm. Furthermore, this new procedure runs in $O(n)$ time and with $O(n)$ storage.

3.3.6 Implementation

The efficient calculations of the prior and posterior mapping moments discussed above are implemented in the R package `phylomoments`, which is available at `https://github.com/dunleavy005/phylomoments`. This package also contains our implementation of the stochastic mapping simulation technique put forth by Nielsen (2002) along with other assorted func-

tions. The computationally intensive parts of our methods are written in C++ and ported to R using the R packages `Rcpp` and `RcppArmadillo` (Eddelbuettel and François, 2011; Eddelbuettel and Sanderson, 2014). In the next section, we present two scientific applications that employ stochastic mapping variances.

3.4 Applications

3.4.1 Testing for Rate Variation Across Sites

Our first application is centered around an across-site rate variation test proposed by Nielsen (2002). Nielsen (2002) uses simulated posterior mapping variances within a posterior predictive approach to model diagnostics. In this subsection, we describe a posterior predictive testing framework that adheres to the principles outlined by Gelman et al. (1996) and test for across-site rate variation in two real datasets using exactly computed posterior mapping variances.

Overview of Posterior Predictive Tests

Conceptually, posterior predictive assessments can be seen as Bayesian analogues of classical frequentist model diagnostics and hypothesis tests. Unlike classical testing procedures, posterior predictive tests permit the use of "test statistics" that depend on both data and unknown parameters. These "test statistics" are otherwise known as discrepancy measures (Meng, 1994; Gelman et al., 1996). In this subsection, we denote the discrepancy measure as $T \equiv T(\mathbf{D}_{1:L}, \boldsymbol{\theta})$. Posterior predictive model testing is based on the following principle: if the assumed model adequately fits the observed data $\mathbf{D}_{1:L}^{obs}$, then simulated datasets $\mathbf{D}_{1:L}^{rep}$ from the assumed model should look like $\mathbf{D}_{1:L}^{obs}$. Similarity between $\mathbf{D}_{1:L}^{obs}$ and $\mathbf{D}_{1:L}^{rep}$ is measured through the discrepancy T. We would like to compare the observed discrepancy $T(\mathbf{D}_{1:L}^{obs}, \boldsymbol{\theta})$ to a reference distribution induced by the hypothesized model. The reference distribution used in posterior predictive diagnostics is derived from the joint posterior distribution of

$\mathbf{D}_{1:L}^{rep}$ and $\boldsymbol{\theta}$:

$$\mathrm{P}(\mathbf{D}_{1:L}^{rep}, \boldsymbol{\theta} | \mathbf{D}_{1:L}^{obs}) = \mathrm{P}(\mathbf{D}_{1:L}^{rep} | \boldsymbol{\theta}) \mathrm{P}(\boldsymbol{\theta} | \mathbf{D}_{1:L}^{obs}). \tag{3.30}$$

Intuitively, this distribution indicates which datasets and parameter values are most plausible if the assumed model holds true. Posterior predictive assessments are usually conducted using simulations as the joint distribution in (3.30) is often analytically intractable. We summarize posterior predictive simulations in the following three steps:

1. Sample $\boldsymbol{\theta}^* \sim \mathrm{P}(\boldsymbol{\theta} | \mathbf{D}_{1:L}^{obs})$;

2. Simulate $\mathbf{D}_{1:L}^{rep,*} \sim \mathrm{P}(\mathbf{D}_{1:L}^{rep} | \boldsymbol{\theta}^*)$;

3. Calculate $T(\mathbf{D}_{1:L}^{obs}, \boldsymbol{\theta}^*)$ and $T(\mathbf{D}_{1:L}^{rep,*}, \boldsymbol{\theta}^*)$.

We repeat these steps N times, where N is a large number, and then compare the N samples of $T(\mathbf{D}_{1:L}^{obs}, \boldsymbol{\theta}^*)$ and $T(\mathbf{D}_{1:L}^{rep,*}, \boldsymbol{\theta}^*)$ by constructing two empirical distributions; a small overlap between these distributions, which could be visualized with histograms, suggests a poor model fit. We can quantify the disagreement between the observed and predicted discrepancies by calculating the posterior predictive p-value (Meng, 1994; Gelman et al., 1996):

$$ppp = \mathrm{P}\big[T(\mathbf{D}_{1:L}^{rep}, \boldsymbol{\theta}) > T(\mathbf{D}_{1:L}^{obs}, \boldsymbol{\theta}) \big| \mathbf{D}_{1:L}^{obs}\big], \tag{3.31}$$

where the probability is computed with respect to the joint posterior distribution $\mathrm{P}(\mathbf{D}_{1:L}^{rep}, \boldsymbol{\theta} | \mathbf{D}_{1:L}^{obs})$. Given N posterior samples of $\mathbf{D}_{1:L}^{rep}$ and $\boldsymbol{\theta}$, we can estimate the posterior predictive p-value by computing:

$$ppp \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\left\{T(\mathbf{D}_{1:L}^{rep,(i)}, \boldsymbol{\theta}^{(i)}) > T(\mathbf{D}_{1:L}^{obs}, \boldsymbol{\theta}^{(i)})\right\}}, \tag{3.32}$$

where $\mathbf{D}_{1:L}^{rep,(i)}$ and $\boldsymbol{\theta}^{(i)}$ represent the ith posterior samples of $\mathbf{D}_{1:L}^{rep}$ and $\boldsymbol{\theta}$, respectively, for $i = 1, ..., N$.

Posterior Predictive Rate Variation Tests

Now, we use the posterior predictive testing framework described above to formulate a test for rate variation across sites in an alignment. We assume that alignment sites evolve independently according to the same distribution provided by the continuous-time Markov process $\psi_{\boldsymbol{\theta}}$. We focus on selecting discrepancy measures that can gauge the variability in substitution rates across sites. One possible discrepancy measure is the variance of the total number of substitutions in the alignment (Nielsen, 2002):

$$T_{var} \equiv T_{var}(\mathbf{D}_{1:L}, \boldsymbol{\theta}) = \mathrm{Var}\left(\sum_{i=1}^{L} H_{\Theta}^{(i)} \middle| \mathbf{D}_{1:L} \right) = \sum_{i=1}^{L} \mathrm{Var}(H_{\Theta}|\mathbf{D}_i), \tag{3.33}$$

where $H_{\Theta}^{(i)}$ represents the number of substitutions at site i. Note that the second equality in (3.33) follows from conditional independence assumptions. Another discrepancy we consider in our analyses is the posterior dispersion index (i.e. posterior variance-to-mean ratio) for substitution counts:

$$\begin{aligned} T_{disp} \equiv T_{disp}(\mathbf{D}_{1:L}, \boldsymbol{\theta}) &= \mathrm{Var}\left(\sum_{i=1}^{L} H_{\Theta}^{(i)} \middle| \mathbf{D}_{1:L} \right) \Bigg/ \mathrm{E}\left(\sum_{i=1}^{L} H_{\Theta}^{(i)} \middle| \mathbf{D}_{1:L} \right) \\ &= \left[\sum_{i=1}^{L} \mathrm{Var}(H_{\Theta}|\mathbf{D}_i) \right] \Bigg/ \left[\sum_{i=1}^{L} \mathrm{E}(H_{\Theta}|\mathbf{D}_i) \right], \end{aligned} \tag{3.34}$$

where $H_{\Theta}^{(i)}$ is specified as above. In the presence of rate variation across sites, the prior dispersion index for substitution counts is often greater than 1 (Yang, 1996). One way to see this is by assuming that the number of substitutions occurring on a particular tree branch follows a Poisson distribution, where the Poisson rate parameter varies across sites according to a gamma distribution. A simple calculation shows that, marginally, the number of substitutions occurring on this branch follows a negative binomial distribution, which has a variance-to-mean ratio that is greater than 1. We can make a similar argument about the number of substitutions across an entire phylogeny. Even though the above explanation

applies only to the prior dispersion index, it does motivate our use of the posterior dispersion index as a discrepancy measure for detecting across-site rate variation in an alignment.

The posterior predictive simulations for our rate variation test can be summarized using the same three steps described above. We can sample $\boldsymbol{\theta}^* \sim \mathrm{P}(\boldsymbol{\theta}|\mathbf{D}_{1:L}^{obs})$ using a Bayesian phylogenetic inference software package; in our examples, we use the computer program MrBayes (Huelsenbeck and Ronquist, 2001) to perform the posterior sampling of $\boldsymbol{\theta}$. For every posterior sample of $\boldsymbol{\theta}$, we can simulate a replicate alignment $\mathbf{D}_{1:L}^{rep,*} \sim \mathrm{P}(\mathbf{D}_{1:L}^{rep}|\boldsymbol{\theta}^*)$ by independently generating $\mathbf{D}_1^{rep,*}, ..., \mathbf{D}_L^{rep,*}$ according to $\psi_{\boldsymbol{\theta}^*}$; we simulate tip data from $\psi_{\boldsymbol{\theta}^*}$ using the standard discretized CTMC approach (Yang, 2006, Chapter 9). We can then analytically calculate the observed and predicted discrepancies (i.e. $T(\mathbf{D}_{1:L}^{obs}, \boldsymbol{\theta}^*)$ and $T(\mathbf{D}_{1:L}^{rep,*}, \boldsymbol{\theta}^*)$) for $T = T_{var}, T_{disp}$ using the algorithm presented in the previous section. Nielsen (2002) also used the discrepancy T_{var} in posterior predictive rate variation tests but could only obtain Monte Carlo estimates of T_{var}. Monte Carlo estimation of T_{var} is computationally intensive because it uses simulations to estimate $\mathrm{Var}(H_\Theta|\mathbf{D}_i)$ for $i = 1, ..., L$. The tree traversal algorithm proposed in this chapter not only eliminates the Monte Carlo error associated with estimating the variance T_{var} but also speeds up the computation of this quantity.

Testing for Rate Variation in Two Sequence Alignments

We analyze the two sequence alignments used by Nielsen (2002) to demonstrate the effectiveness of our posterior predictive testing scheme. Our first dataset contains β-globin sequences for 17 vertebrate species, where each sequence is 432 base pairs long. Our second dataset comprises 28 sequences of the hemagglutinin (HA) gene of human influenza virus A; each sequence has 987 base pairs. In all our analyses, we use a general time-reversible (GTR) substitution model (Tavaré, 1986) with a Dirichlet(1,1,1,1,1,1) prior for the GTR exchangeability rates and a Dirichlet(1,1,1,1) prior for the base frequencies $\boldsymbol{\pi}$. Furthermore, we assume a uniform prior on all possible tree topologies τ and let all branch lengths in $\mathbf{t}$ be a priori uniformly distributed on the interval $[0, 100]$. For each dataset, we generate $N = 1000$ posterior samples of $\mathbf{D}_{1:L}^{rep}$ and $\boldsymbol{\theta}$ using the simulation procedure described previously and

calculate the observed and predicted values of T_{var} and T_{disp}. The 1000 posterior samples of $\boldsymbol{\theta}$ are obtained by running the Markov chain Monte Carlo (MCMC) procedure in MrBayes for 1,100,000 iterations and storing values of $\boldsymbol{\theta}$ every 1000 iterations from iteration 101,000 to iteration 1,100,000. We use trace plots (not shown in this chapter) to assess convergence of the MCMC samplers and find that using 1,100,000 MCMC iterations is sufficient for our purposes.

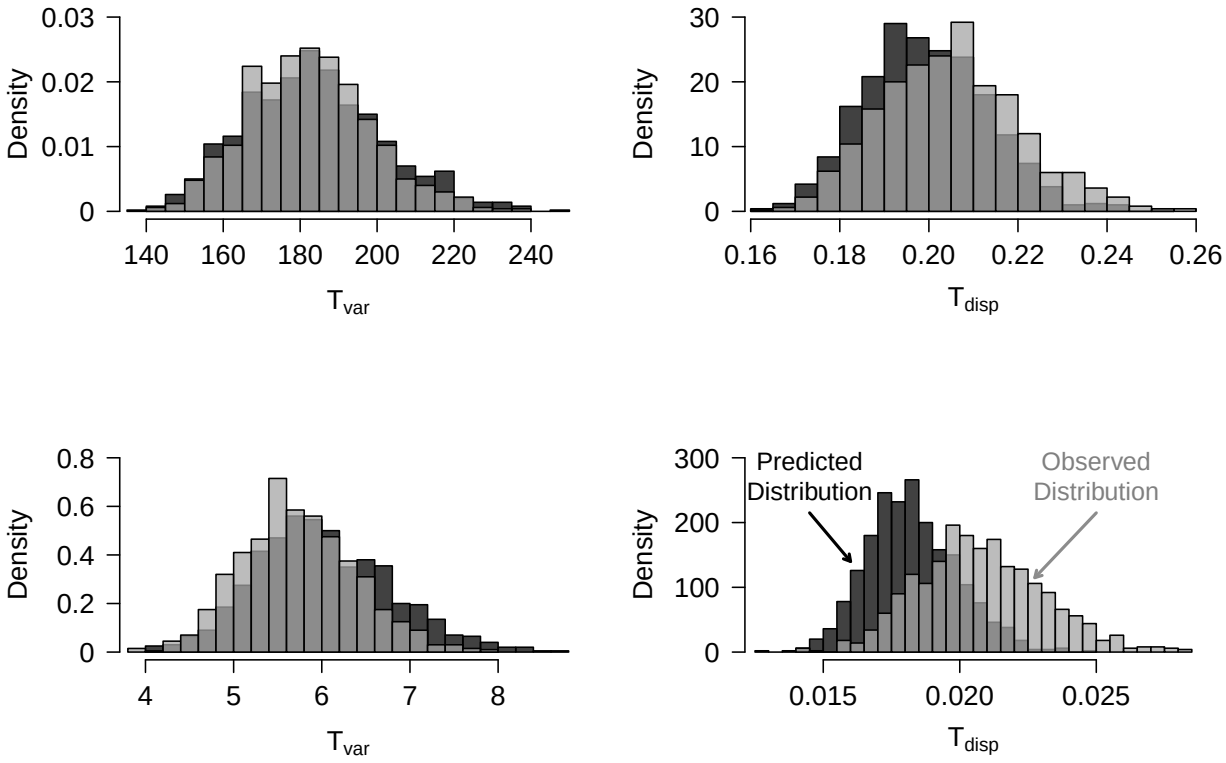


Figure 3.3: Observed and predicted distributions of the discrepancies T_{var} and T_{disp}. In each plot, we superimpose the observed distribution (grey) on top of the predicted distribution (black). The top two plots display the observed and predicted distributions for the β-globin dataset, while the bottom two plots show the observed and predicted distributions for the influenza dataset. These distributions were constructed using $N = 1000$ posterior samples of $\mathbf{D}^{rep}_{1:L}$ and $\boldsymbol{\theta}$.

We summarize the observed and predicted discrepancies in Figure 3.3, which demonstrates that the observed distributions of T_{var} do not deviate much from the corresponding

predicted distributions of T_{var}. The posterior predictive p-values that were computed using the discrepancy T_{var} are approximately 0.60 and 0.80 for the β-globin and influenza datasets, respectively. Thus, it seems that our posterior predictive test based on T_{var} fails to detect across-site rate variation in the two datasets. We note that our posterior predictive test results based on T_{var} do not corroborate the findings by Nielsen (2002), who concluded that there is rate variation across sites in the β-globin and influenza datasets. However, Nielsen (2002) used the posterior variance of substitution counts in a slightly different way without clearly specifying all predictive distributions. We believe our analyses are better aligned with the posterior predictive principles outlined by Gelman et al. (1996).

In Figure 3.3, the observed and predicted distributions of T_{disp} do not completely overlap and appear more separated than the observed and predicted distributions of T_{var}. For both datasets, the observed values of T_{disp} are, on average, greater than the predicted values of T_{disp}. This suggests that the discrepancy T_{disp} is able to detect observed rate variation that is not accounted for by our hypothesized model. The posterior predictive p-values that were computed using T_{disp} are approximately 0.17 and 0.031 for the β-globin and influenza datasets, respectively; note that these p-values are smaller than the corresponding p-values that were computed using T_{var} and as a result provide stronger evidence in support of the rate variation hypothesis.

A popular frequentist approach to modeling rate variation among sites employs a discrete gamma distribution with a fixed number of rate classes (Yang, 1994, 1996). We check our posterior predictive test results by performing likelihood ratio tests that compare discrete gamma models with one rate category (H_0) and four rate categories (H_a). We perform the likelihood ratio tests using the PhyML package (Guindon et al., 2010) and obtain p-values close to 0; these tests also suggest the presence of across-site rate variation in the β-globin and influenza datasets. The posterior predictive p-values that were computed using T_{disp} are not as small as the likelihood ratio p-values, but this should not be surprising because posterior predictive p-values tend to be more conservative than classical frequentist p-values (Meng, 1994). Our posterior predictive analyses suggest that the posterior dispersion index

T_{disp} is better than the posterior variance T_{var} at detecting observed rate variation among sites.

Monte Carlo Error and Timing Analyses

To assess the efficiency gains from using our tree traversal algorithm in this setting, we approximate Monte Carlo standard errors and running times associated with simulation-based T_{var} estimates. For each dataset, we compute these standard errors and running times on randomly subsampled alignments of length L using m Monte Carlo replicates per site, where $L = 50, 100, 200, 400$ and $m = 100, 500, 1000, 10000$; the Monte Carlo standard errors are approximated using well-known formulas for the moments of the sample variance (Mood et al., 1974, Chapter VI). We account for the posterior uncertainty in $\boldsymbol{\theta}$ by first calculating Monte Carlo errors and running times for 200 randomly subsampled posterior $\boldsymbol{\theta}$'s and then averaging these metrics across the $\boldsymbol{\theta}$ samples; the same 200 samples of $\boldsymbol{\theta}$ are used in all our simulations. For each setting of L, we also compute the exact values of T_{var} and track the corresponding running times for the 200 posterior samples of $\boldsymbol{\theta}$ and then average these results over the subsampled $\boldsymbol{\theta}$'s.

Tables 3.1 and 3.2 present the running time comparisons and Monte Carlo error approximations, respectively, for the β-globin and influenza datasets; Table 3.2 can be found in the Appendix. Based on the results shown in Table 3.2, it seems that the Monte Carlo error has a convergence rate of $O\left(\sqrt{L/m}\right)$. This can be justified using a Central Limit Theorem argument if L and m are large; remember that the Monte Carlo estimator of T_{var} is a sum of L independent, non-identically distributed sample variances, where each sample variance is calculated using m independent Monte Carlo replicates. Furthermore, the Monte Carlo error percentages range from 0.15% to 28% across the two datasets. Table 3.1 suggests that the Monte Carlo running time increases linearly in L and m as we might expect. We can also see that our exact computations of T_{var} are at least an order of magnitude faster than the Monte Carlo estimates of T_{var}. Thus, it is clear that our tree traversal algorithm improves the computational efficiency of posterior predictive rate variation tests that utilize Monte

Carlo discrepancy estimates.

	Running times (β-globin)				Running times (influenza)			
	$L = 50$	$L = 100$	$L = 200$	$L = 400$	$L = 50$	$L = 100$	$L = 200$	$L = 400$
$m = 100$	0.70	1.4	3.1	6.1	1.0	2.0	4.1	8.0
$m = 500$	3.3	6.5	15	28	4.7	9.3	19	37
$m = 1000$	7.2	13	28	57	9.0	18	37	74
$m = 10000$	72	140	290	570	93	180	370	740
Exact	0.0040	0.0059	0.0097	0.017	0.0061	0.0091	0.015	0.026

Table 3.1: Monte Carlo running times associated with simulation-based T_{var} estimates for the β-globin and influenza datasets. We compute these running times on randomly subsampled alignments of length L using m Monte Carlo replicates per site. Each table entry, excluding the entries on the bottom row, represents an averaged Monte Carlo running time (in seconds), where the averaging is done over 200 randomly subsampled posterior $\boldsymbol{\theta}$'s. Each table entry on the bottom row denotes an average over running times (in seconds) associated with exact calculations of T_{var}, where the averaging is done over the same 200 posterior samples of $\boldsymbol{\theta}$ mentioned previously. All table entries are rounded to two significant digits.

3.4.2 Detecting Evolutionarily Conserved Regions in Genomic Alignments

Our second application focuses on the detection of conserved elements in genomic alignments; in this setting, conservation refers to evolution that is slower than expected. Detection of conserved DNA sites is of prime interest in comparative genomics because most of the conserved elements in genome-wide sequence alignments are believed to be caused by negative selection and to have evolutionarily important biological functions (Siepel et al., 2005). Computational methods for detecting conserved genomic regions are essential because they are used to flag candidate functional elements, which can then be further examined experimentally (Siepel et al., 2006).

In this subsection, we analyze two statistical tests of conservation presented by Siepel et al. (2006). One test is used to detect conservation across all lineages in a phylogeny, while the other is used to identify lineage-specific conservation; both tests are referred to as SPH conservation tests and implemented in the computer program phyloP (Pollard et al., 2010). Our exact calculations of prior and posterior mapping moments can be used to make the

SPH conservation tests more powerful. We present some modifications to these conservation tests and demonstrate the efficacy of our proposed changes via simulations.

Modifying the SPH Conservation Tests

Let $\psi_{\boldsymbol{\theta}_{\text{neut}}}$ denote the baseline neutral evolutionary model that is assumed to be given; neutral models are commonly estimated using fourfold degenerate sites extracted from genome-wide sequence alignments of interest (Pollard et al., 2010). As in (Pollard et al., 2010), we define $\psi(\rho, \lambda; \Theta_b)$ to be a scaled evolutionary model that is identical to $\psi_{\boldsymbol{\theta}_{\text{neut}}}$ except that it has all its branch lengths scaled by the factor $\rho \in [0, 1]$ and the branch lengths in the subtree defined by Θ_b additionally scaled by the factor $\lambda \in [0, 1]$. For convenience, we let $\psi(\rho) \equiv \psi(\rho, \lambda = 1; \Theta_b)$ for all $b \in \Theta$ and $\rho \in [0, 1]$. Throughout this subsection, we assume that $\mathbf{D}_1, ..., \mathbf{D}_L$ are independent and identically distributed according to $\psi(\cdot)$. The two tests of conservation described in (Siepel et al., 2006) reduce to tests of the models that constrain parameters ρ and λ to particular values.

The SPH "all-branch" conservation test examines conservation across all branches of the phylogeny. Specifically, it tests the null hypothesis $H_0 : \rho = 1$ against the alternative hypothesis $H_a : \rho < 1$ for the evolutionary model $\psi(\rho)$. Siepel et al. (2006) use the following test statistic to distinguish between the two hypotheses:

$$\mathcal{T}_{all} \equiv \mathcal{T}_{all}(\mathbf{D}_{1:L}) = \mathrm{E}\left(\sum_{i=1}^{L} H_{\Theta}^{(i)} \middle| \mathbf{D}_{1:L} \right) = \sum_{i=1}^{L} \mathrm{E}(H_{\Theta} | \mathbf{D}_i), \tag{3.35}$$

where $H_{\Theta}^{(i)}$ represents the number of substitutions at site i. The test statistic in (3.35) serves as a proxy for the "observed" number of substitutions in the genomic alignment $\mathbf{D}_{1:L}$. The prior distribution of $\sum_{i=1}^{L} H_{\Theta}^{(i)}$ is taken to be the null distribution, and the p-value of this test is obtained by first comparing the observed value of $\mathcal{T}_{all}$ to this prior distribution and then computing the corresponding left-tail probability. It turns out that the p-values obtained from this testing procedure are not uniformly distributed and tend to be conservative under

the null hypothesis (Siepel et al., 2006).

To understand why this occurs, we must examine the null distribution chosen for this test. Notice first that $\mathcal{T}_{all}$ is a sum of independent and identically distributed random variables with mean $\mathrm{E}[\mathrm{E}(H_\Theta|\mathbf{D})] = \mathrm{E}(H_\Theta)$ and variance $\mathrm{Var}[\mathrm{E}(H_\Theta|\mathbf{D})] = \mathrm{Var}(H_\Theta) - \mathrm{E}[\mathrm{Var}(H_\Theta|\mathbf{D})]$. For L large, we can invoke the Central Limit Theorem and approximate the sampling distribution of $\mathcal{T}_{all}$ with a normal distribution having mean $L \times \mathrm{E}(H_\Theta)$ and variance $L \times \mathrm{Var}(H_\Theta) - L \times \mathrm{E}[\mathrm{Var}(H_\Theta|\mathbf{D})]$. Using a similar asymptotic argument as above, we can also approximate the null distribution of $\sum_{i=1}^{L} H_\Theta^{(i)}$ — an unobserved quantity — with a normal distribution having mean $L \times \mathrm{E}(H_\Theta)$ and variance $L \times \mathrm{Var}(H_\Theta)$. Siepel et al. (2006) use the exact version of this distribution, which results in the conservative nature of their all-branch test, because the correct null distribution of $\mathcal{T}_{all}$, at least asymptotically, has a variance that is smaller than the one assumed by the authors. For a given significance level, an overdispersed null distribution causes the critical value for rejecting H_0 to be more extreme than it would be for a proper null distribution. This is of practical importance because more extreme critical values make it harder to correctly flag conserved genomic elements.

The SPH all-branch conservation test can be corrected and made more powerful by using the correct asymptotic distribution of $\mathcal{T}_{all}$ as the null distribution. The dynamic programming algorithm discussed in this chapter can be used to calculate the mean and variance of this asymptotic distribution. The quantities $\mathrm{E}(H_\Theta)$ and $\mathrm{Var}(H_\Theta)$ are easily computed using our results for prior mapping moments. Given our efficient computation of the posterior mapping variance $\mathrm{Var}(H_\Theta|\mathbf{D})$, we estimate $\mathrm{E}[\mathrm{Var}(H_\Theta|\mathbf{D})]$ using Monte Carlo simulation of sequence data $\mathbf{D}$. Even though we need Monte Carlo simulations to calculate the asymptotic variance of $\mathcal{T}_{all}$, our approach is still more efficient than directly estimating $\mathrm{Var}[\mathrm{E}(H_\Theta|\mathbf{D})]$ via Monte Carlo sampling.

Siepel et al. (2006) also describe two testing procedures that are used to detect lineage-specific conservation. Both procedures analyze conservation at the subtree level and are referred to as SPH "subtree" tests. Formally, these two approaches test the null hypothesis $H_0 : \lambda = 1, \rho \in [0, 1]$ against the alternative hypothesis $H_a : \lambda < 1, \rho \in [0, 1]$ for the

evolutionary model $\psi(\rho, \lambda; \Theta_b)$. The following test statistic is used in both procedures:

$$\mathcal{T}_{sub} \equiv \mathcal{T}_{sub}(\mathbf{D}_{1:L}) = \mathrm{E}\left(\sum_{i=1}^{L} H_{\Theta_b}^{(i)} \middle| \mathbf{D}_{1:L} \right) = \sum_{i=1}^{L} \mathrm{E}(H_{\Theta_b} | \mathbf{D}_i), \tag{3.36}$$

where $H_{\Theta_b}^{(i)}$ denotes the number of substitutions in the subtree defined by Θ_b at site i. Note that the statistic in (3.36) is the subtree equivalent of that shown in (3.35). The SPH "marginal" subtree test compares the observed value of $\mathcal{T}_{sub}$ to the marginal distribution of $\sum_{i=1}^{L} H_{\Theta_b}^{(i)}$, while the SPH "conditional" subtree test compares the observed value of $\mathcal{T}_{sub}$ to the conditional distribution of $\sum_{i=1}^{L} H_{\Theta_b}^{(i)}$ given $\sum_{i=1}^{L} H_{\Theta}^{(i)}$ is equal to the observed value of $\mathcal{T}_{all}$. Conceptually, the marginal subtree test examines whether the number of substitutions in the subtree is less than would be expected under the null model, whereas the conditional subtree test analyzes whether the number of substitutions in the subtree is surprising given the total number of substitutions in the tree. Even though these subtree tests are intuitively appealing, they suffer from the same problems discussed previously for the all-branch test (Siepel et al., 2006).

The SPH marginal subtree test can be corrected and made more powerful by using the correct asymptotic distribution of $\mathcal{T}_{sub}$ as the null distribution; this asymptotic distribution is obtained using reasoning similar to that used for the asymptotic distribution of $\mathcal{T}_{all}$. We propose our own conditional subtree test based on the following test statistic:

$$\begin{aligned} \mathcal{T}_{ratio} \equiv \mathcal{T}_{ratio}(\mathbf{D}_{1:L}) = \frac{\mathcal{T}_{sub}(\mathbf{D}_{1:L})}{\mathcal{T}_{all}(\mathbf{D}_{1:L})} &= \mathrm{E}\left(\sum_{i=1}^{L} H_{\Theta_b}^{(i)} \middle| \mathbf{D}_{1:L} \right) \Bigg/ \mathrm{E}\left(\sum_{i=1}^{L} H_{\Theta}^{(i)} \middle| \mathbf{D}_{1:L} \right) \\ &= \left[\sum_{i=1}^{L} \mathrm{E}(H_{\Theta_b} | \mathbf{D}_i) \right] \Bigg/ \left[\sum_{i=1}^{L} \mathrm{E}(H_{\Theta} | \mathbf{D}_i) \right], \end{aligned} \tag{3.37}$$

where $H_{\Theta_b}^{(i)}$ and $H_{\Theta}^{(i)}$ are defined as previously. This test statistic serves as a proxy for the "observed" proportion of substitutions in the subtree defined by Θ_b across the alignment $\mathbf{D}_{1:L}$. Using the Central Limit Theorem and the Delta Method, we approximate the sampling distribution of $\mathcal{T}_{ratio}$ with a normal distribution whose moments depend on

$\mathrm{E}(H_{\Theta_b})$, $\mathrm{E}(H_\Theta)$, $\mathrm{Var}[\mathrm{E}(H_{\Theta_b}|\mathbf{D})]$, $\mathrm{Var}[\mathrm{E}(H_\Theta|\mathbf{D})]$, and $\mathrm{Cov}[\mathrm{E}(H_{\Theta_b}|\mathbf{D}), \mathrm{E}(H_\Theta|\mathbf{D})]$; we omit the exact forms of the mean and variance of this asymptotic distribution for brevity. By the Laws of Total Variance and Covariance, we can express $\mathrm{Var}[\mathrm{E}(H_{\Theta_b}|\mathbf{D})]$, $\mathrm{Var}[\mathrm{E}(H_\Theta|\mathbf{D})]$, and $\mathrm{Cov}[\mathrm{E}(H_{\Theta_b}|\mathbf{D}), \mathrm{E}(H_\Theta|\mathbf{D})]$ as follows:

$$\mathrm{Var}[\mathrm{E}(H_{\Theta_b}|\mathbf{D})] = \mathrm{Var}(H_{\Theta_b}) - \mathrm{E}[\mathrm{Var}(H_{\Theta_b}|\mathbf{D})], \tag{3.38}$$

$$\mathrm{Var}[\mathrm{E}(H_\Theta|\mathbf{D})] = \mathrm{Var}(H_\Theta) - \mathrm{E}[\mathrm{Var}(H_\Theta|\mathbf{D})], \tag{3.39}$$

$$\mathrm{Cov}[\mathrm{E}(H_{\Theta_b}|\mathbf{D}), \mathrm{E}(H_\Theta|\mathbf{D})] = \mathrm{Cov}(H_{\Theta_b}, H_\Theta) - \mathrm{E}[\mathrm{Cov}(H_{\Theta_b}, H_\Theta|\mathbf{D})]. \tag{3.40}$$

The prior moments $\mathrm{E}(H_{\Theta_b})$, $\mathrm{E}(H_\Theta)$, $\mathrm{Var}(H_{\Theta_b})$, $\mathrm{Var}(H_\Theta)$, and $\mathrm{Cov}(H_{\Theta_b}, H_\Theta)$ are efficiently computed using the post-order tree traversal algorithm outlined in the Appendix. The quantities $\mathrm{E}[\mathrm{Var}(H_{\Theta_b}|\mathbf{D})]$, $\mathrm{E}[\mathrm{Var}(H_\Theta|\mathbf{D})]$, and $\mathrm{E}[\mathrm{Cov}(H_{\Theta_b}, H_\Theta|\mathbf{D})]$ are approximated using Monte Carlo replicates of $\mathbf{D}$ and our exact calculations of posterior mapping variances and covariances. For both of our lineage-specific conservation tests, we estimate the global scale parameter ρ by numerically maximizing the observed log-likelihood function. In the next subsubsection, we present simulation results that demonstrate the utility of our modified SPH conservation tests.

Simulation Experiments

We evaluate the power and false positive rates of the original and modified SPH conservation tests using simulated alignments. The neutral evolutionary model used by Pollard et al. (2010) is also employed in all our simulation experiments. This model was estimated using fourfold degenerate sites extracted from alignments of the 44 ENCODE regions (Birney et al., 2007) for 36 vertebrate species. For the all-branch and subtree tests, we simulate replicate alignments by independently generating L alignment columns according to $\psi(\rho)$ and $\psi(\rho, \lambda; \Theta_{primates})$, respectively, where *primates* denotes the branch above the primates subtree in the neutral phylogeny. We consider $L = 1, 2, 4, ..., 48, 50$ and $\rho = 0.1, 0.3, 0.5, 0.7, 0.9, 1$ in our all-branch simulations and $L = 1, 5, 10, ..., 45, 50$;

$\rho = 0.1, 0.25, 0.4, ..., 1$; and $\lambda = 0.1, 0.25, 0.4, ..., 1$ in our subtree simulations. We keep the alignment length L relatively small because the primary application of SPH tests is scanning whole genomes in search of short ultra-conserved genetic elements. For each simulation setting under H_a, we generate 1000 replicate datasets, compute the conservation p-values for each dataset using the original and modified SPH tests, and estimate the power by calculating the proportion of p-values less than the given significance level; we construct power curves by varying the significance threshold between 0 and 1. False positive rates are similarly estimated for each simulation setting under H_0. Note that these power curves should not be confused with receiver operating characteristic (ROC) curves; in our simulations, we plot the power (i.e. true positive rates) against the significance levels, whereas ROC curves plot the power against the false positive rates.

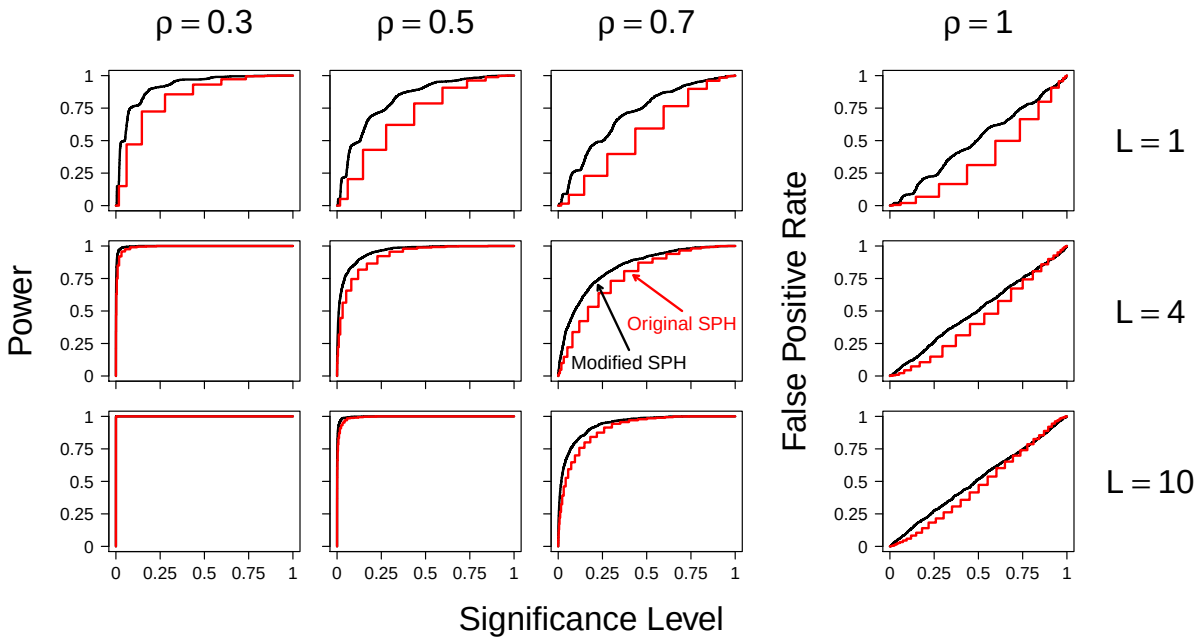
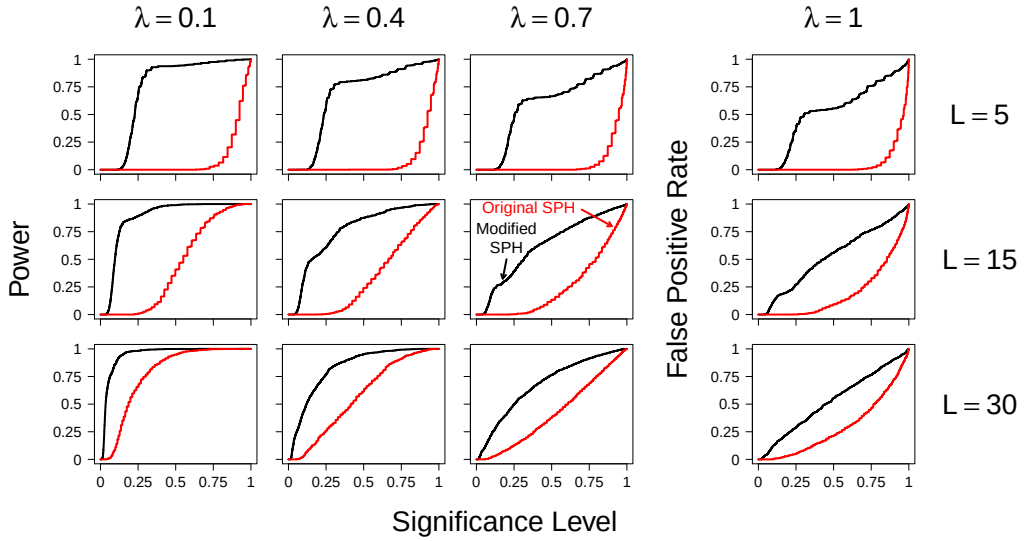


Figure 3.4: Power and false positive rate plots from our all-branch simulation experiments. The power and false positive rate curves for the original SPH all-branch test are shown in red, while the corresponding performance curves for the modified SPH all-branch test are displayed in black. In this figure, we present performance plots for $L = 1, 4, 10$ and $\rho = 0.3, 0.5, 0.7, 1$.

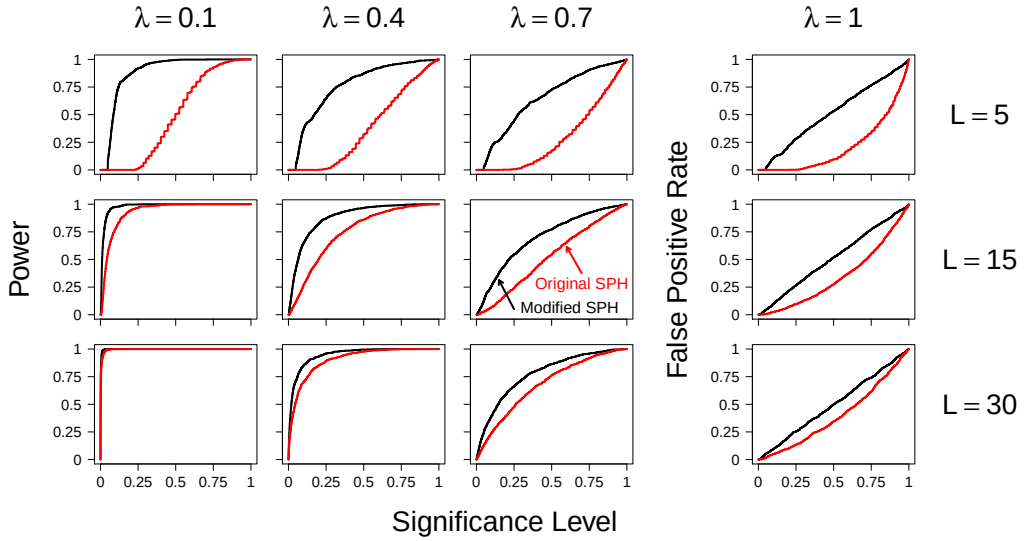
In Figure 3.4, we display some of the power and false positive rate plots from our all-branch simulation experiments. Specifically, we present performance plots for $L = 1, 4, 10$ and $\rho = 0.3, 0.5, 0.7, 1$. These plots suggest that the modified SPH all-branch test is more powerful than the original SPH all-branch test. The gap between the power curves for the two tests is negligible for large L and small ρ but increases as we examine shorter alignments with lower levels of conservation. This latter result is surprising because the null distribution used in the modified SPH all-branch test is based on an asymptotic approximation. The false positive rate plots seem to indicate that the p-values obtained from the modified all-branch test are approximately uniformly distributed under the null hypothesis, even for small L. In addition, it is apparent that the original SPH all-branch p-values are conservative under H_0, confirming the results found in (Siepel et al., 2006).

Figure 3.5 presents power and false positive rate plots from our subtree simulations. We provide performance plots for $L = 5, 15, 30$; $\rho = 0.25, 0.85$; and $\lambda = 0.1, 0.4, 0.7, 1$. The modified SPH subtree tests are more powerful than the original SPH subtree tests in all our simulation experiments. Furthermore, we find that the power curves for the two modified subtree tests are nearly identical; the power curves for the two original subtree tests are similar as well. We only display the power curves for the conditional subtree tests in Figure 3.5; the full set of power curves is shown in Figure 3.6 (see Appendix).

Our subtree simulation experiments suggest that the presence of strong phylogeny-wide conservation (as measured by ρ) makes it more difficult to correctly identify lineage-specific conservation. For ρ close to 1, the separation between the power curves for the original and modified subtree tests is minimal when L is large and λ is small but widens as we analyze shorter elements with lower levels of primate-specific conservation. However, for ρ close to 0, the power curves differ quite substantially for all settings of L and λ. The power of the modified subtree tests is more robust to changes in ρ, while the power of the original subtree tests diminishes greatly as ρ decreases. Thus, it appears that the modified subtree tests can more accurately detect lineage-specific conservation in the presence of strong phylogeny-wide conservation.



(a) Power and false positive rate curves for $\rho = 0.25$



(b) Power and false positive rate curves for $\rho = 0.85$

Figure 3.5: Power and false positive rate plots from our subtree simulation experiments. The power and false positive rate curves for the original SPH conditional subtree test are shown in red, while the corresponding performance curves for the modified SPH conditional subtree test are displayed in black. In this figure, we present performance plots for $L = 5, 15, 30$; $\rho = 0.25, 0.85$; and $\lambda = 0.1, 0.4, 0.7, 1$.

Our results show that testing for conservation in a subtree of interest is more difficult than testing for conservation across the entire phylogeny. This is not too surprising because we have to account for the uncertainty associated with estimating ρ in the subtree tests and the modified subtree test statistics achieve asymptotic normality under H_0 at a rate slower than is observed for the modified all-branch test statistic. The latter can be seen by comparing the false positive rate plots in Figures 3.4 and 3.5; additionally, these plots indicate that the original SPH subtree p-values are more conservative than the original SPH all-branch p-values.

3.5 Discussion

In this chapter, we present a post-order tree traversal algorithm that computes prior and posterior stochastic mapping variances with space and time complexity linear in the number of tips on the phylogeny; prior and posterior mapping covariances are efficiently calculated using a generalized version of this algorithm (see Appendix). In many applications, including the ones presented in this chapter, the posterior distribution of stochastic mapping summaries can be approximated by a normal distribution. Since the normal distribution is fully specified by its mean vector and covariance matrix, our new stochastic mapping (co)variance computation together with an already available efficient way of computing the stochastic mapping mean enables access to the full posterior distribution of stochastic mapping summaries without resorting to costly simulations. Our methodology builds upon the results of Minin and Suchard (2008b) and is inspired by the work of Kenney and Gu (2012), who devised a dynamic programming procedure for calculating second derivatives of phylogenetic likelihood functions.

In fact, our algorithm for computing prior and posterior mapping moments can be adapted to provide a more straightforward description of the Kenney and Gu (2012) algorithm. If we replace the restricted factorial moments in the recursive equations of our algorithm with the appropriate derivatives of CTMC transition probabilities, then we would obtain an algorithm that computes the second derivatives of interest. Even though this reformulation is equivalent

to the approach of Kenney and Gu (2012), we believe our presentation is more streamlined and easier to follow. We also point out that our algorithm and the work by Kenney and Gu (2012) can be viewed as extensions of analogous calculations for hidden Markov models (Lystig and Hughes, 2002; Cappé and Moulines, 2005).

The exact calculations of prior and posterior mapping moments allow us to construct more efficient posterior predictive rate variation tests and more accurate tests of genomic conservation. From our analyses of the β-globin and influenza datasets, we find that the posterior dispersion index for substitution counts is a useful discrepancy measure for detecting observed rate variation across sites. We motivate our use of the posterior dispersion index as a discrepancy measure by alluding to the prior dispersion index for substitution counts often being greater than 1 in the presence of rate variation among sites. The observed and predicted distributions of the posterior dispersion index in Figure 3.3 do not have support outside the interval $(0, 1)$, but this is not entirely inconsistent with the reasoning given above because a simple limiting argument shows that the posterior dispersion index for long sequence alignments is less than or equal to the prior dispersion index, even in the presence of across-site rate variation. From our all-branch and subtree simulation experiments, we see that the modified SPH tests are better than the original SPH tests at correctly identifying phylogeny-wide and lineage-specific conservation in genomic sequence alignments. Specifically, we observe that the differences in power between the original and modified conservation tests are greatest when we analyze short elements with low levels of conservation. Similarly to the techniques found in (Kellis et al., 2003), our modified tests of conservation could aid in the discovery of new transcriptional regulatory motifs in the human genome.

The work presented here can be extended in several different directions. One obvious extension is to generalize our post-order tree traversal algorithm to calculate higher-order moments of stochastic mapping summaries, such as the (co)skewness and (co)kurtosis. These higher-order moments could then be used to construct more complex discrepancy measures for posterior predictive model diagnostics. Similarly to the test of rate variation among sites, it is of interest to develop a posterior predictive approach to testing for rate varia-

tion among branches of the phylogeny. Posterior predictive tests of this type could serve as useful diagnostic tools to assess the appropriateness of relaxed molecular clock models (Drummond et al., 2006). In addition, it would also be beneficial to establish a posterior predictive framework for testing the stationarity and homogeneity assumptions implicit in reversible substitution models. These tests could be used to determine the suitability of nonreversible substitution models (Boussau and Gouy, 2006). Our dynamic programming algorithm can be easily altered to compute the posterior mapping variance of labeled dwelling times, which could be employed as a discrepancy measure in these posterior predictive tests. Finally, another potential avenue for future research is to improve the two tests of genomic acceleration discussed by Pollard et al. (2010), where acceleration refers to evolution that is faster than expected. Pursuing this line of research could lead to the detection of new "human accelerated regions" in our genome (Pollard et al., 2006a,b). Based on the results in this chapter and the promising directions for further study, we think stochastic mapping is and will continue to be essential to making reliable inferences about the latent evolutionary process on the phylogeny.

Appendix

Monte Carlo Summary Tables

	Standard errors (β-globin)				Standard errors (influenza)			
	$L = 50$	$L = 100$	$L = 200$	$L = 400$	$L = 50$	$L = 100$	$L = 200$	$L = 400$
$m = 100$	0.91	1.3	1.8	2.5	0.094	0.13	0.19	0.26
$m = 500$	0.41	0.57	0.82	1.2	0.042	0.060	0.086	0.12
$m = 1000$	0.29	0.41	0.58	0.82	0.030	0.042	0.061	0.085
$m = 10000$	0.092	0.13	0.18	0.26	0.0097	0.013	0.019	0.027

(a) Monte Carlo standard errors for the β-globin and influenza datasets

	$L = 50$	$L = 100$	$L = 200$	$L = 400$
β-globin	22	41	84	170
influenza	0.34	0.54	1.3	2.3

(b) Exact calculations of T_{var} for the β-globin and influenza datasets

Table 3.2: Monte Carlo summary tables for the β-globin and influenza datasets. (a) Monte Carlo standard errors associated with simulation-based T_{var} estimates. We compute these standard errors on randomly subsampled alignments of length L using m Monte Carlo replicates per site. Each table entry represents an averaged Monte Carlo standard error, where the averaging is done over 200 randomly subsampled posterior $\boldsymbol{\theta}$'s. (b) Exact computations of T_{var}. Each table entry denotes an average over exact values of T_{var}, where the averaging is done over the same 200 posterior samples of $\boldsymbol{\theta}$ mentioned above. All table entries in (a) and (b) are rounded to two significant digits.

Prior and Posterior Mapping Covariance Computation

In this subsection, we describe how to efficiently compute prior and posterior mapping covariances. We generalize the post-order tree traversal algorithm discussed in the main part of the chapter and present the necessary formulas for calculating these covariances. Much of the intuition provided for our original tree traversal algorithm carries over to this generalized procedure.

Let $\mathrm{Cov}(H_{\Omega_1}, H_{\Omega_2})$ and $\mathrm{Cov}(H_{\Omega_1}, H_{\Omega_2}|\mathbf{D})$ denote the prior and posterior mapping covari-

ances, respectively, where $\Omega_1, \Omega_2 \subseteq \Theta$ are predefined sets of branch indices. We consider first the calculation of the posterior mapping covariance $\mathrm{Cov}(H_{\Omega_1}, H_{\Omega_2}|\mathbf{D})$. The vectors $\mathbf{F}_u$ and $\mathbf{S}_b$ are defined and computed as in our original tree traversal procedure for all nodes $u \in \{1, ..., n-1, n, ..., 2n-1\}$ and branches $b \in \Theta$. We introduce the m-long vectors $\mathbf{V}_b^{(\Omega_1,[1])}$, $\mathbf{V}_b^{(\Omega_1,[2])}$, $\mathbf{V}_b^{(\Omega_2,[1])}$, $\mathbf{V}_b^{(\Omega_2,[2])}$, $\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[1])}$, $\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[2])}$, $\mathbf{W}_b^{(\Omega_1)}$, $\mathbf{W}_b^{(\Omega_2)}$, and $\mathbf{W}_b^{(\Omega_1,\Omega_2)}$ for all $b \in \Theta$. The ith entries in $\mathbf{V}_b^{(\Omega_1,[1])}$ and $\mathbf{V}_b^{(\Omega_1,[2])}$ are mathematically defined as:

$$\sum_{b^\dagger} \sum_{\mathbf{i}_b} e^{[1]}_{i^*_{p(b^\dagger)} i^*_{c(b^\dagger)}}(h, t_{b^\dagger}) \prod_{b^* \in \Theta_b \setminus \{b^\dagger\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}), \tag{3.41}$$

$$\sum_{b^\dagger} \sum_{\mathbf{i}_b} e^{[2]}_{i^*_{p(b^\dagger)} i^*_{c(b^\dagger)}}(h, t_{b^\dagger}) \prod_{b^* \in \Theta_b \setminus \{b^\dagger\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}), \tag{3.42}$$

respectively, where the state of parent node $p(b)$ is i and $b^\dagger \in \Omega_{1,b}$ for $\Omega_{1,b} = \Omega_1 \cap \Theta_b$. The entries in $\mathbf{V}_b^{(\Omega_2,[1])}$ and $\mathbf{V}_b^{(\Omega_2,[2])}$ and $\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[1])}$ and $\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[2])}$ are defined analogously by replacing Ω_1 with Ω_2 and $\Omega_1 \cap \Omega_2$, respectively, in the above definitions. The ith element of the vector $\mathbf{W}_b^{(\Omega_1)}$ is equal to:

$$\sum_{b^\dagger \neq b^{\dagger\dagger}} \sum_{\mathbf{i}_b} e^{[1]}_{i^*_{p(b^\dagger)} i^*_{c(b^\dagger)}}(h, t_{b^\dagger}) e^{[1]}_{i^*_{p(b^{\dagger\dagger})} i^*_{c(b^{\dagger\dagger})}}(h, t_{b^{\dagger\dagger}}) \prod_{b^* \in \Theta_b \setminus \{b^\dagger, b^{\dagger\dagger}\}} p_{i^*_{p(b^*)} i^*_{c(b^*)}}(t_{b^*}), \tag{3.43}$$

where the state of parent node $p(b)$ is i and $b^\dagger, b^{\dagger\dagger} \in \Omega_{1,b}$ for $\Omega_{1,b}$ defined as above. The elements in $\mathbf{W}_b^{(\Omega_2)}$ and $\mathbf{W}_b^{(\Omega_1,\Omega_2)}$ are similarly defined, except that $b^\dagger, b^{\dagger\dagger} \in \Omega_{2,b}$ and $b^\dagger \in \Omega_{1,b}, b^{\dagger\dagger} \in \Omega_{2,b}$, respectively, for the same $\Omega_{1,b}$ and $\Omega_{2,b} = \Omega_2 \cap \Theta_b$.

For all terminal branches $b \in \mathcal{E}$, we define:

$$\mathbf{V}_b^{(\Omega_1,[1])} = \mathbf{e}^{[1]}(h, t_b)\mathbf{F}_{c(b)} \mathbb{1}_{\{b \in \Omega_1\}}, \tag{3.44}$$

$$\mathbf{V}_b^{(\Omega_1,[2])} = \mathbf{e}^{[2]}(h, t_b)\mathbf{F}_{c(b)} \mathbb{1}_{\{b \in \Omega_1\}}, \tag{3.45}$$

$$\mathbf{V}_b^{(\Omega_2,[1])} = \mathbf{e}^{[1]}(h, t_b)\mathbf{F}_{c(b)} \mathbb{1}_{\{b \in \Omega_2\}}, \tag{3.46}$$

$$\mathbf{V}_b^{(\Omega_2,[2])} = \mathbf{e}^{[2]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_2\}}, \tag{3.47}$$

$$\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[1])} = \mathbf{e}^{[1]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_1\cap\Omega_2\}}, \tag{3.48}$$

$$\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[2])} = \mathbf{e}^{[2]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_1\cap\Omega_2\}}. \tag{3.49}$$

All entries in $\mathbf{W}_b^{(\Omega_1)}$, $\mathbf{W}_b^{(\Omega_2)}$, and $\mathbf{W}_b^{(\Omega_1,\Omega_2)}$ for $b\in\mathcal{E}$ are set to 0. The recursive formulas for calculating these vectors at internal branches $b\in\mathcal{I}$ are:

$$\mathbf{V}_b^{(\Omega_1,[1])} = \mathbf{e}^{[1]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_1\}} + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_1,[1])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_1,[1])}\circ\mathbf{S}_{b_1}\Big), \tag{3.50}$$

$$\mathbf{V}_b^{(\Omega_1,[2])} = \mathbf{e}^{[2]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_1\}} + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_1,[2])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_1,[2])}\circ\mathbf{S}_{b_1}\Big), \tag{3.51}$$

$$\mathbf{V}_b^{(\Omega_2,[1])} = \mathbf{e}^{[1]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_2\}} + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_2,[1])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_2,[1])}\circ\mathbf{S}_{b_1}\Big), \tag{3.52}$$

$$\mathbf{V}_b^{(\Omega_2,[2])} = \mathbf{e}^{[2]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_2\}} + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_2,[2])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_2,[2])}\circ\mathbf{S}_{b_1}\Big), \tag{3.53}$$

$$\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[1])} = \mathbf{e}^{[1]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_1\cap\Omega_2\}} + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_1\cap\Omega_2,[1])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_1\cap\Omega_2,[1])}\circ\mathbf{S}_{b_1}\Big), \tag{3.54}$$

$$\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[2])} = \mathbf{e}^{[2]}(h,t_b)\mathbf{F}_{c(b)}\mathbb{1}_{\{b\in\Omega_1\cap\Omega_2\}} + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_1\cap\Omega_2,[2])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_1\cap\Omega_2,[2])}\circ\mathbf{S}_{b_1}\Big), \tag{3.55}$$

$$\begin{aligned}\mathbf{W}_b^{(\Omega_1)} &= 2\times\mathbf{e}^{[1]}(h,t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_1,[1])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_1,[1])}\circ\mathbf{S}_{b_1}\Big)\mathbb{1}_{\{b\in\Omega_1\}}\\ &\quad + \mathbf{P}(t_b)\Big(2\times\mathbf{V}_{b_1}^{(\Omega_1,[1])}\circ\mathbf{V}_{b_2}^{(\Omega_1,[1])} + \mathbf{W}_{b_1}^{(\Omega_1)}\circ\mathbf{S}_{b_2} + \mathbf{W}_{b_2}^{(\Omega_1)}\circ\mathbf{S}_{b_1}\Big),\end{aligned} \tag{3.56}$$

$$\begin{aligned}\mathbf{W}_b^{(\Omega_2)} &= 2\times\mathbf{e}^{[1]}(h,t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_2,[1])}\circ\mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_2,[1])}\circ\mathbf{S}_{b_1}\Big)\mathbb{1}_{\{b\in\Omega_2\}}\\ &\quad + \mathbf{P}(t_b)\Big(2\times\mathbf{V}_{b_1}^{(\Omega_2,[1])}\circ\mathbf{V}_{b_2}^{(\Omega_2,[1])} + \mathbf{W}_{b_1}^{(\Omega_2)}\circ\mathbf{S}_{b_2} + \mathbf{W}_{b_2}^{(\Omega_2)}\circ\mathbf{S}_{b_1}\Big),\end{aligned} \tag{3.57}$$

$$\begin{aligned}\mathbf{W}_b^{(\Omega_1,\Omega_2)} = \mathbf{e}^{[1]}(h,t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_2,[1])} \circ \mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_2,[1])} \circ \mathbf{S}_{b_1}\Big)\mathbb{1}_{\{b\in\Omega_1\}} \\ + \mathbf{e}^{[1]}(h,t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_1,[1])} \circ \mathbf{S}_{b_2} + \mathbf{V}_{b_2}^{(\Omega_1,[1])} \circ \mathbf{S}_{b_1}\Big)\mathbb{1}_{\{b\in\Omega_2\}} \\ + \mathbf{P}(t_b)\Big(\mathbf{V}_{b_1}^{(\Omega_1,[1])} \circ \mathbf{V}_{b_2}^{(\Omega_2,[1])} + \mathbf{V}_{b_2}^{(\Omega_1,[1])} \circ \mathbf{V}_{b_1}^{(\Omega_2,[1])} \\ + \mathbf{W}_{b_1}^{(\Omega_1,\Omega_2)} \circ \mathbf{S}_{b_2} + \mathbf{W}_{b_2}^{(\Omega_1,\Omega_2)} \circ \mathbf{S}_{b_1}\Big),\end{aligned} \tag{3.58}$$

where b_1 and b_2 represent the two branches that are "below" branch b. This generalized tree traversal algorithm terminates after computing $\mathbf{F}_u$, $\mathbf{S}_b$, $\mathbf{V}_b^{(\Omega_1,[1])}$, $\mathbf{V}_b^{(\Omega_1,[2])}$, $\mathbf{V}_b^{(\Omega_2,[1])}$, $\mathbf{V}_b^{(\Omega_2,[2])}$, $\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[1])}$, $\mathbf{V}_b^{(\Omega_1\cap\Omega_2,[2])}$, $\mathbf{W}_b^{(\Omega_1)}$, $\mathbf{W}_b^{(\Omega_2)}$, and $\mathbf{W}_b^{(\Omega_1,\Omega_2)}$ for $u = root$ and $b \in \{root_1, root_2\}$, where $root$ denotes the root node label and $root_1$ and $root_2$ represent the two branches connecting the root node to its children. The restricted mapping moments of interest are calculated as follows:

$$\mathrm{E}(H_{\Omega_1}\mathbb{1}_{\mathbf{D}}) = \boldsymbol{\pi}^T\Big(\mathbf{V}_{root_1}^{(\Omega_1,[1])} \circ \mathbf{S}_{root_2} + \mathbf{V}_{root_2}^{(\Omega_1,[1])} \circ \mathbf{S}_{root_1}\Big), \tag{3.59}$$

$$\mathrm{E}(H_{\Omega_2}\mathbb{1}_{\mathbf{D}}) = \boldsymbol{\pi}^T\Big(\mathbf{V}_{root_1}^{(\Omega_2,[1])} \circ \mathbf{S}_{root_2} + \mathbf{V}_{root_2}^{(\Omega_2,[1])} \circ \mathbf{S}_{root_1}\Big), \tag{3.60}$$

$$\begin{aligned}\mathrm{E}(H_{\Omega_1}^2\mathbb{1}_{\mathbf{D}}) = \boldsymbol{\pi}^T\Big[2 \times \mathbf{V}_{root_1}^{(\Omega_1,[1])} \circ \mathbf{V}_{root_2}^{(\Omega_1,[1])} + \mathbf{W}_{root_1}^{(\Omega_1)} \circ \mathbf{S}_{root_2} + \mathbf{W}_{root_2}^{(\Omega_1)} \circ \mathbf{S}_{root_1} \\ + \big(\mathbf{V}_{root_1}^{(\Omega_1,[1])} + \mathbf{V}_{root_1}^{(\Omega_1,[2])}\big) \circ \mathbf{S}_{root_2} + \big(\mathbf{V}_{root_2}^{(\Omega_1,[1])} + \mathbf{V}_{root_2}^{(\Omega_1,[2])}\big) \circ \mathbf{S}_{root_1}\Big],\end{aligned} \tag{3.61}$$

$$\begin{aligned}\mathrm{E}(H_{\Omega_2}^2\mathbb{1}_{\mathbf{D}}) = \boldsymbol{\pi}^T\Big[2 \times \mathbf{V}_{root_1}^{(\Omega_2,[1])} \circ \mathbf{V}_{root_2}^{(\Omega_2,[1])} + \mathbf{W}_{root_1}^{(\Omega_2)} \circ \mathbf{S}_{root_2} + \mathbf{W}_{root_2}^{(\Omega_2)} \circ \mathbf{S}_{root_1} \\ + \big(\mathbf{V}_{root_1}^{(\Omega_2,[1])} + \mathbf{V}_{root_1}^{(\Omega_2,[2])}\big) \circ \mathbf{S}_{root_2} + \big(\mathbf{V}_{root_2}^{(\Omega_2,[1])} + \mathbf{V}_{root_2}^{(\Omega_2,[2])}\big) \circ \mathbf{S}_{root_1}\Big],\end{aligned} \tag{3.62}$$

$$\begin{aligned}\mathrm{E}(H_{\Omega_1}H_{\Omega_2}\mathbb{1}_{\mathbf{D}}) = \boldsymbol{\pi}^T\Big[\mathbf{V}_{root_1}^{(\Omega_1,[1])} \circ \mathbf{V}_{root_2}^{(\Omega_2,[1])} + \mathbf{V}_{root_2}^{(\Omega_1,[1])} \circ \mathbf{V}_{root_1}^{(\Omega_2,[1])} \\ + \mathbf{W}_{root_1}^{(\Omega_1,\Omega_2)} \circ \mathbf{S}_{root_2} + \mathbf{W}_{root_2}^{(\Omega_1,\Omega_2)} \circ \mathbf{S}_{root_1} \\ + \big(\mathbf{V}_{root_1}^{(\Omega_1\cap\Omega_2,[1])} + \mathbf{V}_{root_1}^{(\Omega_1\cap\Omega_2,[2])}\big) \circ \mathbf{S}_{root_2} \\ + \big(\mathbf{V}_{root_2}^{(\Omega_1\cap\Omega_2,[1])} + \mathbf{V}_{root_2}^{(\Omega_1\cap\Omega_2,[2])}\big) \circ \mathbf{S}_{root_1}\Big].\end{aligned} \tag{3.63}$$

We also know that $\mathrm{P}(\mathbf{D}) = \boldsymbol{\pi}^T\mathbf{F}_{root}$ (Felsenstein, 1981). Our efficient computations of the

above restricted mapping moments allow us to calculate the associated posterior mapping moments using the following equations:

$$\mathrm{E}(H_{\Omega_1}|\mathbf{D}) = \frac{\mathrm{E}(H_{\Omega_1}\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})}, \tag{3.64}$$

$$\mathrm{E}(H_{\Omega_2}|\mathbf{D}) = \frac{\mathrm{E}(H_{\Omega_2}\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})}, \tag{3.65}$$

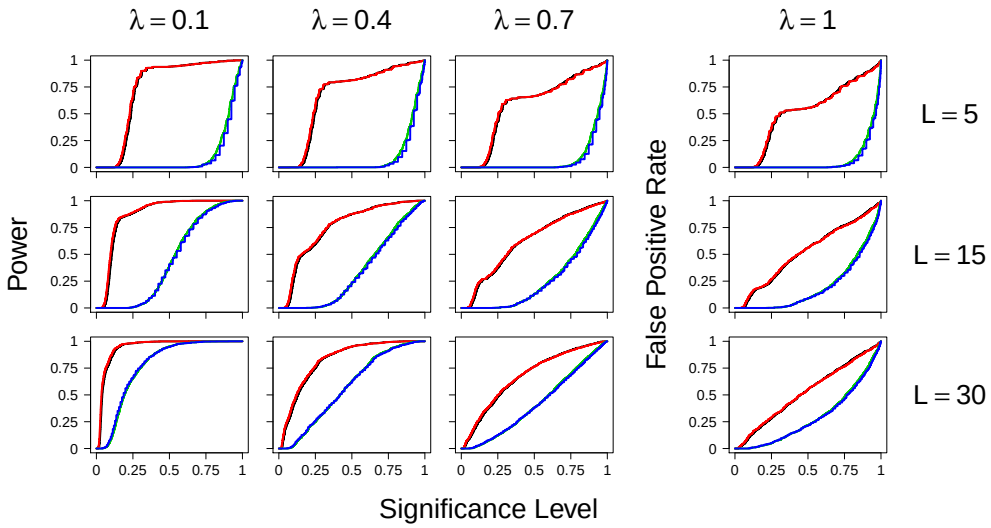
$$\mathrm{Var}(H_{\Omega_1}|\mathbf{D}) = \mathrm{E}(H_{\Omega_1}^2|\mathbf{D}) - \mathrm{E}(H_{\Omega_1}|\mathbf{D})^2 = \frac{\mathrm{E}(H_{\Omega_1}^2\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})} - \left[\frac{\mathrm{E}(H_{\Omega_1}\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})}\right]^2, \tag{3.66}$$

$$\mathrm{Var}(H_{\Omega_2}|\mathbf{D}) = \mathrm{E}(H_{\Omega_2}^2|\mathbf{D}) - \mathrm{E}(H_{\Omega_2}|\mathbf{D})^2 = \frac{\mathrm{E}(H_{\Omega_2}^2\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})} - \left[\frac{\mathrm{E}(H_{\Omega_2}\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})}\right]^2, \tag{3.67}$$

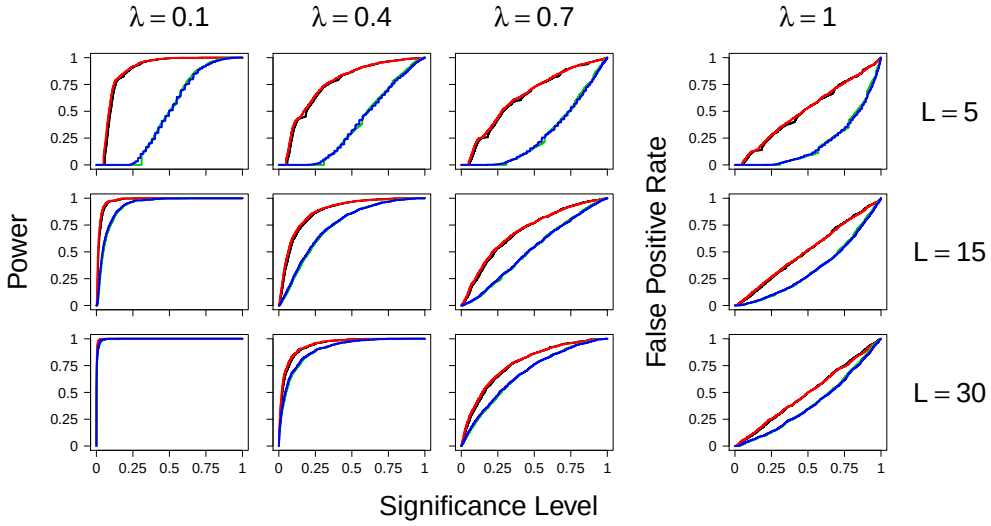
$$\begin{aligned}\mathrm{Cov}(H_{\Omega_1}, H_{\Omega_2}|\mathbf{D}) &= \mathrm{E}(H_{\Omega_1}H_{\Omega_2}|\mathbf{D}) - \mathrm{E}(H_{\Omega_1}|\mathbf{D})\mathrm{E}(H_{\Omega_2}|\mathbf{D}) \\ &= \frac{\mathrm{E}(H_{\Omega_1}H_{\Omega_2}\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})} - \left[\frac{\mathrm{E}(H_{\Omega_1}\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})}\right]\left[\frac{\mathrm{E}(H_{\Omega_2}\mathbb{1}_{\mathbf{D}})}{\mathrm{P}(\mathbf{D})}\right].\end{aligned} \tag{3.68}$$

The prior mapping moments can be obtained by modifying the tree traversal procedure discussed in this subsection. The only changes that need to be made are to the initializations of $\mathbf{F}_u$ at all terminal nodes u. If we set $F_{ui} = 1$ for all terminal nodes $u \in \{n, ..., 2n-1\}$ and $i = 1, ..., m$, then our algorithm will be able to compute the prior moments of interest. In this case, equations (3.64)-(3.68) are used to calculate the prior mapping moments. Thus, the prior and posterior mapping covariances are computed in a similar fashion, but according to different initializations of the $\mathbf{F}_u$ vectors at the tips of the phylogeny.

SPH Subtree Simulation Plots



(a) Power and false positive rate curves for $\rho = 0.25$



(b) Power and false positive rate curves for $\rho = 0.85$

Figure 3.6: Power and false positive rate plots from our subtree simulation experiments. The power and false positive rate curves for the original SPH marginal (conditional) subtree test are shown in green (blue), while the corresponding performance curves for the modified SPH marginal (conditional) subtree test are displayed in black (red). In this figure, we present performance plots for $L = 5, 15, 30$; $\rho = 0.25, 0.85$; and $\lambda = 0.1, 0.4, 0.7, 1$.

Chapter 4

A BAYESIAN PHYLOGENETIC HIDDEN MARKOV MODEL FOR B CELL RECEPTOR SEQUENCE ANALYSIS

4.1 Introduction

One of the most important features of the adaptive immune system is its ability to create a wide variety of high affinity antibodies, the soluble form of B cell receptors (BCRs), that bind to and neutralize pathogens in the body. The initial BCR diversity is generated by randomly joining together various gene segments in a process called VDJ rearrangement; after an initial testing process the cells reach the "naive" state. When stimulated by binding to foreign material called "antigen," B cells diversify further by forming germinal centers (GCs) in the secondary lymphoid organs and going through an affinity maturation process. During the GC reaction, B cells mutate rapidly in a process called somatic hypermutation (SHM), and the high affinity clones are positively selected for via clonal expansion. We would like to better understand the GC mutation and selection processes, because insight into these GC dynamics could aid in the development of vaccines for highly mutable pathogens such as influenza and HIV (Mascola and Haynes, 2013). Inferring the mutational pathways from naive to mature BCR sequences in a GC is an important step in the design of effective vaccines. We have developed a new statistical inference framework that better estimates these mutational pathways and quantifies uncertainty in these estimates.

Rational vaccine design efforts mentioned above depend on accurate inference of full evolutionary paths from a given naive sequence to the corresponding mature BCR sequences in a GC. By understanding the mutational pathways that lead to broadly neutralizing antibodies (bNAbs), vaccines could then be constructed that induce the production of these bNAbs in the body (Stamatatos et al., 2017). For instance, in the case of HIV, most bNAbs are

generally not observed until after a long period of chronic infection, requiring prospective studies to characterize them (Liao et al., 2013). While there have been many such studies that experimentally test longitudinal B cell samples spanning from an initial HIV infection to the development of mature bNAbs (Liao et al., 2013; Doria-Rose et al., 2014, 2016), obtaining early prospective samples is difficult. One way to avoid the process described above is to infer intermediate lineage sequences computationally, synthesize them in the laboratory, and test their binding and neutralization abilities (Doria-Rose et al., 2014; Simonich et al., 2018). Consequently, we will focus on inferring mutational pathways through ancestral sequence reconstruction, a commonly used technique in computational phylogenetics.

Much of the existing BCR sequence analysis literature focuses on modeling either the VDJ recombination process, or the phylogenetic diversification process, but not both. Elhanati et al. (2015) develop a likelihood-based model that encompasses the VDJ rearrangement, SHM, and clonal selection processes; however, they implicitly assume clonal sequences arise independently and do not consider the phylogenetic structure of SHM. Hoehn et al. (2017) introduce a novel codon substitution model for ancestral lineage inference that encodes SHM context-dependent mutational effects but does not account for VDJ rearrangement dynamics in the naive sequence within their maximum likelihood phylogenetic inference framework. Yaari et al. (2013) provide estimates of context-dependent SHM substitution probabilities and motif mutability scores based on an aggregated dataset consisting of the synonymous codon positions of productive antibody sequences. Ralph and Matsen IV (2016a) are able to infer naive BCR sequences using a HMM-based approach that models the VDJ rearrangement process and assumes independent evolution across the different lineages. While these efforts have contributed greatly to our understanding of GC dynamics, we believe that the performance of clonal lineage and ancestral sequence inference procedures can be enhanced by using an evolutionary model for SHM that also accounts for uncertainty in the naive rearrangement process.

Kepler (2013) has developed a likelihood-based SHM modeling framework that jointly estimates the naive sequence and the associated clonal tree and incorporates information

about the VDJ rearrangement process. However, this work does not consider phylogenetic uncertainty in the naive sequence estimation procedure. While there is evidence to suggest that ancestral sequence estimation is robust to phylogenetic uncertainty in other settings (Hanson-Smith et al., 2010), the parameter regime of BCR diversification is quite different than this previous work, leaving open the question of whether incorporating phylogenetic uncertainty would aid in ancestral sequence inference. Therefore, we would like to construct a phylogenetic inference procedure that not only allows for easy quantification of phylogenetic uncertainty but also models the VDJ recombination as an informative prior for the naive sequence at the root of a phylogenetic tree describing the evolution of one GC clone.

In this chapter, we propose a Bayesian approach to phylogenetic inference for clonal sequences that is based on a phylogenetic hidden Markov model (phylo-HMM) (Siepel and Haussler, 2005). Our phylo-HMM models both the naive rearrangement and SHM processes. The Bayesian framework allows us to naturally account for uncertainty in all unobserved variables, including a phylogenetic tree, via posterior distribution sampling. We perform simulation-based experiments to show that naive sequence and phylogenetic inference performed jointly provides higher-quality estimates than those obtained by considering these inferences separately. Our application to real data reveals significant uncertainty in naive sequences, confirming the importance of a Bayesian approach.

4.2 Methods

4.2.1 Overview

The immune system is able to generate a diverse set of naive BCR sequences due to the VDJ rearrangement process (Figure 4.1a). In this process, the B cell first randomly selects V, D, and J gene segments (i.e. DNA sequences) from the respective gene pools in the body. Before joining the gene segments together, the B cell randomly deletes nucleotides at both ends of the V-D and D-J junction regions (i.e. exonuclease deletions) and randomly inserts nucleotides in the same junction regions (i.e. non-templated insertions). Although the VDJ rearrangement

process samples germline genes from the same gene pools for all the naive BCRs in a given individual, different people may have different collections of germline genes (Watson et al., 2017). BCR sequences can be partitioned into framework (FWK) and complementarity-determining (CDR) regions. The BCR binding affinity is largely determined by the sequence segments in the CDR regions, and among all the CDR regions the CDR3 region contributes the most to antigen-binding specificity and has the highest amount of sequence variability. In addition, naive amino-acid sequences are important because BCRs are proteins and protein structure and function are determined by the corresponding amino-acid sequence; these sequences are obtained from the corresponding DNA sequences by mapping each DNA character triplet into an amino-acid letter. The `partis` software program (Ralph and Matsen IV, 2016a,b) treats the naive sequence generative process as a discrete-time Markov chain (DTMC) going from left to right across the sequence bases and also permits maximum likelihood inference of the associated model parameters, which are defined according to the VDJ rearrangement dynamics. We emphasize that the "time" of this DTMC represents position along the sequence, in contrast to the continuous time Markov chain described below modeling the substitution process through chronological time.

As we attempt to characterize the affinity maturation process in GCs, we use the concept of a clonal family (CF) to help analyze BCR repertoire datasets that result from high-throughput sequencing experiments. A clonal family represents a set of BCR sequences that originate from the same naive rearrangement event; in practice, it is simpler to define these families as groups of BCR sequences that share the same naive sequence (Ralph and Matsen IV, 2016b). In this work, we use the latter definition of a CF. The CF definition used here relies on the basic assumption that the unmutated common ancestor of a collection of clonally related sequences can only be identified by the corresponding DNA sequence. There is a chance that two different naive B cells with identical BCR sequences could seed multiple GCs and form their own lineages, but the observed clones from the two GCs would be collapsed into a single CF because both lineages share the same naive sequence. Thus, a correctly inferred CF under this definition will be a cluster of sequences that derive from

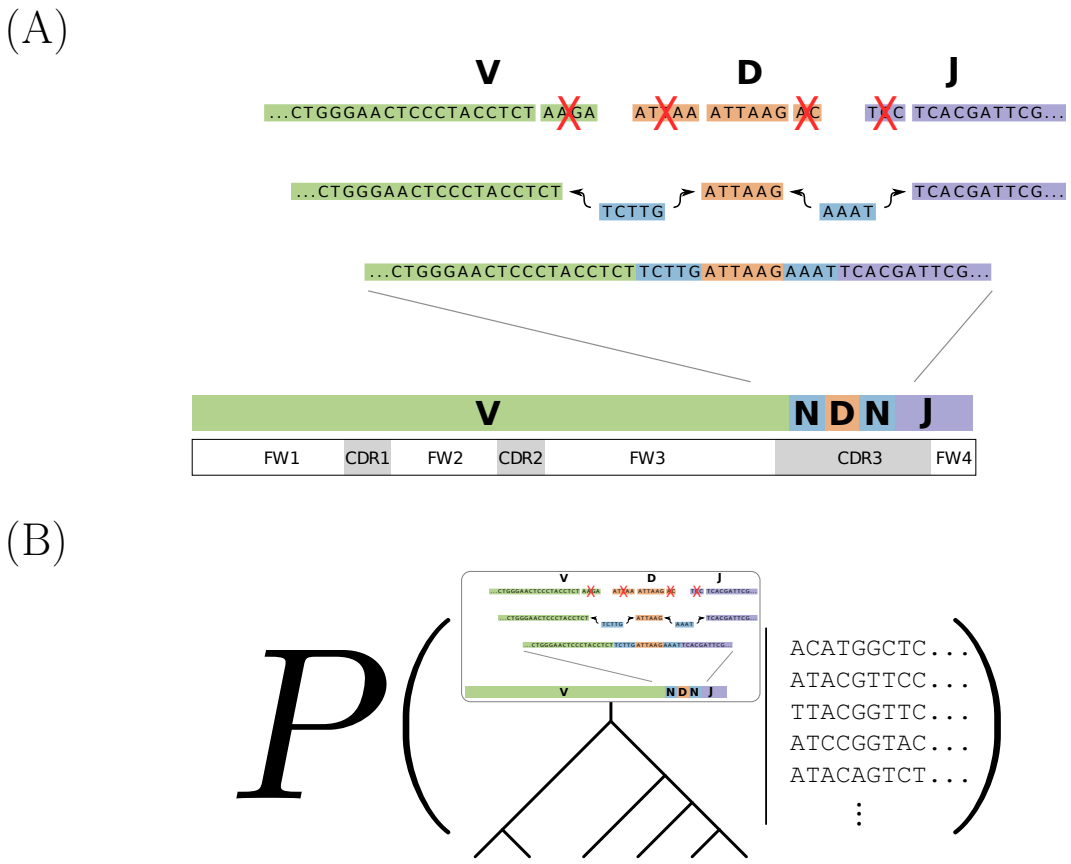


Figure 4.1: (A) A schematic representation of the naive rearrangement process. First, V (green), D (orange), and J (purple) genes are randomly selected from the respective gene pools in the body. Then, nucleotides are randomly deleted (red X's) from both ends of the V-D and D-J junction regions and random bases (blue) are added to the same junction regions before the V, D, and J germline genes can be joined together. The BCR sequences can be partitioned into framework (FWK) and complementarity-determining (CDR) regions. This image was taken from (Ralph and Matsen IV, 2016a). (B) Our Bayesian phylo-HMM jointly models VDJ recombination at the root of the tree (using an HMM) and then subsequent diversification (via a phylogenetic tree). We do posterior inference conditioning on the observed sequence alignment in a clonal family, but not on a fixed inferred naive sequence.

naive B cells with the same BCR sequence; because GC mutation and selection occur at the sequence level, all sequences in a CF go through the same mutation and selection processes. Using this CF definition, Ralph and Matsen IV (2016b) describe how to cluster BCR sequences from large-scale repertoire datasets into CFs with high accuracy using `partis`.

We want to emphasize that repertoire datasets can be clustered and pre-processed in many different ways, but `partis` provides a convenient way to cluster repertoire sequences and produce CF-specific multiple sequence alignments (MSAs).

As mentioned in the opening section, naive sequence bases accumulate mutations via SHM and the corresponding B cells undergo cellular replication. Phylogenetic tree models provide a realistic and mathematically convenient way to represent the aforementioned B cell evolutionary dynamics of a CF. In particular, these models define a likelihood at each MSA position/site as a function of unknown parameters, which consist of a tree topology, branch lengths, and continuous-time Markov chain (CTMC) substitution model parameters. While B cell evolution in a CF is not independent across the different site positions, site-specific phylogenetic likelihoods provide a convenient first approximation in modeling this phenomenon. To help us illustrate how these phylogenetic models work, we provide an example tree visualization for 4 sequences (Figure 4.7).

Given a tree topology with branch lengths, we use a CTMC substitution model to calculate the probabilities of state changes along the branches of the tree. Specifically if t denotes a branch length on a tree, CTMC substitution models allow one to calculate $p_{ij}(t)$, which denotes the probability of going from state i to state j on a branch of length t, where $i, j \in \{A, G, C, T\}$. It is common to use a reversible CTMC substitution model on a tree (Felsenstein, 2004); a reversible substitution model is a Markov substitution model that, if started at stationarity, can be run backwards in time, with the resulting backward Markov model following the same probability law as the original forward model. The standard phylogenetic generative process can be described at each alignment site as follows: 1) a DNA state at the root node is drawn independently according to the same 4-state discrete distribution and 2) the states at the other nodes are sampled in a pre-order traversal using the computed CTMC probabilities at each branch $p_{ij}(t)$; this model is a special case of a directed graphical model (Lauritzen, 1996) that probabilistically generates sequence alignments.

Clearly, standard phylogenetic models do not account for naive rearrangement dynamics at the root because, as we discussed above, the root state at each sequence position is sampled

independently according to an identical distribution. Instead, if we draw root sequence states from the DTMC mentioned earlier, we would obtain a sequence evolution model that more accurately describes B cell evolutionary dynamics. Thus, we formulate our phylo-HMM to consist of a hidden state DTMC model for naive sequences that explicitly incorporates VDJ rearrangement information and an emission distribution generating sequence alignments conditional on the naive sequence that is based on phylogenetic likelihoods. This phylo-HMM hopefully leads to more accurate naive sequence estimates and, as a result of that, higher-quality intermediate ancestral sequence estimates. We introduce a pictorial representation of our Bayesian phylo-HMM to make clear our target of inference (Figure 4.1b).

From a statistical point of view, it is common to assume the naive sequence root node is a leaf node holding naive sequence bases connected to a "virtual root node" (i.e. what we call "root" node for our phylogenetic model above) via a branch length of 0. Even though it may seem like we have described a rooted tree model above, it turns out that under the assumptions of a reversible substitution model and a nucleotide distribution at the virtual root starting at stationarity, the Pulley Principle, first discussed in (Felsenstein, 1981), states that the virtual root may be placed anywhere on the tree without affecting the likelihood. This implies that the model described above does not correspond to a single rooted tree, but an equivalence class of rooted trees that maps to a unique unrooted tree. This is an important distinction as our phylo-HMM will in fact use an unrooted tree model, which will be justified when we describe our posterior sampling approach.

4.2.2 Notation and Assumptions

We now introduce some notation and assumptions that will be used throughout this chapter. Let $\mathbf{D} = \{D_i^{(j)}\}_{i=1:m,j=1:n}$ denote the MSA of m clonal DNA sequences of length n. We define $\mathbf{Y}_{\text{naive}} = \{Y_{\text{naive}}^{(j)}\}_{j=1:n}$ and $\widehat{\mathbf{Y}}_{\text{naive}} = \{\widehat{Y}_{\text{naive}}^{(j)}\}_{j=1:n}$ to be the corresponding length-n naive sequence random variable and point estimate, respectively. We let τ represent a tree topology with m tips and a root branch length; in total, this topology has m internal nodes and $2m-1$ branch lengths. We assume that the ancestral sequence at the root of

τ is $\mathbf{Y}_{\text{naive}}$. Furthermore, we define $\mathbf{t} = \{t_i\}_{i=1:(2m-1)}$ to be the branch lengths associated with τ. Let $\mathbf{Y}_{\text{int}} = \{Y_i^{(j)}\}_{i=1:(m-1),j=1:n}$ denote the internal nodes of τ excluding the naive sequence $\mathbf{Y}_{\text{naive}}$. For convenience, we let $\mathbf{D}^{(j)} = \{D_i^{(j)}\}_{i=1:m}$ and $\mathbf{Y}_{\text{int}}^{(j)} = \{Y_i^{(j)}\}_{i=1:(m-1)}$ symbolize the observed sequence data and unobserved ancestral sequence data, respectively, at MSA site $j \in \{1, ..., n\}$. Conditioned on the root sequence $\mathbf{Y}_{\text{naive}}$, we assume that the ancestral states at each site in the MSA evolve independently along the phylogeny τ according to a general time-reversible (GTR) substitution model (Tavaré, 1986). Let $\mathbf{e} = \{e_{AC}, e_{AG}, e_{AT}, e_{CG}, e_{CT}, e_{GT}\}$ and $\boldsymbol{\pi} = \{\pi_A, \pi_C, \pi_G, \pi_T\}$ represent the GTR exchangeability rates and equilibrium base frequencies, respectively. We also account for phylogenetic rate variation among sites by employing a discrete gamma distribution with a fixed number of rate classes K (Yang, 1994, 1996) and define α to be the associated gamma shape parameter, denote $\mathbf{r} = \{r_1(\alpha), ..., r_K(\alpha)\}$ as the set of discrete rates deterministically induced by α, and let $\mathbf{r}^* = \{r^*_{(j)}\}_{j=1:n}$ represent the discrete rates chosen at each site in the MSA. In theory, we would like to compute phylogenetic likelihoods with branch lengths scaled by $r^*_{(j)}$ and mix over $r^*_{(j)} \sim \text{Gamma}(\alpha, \alpha)$ for $j \in \{1, ..., n\}$. However, these integrals are generally intractable so Yang (1994) suggested dividing the Gamma(α, α) distribution into K equal-probability rate classes, with the mean rate in each class used to represent all rates in that class. In practice, we use the Categorical$(\mathbf{r}, \mathbf{p})$ distribution to define the models on $r^*_{(j)}$ for $j \in \{1, ..., n\}$, where $\mathbf{p}$ is a, possibly unnormalized, probability vector; in addition, this model is used to represent more general discrete distributions.

4.2.3 Phylo-HMM Description

Phylo-HMMs are special cases of directed graphical models (Lauritzen, 1996) and treat evolution as a combination of two Markov processes: one across the sites in the MSA and one down the phylogeny. They are commonly used for sequence-level segmentation problems such as gene prediction and detection of highly-conserved regions (Siepel and Haussler, 2005). In fact, a phylo-HMM is similar in structure to a standard HMM; the main difference between the two model classes is that a phylo-HMM uses a phylogenetic likelihood as its emission

probability distribution, while standard HMMs usually specify simpler emission distributions. Our BCR-specific phylo-HMM specifies a Markov process along $\mathbf{Y}_{\text{naive}}$ and, conditional on $\mathbf{Y}_{\text{naive}}$, a phylogenetic evolutionary process down the given tree. Phylo-HMMs have not been applied to BCR sequence analysis before and we believe this biologically realistic probabilistic model is uniquely suited to provide higher-quality naive sequence and ancestral sequence estimates compared to those obtained under current state-of-the-art methods.

To help us describe the phylo-HMM generative process, we provide an illustration of the associated graphical model diagram for an example alignment with $m = 3$ sequences and $n = 3$ sites (Figure 4.2). The naive sequence "hidden state" prior distribution $p(\mathbf{Y}_{\text{naive}})$ decomposes to $p\big(Y^{(1)}_{\text{naive}}\big) \prod_{j=2}^{n} p\big(Y^{(j)}_{\text{naive}} \,\big|\, Y^{(j-1)}_{\text{naive}}\big)$ and the bases are generated sequentially; these prior probabilities depend on hyperparameters that can be set using the `partis` software package (Ralph and Matsen IV, 2016a,b). For the tree topology τ, we assume that a tree is drawn from the Uniform distribution over $(m+1)$-tip unrooted trees; this seems like a strange choice given that we described τ as a rooted topology above, but this decision will be justified when we discuss how to perform Bayesian inference under the phylo-HMM. The branch lengths $\mathbf{t}$ and the gamma shape parameter α are assumed to be *a priori* independent and to follow Exponential(λ) distributions, where λ is some prespecified rate. The GTR exchangeability rates $\mathbf{e}$ and equilibrium base frequencies $\boldsymbol{\pi}$ are usually assumed to come from six-dimensional and four-dimensional Dirichlet distributions, respectively.

For each MSA site $j \in \{1, ..., n\}$, $r^*_{(j)}$ *a priori* follows the Categorical$\big(\mathbf{r}, \big(\frac{1}{K}, ..., \frac{1}{K}\big)\big)$ distribution. Then, at each site $j \in \{1, ..., n\}$, we assume that $\mathbf{Y}^{(j)}_{\text{int}}$ and $\mathbf{D}^{(j)}$ are generated by drawing DNA states from CTMC transition probability matrices based on augmented branch lengths $\mathbf{t} \times r^*_{(j)}$. For example, in Figure 4.2, we first sample $Y^{(j)}_1$ from $p\big(Y^{(j)}_1 \,\big|\, Y^{(j)}_{\text{naive}}\big)$, which is a row vector distribution in the CTMC transition probability matrix for the "branch length" $t_1 \times r^*_{(j)}$, where $j \in \{1, ..., n\}$. Once we have sampled $Y^{(j)}_1$ for $j = 1, ..., n$, we can draw $Y^{(j)}_2$ and $D^{(j)}_3$ using similar row vector distributions from CTMC transition probability matrices for the "branch lengths" $t_2 \times r^*_{(j)}$ and $t_3 \times r^*_{(j)}$, respectively. We can recursively continue this process until we generate states at $D^{(j)}_1$ and $D^{(j)}_2$ for $j \in \{1, ..., n\}$.

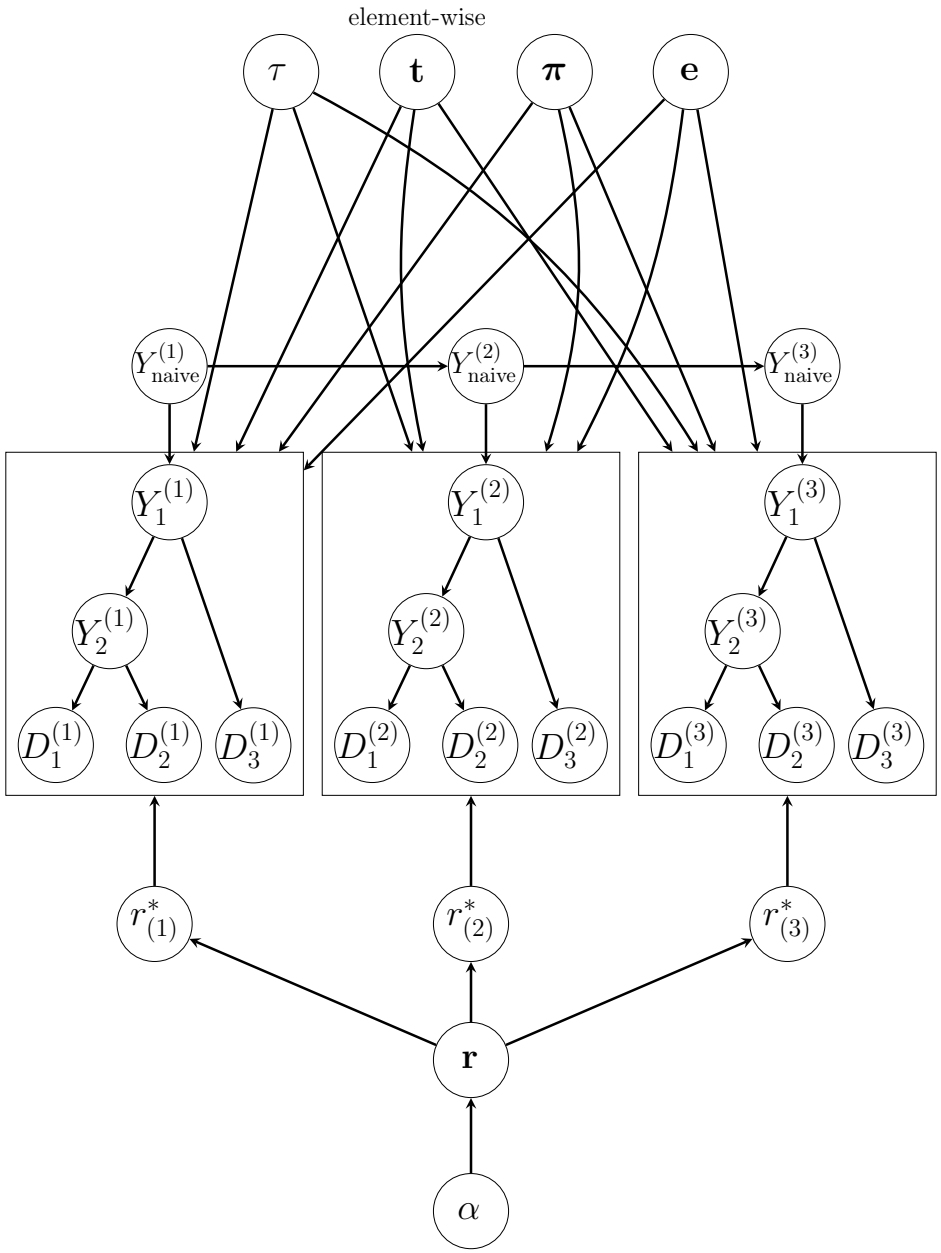


Figure 4.2: The phylo-HMM graphical model diagram for an example alignment with $m = 3$ sequences and $n = 3$ sites. The τ, $\mathbf{t}$, $\boldsymbol{\pi}$, and $\mathbf{e}$ nodes represent the 4-tip unrooted tree topology, the associated 5 branch lengths, the GTR exchangeability rates, and GTR equilibrium base frequencies, respectively. The parameter α denotes the gamma shape parameter associated with the K-class discrete gamma distribution, which is used to model phylogenetic rate variation among sites; $\mathbf{r}$ symbolizes the vector of K discrete rates that is deterministically induced by α. The set of nodes $\mathbf{r}^* = \{r^*_{(1)}, r^*_{(2)}, r^*_{(3)}\}$ defines the rates that are drawn from $\mathbf{r}$ at each particular site. The $\mathbf{Y}_{\text{naive}} = \{Y^{(1)}_{\text{naive}}, Y^{(2)}_{\text{naive}}, Y^{(3)}_{\text{naive}}\}$ "hidden state" node collection represents the Markov process that stochastically generates the naive sequence in our phylo-HMM. The node sets $\{Y^{(j)}_i\}_{i=1:2,j=1:3}$ and $\mathbf{D} = \{D^{(j)}_i\}_{i=1:3,j=1:3}$ denote the internal nodes of τ excluding the naive sequence $\mathbf{Y}_{\text{naive}}$ and the observed MSA, respectively. We draw plates around the $\mathbf{Y}^{(j)}_{\text{int}}$ and $\mathbf{D}^{(j)}$ node sets for $j \in \{1, 2, 3\}$ to indicate that any directed edges touching a plate apply to all nodes in the plate (except for edges that originate from $\mathbf{t}$, which apply element-wise to the nodes in the plate).

4.2.4 Posterior Distribution Inference

We are interested in sampling from the posterior distribution $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}} \,|\, \mathbf{D})$:

$$
\begin{aligned}
&p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}} \,|\, \mathbf{D}) \\
&\propto p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}}, \mathbf{D}) \\
&= p(\mathbf{r}^*, \mathbf{Y}_{\text{int}}, \mathbf{D} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}) p(\tau) p(\mathbf{t}) p(\boldsymbol{\pi}) p(\mathbf{e}) p(\alpha) p(\mathbf{r} \,|\, \alpha) p\left(\mathbf{Y}_{\text{naive}}\right) \\
&= \left\{ \prod_{j=1}^{n} p\Big(\mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, r^*_{(j)}, \mathbf{Y}^{(j)}_{\text{int}}\Big) p\Big(\mathbf{Y}^{(j)}_{\text{int}} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, r^*_{(j)}, Y^{(j)}_{\text{naive}}\Big) p\big(r^*_{(j)} \,\big|\, \mathbf{r}\big) \right\} \\
&\qquad \times p(\tau) p(\mathbf{t}) p(\boldsymbol{\pi}) p(\mathbf{e}) p(\alpha) p(\mathbf{r} \,|\, \alpha) p(\mathbf{Y}_{\text{naive}}),
\end{aligned}
$$

where this model decomposition results from the definition of a directed graphical model. We also factorize the posterior distribution in the following way:

$$
\begin{aligned}
&p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}} \,|\, \mathbf{D}) \\
&\quad = p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}) \\
&\qquad \times p(\mathbf{Y}_{\text{naive}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{D}) \\
&\qquad \times p(\mathbf{r}^*, \mathbf{Y}_{\text{int}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}, \mathbf{D}).
\end{aligned}
$$

This formulation is useful because it suggests that we can generate draws from the posterior distribution by sampling sequentially from three conditional probability distributions. Conceptually, to sample from the posterior, we have to draw in-order: 1) the phylogeny-related parameters, 2) the naive sequence, and 3) the ancestral sequences. We describe how to perform these three sampling steps in the following subsections and provide a complete summary of the sampling process in Algorithm 1.

Tree Sampling

Our strategy for sampling from $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$ is to first draw a large pool of observations from an easy-to-sample proposal distribution q and then perform weighted bootstrap

Algorithm 1: Posterior Sampling of $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}} \,|\, \mathbf{D})$

Input: CF multiple sequence alignment $\mathbf{D}$, number of discrete rates K
N_{pool}, N_{final} ($N_{\text{pool}}/N_{\text{final}} \approx 20$)

Output: N_{final} samples of
$(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}}) \sim p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}} \,|\, \mathbf{D})$

Tree Sampling — $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$

1. Run `partis` on input data $\mathbf{D}$.

$\Rightarrow \mathbf{D}^* = \{\mathbf{D}, \widehat{\mathbf{Y}}^{\text{partis}}_{\text{naive}}\}, \widehat{p}(\mathbf{Y}_{\text{naive}})$

2. Run `RevBayes` MCMC on the augmented MSA $\mathbf{D}^*$.

$\Rightarrow N_{\text{pool}}$ samples of $(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}) \sim q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)$

3. Run the SIR algorithm without replacement on the N_{pool} $(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})$ proposal samples with weights $w = \frac{p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})}{q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)}$.

$\Rightarrow N_{\text{final}}$ samples of $(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}) \sim p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$

Naive Sequence Sampling — $p(\mathbf{Y}_{\text{naive}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{D})$

For each sample $(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}) \sim p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$:

For each site $j \in \{n, ..., 1\}$:

1. Draw $Y^{(j)}_{\text{naive}}$ using our phylo-HMM-based backward sampling procedure.

$\Rightarrow N_{\text{final}}$ samples of $(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}) \sim p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}} \,|\, \mathbf{D})$

Intermediate Ancestral Sequence Sampling — $p(\mathbf{r}^*, \mathbf{Y}_{\text{int}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}, \mathbf{D})$

For each sample $(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}) \sim p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}} \,|\, \mathbf{D})$:

For each site $j \in \{1, ..., n\}$:

1. Sample $r^*_{(j)}$ according to probabilities proportional to $p\big(\mathbf{D}^{(j)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, Y^{(j)}_{\text{naive}}, r^*_{(j)}\big)$.

2. Sample $\mathbf{Y}^{(j)}_{\text{int}}$ in a pre-order fashion using the standard ASR distribution at internal nodes on a $r^*_{(j)}$-scaled tree.

$\Rightarrow N_{\text{final}}$ samples of $(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}}) \sim p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^*, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}} \,|\, \mathbf{D})$

with weights $\frac{p}{q}$ on those samples to obtain approximate draws from the correct distribution. This "sampling-importance-resampling" (SIR) algorithm is a sample filtering method that finds use in a wide variety of statistical applications (Smith and Gelfand, 1992; Gordon et al., 1993; Andrieu et al., 2010). The original SIR algorithm resampled observations with replacement, but there are theoretical and practical considerations that make resampling without replacement more attractive (Skare et al., 2003; Gelman et al., 2013). A thorough review of SIR sampling can be found in (Gelman and Meng, 2004, Chapter 24). We use the SIR algorithm to sample from $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$ because we want to take advantage of already-existing software programs for Bayesian phylogenetic inference while incorporating biologically-realistic details into our phylo-HMM.

Our phylogeny proposal distribution q comes from the `RevBayes` software package (Höhna et al., 2016) because this is a proposal that is close to the target distribution p while also being easy to sample from. In short, this q is traditional Bayesian phylogenetic analysis with a point estimate of the naive sequence. In more detail, we first input the MSA $\mathbf{D}$ into the `partis` package, obtain a naive sequence point estimate $\widehat{\mathbf{Y}}^{\text{partis}}_{\text{naive}}$, and create an augmented MSA $\mathbf{D}^* = \{\mathbf{D}, \widehat{\mathbf{Y}}^{\text{partis}}_{\text{naive}}\}$. This allows us to generate Markov chain Monte Carlo (MCMC) samples from $q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)$ and provides a convenient way to sample trees that have m tips and an informative root branch length emanating from the naive sequence. In our `RevBayes` analysis, we require that the prior components of $q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)$ be defined as we discussed in the previous subsection for $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$ and as we will see, this assumption is critical to the validity of our technique.

For the purposes of this discussion, let us assume that we sample a large number (say N_{pool}) of parameter values from $q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)$ in `RevBayes`. As we briefly mentioned above, we resample N_{final} times without replacement from the set of N_{pool} $q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)$ draws. Each $q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)$ sample is assigned a sampling weight $w = \frac{p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})}{q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)}$. While it seems odd to use the ratio of posterior probabilities that are conditional on different datasets as bootstrap weights, the only technical requirement on p and q is that the parameters of interest (i.e. τ, $\mathbf{t}$, $\boldsymbol{\pi}$, $\mathbf{e}$, α, and $\mathbf{r}$) have the same support, which is indeed the case in

our situation. Smith and Gelfand (1992) suggest picking N_{pool} and N_{final} so $\frac{N_{\text{pool}}}{N_{\text{final}}} \geq 10$ while Rubin (1987) proposed a safe rule-of-thumb to be $\frac{N_{\text{pool}}}{N_{\text{final}}} = 20$; we use $\frac{N_{\text{pool}}}{N_{\text{final}}} = 20$ in all the applied experiments conducted in this chapter. It may not be immediately clear how one would efficiently compute the numerator in w so we express w in the following form:

$$
\begin{aligned}
w &= \frac{p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})}{q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D}^*)} \\
&= \frac{p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{D})/p(\mathbf{D})}{q(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{D}^*)/q(\mathbf{D}^*)} \\
&= \frac{p(\mathbf{D} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})}{q(\mathbf{D}^* \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})} \frac{q(\mathbf{D}^*)}{p(\mathbf{D})} \\
&= \frac{\sum_{\mathbf{Y}_{\text{naive}}} p(\mathbf{D} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}) p(\mathbf{Y}_{\text{naive}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})}{q(\mathbf{D}^* \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})} \frac{q(\mathbf{D}^*)}{p(\mathbf{D})} \\
&= \frac{\sum_{\mathbf{Y}_{\text{naive}}} p(\mathbf{D} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \mathbf{r}, \mathbf{Y}_{\text{naive}}) p(\mathbf{Y}_{\text{naive}})}{q(\mathbf{D}^* \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})} \frac{q(\mathbf{D}^*)}{p(\mathbf{D})} \\
&= \frac{\sum_{\mathbf{Y}_{\text{naive}}} \left\{ p\left(Y^{(1)}_{\text{naive}}\right) \prod_{j=2}^{n} p\left(Y^{(j)}_{\text{naive}} \,\middle|\, Y^{(j-1)}_{\text{naive}}\right) \prod_{k=1}^{n} p\left(\mathbf{D}^{(k)} \,\middle|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \mathbf{r}, Y^{(k)}_{\text{naive}}\right) \right\}}{q(\mathbf{D}^* \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})} \frac{q(\mathbf{D}^*)}{p(\mathbf{D})},
\end{aligned}
$$

where the transition from the second line to the third line is due to the priors on τ, $\mathbf{t}$, $\boldsymbol{\pi}$, $\mathbf{e}$, α, and $\mathbf{r}$ for both p and q being equal and the decomposition between the fourth and sixth lines is a result of d-separation conditional independencies (Lauritzen, 1996) as follows. Specifically, $\mathbf{Y}_{\text{naive}} \perp \{\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}\}$ because every undirected path in the graphical model between $\mathbf{Y}_{\text{naive}}$ and $\{\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}\}$ does not contain any "conditioned" child nodes and $\mathbf{D} \perp \alpha \,|\, \{\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \mathbf{r}, \mathbf{Y}_{\text{naive}}\}$ as all paths between $\mathbf{D}$ and α are "blocked" by the intermediate node $\mathbf{r}$. The final denominator term above $q(\mathbf{D}^* \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r})$ is the usual phylogenetic likelihood, here calculated by `RevBayes`. Note that the marginal likelihood ratio $\frac{q(\mathbf{D}^*)}{p(\mathbf{D})}$ does not affect the bootstrap sampling because the sampling probabilities are proportional to w and the likelihood ratio is a constant with respect to τ, $\mathbf{t}$, $\boldsymbol{\pi}$, $\mathbf{e}$, α, and $\mathbf{r}$ in the N_{pool} sampling weights. The numerator term in the final equation for w looks complicated, but is actually the phylo-HMM likelihood with the "hidden state" probabilities $p\left(Y^{(1)}_{\text{naive}}\right)$ and $p\left(Y^{(j)}_{\text{naive}} \,\middle|\, Y^{(j-1)}_{\text{naive}}\right)$ for $j \in \{2, ..., n\}$ and the "emission probabilities" $p\left(\mathbf{D}^{(k)} \,\middle|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \mathbf{r}, Y^{(k)}_{\text{naive}}\right)$

that marginalize over the site-wise rates $r^*_{(k)}$ for $k \in \{1, ..., n\}$. We can efficiently calculate the phylo-HMM likelihood using the forward algorithm (Rabiner, 1986), but this approach requires us to be able to compute the phylo-HMM emission probabilities in a straightforward manner as the hidden state probabilities in $p(\mathbf{Y}_{\text{naive}})$ can be easily inferred using `partis`, which we now call $\widehat{p}(\mathbf{Y}_{\text{naive}})$.

It turns out that we can, again, leverage existing software tools to help us efficiently compute the emission probabilities $p\big(\mathbf{D}^{(k)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \mathbf{r}, Y^{(k)}_{\text{naive}}\big)$ for $k \in \{1, ..., n\}$. The key point is to recognize that $p\big(\mathbf{D}^{(k)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \mathbf{r}, Y^{(k)}_{\text{naive}}\big)$ is an entry in the Felsenstein likelihood vector at the $Y^{(k)}_{\text{naive}}$ node, which denotes the probability of the observed data at only the tips that descend from node $Y^{(k)}_{\text{naive}}$, given the conditioned state of node $Y^{(k)}_{\text{naive}}$. These vectors are commonly used within a post-order tree traversal algorithm to compute standard phylogenetic likelihoods (Felsenstein, 1981). Let $\mathbf{F}_u = (F_{u1}, ..., F_{um})^T$ be the vector of partial likelihoods at node u, where F_{ui} denotes the probability of the observed data at only the tips that descend from node u, given that the state of node u is i. Because we utilize an unrooted tree model in our phylo-HMM, we can use a Pulley Principle argument (Felsenstein, 1981) to show that a standard phylogenetic likelihood on our tree can be represented as $p\big(\mathbf{D}^{(k)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \mathbf{r}, Y^{(k)}_{\text{naive}}\big)\pi_{Y^{(k)}_{\text{naive}}}$. Thus, we can compute the phylo-HMM emission probabilities by first calculating the standard site-wise phylogenetic likelihoods on the same tree and then dividing out the $Y^{(k)}_{\text{naive}}$ stationary probabilities. To compute these standard site-specific phylogenetic likelihoods, we make use of the `libpll` C library (Flouri et al., 2014), which is a versatile high-performance software library for phylogenetic analysis. Of course, we could have instead used a rooted tree model in our phylo-HMM, performed our own post-order tree traversal algorithm for likelihood computations, and extracted out the appropriate entries in the site-wise Felsenstein likelihood vectors, but we want our inference technique to scale to the large datasets that result from high-throughput BCR sequencing and no currently-existing software package meets this requirement. In the next subsection, we describe how to generate naive sequence draws given the approximate phylogenetic tree samples from $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$.

Naive Sequence Sampling

To sample naive sequences, we exploit the fact that our phylo-HMM is essentially a standard HMM with a naive-conditional phylogenetic likelihood as its emission distribution. We draw the "hidden state" naive sequence $\mathbf{Y}_{\text{naive}}$ from $p(\mathbf{Y}_{\text{naive}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{D})$ by adapting the hidden state posterior sampling technique for standard HMMs to be used for our specialized phylo-HMM (Scott, 2002). Just as we perform the forward algorithm by recursively computing and caching intermediate phylo-HMM likelihoods (i.e. forward probabilities) going left to right across the MSA, we can sample $\mathbf{Y}_{\text{naive}}$ by doing a backward pass through the phylo-HMM and drawing the $Y^{(j)}_{\text{naive}}$ states starting at site n and ending at the first alignment site. In fact, the maximum a posteriori probability (MAP) estimate of the hidden state sequence $\mathbf{Y}_{\text{naive}}$ is obtained using a similar procedure called the Viterbi algorithm (Rabiner, 1986; Scott, 2002). This backward sampling procedure can actually use the previously cached forward probabilities in the calculation of the sampling probabilities at each site, which is convenient because we already had to run the forward algorithm to sample the phylogeny-related parameters from $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r} \,|\, \mathbf{D})$. Once $\mathbf{Y}_{\text{naive}}$ has been sampled, we can proceed to draw the intermediate ancestral states $\mathbf{Y}_{\text{int}}$ from the conditional distribution $p(\mathbf{r}^*, \mathbf{Y}_{\text{int}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}, \mathbf{D})$.

Intermediate Ancestral Sequence Sampling

Just as we did for our naive sequence sampling, we sample the intermediate ancestral states $\mathbf{Y}_{\text{int}}$ by utilizing a modified version of the standard ancestral sequence reconstruction (ASR) technique used in phylogenetics (Nielsen, 2002). It is important to note that sampling from $p(\mathbf{r}^*, \mathbf{Y}_{\text{int}} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{Y}_{\text{naive}}, \mathbf{D})$ can be reduced to drawing $\big(r^*_{(j)}, \mathbf{Y}^{(j)}_{\text{int}}\big)$ pairs from $p\big(r^*_{(j)}, \mathbf{Y}^{(j)}_{\text{int}} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}, \mathbf{D}^{(j)}\big)$ for each MSA site $j \in \{1, ..., n\}$, which is justified by d-separation conditional independencies (Lauritzen, 1996). At each site $j \in \{1, ..., n\}$, we

first sample $r^*_{(j)}$ and then $\mathbf{Y}^{(j)}_{\text{int}}$ according to the previously-described distribution:

$$
\begin{aligned}
&p\Big(r^*_{(j)}, \mathbf{Y}^{(j)}_{\text{int}} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}, \mathbf{D}^{(j)}\Big) \\
&= p\Big(r^*_{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}, \mathbf{D}^{(j)}\Big)\, p\Big(\mathbf{Y}^{(j)}_{\text{int}} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}, r^*_{(j)}, \mathbf{D}^{(j)}\Big),
\end{aligned}
$$

where the above decomposition is based on the definition of conditional probability. We draw the site-specific rates before the site-wise intermediate ancestral states because conditioning on the rates allows for simpler and more efficient ancestral state sampling using an already-existing software package.

It turns out that we can draw $r^*_{(j)}$ values from $\mathbf{r}$ with probabilities proportional to $p\big(\mathbf{D}^{(j)} \,|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, Y^{(j)}_{\text{naive}}, r^*_{(j)}\big)$, which makes intuitive sense because it implies rates should be sampled according to the site-specific likelihoods with branch lengths scaled by the corresponding rates. To understand why the above statement holds true, we express $p\big(r^*_{(j)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}, \mathbf{D}^{(j)}\big)$ as follows:

$$
\begin{aligned}
&p\Big(r^*_{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}, \mathbf{D}^{(j)}\Big) \\
&\propto p\Big(r^*_{(j)}, \mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}\Big) \\
&= p\Big(r^*_{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}\Big)\, p\Big(\mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y^{(j)}_{\text{naive}}, r^*_{(j)}\Big) \\
&= p\Big(r^*_{(j)} \,\Big|\, \mathbf{r}\Big)\, p\Big(\mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, Y^{(j)}_{\text{naive}}, r^*_{(j)}\Big) \\
&\propto p\Big(\mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, Y^{(j)}_{\text{naive}}, r^*_{(j)}\Big),
\end{aligned}
$$

where the transition from the second line to the third line stems from the definition of conditional probability, the transition from the third to fourth line is a result of d-separation (Lauritzen, 1996), and the fourth-to-fifth line transition is due to the fact that $r^*_{(j)} \,|\, \mathbf{r} \sim \text{Categorical}\big(\mathbf{r}, \big(\frac{1}{K}, ..., \frac{1}{K}\big)\big)$ by assumption. These site-specific likelihoods are in fact almost identical to the naive-conditional phylogenetic likelihoods discussed in the previous two subsections with the only difference being that we now condition on the site-wise rates instead

of marginalizing over them.

To illustrate why sampling $\mathbf{Y}_{\text{int}}^{(j)}$ from $p\big(\mathbf{Y}_{\text{int}}^{(j)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}, \mathbf{D}^{(j)}\big)$ is similar to drawing ASRs according to the procedure outlined by Nielsen (2002), we derive the sampling distribution of $Y_1^{(j)}$, the most recent common ancestor of $\mathbf{D}^{(j)}$. The distribution $p\big(Y_1^{(j)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}, \mathbf{D}^{(j)}\big)$ can be decomposed in the following manner:

$$
\begin{aligned}
&p\Big(Y_1^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}, \mathbf{D}^{(j)}\Big) \\
&\propto p\Big(Y_1^{(j)}, \mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}\Big) \\
&= p\Big(Y_1^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}\Big)\, p\Big(\mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}, Y_1^{(j)}\Big) \\
&= p\Big(Y_1^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}\Big)\, p\Big(\mathbf{D}^{(j)} \,\Big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, r_{(j)}^{*}, Y_1^{(j)}\Big),
\end{aligned}
$$

where, as before, the transition between the second and third lines are due to the definition of conditional probability and the transition from the third line to the fourth line is a result of d-separation (Lauritzen, 1996). The first term in the resulting expression above, $p\big(Y_1^{(j)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}\big)$, is the CTMC transition probability between $Y_{\text{naive}}^{(j)}$ and $Y_1^{(j)}$ on a scaled branch length $t_1 \times r_{(j)}^{*}$ and the second term, $p\big(\mathbf{D}^{(j)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, r_{(j)}^{*}, Y_1^{(j)}\big)$, is an entry in the Felsenstein likelihood vector at the $Y_1^{(j)}$ node on the given tree with $r_{(j)}^{*}$-scaled branch lengths. This expression for the sampling distribution of $Y_1^{(j)}$ corresponds with the standard ASR distribution at internal nodes described in (Nielsen, 2002). Thus, by conceptually treating $Y_{\text{naive}}^{(j)}$ as a sampled root node and scaling the branch lengths by $r_{(j)}^{*}$, we can draw $Y_1^{(j)} \sim p\big(Y_1^{(j)} \,\big|\, \tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, Y_{\text{naive}}^{(j)}, r_{(j)}^{*}, \mathbf{D}^{(j)}\big)$ using equation (10) presented by Nielsen (2002). By recursively using d-separation arguments (Lauritzen, 1996), we can use similar logic to that described above to sample states at the other internal nodes $\mathbf{Y}_{\text{int}}^{(j)} \setminus Y_1^{(j)}$ in a pre-order fashion as well. Once we have drawn $\mathbf{Y}_{\text{int}}^{(j)}$ at each alignment site $j \in \{1, ..., n\}$, our three-stage sampling process of $p(\tau, \mathbf{t}, \boldsymbol{\pi}, \mathbf{e}, \alpha, \mathbf{r}, \mathbf{r}^{*}, \mathbf{Y}_{\text{naive}}, \mathbf{Y}_{\text{int}} \,|\, \mathbf{D})$ is completed.

4.2.5 Implementation

The entire Bayesian phylo-HMM sampling process is implemented in a pipeline called `linearham`, which is available at `https://github.com/matsengrp/linearham`. We developed our phylo-HMM data structure in C++ and, as mentioned before, used the tree structures from the `libpll` C library (Flouri et al., 2014) to ensure our phylogenetic likelihood computations were as fast as possible. We have provided a Docker container (`https://hub.docker.com/r/matsengrp/linearham`) so users can try out the software without installation, as well as specifying the installation dependencies and provide an example command in a Dockerfile.

Our `linearham` program also provides an interface to `partis` to obtain $\widehat{p}(\mathbf{Y}_{\text{naive}})$. It accepts a repertoire FASTA file as input and can infer the hidden state transition probabilities assuming either the repertoire needs to be partitioned into individual CFs before phylogenetic inference or the "repertoire" is a single CF already, which is useful for researchers that want to run `linearham` inference on a hand-curated CF. It is also possible to specify an external set of $p(\mathbf{Y}_{\text{naive}})$ parameters.

In addition, `linearham` summarizes its phylogenetic inference output in a user-friendly format. It provides an output FASTA file that contains each sampled amino acid naive sequence and its associated posterior probability, creates a FASTA-like file that maps each sampled amino acid naive sequence to its corresponding set of DNA naive sequences and posterior probabilities, and generates an amino acid naive sequence posterior probability logo using `WebLogo` (Crooks et al., 2004) to visualize the per-site uncertainties. Furthermore, we provide similarly-styled output files for particular naive-to-tip tree lineages of interest by tabulating the posterior probabilities of sampled naive sequences and intermediate ancestral sequences on the lineage. For naive-to-tip lineage analysis, we also create a posterior probability lineage graphic using `Graphviz` (Gansner and North, 2000) that summarizes the different inferred naive-to-tip sequence trajectories and their relative confidences (Gong et al., 2013).

4.3 Simulation Experiments

In our simulation experiments, we focus on validating the accuracy of the naive sequence and ancestral sequence estimates produced by `linearham`. For naive sequence validations, we compare and contrast the performance between `linearham`, `partis`, and the `ARPP` program (Kepler, 2013). The `partis` package provides maximum likelihood naive sequence estimates (Ralph and Matsen IV, 2016a), but its model assumes a star-tree configuration, which is unrealistic for many CFs going through long periods of SHM and affinity maturation (Liao et al., 2013; Doria-Rose et al., 2014, 2016). The `ARPP` program is also a likelihood-based framework that jointly infers a CF tree and the associated naive sequence using information about the VDJ rearrangement process, but does not quantify phylogenetic uncertainty; we use this program in our validations because it is one of the only other programs that estimates CF phylogenies and naive sequences at the same time. We did attempt to use the newer version of `ARPP` (called `Cloanalyst`), but ran into difficulties with the program crashing and were unable to successfully resolve this issue when we contacted the program author.

Similarly, for our ancestral sequence validations, we restrict our comparisons to the `linearham`, `RevBayes`, and `dnaml` (Felsenstein, 2005) packages. As discussed before, the `RevBayes` program performs Bayesian phylogenetic inference on a given MSA, but in this case, we sample CF trees and ASRs from `RevBayes` using an augmented CF sequence alignment that contains the `partis`-inferred naive sequence (i.e. $\mathbf{D}^*$ from above). This approach to ASR was used in (Simonich et al., 2018) and is similar to that of `linearham` with the main difference being that `RevBayes` ASR sampling is conditional on the `partis`-inferred naive sequence whereas `linearham` ASR inference conditions on naive sequences drawn from a posterior distribution. For all the experiments conducted in this section, we run `RevBayes` using 50,000 MCMC iterations, sampling every 10 iterations, discard the first 500 `RevBayes` samples as burn-in, and sample without replacement 225 times from the 4,500 effective `RevBayes` samples in the case of `linearham` inference. The `dnaml` package performs maximum likelihood phylogenetic inference and generates an ASR conditional on this likelihood-based tree

estimate. While it is possible to sample ASRs on a maximum likelihood tree (Nielsen, 2002), `dnaml` only reports the most probable ancestral sequences. We run `dnaml` on an augmented CF sequence alignment that contains the `ARPP` maximum likelihood estimate of the naive sequence. In the following subsections, we describe our simulation experiments in more detail from the data-generating mechanism to the validation results.

4.3.1 Simulation Setup

To simulate tree topologies with a fixed number of tips in our experiments, we used the single-parameter beta-splitting generative process (Aldous, 1996). The beta-splitting process is able to generate a wide variety of tree topologies ranging from balanced topologies (i.e. trees with approximately equal root-to-tip distances) to unbalanced topologies (i.e. trees with highly variable root-to-tip distances) by varying the associated "balance" parameter β. Intuitively, this process can be seen as a recursive partitioning procedure that, at each tree split, partitions the Uniform$(0, 1)$ random numbers corresponding to the "tips" on the current sub-interval according to a Beta$(\beta + 1, \beta + 1)$ distribution. As $\beta \to \infty$, the generated trees get closer and closer to balanced binary trees and, as $\beta \to -2$, the simulated topologies look more and more "comb-like" (i.e. unbalanced) (Aldous, 1996).

We are motivated to use this topology-generating process because the level of balance of the tree determines the extent to which a phylogenetic approach to naive sequence estimation improves over a star-tree model. Informally speaking, a phylogenetic approach weights the information coming from tips close to the root (in the imbalanced case) more strongly than tips more distant from the root. Thus we expect a phylogenetic approach to be superior in the imbalanced case. On the other hand, `partis` assumes evolution occurs according to a star tree, which implies the expected number of substitutions from the root to each of the tip sequences should be approximately equal. Thus, for imbalanced trees, that have a large variance in the root-to-tip branch length distances, we would expect `partis` to provide poor naive sequence estimates for sequence datasets generated on those trees compared to a phylogenetic approach. Throughout the rest of this section, we define "tree imbalance" to

be the standard deviation of a tree's root-to-tip distances.

To generate branch lengths for our simulated trees that preserve the shapes induced by the beta-splitting topology prior, we independently draw values from a Uniform$(0, 2M)$ distribution, where M is a constant derived from HIV bnAb lineage trees. Specifically, we ran the PC64 (Landais et al., 2017) and VRC01 (Wu et al., 2015) datasets through `partis` to obtain augmented CF sequence alignments with the `partis`-inferred naive sequence, inferred approximate maximum likelihood CF trees using `FastTree` (Price et al., 2009, 2010), and set M to be the average across all estimated branch lengths in the two trees (≈ 0.0179). We describe these two datasets in more detail in the next section when we discuss our real-world dataset applications. In the subsequent parts of this section, we refer to the inferred `FastTree` trees described above as the PC64 and VRC01 phylogenies, respectively. We emphasize that these trees are only used as the basis for a simulation study and their level of accuracy is not especially important.

Each simulated phylogeny also receives a root branch length, which is not accounted for by the above generative processes. We use the mean of the PC64 and VRC01 root branch lengths (≈ 0.01759) as the default root branch length for simulation and also assign simulated trees root branch lengths of 0.1 to validate inference in settings with long periods of shared mutational history. For each simulated tree, we draw a naive sequence at the root according to the `partis` prior distribution that, as mentioned previously, models the VDJ rearrangement process. We use the default settings in `partis` for human heavy chain data, as that is our target regime of interest.

To simulate DNA sequence data on the selected trees given the `partis`-generated naive sequences, we use the simulator in the `samm` package (Feng et al., 2017). A complex collection of enzymes introduces mutations to affinity-maturing sequences in a random pattern that is known to be highly sensitive to the sequence motif (i.e. the subsequence of DNA bases surrounding the mutating position) (Rogozin and Kolchanov, 1992; Dunn-Walters et al., 1998; Chahwan et al., 2012; Methot and Di Noia, 2017). The `samm` program estimates these motif mutabilities (i.e. the probability a position will mutate given the motif at that position)

and substitution probabilities (i.e. the probability a position will mutate to a new base given the motif at that position) using a penalized Cox proportional hazards model and simulates sequence mutations given these inferred parameters. In our work, we use the default `samm` parameters inferred for human heavy chain sequences (Feng et al., 2017).

In our simulation experiments, we set β to -1.5, -1.25, and -1; the CF sequence count N_{CF} to 40 and 80; and the root branch length t_0 to 0.01759 and 0.1. There are 12 different parameter settings and in each setting, we simulate 15 trees for a grand total of 180 simulated trees. While 180 simulated trees does not seem like a large amount of Monte Carlo replicates, we are forced to constrain the number of trees we perform inference on because the `ARPP` program has to be run by hand via a GUI. Using these parameter settings allows for the simulation of both balanced and unbalanced trees.

4.3.2 Naive Sequence Validations

For each of the 180 simulated trees, we validate the accuracy of the naive sequence estimates generated from the `linearham`, `partis`, and `ARPP` programs by computing the hamming distances (i.e. the number of mismatched characters between two equal-length strings) between the estimates and the true naive sequence. We did so for the inferred DNA sequences directly and also for those same sequences translated to amino acids after inference was complete. The `partis` and `ARPP` packages provide naive sequence point estimates, but `linearham` samples naive sequences from a posterior distribution so we take the naive sequence with the highest posterior probability as the `linearham` "point estimate".

We summarize the hamming distance results for all 180 simulated trees described above and plot this performance metric against the corresponding values of tree imbalance (Figure 4.3); for reference, we plot the tree imbalance values for the PC64 and VRC01 trees as well. The performance of `partis` clearly worsens as trees become more unbalanced, which makes sense given that `partis` assumes a star-tree configuration for clonal family evolution. The `linearham` and `ARPP` programs provide accurate naive sequence estimates and perform similarly across the observed tree imbalance spectrum, which is not too surprising because

they both incorporate phylogenetic information into their estimates. We also show the mean

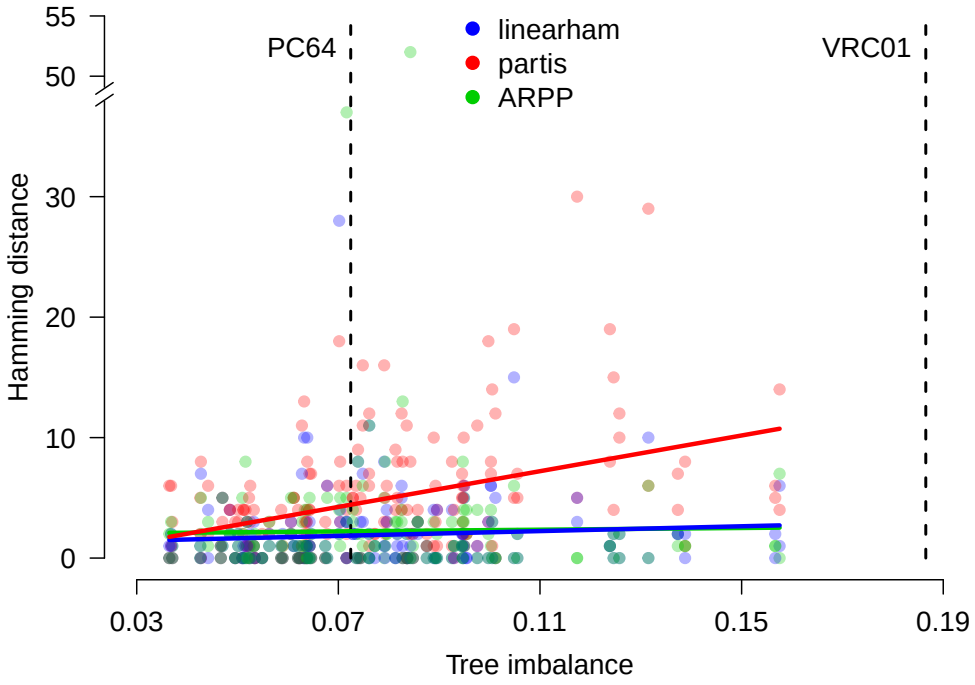


Figure 4.3: The hamming distances between the simulated naive DNA sequences and their corresponding `linearham`, `partis`, and `ARPP` estimates versus the tree imbalance values of the simulated trees. Linear regression lines are superimposed for each method to indicate how the results vary as trees get more unbalanced. For reference, we plot the tree imbalance values for the PC64 and VRC01 trees.

hamming distance results aggregated over all 180 trees for the same three programs, the full-sequence and CDR3 regions, and the DNA/amino-acid sequence types (Table 4.1). Table 4.1 suggests that `linearham` and `ARPP` both perform better than `partis` does by approximately 2-3 nucleotides (1.6-1.8 amino acids) in the whole sequence and in the CDR3 region. Furthermore, it seems `linearham` and `ARPP` perform similarly across all the different settings in Table 4.1.

We now present the mean hamming distance results, averaging over all trees generated under the different simulation parameter settings (Tables 4.3 to 4.5). Specifically, we average over the trees that were simulated using beta-splitting "balance" parameter

Sequence Region	Program	Sequence Type	
		DNA	Amino-Acid
Full-sequence	`linearham`	1.92 (3.14)	1.17 (1.76)
	`partis`	4.81 (4.91)	3.02 (2.84)
	`ARPP`	2.24 (5.06)	1.39 (3.32)
CDR3-only	`linearham`	1.66 (2.89)	1.08 (1.69)
	`partis`	4.17 (3.70)	2.69 (2.29)
	`ARPP`	1.27 (1.78)	0.856 (1.18)

Table 4.1: Mean hamming distances between the simulated naive sequences and their corresponding estimates, where the hamming distances are averaged over all 180 simulated trees. Results are provided for the `linearham`, `partis`, and `ARPP` programs; the full-sequence and CDR3 regions; and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

values $\beta = -1, -1.25, -1.5$, CF sequence counts $N_{\text{CF}} = 40, 80$, and root branch lengths $t_0 = 0.01759, 0.1$. The mean hamming distance values seem to increase slightly for `linearham` and `ARPP` and considerably for `partis` as we go from $\beta = -1, -1.25$ to $\beta = -1.5$, which confirms the results seen in Figure 4.3. The performance of `linearham` and `partis` deteriorates as N_{CF} goes from 40 to 80, while the opposite result is true for `ARPP`. Despite this, `linearham` is still better than `ARPP` at predicting the whole naive sequence. As the root branch length t_0 increases from 0.01759 to 0.1, the mean hamming distances increase substantially for all three programs, which is intuitive because we are essentially introducing a higher amount of mutations that is shared across all clonal sequences. While the `linearham` naive sequence validations did not take advantage of the full naive sequence posterior distribution, we demonstrate the usefulness of accounting for phylogenetic uncertainty in our ancestral sequence validations described below.

4.3.3 Intermediate Ancestral Sequence Validations

Our ancestral sequence validation experiments are centered around accurate inference of particular root-to-tip ancestral sequence lineages of interest because immunologists frequently use ASR to estimate the mutational pathways associated with antibody development (Doria-Rose et al., 2014; Simonich et al., 2018). For each of our simulated trees, we determine the root-to-tip ancestral lineage of interest by identifying the tip sequence that is farthest from the naive sequence in terms of branch length distance.

We quantify the results of our ancestral sequence validation by treating it as a machine learning classification problem: do the posterior probabilities aid us in deciding if the ancestral lineage sequences are correct? In all our experiments, we measure classification performance by recording the positive predictive value (i.e. the fraction of sequences in the ancestral lineage prediction set that are on the true ancestral lineage) and the true positive rate (i.e. the fraction of sequences on the true ancestral lineage that are in the ancestral lineage prediction set). The "predicted classification" of these sequences is obtained by applying a decision boundary $\rho \in \{0.25, 0.5, 0.75\}$ to the posterior probability. Thus, for example, if $\rho = 0.75$ and a given ancestral sequence is on the true ancestral lineage and has posterior probability 0.8, it is considered to be a "true positive" prediction. This analysis is straightforward for the `linearham` and `RevBayes` programs, that do estimate posterior probabilities. We define `dnaml` "posterior probabilities" to be either 0 or 1 depending on whether a lineage sequence is out of or in the `dnaml`-inferred set of most probable reconstructed lineage sequences.

We report the positive predictive values and the true positive rates for all 180 simulated trees and plot these performance metrics against the corresponding values of tree imbalance (Figure 4.4); for reference, we plot the tree imbalance values for the PC64 and VRC01 trees as well. Notice that the superimposed linear regression lines have negative slopes close to 0, which suggests that ancestral lineage inference is not clearly sensitive to the "balance" of the tree. We also display the mean positive predictive values and mean true positive rates

aggregated over all 180 trees for the same three programs, the different decision boundaries $\rho \in \{0.25, 0.5, 0.75\}$, and the DNA/amino-acid sequence types (Table 4.2). Table 4.2 indicates that `linearham` performs better than `RevBayes` in every setting, indicating that accounting for naive rearrangement uncertainty in our posterior distribution rather than conditioning on the `partis`-inferred naive sequence leads to more accurate ancestral lineage sequence estimates. At the lowest decision boundary $\rho = 0.25$, `linearham` obtains slightly better positive predictive values and true positive rates than either `RevBayes` or `dnaml` does (Table 4.2). As the decision boundary ρ increases, `linearham` and `RevBayes` achieve higher positive predictive values at the expense of lower true positive rates, which makes sense because high values of ρ imply that only lineage sequences with high posterior probabilities are predicted to be on the true lineage. In addition, note that the increases in positive predictive values are greater than the decreases in true positive rates for `linearham` as ρ increases. Of course, `dnaml` obtains the same positive predictive values and true positive rates regardless of ρ because its "posterior probabilities" are either 0 or 1.

These results help demonstrate why Bayesian ancestral lineage inference should be favored over likelihood-based approaches to intermediate lineage inference as the Bayesian posterior probabilities quantify the uncertainty in our sequence estimates. In a real-life experimental setting, the decision boundary ρ should be chosen based on the desired level of positive predictive values or true positive rates and our analysis provides some insight into the mapping between ρ and these classification metrics. In practice, immunologists are probably more interested in controlling positive predictive values than true positive rates because synthesizing computationally-inferred lineage sequences and testing their binding and neutralization abilities is a laborious and expensive endeavor. Thus, knowing the approximate fraction of inferred intermediate lineage sequences that are on the true lineage is of the utmost importance to an immunologist.

We also present the mean positive predictive values and mean true positive rates for decision boundary $\rho = 0.5$, averaging over all trees generated under the different simulation parameter settings (Tables 4.6 to 4.8). The validation performance of all the programs seems

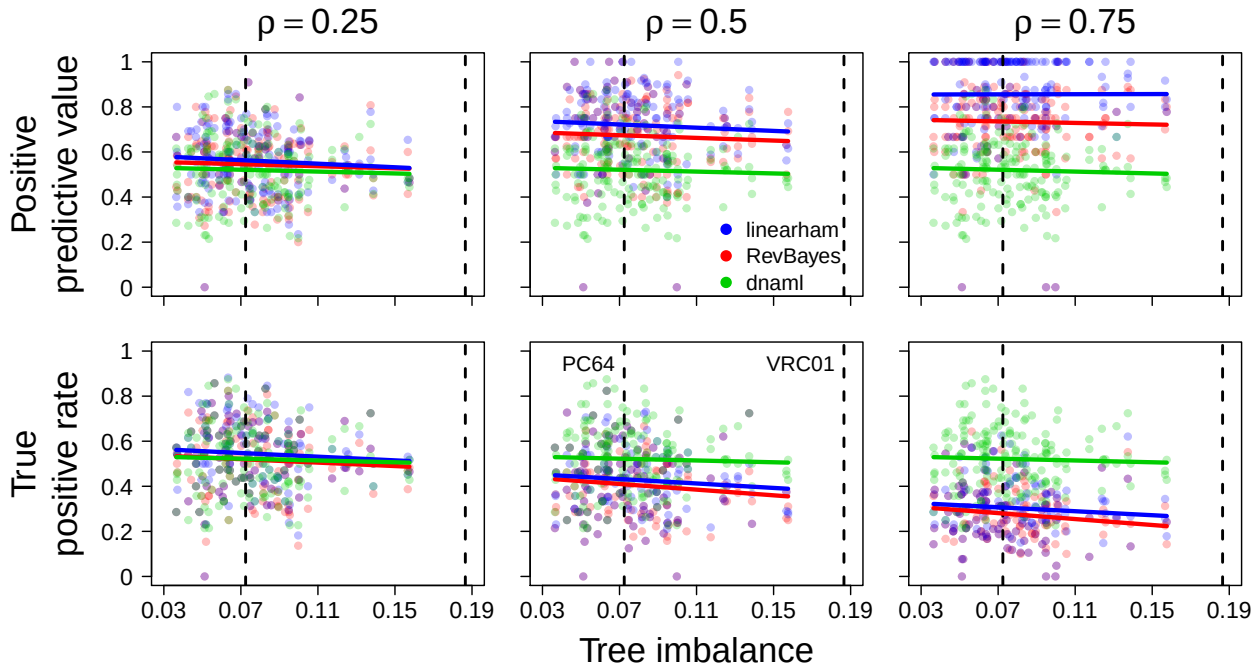


Figure 4.4: The positive predictive values and the true positive rates versus the tree imbalance values of the simulated trees, stratified by decision boundary ρ. Positive predictive values and true positive rates are computed on the DNA sequences and for the `linearham`, `RevBayes`, and `dnaml` programs. Linear regression lines are superimposed for each package to indicate how the results vary as trees get more unbalanced. For reference, we plot the tree imbalance values for the PC64 and VRC01 trees (vertical dashed lines).

to decline, albeit slightly, from $\beta = -1$ to $\beta = -1.5$, which makes sense given the trends in Figure 4.4. For the most part, mean positive predictive values and mean true positive rates increase for `linearham`, `RevBayes`, and `dnaml` as N_{CF} goes from 40 to 80, which seems surprising but for larger CFs, the root-to-tip lineage becomes larger and may explain this observed pattern. As the root branch length t_0 increases from 0.01759 to 0.1, the ancestral lineage validation results worsen considerably for all three programs, which is logical because, as we stated above, we are essentially introducing a higher amount of mutations shared across all clonal sequences when we utilize longer root branch lengths in our simulations.

Performance Metric	Program	Sequence Type					
		DNA			Amino-Acid		
		$\rho = 0.25$	$\rho = 0.5$	$\rho = 0.75$	$\rho = 0.25$	$\rho = 0.5$	$\rho = 0.75$
Positive predictive value	`linearham`	0.561 (0.139)	0.719 (0.157)	0.856 (0.176)	0.613 (0.131)	0.758 (0.145)	0.858 (0.146)
	`RevBayes`	0.544 (0.141)	0.672 (0.160)	0.734 (0.166)	0.590 (0.134)	0.713 (0.145)	0.774 (0.139)
	`dnaml`	0.520 (0.139)	0.520 (0.139)	0.520 (0.139)	0.590 (0.165)	0.590 (0.165)	0.590 (0.165)
True positive rate	`linearham`	0.545 (0.146)	0.428 (0.147)	0.304 (0.125)	0.640 (0.139)	0.533 (0.144)	0.398 (0.138)
	`RevBayes`	0.517 (0.147)	0.406 (0.144)	0.276 (0.116)	0.606 (0.140)	0.505 (0.144)	0.370 (0.134)
	`dnaml`	0.521 (0.142)	0.521 (0.142)	0.521 (0.142)	0.584 (0.166)	0.584 (0.166)	0.584 (0.166)

Table 4.2: Mean positive predictive values and mean true positive rates, averaged over all 180 simulated trees. Results are provided for the `linearham`, `RevBayes`, and `dnaml` programs; the different decision boundaries $\rho \in \{0.25, 0.5, 0.75\}$; and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

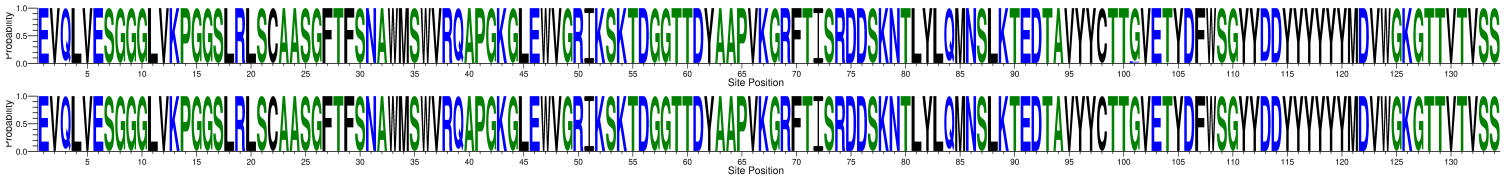
4.4 PC64/VRC01 Ancestral Lineage Analysis

We illustrate the capabilities of `linearham` and `ARPP`/`dnaml` on real-world datasets by applying the three methods to subsets of the PC64 and VRC01 datasets. The PC64 dataset contains a set of clonal sequences with multiple HIV-binding bNAbs that results from a longitudinal study over 46 months on an African donor (i.e. donor PC64) within the International AIDS Vaccine Initiative Protocol C cohort (Landais et al., 2016). Our VRC01 clonal family dataset also originates from an HIV-infected donor and contains many bNAb sequences that are part of a well-known class of HIV-binding antibodies (i.e. the VRC01 class) (Wu et al., 2011; West et al., 2012; Zhou et al., 2013). The tip sequences of interest for the PC64 and VRC01 datasets are chosen to be PCT64-35M and NIH45-46, respectively, which are both monoclonal antibody sequences that accumulated a large amount of SHM. We use 100 sequences from the PC64 CF dataset using a pruning strategy discussed in (Si-

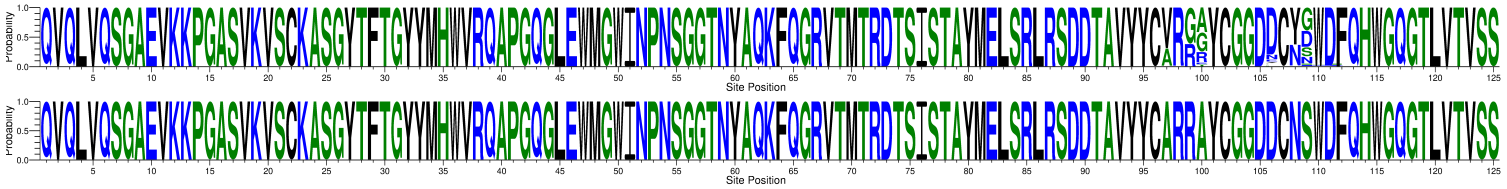
monich et al., 2018) and trim the VRC01 CF dataset to 268 sequences using the `cd-hit` sequence clustering program (Li and Godzik, 2006) with a 95% sequence identity cutoff. We perform inference on these subsetted datasets using the same settings as described for our simulation experiments.

The PC64 amino acid naive sequence posterior probability logos suggest that there is little uncertainty in the naive sequence reconstruction (Figure 4.5a) and, in fact, the most probable `linearham` amino acid naive sequence has a probability of approximately 0.92. However, the VRC01 naive sequence seems to have considerable uncertainty in the CDR3 region (Figure 4.5b), which shows that properly modeling ancestral sequence and phylogenetic uncertainty is important for real-world datasets with highly-mutated sequences. The most probable `linearham` VRC01 naive sequence estimate has a posterior probability approximately equal to 0.036. It is important to note that the VRC01 CF sequences were first collected 5 years after the diagnosis of the associated HIV-1 infection, whereas the PC64 CF sequences contain samples drawn from the corresponding donor as early as 4 months post infection. This indicates that the VRC01 naive sequence reconstruction inherently has more uncertainty because there are not any early time-point samples in the corresponding dataset. These results suggest that in the absence of uncertainty, `linearham` and `ARPP` produce similar naive sequence reconstructions. However, when there is a significant amount of naive sequence uncertainty, `linearham`, unlike `ARPP`, provides alternate hypotheses that should be considered along with corresponding uncertainty estimates.

Our `linearham` analysis demonstrates that there are many probable naive-to-tip sequence paths (Figure 4.6), suggesting that intermediate ancestral sequences have high levels of uncertainty; we use 0.04 probability cutoffs in each `linearham`-inferred lineage graphic. In particular, the `linearham` lineage diagram for the PC64 dataset (Figure 4.6a) shows many possible routes of evolution from the different naive sequences to the PCT64-35M mature sequence and displays confidence values via posterior probabilities. The VRC01 lineage graphic in Figure 4.6b does not display connections between any naive sequences and intermediate ancestral sequences, which reflects the fact that naive sequence inference is extremely



(a) PC64 naive sequence posterior probability logos



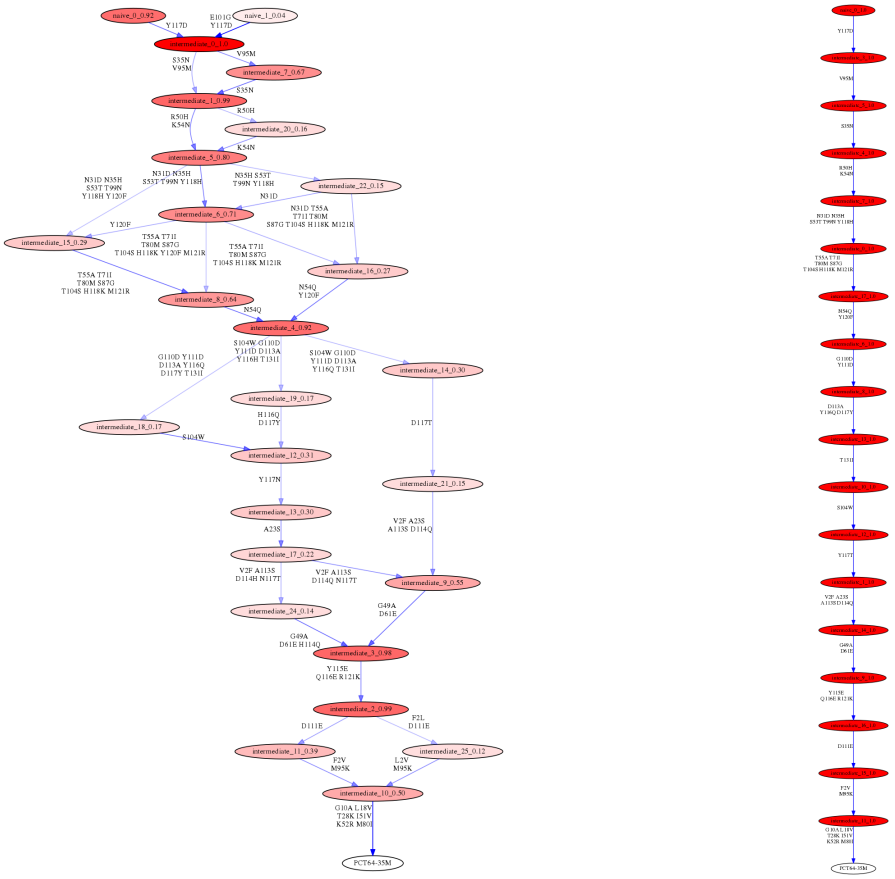
(b) VRC01 naive sequence posterior probability logos

Figure 4.5: The `linearham`-inferred (top) and `ARPP`-inferred (bottom) amino acid naive sequence posterior probability logos for the pruned PC64 dataset of 100 sequences and the trimmed VRC01 alignment of 268 sequences.

challenging on this dataset. In summary, there is considerable uncertainty in naive-to-tip mutational trajectory inference in real-world BCR datasets. This finding contradicts the assertions of Kepler (2013), who states that ancestral sequence and phylogenetic uncertainty is unimportant, and proceeds with a single point estimate.

4.5 Discussion

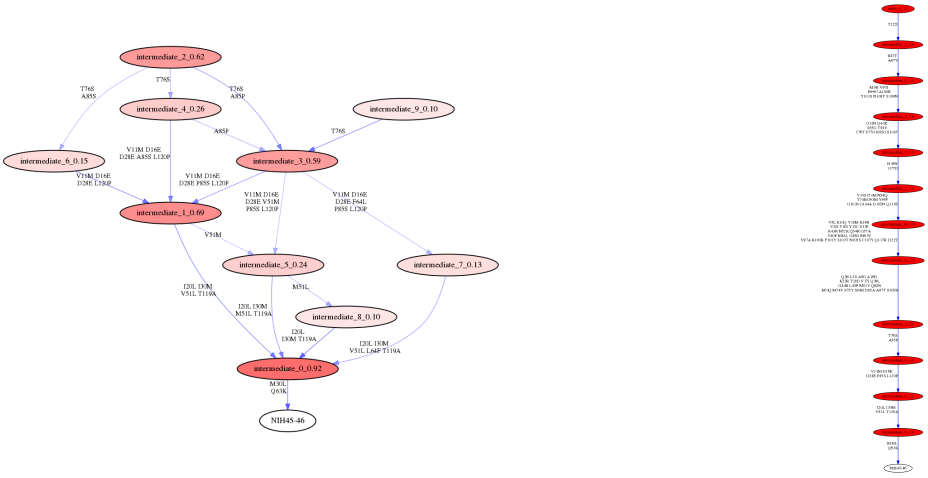
In this chapter, we introduce a novel Bayesian approach to CF phylogenetic inference that is based on a phylo-HMM. Our phylo-HMM posterior sampling methodology not only allows for easy quantification of phylogenetic uncertainty but also models the VDJ recombination process as an informative prior on the root sequence. Specifically, our phylo-HMM models both the naive rearrangement and SHM processes by using a hidden state discrete-time Markov model on naive sequences that explicitly incorporates VDJ rearrangement information and an emission distribution generating the clonal sequences conditional on the naive sequence that is based on phylogenetic likelihoods. We show that our inference procedure,



(a) PC64 posterior probability lineage inference

implemented in the software package `linearham`, provides higher-quality naive sequence and ancestral sequence estimates compared to those obtained under current state-of-the-art methods and augments these estimates with relative confidence values by reporting the associated posterior probabilities.

From our simulation experiments, we see that the `partis` naive sequence estimates get substantially worse as trees get more unbalanced unlike that of `linearham` and `ARPP`, which is intuitive because `partis` assumes a star-tree configuration whereas `linearham` and `ARPP` leverage phylogenetic models. The `linearham` and `ARPP` programs perform similarly in our naive sequence validations regarding their most highly supported inferences, but only `linearham` is capable of characterizing naive sequence uncertainty. This is important be-



(b) VRC01 posterior probability lineage inference

Figure 4.6: The `linearham`-inferred (left) and `dnaml`-inferred (right) naive-to-tip amino acid sequence trajectories for the pruned PC64 dataset of 100 sequences and the trimmed VRC01 alignment of 268 sequences. The tip sequences of interest for the PC64 and VRC01 datasets are chosen to be PCT64-35M and NIH45-46, respectively, and we use 0.04 probability cutoffs for these lineage graphics. The nodes correspond to unique ancestral sequences filled with red color, where the opacity is proportional to the posterior probability of the associated sequence. Each node has a label that denotes whether the associated sequence is a naive or intermediate ancestral sequence, the posterior probability rank of the sequence among all sampled naive or intermediate ancestral sequences, and the sequence-specific posterior probability itself. The directed edges connecting nodes represent ancestral sequence transitions, are shaded blue with an opacity proportional to the posterior probability of the associated sequence transition, and are annotated with the site-specific mutations between the two sequences.

cause, as we see in our VRC01 analysis, there is a significant amount of uncertainty in naive sequence estimates obtained from real datasets, which is not a conclusion shared by Kepler (2013). In addition, we demonstrate that `linearham` ancestral lineage inference performs better, via mean positive predictive values and mean true positive rates, than either `RevBayes` or `dnaml` does at the lowest decision boundary $\rho = 0.25$, which suggests that accounting for naive sequence and phylogenetic uncertainty does lead to slightly improved ASR performance. Furthermore, our Bayesian ASR results indicate that ρ can be chosen according to

a prespecified trade-off between positive predictive values and true positive rates, which is an important consideration for immunologists looking to synthesize computationally-inferred lineage sequences.

Based on all the evidence in this manuscript, we recommend using `linearham` to infer and visualize naive-to-bNAb mutational pathways in CFs of interest. Bayesian phylogenetic inference has already been shown to be useful for identifying different possible routes of evolution from a fixed naive sequence to a bNAb with relative confidence values (Simonich et al., 2018) and we believe our Bayesian phylo-HMM analysis pipeline in `linearham` can be used to not only infer similarly-styled maturation pathways but also visualize the uncertainties inherent in the naive sequence and ancestral sequence estimates. From a practical standpoint, these different possible ancestral lineages allow immunologists to generate many different intermediate antibody candidates that bind and/or neutralize HIV.

Our Bayesian phylo-HMM inference procedure admits a number of possible future extensions to enhance the effectiveness of our technique. For instance, we have developed our approach for heavy chain sequence data, but BCRs comprise both heavy chain and light chain sequences, both of which are essential to understanding bNAb development. Importantly, light chain naive sequences undergo VJ recombination, which is similar to VDJ rearrangement but without D germline genes, and the inference procedure in `linearham` could be modified to account for this different naive sequence generative process. Another drawback of our method is that it does not sample the parameters of $p(\mathbf{Y}_{\text{naive}})$ and uses `partis` (and its star-tree assumption) to estimate them. Ideally, our Bayesian inference procedure would jointly sample all the model parameters, but currently this is not practically feasible. In addition, our method implicitly assumes the CFs have been obtained from `partis`, which uses the star-tree assumption to cluster repertoire sequences. It may be possible to, as Ralph and Matsen IV (2016b) state, incorporate the phylo-HMM in the CF clustering procedure within `partis` to obtain higher-quality CFs, but in our current Bayesian implementation that would be computationally costly.

Supplementary Figures and Tables

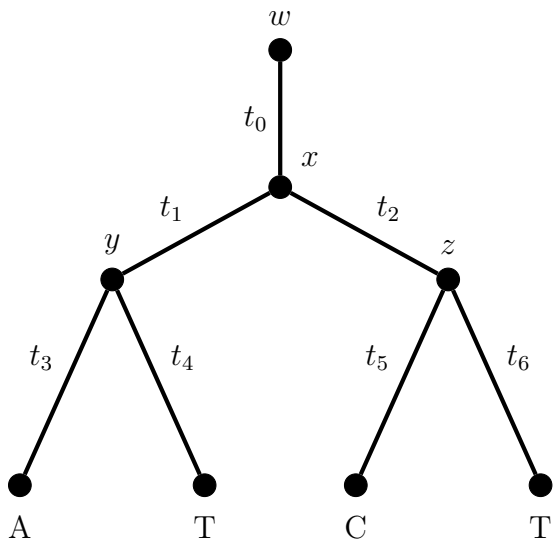


Figure 4.7: An example phylogenetic tree. Letters x, y, z, w represent the unobserved internal node states where w is associated with the root node, t_0 defines the root branch length, and $(t_0, t_1, t_2, ..., t_6)$ denotes the entire vector of branch lengths. Given this tree topology and set of branch lengths, we can calculate the likelihood of observing the nucleotide vector (A, T, C, T) by marginalizing probabilities over the unobserved states x, y, z, w.

Sequence Region	Program	Sequence Type					
		DNA			Amino-Acid		
		$\beta = -1$	$\beta = -1.25$	$\beta = -1.5$	$\beta = -1$	$\beta = -1.25$	$\beta = -1.5$
Full-sequence	`linearham`	1.73 (3.94)	1.92 (2.66)	2.10 (2.71)	0.917 (1.77)	1.18 (1.66)	1.40 (1.84)
	`partis`	3.85 (3.26)	3.82 (3.85)	6.77 (6.48)	2.48 (1.90)	2.42 (2.41)	4.15 (3.61)
	`ARPP`	2.42 (6.89)	2.52 (5.11)	1.78 (1.98)	1.38 (4.41)	1.52 (3.47)	1.27 (1.39)
CDR3-only	`linearham`	1.45 (3.42)	1.67 (2.55)	1.87 (2.66)	0.833 (1.63)	1.10 (1.63)	1.30 (1.81)
	`partis`	3.53 (2.84)	3.47 (3.45)	5.50 (4.34)	2.38 (1.81)	2.25 (2.20)	3.45 (2.64)
	`ARPP`	1.12 (1.76)	1.32 (1.95)	1.38 (1.63)	0.667 (1.07)	0.867 (1.20)	1.03 (1.26)

Table 4.3: Mean hamming distances between the simulated naive sequences and their corresponding estimates, where the hamming distances are averaged over all trees generated under the different beta-splitting "balance" parameter value settings. Results are provided for the `linearham`, `partis`, and `ARPP` programs; the full-sequence and CDR3 regions; and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

Sequence Region	Program	Sequence Type			
		DNA		Amino-Acid	
		$N_{\text{CF}} = 40$	$N_{\text{CF}} = 80$	$N_{\text{CF}} = 40$	$N_{\text{CF}} = 80$
Full-sequence	`linearham`	1.73 (2.36)	2.10 (3.77)	1.17 (1.67)	1.17 (1.86)
	`partis`	4.12 (3.58)	5.50 (5.90)	2.72 (2.40)	3.31 (3.20)
	`ARPP`	2.39 (4.36)	2.09 (5.70)	1.52 (2.92)	1.26 (3.69)
CDR3-only	`linearham`	1.51 (2.26)	1.81 (3.41)	1.04 (1.59)	1.11 (1.80)
	`partis`	3.84 (3.38)	4.49 (3.98)	2.57 (2.29)	2.82 (2.30)
	`ARPP`	1.33 (1.63)	1.21 (1.92)	0.944 (1.21)	0.767 (1.15)

Table 4.4: Mean hamming distances between the simulated naive sequences and their corresponding estimates, where the hamming distances are averaged over all trees generated under the different CF sequence count settings. Results are provided for the `linearham`, `partis`, and `ARPP` programs; the full-sequence and CDR3 regions; and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

Sequence Region	Program	Sequence Type			
		DNA		Amino-Acid	
		$t_0 = 0.01759$	$t_0 = 0.1$	$t_0 = 0.01759$	$t_0 = 0.1$
Full-sequence	`linearham`	0.744 (1.27)	3.09 (3.94)	0.456 (0.889)	1.88 (2.10)
	`partis`	2.69 (2.97)	6.93 (5.54)	1.77 (1.91)	4.27 (3.06)
	`ARPP`	0.80 (1.26)	3.68 (6.76)	0.467 (0.737)	2.31 (4.46)
CDR3-only	`linearham`	0.644 (1.17)	2.68 (3.65)	0.433 (0.875)	1.72 (2.04)
	`partis`	2.50 (2.47)	5.83 (3.97)	1.68 (1.62)	3.71 (2.42)
	`ARPP`	0.444 (0.751)	2.10 (2.10)	0.311 (0.574)	1.40 (1.37)

Table 4.5: Mean hamming distances between the simulated naive sequences and their corresponding estimates, where the hamming distances are averaged over all trees generated under the different root branch length settings. Results are provided for the `linearham`, `partis`, and `ARPP` programs; the full-sequence and CDR3 regions; and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

Performance Metric	Program	Sequence Type					
		DNA			Amino-Acid		
		$\beta = -1$	$\beta = -1.25$	$\beta = -1.5$	$\beta = -1$	$\beta = -1.25$	$\beta = -1.5$
Positive predictive value	`linearham`	0.742 (0.157)	0.715 (0.157)	0.700 (0.158)	0.782 (0.164)	0.759 (0.147)	0.733 (0.118)
	`RevBayes`	0.691 (0.162)	0.676 (0.152)	0.649 (0.165)	0.725 (0.166)	0.726 (0.140)	0.689 (0.124)
	`dnaml`	0.519 (0.141)	0.528 (0.149)	0.512 (0.129)	0.575 (0.179)	0.605 (0.173)	0.590 (0.141)
True positive rate	`linearham`	0.435 (0.149)	0.443 (0.139)	0.407 (0.152)	0.543 (0.159)	0.544 (0.139)	0.512 (0.131)
	`RevBayes`	0.413 (0.142)	0.425 (0.133)	0.378 (0.154)	0.510 (0.160)	0.522 (0.136)	0.482 (0.133)
	`dnaml`	0.521 (0.144)	0.531 (0.152)	0.512 (0.132)	0.566 (0.182)	0.600 (0.166)	0.586 (0.148)

Table 4.6: Mean positive predictive values and mean true positive rates for decision boundary $\rho = 0.5$, where we average over all trees generated under the different beta-splitting "balance" parameter value settings. Results are provided for the `linearham`, `RevBayes`, and `dnaml` programs and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

Performance Metric	Program	Sequence Type			
		DNA		Amino-Acid	
		$N_{\text{CF}} = 40$	$N_{\text{CF}} = 80$	$N_{\text{CF}} = 40$	$N_{\text{CF}} = 80$
Positive predictive value	`linearham`	0.714 (0.165)	0.725 (0.150)	0.759 (0.166)	0.756 (0.121)
	`RevBayes`	0.654 (0.171)	0.690 (0.147)	0.709 (0.168)	0.717 (0.117)
	`dnaml`	0.513 (0.146)	0.526 (0.132)	0.586 (0.177)	0.594 (0.153)
True positive rate	`linearham`	0.430 (0.153)	0.427 (0.141)	0.527 (0.159)	0.539 (0.127)
	`RevBayes`	0.403 (0.151)	0.408 (0.137)	0.501 (0.158)	0.509 (0.129)
	`dnaml`	0.514 (0.152)	0.528 (0.132)	0.573 (0.174)	0.595 (0.156)

Table 4.7: Mean positive predictive values and mean true positive rates for decision boundary $\rho = 0.5$, where we average over all trees generated under the different CF sequence count settings. Results are provided for the `linearham`, `RevBayes`, and `dnaml` programs and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

Performance Metric	Program	Sequence Type			
		DNA		Amino-Acid	
		$t_0 = 0.01759$	$t_0 = 0.1$	$t_0 = 0.01759$	$t_0 = 0.1$
Positive predictive value	`linearham`	0.728	0.711	0.768	0.748
		(0.141)	(0.173)	(0.140)	(0.150)
	`RevBayes`	0.682	0.662	0.727	0.700
		(0.146)	(0.173)	(0.135)	(0.153)
	`dnaml`	0.537	0.503	0.616	0.564
		(0.133)	(0.144)	(0.154)	(0.172)
True positive rate	`linearham`	0.450	0.407	0.560	0.506
		(0.136)	(0.154)	(0.140)	(0.143)
	`RevBayes`	0.415	0.396	0.523	0.487
		(0.131)	(0.155)	(0.135)	(0.151)
	`dnaml`	0.536	0.506	0.609	0.559
		(0.136)	(0.147)	(0.157)	(0.171)

Table 4.8: Mean positive predictive values and mean true positive rates for decision boundary $\rho = 0.5$, where we average over all trees generated under the different root branch length settings. Results are provided for the `linearham`, `RevBayes`, and `dnaml` programs and the DNA/amino-acid sequence types. Standard errors are also presented in parentheses.

Chapter 5

PREDICTING B CELL RECEPTOR SUBSTITUTION PROFILES USING PUBLIC REPERTOIRE DATA

5.1 Introduction

In the therapeutic antibody discovery and engineering field, researchers commonly isolate antibodies from animal or human immunizations and screen for functional properties such as binding to a target protein. Following the initial screening process, a small number of well-behaving antibodies (hits) are isolated for more rigorous examination of their biophysical properties in order to determine their potential as a therapeutic. After this stage, only a few final antibodies remain as lead candidates. However, even these carefully selected antibodies often have immunogenic peptides or other undesirable properties such as poor thermo/chemical stability and aggregation tendencies. To address these problems, the art of antibody engineering has emerged (Igawa et al., 2011), with numerous rational design strategies developed to mitigate aggregation. Researchers have removed hydrophobic surface patches to avoid aggregation (Clark et al., 2014; Casaz et al., 2014; Courtois et al., 2016; Geoghegan et al., 2016), "deimmunized" complementarity-determining regions by screening immunogenic peptides and mutating positions detrimental for peptide MHCII binding (Harding et al., 2010), and improved thermostability through stable framework grafting (McConnell et al., 2014) and targeted mutagenesis using predictions from proprietary structure/sequence analysis software (Seeliger et al., 2015). Although referred to as "rational", the choice of which amino acid to use for a site-directed mutation is often made using 1) the germline as a reference, 2) biochemical similarity between amino acids, or 3) the highest probability amino acid from a generic substitution matrix (e.g. BLOSUM) (Henikoff and Henikoff, 1992). However, neither of these three methods are explicitly designed to conserve antibody functionality (i.e. binding

to the same epitope with the same kinetics), so mutations are likely to have negative side effects on affinity. These considerations motivate a prediction problem: given a B cell receptor (BCR) sequence, which positions can be modified, and to which amino acids, without drastically changing the binding properties of the resulting BCR?

An immunization-derived antibody has already implicitly explored the mutational space through the population of B cells sharing the same naive ancestor, referred to as its clonal family (CF). The members of a CF arise during affinity maturation in a germinal center and carry fitness information about the effect of amino acid substitutions. A profile of the observed substitutions aggregated over all the B cells in a CF reveals which sites are more conserved, which sites can be more freely edited, and which amino acids can be used for replacements. However, we generally do not sequence all the B cells that are released from a germinal center so the information to make such a substitution profile is lost. Thus, we can formulate a more specific version of our prediction problem: given bulk BCR data and a single input sequence, can we infer the most likely per-site substitutions that are allowed in its true germinal center clonal family?

We begin by reviewing the natural mutation and selection process of germinal center affinity maturation. The Darwinian selection undertaken inside a germinal center is driven by B cells' ability to bind the antigen through the membrane-embedded BCR. The highly-mutated population of B cells in a germinal center is under stringent selection, driving the cell population towards higher and higher affinity until the germinal center is dissolved. Each germinal center is seeded by around one hundred naive B cells, but eventually internal competition makes one or a few of these lineages take over the whole germinal center (Tas et al., 2016). Although B cells in the germinal center reaction experience an extraordinarily high mutation rate (10^6 fold higher than the regular somatic mutation rate (Victora and Nussenzweig, 2012)), they rarely harbor more than 15% mutations at the DNA level (Briggs et al., 2017). However, since they must maintain some degree of antigen specificity to survive during the course of the germinal center reaction, lineages evolve in small incremental steps (Kepler et al., 2014; Kuraoka et al., 2016) and therefore, even lineages that drift far away

from their naive B cell ancestor most likely maintain the same epitope specificity throughout the germinal center reaction (Schmidt et al., 2013).

We can describe the combination of germinal center mutation and selection dynamics by computing per-site amino acid frequency vectors from observed BCR sequence data. We follow previous authors in calling site-specific amino acid probability vectors "substitution profiles", where each vector in a profile stores the probabilities of observing the 20 different amino acids at a given site (Sheng et al., 2017). We use the concept of a clonal family, defined by a shared heavy chain inferred naive DNA sequence, to segment BCR sequences into evolutionarily-related groups (Ralph and Matsen IV, 2016b); some practitioners refer to these groups as lineages. CF inference is highly informed by nucleotide sequences and therefore performed using DNA sequences. This makes DNA-level information necessary even though germinal center selection operates at the protein level and synonymous codons do not possess any fitness advantages (modulo transcription rate differences and codon bias, which we follow many others in ignoring here). The per-site amino acid frequency vectors described above form the substitution profile estimates; the substitution profile estimates converge to the true substitution profiles as the number of sequences sampled from the same CF tends to infinity.

Most CFs do not contain enough sequences in order to get a detailed substitution profile estimate. Indeed, most CFs in repertoire sequencing (Rep-Seq) samples have few members and a large fraction are singletons due to the exponential nature of the CF size distribution (Ralph and Matsen IV, 2016b). Additionally, many antibody screening methods are not geared towards whole repertoire sequencing. One may wish, then, to enhance the substitution profile estimates for data-sparse CFs with substitution profile information from similar CFs.

In this chapter, we present "Substitution Profiles Using Related Families" (SPURF), a penalized tensor regression framework that integrates multiple sources of information to predict the CF-specific amino acid frequency profile for a single input BCR sequence (Figure 5.1). Some of these information sources include substitution profiles for CFs in large, publicly available BCR sequence datasets and germline gene sequence information. We combine the

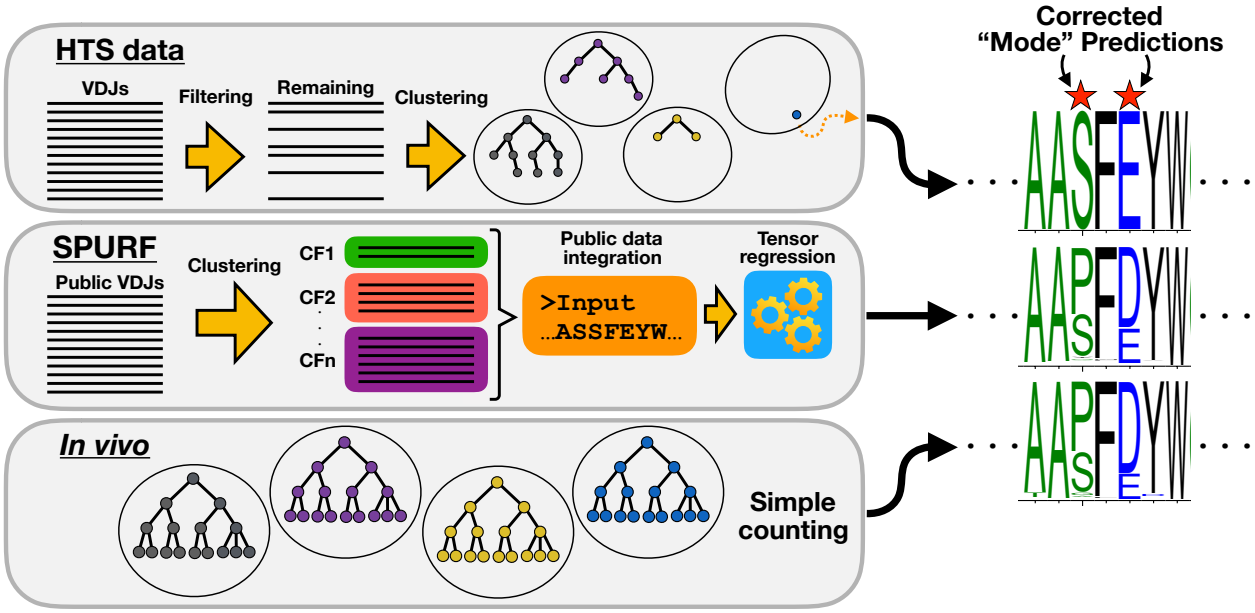


Figure 5.1: Amino acid substitution profiles viewed from three different perspectives: High-throughput sequencing data (HTS data) yields large amounts of VDJ sequences, but because of uneven sampling many CFs will be sampled just once, resulting in poor representations of the amino acid substitution profiles of those true CFs. "Substitution Profiles Using Related Families" (SPURF) is a statistical framework that integrates large scale Rep-Seq data to predict amino acid substitution profiles for singleton CFs. *In vivo* affinity maturation will test many different mutations and the resulting CFs reflect the amino acid substitution profiles that we attempt to predict.

local context-specific profile information with global profile information derived from other related germinal centers by regularizing the noisy local substitution profile estimate and pooling it closer towards more robust global profile estimates. Even though each germinal center focuses on binding to a unique epitope context, there are structural and possibly functional properties associated with BCR sequences that are common across germinal centers that we can leverage.

In addition, our inference machinery uses both standard and spatial lasso penalties as model regularizers and, as a result, furnishes sparse, interpretable parameter estimates. While our output type shares some similarities to that described by Sheng et al. (2017), the proposed objective, approach, and details differ (e.g. they predict substitution profiles

for gene families, we predict substitution profiles for CFs). We enable substitution profile prediction for single input BCR sequences based on profiles derived from a high-quality repertoire dataset that contains B cell samples from many human donors. To demonstrate the usefulness of our technique, we validate SPURF on two external datasets — one containing CFs extracted from a single human donor and the other focusing on a single CF of a HIV broadly neutralizing antibody. Lastly, we implement SPURF in an open-source software package (`https://github.com/krdav/SPURF`), which outputs a predicted CF-specific substitution profile and an associated logo plot based on a single input BCR sequence.

5.2 Methods

5.2.1 Overview

The aim of our model is to take a single sequence and predict the site-wise amino acid frequencies as would be found in the full CF from which this single sequence derived. We will refer to this as the sequence's CF-specific substitution profile. For this prediction problem, we have no direct information about this desired substitution profile other than the information contained in the input sequence itself, but we may use other information (e.g. from the inferred germline gene, simulated substitutions, or information derived from published BCR sequence datasets). For large CFs, a CF-specific substitution profile can be constructed simply by counting and making a per-site frequency matrix, with the rows of the matrix representing each of the 20 amino acids, and the columns being the sequence positions.

For training, we extract a collection of such large CFs and use them to build "ground truth" CF-specific substitution profiles as a training set for fitting the model. A randomly sampled single sequence is then taken out from each of these large CFs to predict the substitution profile, which is compared to the ground truth. We refer to these single sequences, sampled from large CFs, as subsamples.

To make the best possible prediction, we need a flexible model framework that can accommodate different sources of information seamlessly (Figure 5.2). For example, previous

work by Sheng et al. (2016) and Kirik et al. (2017) suggests that the various V genes have different characteristic paths of diversification. We can obtain a data-driven summary of that intuition by building profiles from large Rep-Seq data sets stratified by V gene. We may also think that the neutral substitution process is an important factor in determining substitution profiles (Sheng et al., 2016). We can quantify that sort of information by repeatedly simulating the neutral substitution process using a context-sensitive model (Cui et al., 2016). We call each external data set (e.g., V gene alignments and sequences simulated from the neutral substitution process) that can be used to predict the CF substitution profile of interest a source of profile information.

To make predictions using these types of information, we need a way of describing the various sites, and a way of integrating the information across the sites. We use the AHo numbering scheme (Honegger and PluÈckthun, 2001) to provide a single coordinate system to all sequences via its fixed-length numbering vector going from 1 to 149. Given this coordinate system, we use a site-wise weighted average of the input predictive profiles using a $\boldsymbol{\alpha}$ weight vector for each source of profile information.

To train this model, we fit the $\boldsymbol{\alpha}$ vectors by minimizing some objective function that quantifies the difference between the predicted profiles (where the prediction uses the subsampled sequence and the external profile information) and the "ground truth" substitution profiles from the large CFs. Any objective function could be used, but here we provide implementations of two such functions, a "fine-grained" L_2-error-based objective and a "coarse-grained" Jaccard-similarity-based objective (Jaccard, 1912).

We use two forms of regularization to avoid overfitting the many parameters of this model. This includes a standard lasso penalty to shrink weights to zero that do not contribute significantly to prediction performance (Tibshirani, 1996). We also use a fused lasso penalty (Tibshirani et al., 2005, 2014) to smooth differences between parameters at nearby sites in the sequence. These regularization terms have tuning parameters that regulate the strength of the penalties and are estimated using cross-validation.

Given this setup, a forward stepwise selection procedure is run with cross-validation to

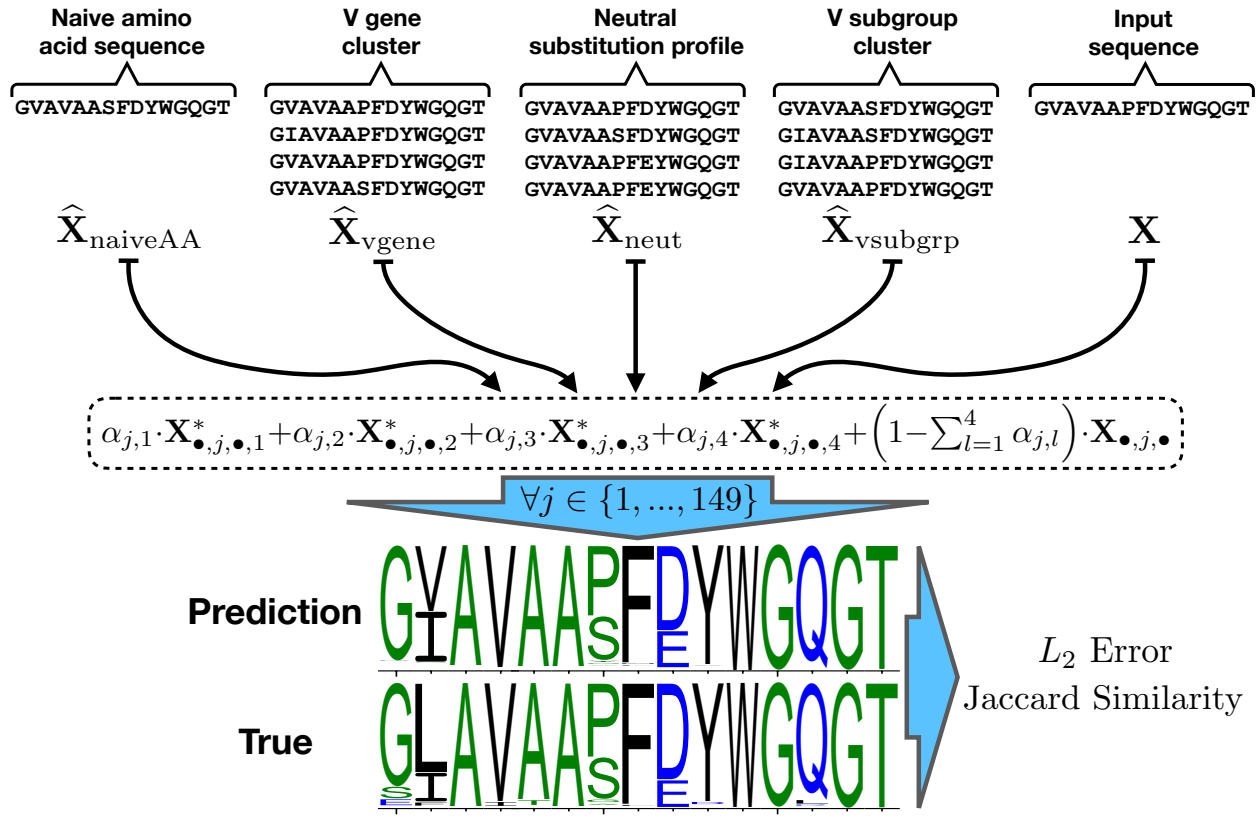


Figure 5.2: SPURF uses a per-site linear combination of substitution profiles from diverse sources to predict complete substitution profiles from a single member of a CF. At the top are the different profiles that serve as inputs to the model, some directly related to the naive sequence ($\widehat{\mathbf{X}}_{\text{naiveAA}}$ and $\widehat{\mathbf{X}}_{\text{neut}}$), and others partitions of the public Rep-Seq datasets ($\widehat{\mathbf{X}}_{\text{vgene}}$ and $\widehat{\mathbf{X}}_{\text{vsubgrp}}$). To predict a substitution profile, a weighted average is taken over the input sequence $\mathbf{X}$ and external profiles $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\right\}$ (see the dashed line bubble). The vertical blue arrow indicates that the weighted average (in the dashed line bubble) occurs at each of the 149 AHo positions. Once a predicted profile is generated, this is compared to ground truth using either L_2 error or Jaccard similarity as a performance metric. The $\boldsymbol{\alpha}$ vectors are estimated by optimizing the objective function, which also includes a statistical regularization term to prevent overfitting (not shown for simplicity).

pick the set of external profiles to use in the final model. As a last check, this model is tested on two external datasets to give a fair estimate of the prediction performance.

5.2.2 Data

We divide input data into two parts, with each part for a respective purpose: 1) model fitting and model testing and 2) providing "public" substitution profiles over clustered data to be used by our model. Throughout this work, we are careful to not use the same data for both purposes as this would bias our estimates; as a final validation, we test SPURF on two external datasets which are only used in this validation. Because we do not model sequence error, we only include high-quality data that we have high confidence in. We collect post-processed data files from 6 published works on Rep-Seq, which we refer to as repertoire data 1 to 6 (RD1-6):

1. RD1 from Gupta et al. (2017), which is an Illumina MiSeq re-sequencing of the samples in Laserson et al. (2014), where they sequence multiple time-points before and after influenza vaccination of 3 donors using the 454 pyrosequencing platform.

2. RD2 from Vander Heiden et al. (2017), from a study of the auto-immune disease Myasthenia Gravis (MG), in which 9 MG patients and 4 healthy donors participate.

3. RD3 from Stern et al. (2014), containing data from different tissues in a study of B cell response in 4 multiple sclerosis patients.

4. RD4 from Tsioris et al. (2015), from a study of neutralizing antibodies against the West Nile virus by sequencing naive and memory cells from 7 virus infected donors.

5. RD5 from Turchaninova et al. (2016), from a study of Rep-Seq error correction by sequencing naive, plasma, and memory cells from a single healthy donor.

6. RD6 from Meng et al. (2017), from the "B cell tissue atlas" acquired from the ImmuneDB web portal.

All datasets are acquired in their post-processed form with read processing performed as described in their respective publications.

The first five datasets (RD1-5) are prepared from unique molecular identifier (UMI) barcoded cDNA spanning the whole VDJ region and sequenced on the Illumina MiSeq platform using overlapping paired-end reads. Using the UMI, these reads are processed to address both PCR and sequencing errors giving high confidence reads (Shugay et al., 2014). Briefly, UMIs are used for error correction in conjunction with either of the pRESTO (Vander Heiden et al., 2014) or MIGEC (Shugay et al., 2014) processing pipelines and an appropriate Phred quality score cutoff. Paired-end reads are assembled using pRESTO and only the set of high confidence assembled reads constitute the final dataset used in this work. RD6 is the only dataset not prepared with UMIs; however, it is sequenced directly from genomic DNA (gDNA) instead of the more common practice of sequencing mRNA. Sequencing gDNA has the benefit of avoiding mutations introduced by the transcription machinery as well as mutations introduced in the RT-PCR step. On the other hand, DNA sequencing is not able to discriminate between expressed versus unexpressed BCRs (e.g. in the case of faulty VDJ recombination) and therefore we apply aggressive filtering of non-functional BCR sequences. We prefer quality over quantity and therefore avoid datasets from the 454 technology because of their higher indel frequencies compared to those from Illumina technologies (Loman et al., 2012).

Individual sequence files are merged based on donor identity so that the number of sample files matches the number of donors; this process yields 33 donor files. The donor files are then annotated and partitioned into CFs using the `partis` software (Ralph and Matsen IV, 2016a,b). Each donor file is run separately from the other files so CFs are defined by their unique `partis`-inferred naive sequence and donor identity. To ensure we obtain the highest quality and most biologically relevant sequences, `partis` is run in its most restrictive mode, discarding all reads with VDJ recombinations that are deemed as unproductive because of out-of-frame N/P junction nucleotides, missing invariant codons, or stop codons inside the VDJ region; furthermore, the most accurate `partis` partitioning mode ("full") is used to get the best CF estimates. Lastly, productive VDJ-recombined sequences are removed if they contain indels to assure concordance between the length of the naive sequence and the length

of the read sequences in its CF.

At this stage, some sequences contain ambiguous bases (e.g., because of primer masking); these are allowed to pass only if the ambiguous bases are inside the first or last 30 nucleotides of the VDJ region (equivalent to the length of the potentially masked PCR primers), otherwise they are discarded. This is a way of substituting the error-prone ends with neutral bases that minimize variance and maintain a conservative estimate of the substitutions; we also note that this has no apparent effect on the subsequently-described estimates (Figure 5.3 and Figure 5.4). For all sequences that pass this requirement, ambiguous bases are substituted with bases from the naive sequence in batches of 3 nucleotides (i.e. one codon) at a time until all ambiguous bases are resolved. Sequences are then translated into their respective amino acid sequences and de-duplication of repeated amino acid sequences is done within each CF. Because our statistical methodology operates on these amino acid sequences, we use the word "sequence" in subsequent sections to refer to these amino acid sequences. All CFs with fewer than 5 unique sequences are discarded. From these remaining CFs, their inferred naive sequences are used for antibody sequence numbering with the ANARCI software (Dunbar and Deane, 2015) under the AHo numbering scheme (Honegger and PluÈckthun, 2001). As a result of our restriction to non-indel sequences, all sequences within a given CF have equal length; thus, the AHo numbering from the naive sequence can be positionally transferred to all its CF-related read sequences. Finally, for each CF, the amino acid usage is extracted as a vector of counts at each AHo position. This overall dataset, which we call the "aggregated" dataset, contains 518,174 sequences distributed over 31,893 CFs and is built as a matrix of counts with rows denoting CFs and columns representing AHo positions and amino acid identities. All data used to build this aggregated dataset is public and freely available. We provide the data partitioned into CFs and numbered into AHo numbering for download on Zenodo (`https://doi.org/10.5281/zenodo.1289984`).

Model Fitting Dataset

To fit our CF-specific substitution profile prediction model, it is desirable to use the CFs from the aggregated dataset with the most sequence members so we can train using the observed substitution profiles with the least amount of noise; on the other hand, it is also desirable to extract CFs from as many donors as possible to avoid overfitting towards a few similar donors. To achieve both goals, we pick 500 CFs as a "model fitting" dataset as follows. We first exclude any CFs with less than 100 sequences from being eligible to be picked. We then cycle through donors, each time picking the largest remaining eligible CF. If a donor does not have any remaining eligible CFs, it is skipped. The process ends when 500 CFs are found; all unpicked CFs are used as the "public" dataset.

In addition, we perform subsampling for each CF in the model fitting dataset; this is the information from which we would like to predict the full profile. First, a single sequence is randomly chosen from each CF, then `partis` is re-run using each of these subsampled sequences to re-do the VDJ annotation and naive sequence inference. For some inferred naive DNA sequences, a stop codon is incidentally present in the N/P nucleotides of the junction region; these are considered spurious and replaced by the identically positioned codon from the input sequence. We stress that the CF-specific annotation and naive sequence are inferred solely based on the subsampled sequence itself and are not determined using information from the other CF sequence members. Additionally, the parameters used within the `partis` clustering and annotation procedure are derived from an external dataset. Once we finish the `partis` inference process on the subsampled sequences, we construct the amino acid count matrix for these same sequences; we denote these substitution profiles as the "subsampled" profiles because they are subsampled from the "full" profiles in the model fitting dataset.

Simulation of Neutral Substitution Profiles

For each of the 500 subsampled substitution profiles, we also simulate a neutral substitution profile via a context-sensitive model. For each subsampled sequence, we calculate its number

of somatic hypermutations (SHMs) and introduce that number of mutations sequentially into the inferred naive DNA sequence according to the BCR-specific neutral substitution model S5F (Cui et al., 2016). Once the last mutation is introduced, the simulated DNA sequence is translated into an amino acid sequence and stored as a sample of the neutral substitution process. This procedure is repeated 10,000 times and the count profile aggregated over all the samples is referred to as the "neutral" profile.

External Validation Datasets

For validation, two test sets are generated: the first called "Briggs", is made from the healthy donor single cell droplet sequencing dataset described in (Briggs et al., 2017). Briefly, the data is made by passing 3 million B cells into 6 emulsion pools, each droplet with a unique barcode, and then reverse transcribing mRNA inside these droplets, attaching both a droplet and a molecular barcode. After breaking the emulsion, cDNA is sequenced and processed using UMI consensus building using pRESTO. The highest-quality UMI consensus sequence is extracted from each drop and aggregated into the final heavy chain dataset, which is then further partitioned into CFs using `partis`. Finally, this validation dataset is built up in the same manner as the model fitting dataset, where the only difference is that we allow smaller CFs to enter this dataset (minimum 28 sequences; Table 5.1) in order to increase the number of extracted CFs to 100.

As all the above described datasets are repertoire wide datasets with hundreds of clonal families, we sought to find a suitable dataset with focus on a single large CF. For this, we created the second test set curated from the "Liao" dataset which comes from a well studied broadly neutralizing HIV clone, CH103, described in (Liao et al., 2013). To prepare the raw heavy chain sequences from (Liao et al., 2013), they were annotated and indel reversed by `partis`, following reconstruction of the whole VDJ region by substituting ambiguous bases with bases from the `partis`-inferred naive sequence if necessary. Finally, sequences unable to be annotated within the standard 149 position AHo numbering scheme were filtered out, leaving 312 sequences (available on the SPURF GitHub repository). For the Liao dataset, the

prediction error is measured using all sequences as input samples, contrary to the repertoire datasets where only a single input sequence from each CF is used.

	Dataset summary statistics					
Dataset	N_{donors}	N_{CF}	Total N_{seq}	Min N_{seq}	Median N_{seq}	Max N_{seq}
Aggregated	33	31,893	518,174	5	9	2,709
Model fitting	15	500	98,887	100	147	2,709
Public	33	31,393	419,287	5	8	104
Briggs	1	100	6,702	28	44	370
Liao	1	1	312	312	312	312

Table 5.1: Number of donors (N_{donors}), number of CFs (N_{CF}), number of sequences from all CFs (Total N_{seq}), smallest CF size (Min N_{seq}), median CF size (Median N_{seq}), and maximum CF size (Max N_{seq}). "Aggregated" is the base dataset aggregating RD1-6. "Model fitting" refers to the dataset with the 500 largest CFs from the "Aggregated" dataset. "Public" is the dataset left after the "Model fitting" dataset is extracted from the "Aggregated" dataset. "Briggs" and "Liao" are the external validation datasets used for testing.

In summary, our datasets span 35 different donors, ∼32,000 clonal families and ∼500,000 sequences (Table 5.1). We note that the distribution of VDJ gene usage is non-uniform but that the "Model fitting" and "Public" datasets have very similar V/J gene usage (Figure 5.10 and Figure 5.11). On the other hand, the "Briggs" dataset does have a distinctly different V/J gene usage distribution compared to the other datasets, which we attribute to the fact that it comes from a single donor.

5.2.3 Input Data Tensor

Before we present our penalized tensor regression model, we first describe how the input data for the model is constructed, building off the data descriptions in the last subsection. Throughout the rest of this section, we assume the count matrices are normalized to frequencies and reorganized into three-dimensional tensors (i.e. arrays) as follows. For any substitution profile tensor $\mathbf{T} = \{T_{i,j,k}\}$, let $T_{i,j,k}$ denote the frequency of the kth amino acid at the jth AHo position for the ith CF; we represent the subsampled, full, and public sub-

stitution profile tensors as $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$, respectively. Our goal is to use the subsampled profiles $\mathbf{X}$ to predict the corresponding full substitution profiles $\mathbf{Y}$ (i.e. we want to construct a function $F(\mathbf{X})$ such that $F(\mathbf{X}) \approx \mathbf{Y}$). We incorporate information from the public dataset $\mathbf{Z}$ to enhance these predictions. In addition to the subsampled profiles, we use other types of substitution profiles within $F(\mathbf{X})$:

1. Public substitution profiles segmented by the inferred V-subgroup label ($\widehat{\mathbf{X}}_{\text{vsubgrp}}$);
2. Public substitution profiles segmented by the inferred V-gene label ($\widehat{\mathbf{X}}_{\text{vgene}}$);
3. Inferred naive sequence "substitution profiles" ($\widehat{\mathbf{X}}_{\text{naiveAA}}$);
4. Public substitution profiles segmented by the inferred naive sequence ($\widehat{\mathbf{X}}_{\text{naiveAA-clust}}$);
5. Public substitution profiles segmented by the original frequency profiles ($\widehat{\mathbf{X}}_{\text{clust}}$);
6. Neutral substitution profiles ($\widehat{\mathbf{X}}_{\text{neut}}$).

To compute the external profiles in $\widehat{\mathbf{X}}_{\text{vsubgrp}}$ (resp. $\widehat{\mathbf{X}}_{\text{vgene}}$), we cluster the public dataset $\mathbf{Z}$ by averaging its CF-specific substitution profiles according to the `partis`-inferred (Ralph and Matsen IV, 2016a) IMGT defined (Lefranc, 2001) V-subgroup (resp. V-gene) labels and then assign each row in $\mathbf{X}$ to a V-subgroup (resp. V-gene) cluster profile according to its V-subgroup (resp. V-gene) identity. We obtain the second set of profiles $\widehat{\mathbf{X}}_{\text{naiveAA}}$ by using the `partis`-inferred naive sequences as substitution profiles (these profiles contain zeros and ones because they are based on one sequence only); we re-emphasize that these naive sequences are inferred based only on the corresponding subsampled sequences in $\mathbf{X}$. We cluster the public dataset $\mathbf{Z}$ once more by running K-means clustering based on the inferred naive sequences in $\mathbf{Z}$ and obtain our third set of substitution profiles $\widehat{\mathbf{X}}_{\text{naiveAA-clust}}$ by assigning each CF in $\mathbf{X}$ to its closest cluster centroid. The additional cluster profiles $\widehat{\mathbf{X}}_{\text{clust}}$ are obtained similarly as above, except in this case, we run K-means clustering based on the original frequency profiles in $\mathbf{Z}$. The K-means clustering procedure is run over a grid

of cluster sizes ranging from 2 to 120 using the algorithm described by Hartigan and Wong (1979) with the standard euclidean distance metric. Lastly, the tensor $\widehat{\mathbf{X}}_{\text{neut}}$ contains the simulated S5F neutral substitution profiles, which are described in the previous subsection.

The frequency tensors $\widehat{\mathbf{X}}_{\text{vsubgrp}}$ and $\widehat{\mathbf{X}}_{\text{vgene}}$ are important to include in our analysis because these profiles capture substitution information at the level of the V subgroup (V1, V2, ...) and V gene (V1-5, V2-2, ...), respectively; this is similar to the types of profiles obtained in (Sheng et al., 2017). Even though we expect the $\widehat{\mathbf{X}}_{\text{vsubgrp}}$ and $\widehat{\mathbf{X}}_{\text{vgene}}$ tensors to be correlated, we are interested in seeing whether either of these profiles will dominate the other in our regression model. As described in the introduction, most germinal center lineages do not accumulate many mutations relative to the naive sequence so substitution profiles based solely on the naive sequence (like $\widehat{\mathbf{X}}_{\text{naiveAA}}$) may be informative for predicting the mutational patterns at conserved residue positions. In addition, we believe that the $\widehat{\mathbf{X}}_{\text{naiveAA-clust}}$ cluster profiles are useful as the naive sequence can greatly influence the pattern of substitutions in a CF due to local sequence context. Unlike the $\widehat{\mathbf{X}}_{\text{vsubgrp}}$ and $\widehat{\mathbf{X}}_{\text{vgene}}$ substitution profiles, which are based on IMGT labeling schemes, the profiles in $\widehat{\mathbf{X}}_{\text{naiveAA-clust}}$ (and $\widehat{\mathbf{X}}_{\text{clust}}$) are determined by a data-driven clustering procedure, which allows us to group CFs in $\mathbf{Z}$ in a more intricate fashion. The simulated neutral substitution profiles $\widehat{\mathbf{X}}_{\text{neut}}$ are able to provide some insight into the CF-specific SHM processes without the corresponding clonal selection effects.

To condense our model presentation, we introduce a four-dimensional tensor $\mathbf{X}^*$ that combines as many of the input profiles mentioned previously as we would like, where p, the size of the fourth tensor dimension, represents the number of external profiles used. We define $\mathbf{X}^* \equiv \{X^*_{i,j,k,l}\}$ to be the input data tensor that incorporates all the external information we want to use in our substitution profile predictions; note that $i \in \{1, ..., N_{CF}\}$ (N_{CF} CFs in the tensors), $j \in \{1, ..., 149\}$ (149 AHo positions), $k \in \{1, ..., 20\}$ (20 amino acids), and $l \in \{1, ..., p\}$ (p external profiles). Each element $X^*_{i,j,k,l}$ represents an amino acid frequency as described above for $\mathbf{T}_{i,j,k}$; for instance, $X^*_{5,130,1,4}$ represents the amino acid frequency of the first amino acid (i.e. alanine) at the 130th AHo position for the 5th CF in the 4th profile in the tensor. In addition, we use the indexing symbol $\bullet$ to extract all elements of a particular

array dimension of a tensor (i.e. $\mathbf{X}^*_{10,50,\bullet,2}$ specifies the full substitution profile of the 20 amino acids at the 50th AHo position for the 10th CF in the 2nd profile in the tensor). This setup allows us to easily include as many external profiles as we would like.

5.2.4 Model Formulation

Given the subsampled profiles $\mathbf{X}$ and all the external profiles $\mathbf{X}^*$, we compute a weighted average to form an estimator of $\mathbf{Y}$. Our independent-across-sites model $F(\mathbf{X}) = \left[f(\mathbf{X}_{\bullet,1,\bullet}), ..., f(\mathbf{X}_{\bullet,149,\bullet})\right]$ is specified as follows:

$$f(\mathbf{X}_{\bullet,j,\bullet}) \equiv f(\mathbf{X}_{\bullet,j,\bullet}; \boldsymbol{\alpha}_{j,\bullet}) = \sum_{l=1}^{p} \alpha_{j,l} \cdot \mathbf{X}^*_{\bullet,j,\bullet,l} + \left(1 - \sum_{l=1}^{p} \alpha_{j,l}\right) \cdot \mathbf{X}_{\bullet,j,\bullet}, \tag{5.1}$$

where $\boldsymbol{\alpha} = \{\alpha_{j,l}\}$; $0 \leq \alpha_{j,l} \leq 1$; $0 \leq \sum_{l=1}^{p} \alpha_{j,l} \leq 1$ represents the site-specific weights of the different external profiles for $j = 1, ..., 149$ and $l = 1, ..., p$. Although we consider f to be a function of the per-site data $\mathbf{X}_{\bullet,j,\bullet}$, the frequencies $\mathbf{X}^*_{\bullet,j,\bullet,l}$ are computed using sequence-level, site-dependent information. With $149 \times p$ parameter values of $\boldsymbol{\alpha}$, this is a highly parameterized model so we include regularization terms to prevent overfitting and obtain sparse, interpretable parameter estimates. Specifically, we use standard and spatial (fused) lasso penalties to achieve these goals.

Standard lasso penalties shrink individual parameters to zero and are commonly used to obtain sparse solutions in regression problems (Tibshirani, 1996). It has been shown that regression models using standard lasso penalties provide more accurate predictions than models using best subset selection penalties when there is a low signal-to-noise ratio (Hastie et al., 2017), which probably holds true in our problem as well. In addition, standard lasso penalties are convex functions, which is important in a regression problem as it guarantees that a local minimum is indeed a unique global solution (Boyd and Vandenberghe, 2004).

On the other hand, fused lasso penalties shrink the differences between parameters to zero and are useful in regression problems with spatially-related covariates (Tibshirani et al., 2005). We believe that the $\boldsymbol{\alpha}$ parameters have a spatial relationship (i.e. adjacent residues are

under similar constraints); for instance, given that the mutations in the framework regions are largely related to antibody stability, it makes sense that we would weight external profile information similarly in those regions. The fusion penalty in this setting enforces smoothness of the $\boldsymbol{\alpha}$ trend across the AHo positions. For example, if we penalize first-order differences of the $\boldsymbol{\alpha}$ trend, the fitting procedure will necessarily favor trends that have no slope (i.e. that are piecewise constant). We can obtain more flexible piecewise polynomial $\boldsymbol{\alpha}$ trends by penalizing higher-order successive differences of $\boldsymbol{\alpha}$ (Tibshirani et al., 2014).

In our modeling framework, the standard lasso penalty is represented as $\sum_{j=1}^{149}\sum_{l=1}^{p}|\alpha_{j,l}| = \big|\big|\boldsymbol{\alpha}\big|\big|_1$ and the fused lasso penalty is specified by $\sum_{l=1}^{p}\big|\big|\nabla^d(\boldsymbol{\alpha}_{\bullet,l})\big|\big|_1$, where $||\cdot||_q$ denotes the L_q norm and $\nabla^d(\cdot)$ represents the dth difference operator. This $\nabla^d(\cdot)$ operator accepts a vector $\mathbf{v}$ as input (call its length $n_{\mathbf{v}}$) and outputs a length-$(n_{\mathbf{v}}-d)$ vector that results from successively differencing adjacent elements d times. In the special case when $d=1$, the fusion penalty becomes $\sum_{l=1}^{p}\big|\big|\nabla^1(\boldsymbol{\alpha}_{\bullet,l})\big|\big|_1 = \sum_{j=2}^{149}\sum_{l=1}^{p}|\alpha_{j,l}-\alpha_{j-1,l}|$; the $|\alpha_{j,l}-\alpha_{j-1,l}|$ terms can be interpreted as first-order discrete derivatives.

Our unpenalized objective function can be written as:

$$L_2^{\boldsymbol{\alpha}} \equiv L_2^{\boldsymbol{\alpha}}(\mathbf{Y}, F(\mathbf{X})) = \frac{1}{149 \cdot N_{CF}}\sum_{j=1}^{149}\big|\big|\mathbf{Y}_{\bullet,j,\bullet} - f(\mathbf{X}_{\bullet,j,\bullet};\boldsymbol{\alpha}_{j,\bullet})\big|\big|_2^2, \tag{5.2}$$

where, as in the last subsection, N_{CF} denotes the number of CFs in $\mathbf{X}$ and $\mathbf{Y}$; we refer to this objective as "L_2 Error". Our penalized estimation problem is defined in the following manner:

$$\begin{gathered}\widehat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}}\ L_2^{\boldsymbol{\alpha}}(\mathbf{Y}, F(\mathbf{X})) + \lambda_1\big|\big|\boldsymbol{\alpha}\big|\big|_1 + \lambda_2\sum_{l=1}^{p}\big|\big|\nabla^d(\boldsymbol{\alpha}_{\bullet,l})\big|\big|_1, \\ \text{s.t. } 0 \le \alpha_{j,l} \le 1,\ 0 \le \sum_{l=1}^{p}\alpha_{j,l} \le 1,\ \forall j,l,\end{gathered} \tag{5.3}$$

where $\lambda_1, \lambda_2 \ge 0$ and $d \in \mathbb{N}$ signify tuning parameters. The differencing order d is used to specify a given level of smoothness in the spatial $\boldsymbol{\alpha}$ trend estimates because the $\sum_{l=1}^{p}\big|\big|\nabla^d(\boldsymbol{\alpha}_{\bullet,l})\big|\big|_1$

term in the above minimization problem encourages $\boldsymbol{\alpha}$ trends that have dth order discrete derivatives close to 0 (i.e. that are piecewise polynomials of order $d-1$). In addition, careful selection of λ_1 and λ_2 is required to obtain an adequate model fit. Unfortunately, this is a constrained optimization problem with a multivariate output and there are not any obvious ways to minimize such an objective without resorting to general-purpose optimizers. Therefore, in all our experiments, we use the L-BFGS-B algorithm (Byrd et al., 1995) to fit the above model. We note that the above penalized optimization problem is (non-strictly) convex so any local minimum is, in fact, a global solution too.

5.2.5 Jaccard Similarity

While the model described above has computational and statistical appeal, in engineering applications it is mostly interesting to know the high-frequency amino acid predictions; however, our penalized objective function focuses attention on the complete substitution profiles and not exclusively the high-frequency amino acids. To provide a metric with exclusive focus on high-frequency amino acids, we utilize the Jaccard similarity metric, which can be used to measure differences between predicted and observed sets. Sets of high-frequency amino acids are defined at each position by a minimum frequency cutoff t; Jaccard similarities are then computed between the observed and predicted sets and averaged across each CF and AHo position in the dataset.

The Jaccard similarity metric (Jaccard, 1912) measures the similarity between two finite sets. Specifically, for any sets A and B, the similarity metric $J(A,B)$ is defined as the ratio of the intersection size $|A \cap B|$ to the union size $|A \cup B|$. It has these properties: $0 \leq J(A,B) \leq 1$; $J(A,B) = 1$ when $A = B$ and $J(A,B) = 0$ when $A \cap B = \emptyset$ (empty set). To formally establish our use of Jaccard similarity, we define the following notation. Let $\mathcal{Y}_{i,j} = \{y \in \mathbf{Y}_{i,j,\bullet} \mid y \geq t\}$ represent the set of amino acid frequencies at AHo position j for CF i that has observed frequencies greater than or equal to the cutoff t and denote $\boldsymbol{\mathcal{Y}} \equiv \{\mathcal{Y}_{i,j}\}$ for $i = 1, ..., N_{CF}$ and $j = 1, ..., 149$. We define $\widehat{\mathcal{F}}^{\mathbf{X}}_{i,j}$ and $\widehat{\boldsymbol{\mathcal{F}}}^{\mathbf{X}} \equiv \{\widehat{\mathcal{F}}^{\mathbf{X}}_{i,j}\}$ to be the analogous quantities for the predicted amino acid frequencies. If we let $\mathcal{A}(\mathcal{Y}')$ denote a

function that accepts as input an amino acid frequency set $\mathcal{Y}'$ (i.e. $\mathcal{Y}_{i,j}$ or $\widehat{\mathcal{F}}^{\mathbf{X}}_{i,j}$) and outputs the corresponding set of amino acid identities, then our Jaccard similarity objective can be written as:

$$J^{\boldsymbol{\alpha}}_t \equiv J^{\boldsymbol{\alpha}}_t(\mathbf{Y}, F(\mathbf{X})) = \frac{1}{149 \cdot N_{CF}} \sum_{i=1}^{N_{CF}} \sum_{j=1}^{149} J\big(\mathcal{A}(\mathcal{Y}_{i,j}), \mathcal{A}(\widehat{\mathcal{F}}^{\mathbf{X}}_{i,j})\big), \tag{5.4}$$

which is referred to as the "Jaccard Similarity" objective. We can define a penalized Jaccard estimation problem by substituting $-J^{\boldsymbol{\alpha}}_t(\mathbf{Y}, F(\mathbf{X}))$ for $L^{\boldsymbol{\alpha}}_2(\mathbf{Y}, F(\mathbf{X}))$ in Equation (5.3). Jaccard similarity optimization is difficult using derivative-based optimization because of its discrete nature, so we use a smooth approximation of the aforementioned metric for model fitting in our experiments (see Supplementary subsection Smoothed Jaccard Similarity).

5.2.6 Forward Stepwise Selection

We devise a forward stepwise selection procedure to help us determine the combination of external profiles that best predict the outcome of interest, which can be penalized L_2 Error or Jaccard Similarity. In this procedure, we initially try all possible external profiles in the model separately and determine the best fit using 5-fold cross-validation. We cache the best model from the initial step and continue fitting models with two external profiles; the first external profile is fixed to be the best profile from the previous round and the second profile can be any possible remaining external profile. We continue this iterative scheme until we reach a prespecified limit on the number of external profiles allowed in $\mathbf{X}^*$. It is important to note that to ease computation, we perform forward selection using the unpenalized variants of our models. Even though this procedure is greedy and not as thorough as all-subsets selection, we believe this technique provides the best trade-off between accuracy and efficiency. We provide the implementation of our stepwise procedures at `https://github.com/krdav/SPURF`.

5.2.7 Inference Pipeline

We apply a 80%/20% training/test split to the model fitting dataset described above. We first run the forward stepwise selection procedure with a maximum profile limit of five to approximately determine the best profile groupings starting with a single profile and ending with a group of five profiles. Using the profile groupings from the previous step, we fit the penalized version of the model and use 5-fold cross-validation to obtain estimates of the relevant tuning parameters, which consist of the lasso penalty weights λ_1, λ_2 and the differencing order d; note that we report unpenalized performance estimates when we run cross-validation. After we determine the optimal tuning parameters via cross-validation, we fit the penalized model using the entire training portion of the model fitting dataset and the best tuning parameters and cache the resulting parameter estimates of $\boldsymbol{\alpha}$. Once we obtain the estimates of $\boldsymbol{\alpha}$ from the penalized model, we can use them to compute the chosen performance metric on the testing portion of the model fitting dataset and any other validation dataset of interest.

5.3 Results

As described in the methods (the Inference Pipeline subsection), we first need to infer the best profile groupings to use in penalized model fitting. To determine these groupings, we run the forward stepwise selection procedure for both the L_2 error function and the smoothed Jaccard objective function with a frequency cutoff $t = 0.2$ (Table 5.2). For both objective functions, the forward selection path is the same until $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\right\}$. For the L_2 loss function, model performance is the best when $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\right\}$ even though there are diminishing returns for using profiles beyond $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}\right\}$. In a similar fashion, the Jaccard similarity estimates tend to be highest when $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}\right\}$, despite the almost identical model performance from just using $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}\right\}$. For the subsequent penalized model fitting step, we choose to evaluate the $\left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}\right\}$ and $\left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\right\}$ profile groupings with the L_2 objective and $\left\{\widehat{\mathbf{X}}_{\text{naiveAA}}\right\}$

and $\left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}\right\}$ with the smoothed Jaccard similarity objective. The inclusion of the $\widehat{\mathbf{X}}_{\text{vgene}}$ tensor puts a notable restriction on the model; no prediction can be made for a sequence annotated to a V gene which has not been observed in our Public dataset.

Objective Function	Unregularized CV					
L_2 Error	$\varnothing$	$\widehat{\mathbf{X}}_{\text{naiveAA}}$	$\widehat{\mathbf{X}}_{\text{vgene}}$	$\widehat{\mathbf{X}}_{\text{neut}}$	$\widehat{\mathbf{X}}_{\text{vsubgrp}}$	$\widehat{\mathbf{X}}_{\text{naiveAA-clust-5}}$
	0.110	0.0542	0.0459	0.0456	0.0455	0.0456
Jaccard Similarity ($t = 0.2$)	$\varnothing$	$\widehat{\mathbf{X}}_{\text{naiveAA}}$	$\widehat{\mathbf{X}}_{\text{vgene}}$	$\widehat{\mathbf{X}}_{\text{neut}}$	$\widehat{\mathbf{X}}_{\text{vsubgrp}}$	$\widehat{\mathbf{X}}_{\text{naiveAA-clust-85}}$
	0.9170	0.9322	0.9324	0.9323	0.9319	0.9318

Table 5.2: Results of forward stepwise selection on our L_2 and smooth Jaccard objective functions. The performance estimates shown in the table are obtained using 5-fold cross-validation. Going from left to right, each column represents the best profile addition into $\mathbf{X}^*$ with the associated CV performance estimate. For Jaccard, we fit using the smooth Jaccard objective, but report exact Jaccard similarity estimates, both using frequency cutoff $t = 0.2$. Note that we fix the prespecified limit on the number of external profiles allowed in $\mathbf{X}^*$ to be 5. $\varnothing$ represents the model using only the input sequence.

We now use the approximate profile groupings obtained from the forward stepwise selection procedure to fit our regularized models. The penalized estimation problem has additional tuning parameters that must be determined. In our experiments, we cross-validate over penalty parameters; $\lambda_1, \lambda_2 = 10^{-7}, 5.05 \times 10^{-6}, 10^{-5}$; the differencing order, $d = 1, 2, 3$; and the two profile groupings specified above for both the L_2 error and Jaccard similarity objectives. The best regularized L_2 model uses $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\right\}$, while the best regularized Jaccard model utilizes $\mathbf{X}^* = \left\{\widehat{\mathbf{X}}_{\text{naiveAA}}\right\}$ (Table 5.5, Figure 5.13). In summary, using many external profiles is important for predicting the complete substitution profiles, while the inferred naive sequence is the only external profile deemed useful for our model to accurately predict the observed high-frequency amino acids (where high-frequency is defined by being at least 20% of the observed amino acids).

Our optimization times for both L_2 loss and Jaccard Similarity on the 500 CFs ranged from 12 to 15 minutes. Our optimization is based on evaluating the objective function

at different points and each objective function call has linear complexity in the number of CFs so increasing the number of CFs should result, on average, in a linear increase in time complexity. Computational time invested in pre-processing is one-time and negligible.

In addition to predictive performance, we are also interested in understanding how the estimated parameter weights from our best regularized L_2 model vary across the different external profiles in $\mathbf{X}^*$ and antibody regions. For convenience, we aggregate the estimates of $\boldsymbol{\alpha}$ associated with the V gene ($\widehat{\mathbf{X}}_{\text{vgene}}$ and $\widehat{\mathbf{X}}_{\text{vsubgrp}}$) and with the full naive sequence ($\widehat{\mathbf{X}}_{\text{naiveAA}}$ and $\widehat{\mathbf{X}}_{\text{neut}}$) as these sets of profiles are intuitively similar (Figure 5.3); the V-gene and V-subgroup profiles are both derived by averaging over different IMGT V germline gene labeling schemes and the simulated S5F neutral substitution profiles originate from the CF-specific inferred naive sequence. Antibody heavy chain (and light chain) sequences can be partitioned into framework regions (FWKs) and complementarity-determining regions (CDRs) by the AHo definitions (Honegger and PluÈckthun, 2001); the BCR binding affinity is largely determined by the CDRs (especially by the heavy chain CDR3), while the FWKs encode the structural constraints of the BCR and thus can be strongly conserved (Tomlinson et al., 1995). The $\widehat{\mathbf{X}}_{\text{vgene}}$ and $\widehat{\mathbf{X}}_{\text{vsubgrp}}$ profiles are extremely important for prediction at FWK1-FWK3, which is not surprising as V germline genes extend from the FWK1 to the beginning of the CDR3. In contrast, the $\widehat{\mathbf{X}}_{\text{naiveAA}}$ and $\widehat{\mathbf{X}}_{\text{neut}}$ external profiles are heavily weighted in the CDR3 and FWK4; this result is also intuitive because the CDR3 is highly variable across CFs as it is a strong determinant of antigen-binding specificity, the $\widehat{\mathbf{X}}_{\text{naiveAA}}$ and $\widehat{\mathbf{X}}_{\text{neut}}$ profiles are our only CF-specific sources of external information, and the V gene specific profiles cannot provide any information beyond the end of the V gene. Furthermore, the FWKs have, on average, more support from the external profiles compared to the CDRs, which is consistent with our understanding of antibody biochemistry as the FWKs are structurally constrained and thus need to be more conserved compared to the more flexible CDRs. We note that the middle of the CDR3 has artificially low estimates of $\boldsymbol{\alpha}$ because most of the AHo positions in the CDR3 have only a few or no defined sequence positions in the dataset (Figure 5.7).

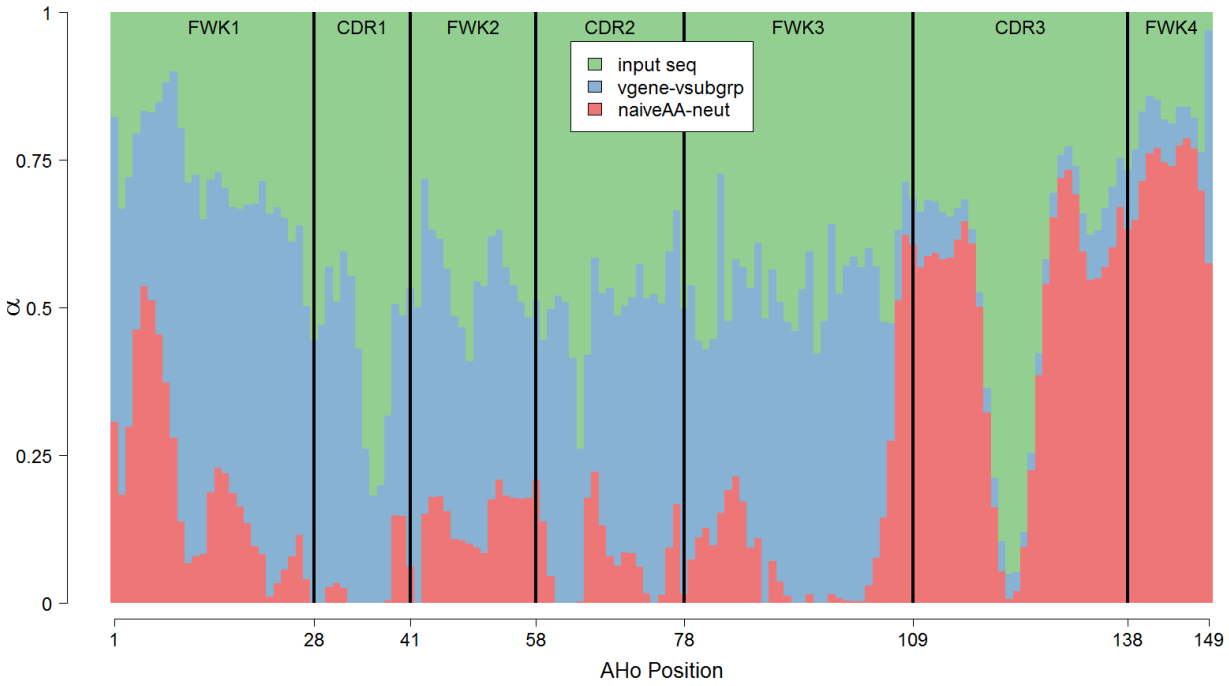


Figure 5.3: A stacked barplot of the estimated parameter values of $\boldsymbol{\alpha}$ from the best regularized L_2 model. For convenience, we aggregate the estimates of $\boldsymbol{\alpha}$ associated with $\widehat{\mathbf{X}}_{\text{vgene}}$ and $\widehat{\mathbf{X}}_{\text{vsubgrp}}$ (blue) and with $\widehat{\mathbf{X}}_{\text{naiveAA}}$ and $\widehat{\mathbf{X}}_{\text{neut}}$ (red). The black vertical lines represent the boundaries between the different CDRs and FWKs.

While our penalized modeling framework allows for easy interpretation of the parameter estimates, ultimately the quality of the $\boldsymbol{\alpha}$ estimates is determined by their performance on independent test datasets. Specifically, we compute the L_2 error ($L_2^{\boldsymbol{\alpha}}$) and Jaccard similarity ($J_{0.2}^{\boldsymbol{\alpha}}$) between the predicted and observed profiles associated with both the testing portion of the model fitting dataset and the Briggs validation dataset (Table 5.3); we remind readers that these predictions are made based on the subsampled (i.e. single-sequence) profiles in the aforementioned datasets and compared to the corresponding actual amino acid frequencies through the $L_2^{\boldsymbol{\alpha}}$ and $J_{0.2}^{\boldsymbol{\alpha}}$ performance metrics (Figure 5.2). Additionally, we compute the L_2 error and Jaccard similarity on all sequences in the Liao dataset, comparing the baseline and SPURF predictions to the full amino acid frequencies (Table 5.3, Table 5.6, and Figure 5.12). Our model improves upon the "baseline" prediction performance, where "baseline" refers to

predictions made using only the input sequence (i.e. model predictions with all parameter values of $\boldsymbol{\alpha}$ set to 0).

Objective Function	Model Type	Objective Function Values		
		Model fitting: test	Briggs	Liao
L_2 Error	Best	0.0492	0.0511	0.0991
	Baseline	0.114	0.129	0.183
Jaccard Similarity ($t = 0.2$)	Best	0.9289	0.9227	0.8516
	Baseline	0.9156	0.9053	0.8439

Table 5.3: The model performance using either L_2 Error or Jaccard Similarity resulting from predicting on independent datasets. We provide results for the testing portion of the model fitting dataset, the Briggs validation dataset, and the Liao dataset. Note that the term "baseline" refers to predictions made using only the input sequence (i.e. model predictions with all parameter values of $\boldsymbol{\alpha}$ set to 0). Lower L_2 error and higher Jaccard Similarity mean higher accuracy.

In addition, we also want to know how well our model performs in the different antibody regions (i.e. FWKs/CDRs). To answer this question, we compute the same metrics as shown in Table 5.3 for the different FWKs and CDRs (Figure 5.4). To provide some insight into the variability of the model performance estimates in the different regions, we calculate bootstrap standard errors, which are expressed as error bars in Figure 5.4.

We see that our substitution profile prediction model performs well in the CDRs relative to the baseline model. This is an important finding because antigen binding is largely determined by the sequence segments in the CDRs, and especially CDR3. In fact, our models seem to provide the greatest improvement in performance in the CDR3, which is also the hardest region to predict because it has the highest amount of sequence variability. Another important takeaway is that the prediction performance is better in FWKs than CDRs, which is presumably because FWKs have lower variance and are more conserved compared to CDRs. In summary, our prediction models are able to systematically integrate different data sources to make better predictions of the per-site amino acid compositions in CFs.

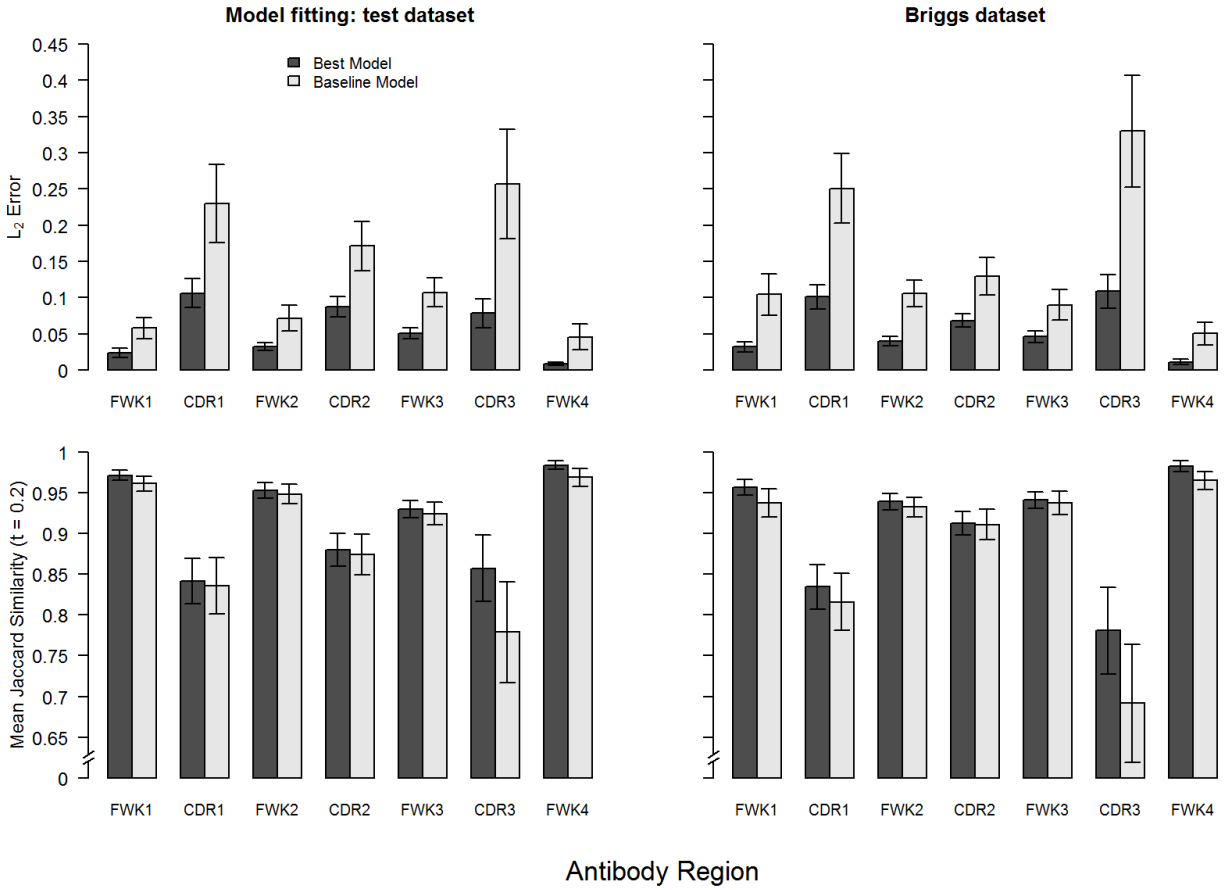


Figure 5.4: The model performance results across the different antibody regions on the model fitting test dataset and the Briggs validation dataset. In these plots, we compare the performances from our best models to the baseline predictive performances using only the input sequence (i.e. model predictions with all parameter values of $\boldsymbol{\alpha}$ set to 0). The error bars show bootstrap standard errors.

Our model also improves the prediction of the highest-frequency amino acid at a given position, referred to here as the mode (Table 5.4). Indeed, the counts in the bottom-left cells (cases where the model is correctly predicting the actual mode given an incorrect input sequence amino acid) are larger than the counts in the top-right cells (vice-versa). In addition, the input sequence amino acids that are not the true modes but correctly predicted by the model to be the actual modes are all germline reversions, which is consistent with the $\widehat{\mathbf{X}}_{\text{naiveAA}}$

profile being heavily weighted in our prediction model (Figure 5.3). In the opposite case, where the input sequence amino acid is correct but the model prediction is wrong, all the counts consist of germline predictions as well. In summary, many of the mode predictions are just germline reversions and, in fact, most of these predictions are to the true modes (i.e. the actual highest-frequency amino acids); however, most of the input sequence amino acids are the true modes already ($\approx 99\%$).

germline | non-germline		Correct SPURF Mode Prediction?	
		Yes	No
Is input amino acid the mode?	Yes	10,473 | 465	156 | 0
	No	349 | 0	170 | 395

(a) Model fitting: test dataset

germline | non-germline		Correct SPURF Mode Prediction?	
		Yes	No
Is input amino acid the mode?	Yes	10,541 | 376	178 | 1
	No	474 | 0	196 | 393

(b) Briggs dataset

Table 5.4: Mode prediction results from both the testing portion of the model fitting dataset and the Briggs dataset fitted using the L_2 objective function. For each CF and AHo position in a given dataset, we determine whether the predicted mode (i.e. highest-frequency amino acid) from our best model is the same as the actual mode. Results are aggregated based on whether or not the input sequence has the correct mode. At the left side of the vertical bar (|) is the count for the germline predicted modes (i.e. situations when the predicted amino acid mode is the naive sequence amino acid) and at the right side is the count for the non-germline predicted modes (vice-versa).

The in-sample and out-of-sample prediction performances demonstrate that our SPURF inference pipeline is able to obtain accurate and robust estimates of $\boldsymbol{\alpha}$. Specifically, prediction

performance is consistently similar but slightly worse when comparing the Briggs dataset to the model fitting test set, which likely reflects two things: 1) the median number of sequences per CF in the Briggs set is lower than in the test set (Table 5.1) and 2) the model fitting dataset is sampled from the same donors as the dataset for cross-validation. Regardless, the differences between the test and Briggs datasets are small, which provides evidence in support of our model performance estimates. We also note that the test on the Liao data yielded results strongly favoring SPURF over baseline. Since the Liao dataset carries a high mutation frequency compared to the average CF of the other dataset it is (as expected) harder to predict the amino acid frequency, which is refelcted in the magnitude of both the L_2 error and Jaccard similarity for all predictions. Subjective assessments of the inferred substitution profiles coincide with our description of the L_2 error metric, namely that fine-grained amino acid substitution information is captured by SPURF (Figure 5.8).

The SPURF model setup produces interpretable and meaningful profile weights (Figure 5.5; per-profile decomposition in Figure 5.9). The input sequence is strongly weighted in the CDRs, indicating that substitutions in these regions are both specific and conserved within the CF and, therefore, cannot easily utilize the information from other CFs. The weight on the V gene specific profiles spikes at CDR1 and at the end of FWK3, which is at the heavy and light chain interface. We note that, as expected, the weight on the V gene specific profiles is minimal downstream of FWK3 as this is the end of the V gene and the beginning of the V-D junction region. As such, nothing prevents the V gene profiles from having a high weight downstream of FWK3, but the model framework has chosen these meaningful weights without any manual interference. We ascribe this shrinkage feature of the weights to the standard lasso penalty built into SPURF. The profiles that are derived from the inferred naive sequence ($\widehat{\mathbf{X}}_{\text{naiveAA}}$, $\widehat{\mathbf{X}}_{\text{neut}}$) take up the missing weight of the V gene profiles as these are highly weighted in the CDR3 and FWK4.

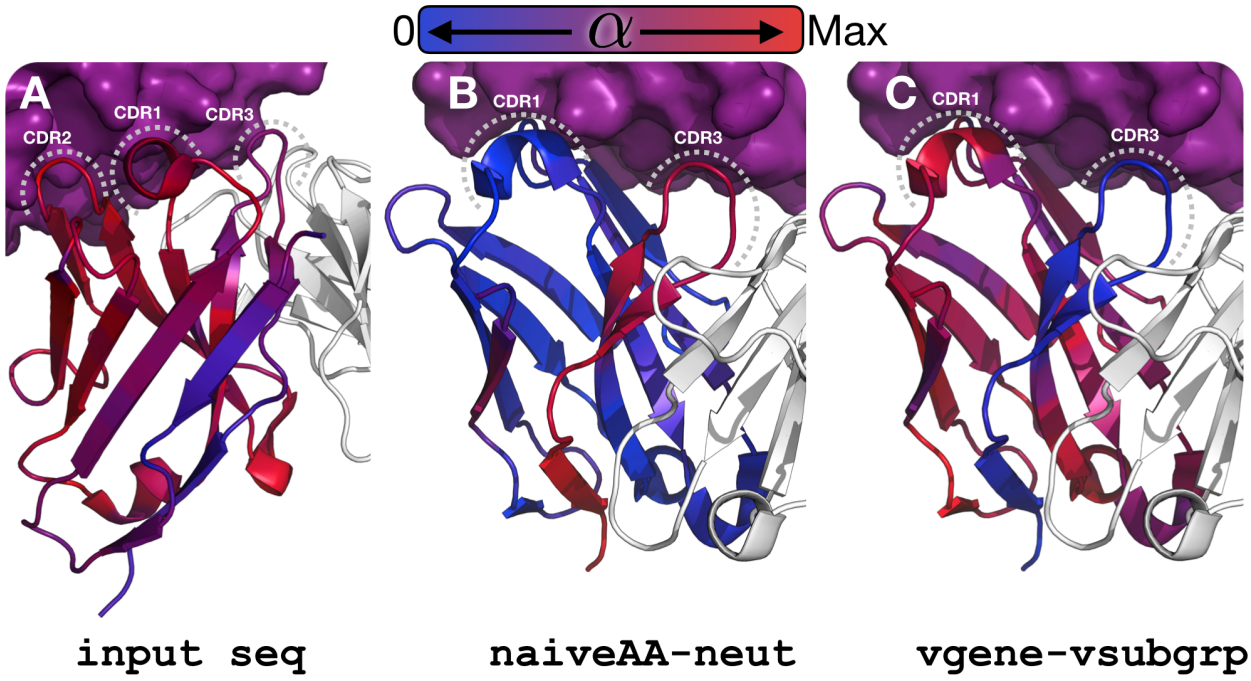


Figure 5.5: Positional profile weights $\boldsymbol{\alpha}$ mapped to an antibody protein structure (PDB: 5X8L). The antigen (PD-L1) appears as a purple surface at the top of the images, the light chain appears in white cartoon, and the heavy chain is displayed using a blue to red color gradient; the grey dashed lines mark the CDR loops. The color gradient represents the possible values of profile weights in $\boldsymbol{\alpha}$ and goes from blue at a zero weight to red at the maximum weight for the profile. The display in panels **B** and **C** is rotated relative to panel **A** to better show results for CDR1 and CDR3; as a consequence, the CDR2 loop is hidden behind the CDR1. Panel **A** shows that the input sequence has high weight at the CDR1 and CDR2, panel **B** illustrates that the naive sequence and the neutral substitution profile have high weight at the CDR3 and FWK4, and panel **C** demonstrates that the V gene and V subgroup profiles are highly weighted in parts of the CDR1 but more generally in the FWKs, especially at the heavy and light chain interface.

5.4 Discussion

In this chapter, we present SPURF, a statistical framework for predicting CF-specific amino acid frequency profiles from single input BCR sequences by leveraging multiple sources of external information. We use standard and spatial lasso penalties to prevent our model from overfitting and obtain sparse, interpretable estimates of the profile weights, expressed by an

$\boldsymbol{\alpha}$ matrix. The spatial lasso penalizes extreme differences between spatially-adjacent profile weights, while the standard lasso penalties promote simpler models by shrinking parameter values in $\boldsymbol{\alpha}$ to 0 if the associated external profiles are not useful predictors. We show that our method not only performs well on the held-out (test) portion of our model fitting dataset but also provides accurate predictions on the Briggs and Liao external validation datasets. Indeed, we did not obtain the Briggs or Liao validation datasets until after we ran our model inference pipeline on the model fitting dataset.

Using two different objective functions we fitted SPURF to predict the frequencies of all amino acids (L_2 objective) and only the >20% frequency amino acids (Jaccard similarity objective). With the L_2 objective we obtained a large difference (0.114 to 0.0492) between the baseline model and SPURF, which was confirmed using repertoire wide data from (Briggs et al., 2017) and single clone data from (Liao et al., 2013) (Figure 5.4 and Figure 5.12). With the Jaccard similarity objective improvements over baseline were more modest (0.9156 to 0.9289) showing that SPURF is strongest at predicting the full spectrum of amino acid frequencies (Table 5.3). Still, fitted using the L_2 objective, SPURF can recover the highest frequency amino acid of a clonal family (mode prediction) much better than a random sequence from the corresponding clonal family (Table 5.4), showing the versatility of the L_2 objective.

Our work can be seen as a prediction-based extension of the work of Sheng et al. (2017) and Kirik et al. (2017). This previous work illustrates that amino acid substitution profiles differ between germline genes, a finding supported by the context specificity of somatic hypermutation (Cui et al., 2016). In our work, we provide a prediction algorithm that takes a single BCR sequence from a clonal family as input and outputs a CF-specific substitution profile estimate for the whole VDJ region. As SPURF relies on large CFs to establish a ground truth substitution profile it is possible that certain types of rare clones or V/J gene combinations are not included in our training/test data. For such rare events the error estimates reported cannot be reliably used, however, we note that our training/test data cover a broad set of V/J combinations (Figure 5.10) and that the substitution profile of a

rare, but expanded, broadly HIV neutralizing clone is well predicted (Figure 5.12).

We believe that this work will be a useful tool for antibody engineering in situations when it is important to maintain antibody binding affinity to the same epitope. The predicted profiles from SPURF can be used to choose the sites that are most tolerable for mutagenesis and the substitutions that are most likely to maintain binding specificity; as such, this information can be used to engineer antibodies with better biophysical properties.

The seven datasets utilized in the present study were all derived from different laboratories employing varying strategies to obtain their processed data which served as input for SPURF. We carefully examined available resources and selected the datasets to be used in our model. However, our approach would greatly benefit from a large and uniformly accessible repository of Rep-seq datasets. For this to happen, data has to be discoverable and usable, including having all information about the study and data processing available together with the raw and processed data in publicly accessible data repositories. Recently, the Adaptive Immune Receptor Repertoire (AIRR) community (Breden et al., 2017) proposed MiAIRR (Rubelt et al., 2017), a set of minimal standard elements to be published alongside the raw and processed data. Future Rep-seq studies following this initiative and making their data available under the MiAIRR-standard will facilitate the development of SPURF and future approaches with similar goals.

To our knowledge, SPURF is the first prediction algorithm for B cell CF substitution profiles. There are many possible extensions; in our SPURF inference pipeline, we subsample single BCR sequences from CFs to use as model input; unfortunately, this means that our modeling analysis is conditional on a dataset that does not account for the variability associated with the subsampling process. One obvious means of fixing the above problem is to draw multiple subsamples from each CF and treat these multiple "observations" per CF within a dataset as a clustered data or weighted least squares problem. In addition, our model fitting dataset consists of only the largest CFs because we need accurate CF-specific substitution profile estimates to serve as the ground truth. This non-random sampling technique could potentially bias our analysis results; however, this appears unlikely given our

model's performance on the external Briggs and Liao validation datasets. Furthermore, our approach models per-site amino acid composition in a CF and accounts for interactions between sites only through the fusion lasso penalties. It is well known from other protein studies that spatially-adjacent amino acid residues evolve jointly (Jones et al., 2011; Ekeberg et al., 2013), presumably to maintain structural stability, or in the case of antibodies to stabilize the interface between heavy and light chains (Wang et al., 2009). In the context of antibodies, residues in the FWKs have the potential to co-evolve (e.g. FWK residues flanking the CDRs could co-evolve to stabilize the stem leading to the more flexible CDRs). Thus, figuring out how to incorporate more detailed interaction effects in our model is an important avenue for future research.

Supplementary Materials

Model Interpretation

In this subsection, we provide statistical motivation for our penalized regression model, which can be interpreted as specifying an ensemble of multinomial logistic regression models at each AHo position. We use some of the notation mentioned in the methods section and introduce new notation as needed. We begin by describing the structure of the multinomial logistic regression models and then discuss how we perform model averaging with these component models to form the estimator $F(\mathbf{X})$ as stated in the methods section. We conclude this subsection by showing that our regularized minimization problem can be characterized as a maximum a posteriori (MAP) inference problem.

Suppose we observe M amino acids at the jth AHo position for the ith CF; for simplicity, we let $y_1, ..., y_M$ denote the observed amino acids. We assume that $y_1, ..., y_M$ are drawn independently from a common multinomial distribution with 20 possible categories and define a logistic regression model for the amino acid probabilities that does not include covariates. The standard way to formulate such a model is as follows:

$$\log\left(\frac{\mathrm{P}(y_m = c)}{\mathrm{P}(y_m = 20)}\right) = \beta^{(c)}, \qquad \forall c \in \{1, ..., 20\},$$

where c indexes a particular amino acid, $\beta^{(20)} \equiv 0$, and $m = 1, ..., M$. We can equivalently represent the model as:

$$\mathrm{P}(y_m = c) = \frac{\exp(\beta^{(c)})}{\sum_{c'=1}^{20} \exp(\beta^{(c')})}, \qquad \forall c \in \{1, ..., 20\},$$

where $m = 1, ..., M$. If we let $\widehat{p}_c$ denote the observed proportion of amino acid c in the sample, then it is easy to show that maximizing the multinomial likelihood of the M observations

with respect to the $\beta^{(c)}$ parameters leads to the following parameter estimates:

$$\widehat{\beta}^{(c)} = \log\left(\frac{\widehat{p}_c}{\widehat{p}_{20}}\right), \qquad \forall c \in \{1, ..., 20\},$$

which implies that:

$$\widehat{\mathrm{P}}(y_m = c) = \widehat{p}_c, \qquad \forall c \in \{1, ..., 20\},$$

where $\widehat{\mathrm{P}}(y_m = c)$ represents the logistic regression estimate of $\mathrm{P}(y_m = c)$. Therefore, this multinomial logistic regression model provides simple, intuitive estimates of the amino acid probabilities. While these logistic regression estimates may seem trivial, the underlying framework allows for easy integration of CF-specific and site-specific information into our model.

The above model considers the amino acid probabilities at only one AHo position so one could fit this multinomial logistic regression model at each of the 149 AHo positions in a CF to obtain a complete substitution profile estimate. In this chapter, each CF-specific substitution profile estimate is a maximum likelihood estimate (MLE) obtained by fitting 149 multinomial regression models to the observed amino acid data in the CF. Given that our proposed modeling procedure computes a per-site weighted average between the subsampled profiles $\mathbf{X}$ and all the external profile estimates $\mathbf{X}^*$, one can interpret this model $F(\mathbf{X})$ as defining an ensemble of multinomial logistic regression estimates at each AHo position.

To specify this relationship more clearly, we denote the likelihood of $\mathbf{Y}_{i,j,\bullet}$ as follows:

$$\mathbf{Y}_{i,j,\bullet} \sim \mathrm{MVN}\big(\boldsymbol{\mu}_{i,j,\bullet}, \sigma^2 \mathbb{I}_{20}\big),$$
$$\boldsymbol{\mu}_{i,j,\bullet} \equiv \sum_{l=1}^{p} \alpha_{j,l} \cdot \mathbf{X}^*_{i,j,\bullet,l} + \Big(1 - \sum_{l=1}^{p} \alpha_{j,l}\Big) \cdot \mathbf{X}_{i,j,\bullet},$$

where MVN means multivariate normal, $\boldsymbol{\mu}_{i,j,\bullet}$ signifies the mean vector of $\mathbf{Y}_{i,j,\bullet}$ and defines our ensemble model, σ^2 represents an unknown variance parameter, $\mathbb{I}_{20}$ symbolizes the 20×20 identity matrix, and p represents the number of external profiles in $\mathbf{X}^*$. Note that $\boldsymbol{\mu}_{i,j,\bullet}$

depends on the multinomial logistic regression estimates described previously as both $\mathbf{X}$ and $\mathbf{X}^*$ contain the MLE-based substitution profile estimates. In addition, the form of $\boldsymbol{\mu}_{i,j,\bullet}$ relates to our previous definition of $F(\mathbf{X})$ by observing that $\boldsymbol{\mu}_{\bullet,j,\bullet} = f(\mathbf{X}_{\bullet,j,\bullet}; \boldsymbol{\alpha}_{j,\bullet})$. We can also integrate the inequality constraints of the ensemble weights $\alpha_{j,l}$ into the likelihood function by including the indicator term $\mathbb{1}_{\boldsymbol{\alpha}} \equiv \mathbb{1}\{0 \leq \alpha_{j,l} \leq 1;\ 0 \leq \sum_{l=1}^{p} \alpha_{j,l} \leq 1;\ \forall j, l\}$. The lasso penalties can be incorporated into our model through the use of sparsity-inducing prior distributions.

Specifically, the priors placed on $\boldsymbol{\alpha}$ can be represented as:

$$\mathrm{P}(\boldsymbol{\alpha}_{\bullet,l}) \propto \exp\Big(-\lambda_1 \big|\big|\boldsymbol{\alpha}_{\bullet,l}\big|\big|_1 - \lambda_2 \big|\big|\nabla^d(\boldsymbol{\alpha}_{\bullet,l})\big|\big|_1\Big), \qquad \forall l \in \{1, ..., p\},$$

where $\lambda_1, \lambda_2 \geq 0$ and $d \in \mathbb{N}$ are the same tuning parameters specified in the methods section (Park and Casella, 2008; Kyung et al., 2010; Faulkner et al., 2018). These Laplace-like prior distributions can be expressed as scale mixtures of normal distributions with independent gamma distributed variances (Kyung et al., 2010); for a more comprehensive discussion on shrinkage priors of this form, we refer readers to (Faulkner et al., 2018).

The posterior $\mathrm{P}(\boldsymbol{\alpha}|\mathbf{Y})$ can be presented in the following manner:

$$\begin{aligned}\mathrm{P}(\boldsymbol{\alpha}|\mathbf{Y}) &\propto \mathrm{P}(\mathbf{Y}|\boldsymbol{\alpha})\mathrm{P}(\boldsymbol{\alpha}) \\ &\propto \prod_{i=1}^{500}\prod_{j=1}^{149} \mathrm{P}(\mathbf{Y}_{i,j,\bullet}|\boldsymbol{\alpha}_{j,\bullet}) \prod_{l=1}^{p} \mathrm{P}(\boldsymbol{\alpha}_{\bullet,l}).\end{aligned}$$

Note that we assume the $\mathbf{Y}_{i,j,\bullet}$ vectors are independent conditional on $\boldsymbol{\alpha}_{j,\bullet}$ and the prior distribution on $\boldsymbol{\alpha}$ factorizes across $l \in \{1, ..., p\}$. The MAP estimate of the ensemble weights $\boldsymbol{\alpha}$ is obtained by maximizing $\mathrm{P}(\boldsymbol{\alpha}|\mathbf{Y})$ and is equivalent to the estimate that minimizes the regularized objective function shown in the methods section. The latter assertion can be seen as the posterior on $\boldsymbol{\alpha}$ can be monotonically transformed into our penalized minimization problem (up to a constant factor in $\boldsymbol{\alpha}$).

Of course, there are limitations to this interpretation of our modeling framework. For

instance, $\mathbf{Y}_{i,j,\bullet}$ is a frequency vector, yet we model the likelihood of $\mathbf{Y}_{i,j,\bullet}$ using a multivariate normal distribution, which has support over all real numbers; we could potentially remedy this problem by modeling the likelihood of $\mathbf{Y}_{i,j,\bullet}$ as a Dirichlet distribution as its negative log-likelihood looks similar to a cross-entropy loss function. In addition, we specify priors on $\boldsymbol{\alpha}$ that also have support over the real line, which is not realistic. Our assumption that the $\mathbf{Y}_{i,j,\bullet}$ vectors are conditionally independent given $\boldsymbol{\alpha}_{j,\bullet}$ is used solely for presentation purposes and does not hold in practice because the substitution profile data in both $\mathbf{X}$ and $\mathbf{X}^*$ are correlated across AHo positions. Despite these issues with our statistical representation of the penalized regression model, our results demonstrate that the model is useful for predicting CF-specific substitution profiles in data-sparse situations.

Smoothed Jaccard Similarity

As we stated in the methods section, optimization on the Jaccard similarity objective function is difficult because this metric is locally flat with respect to our parameter values of $\boldsymbol{\alpha}$. For some small changes in $\boldsymbol{\alpha}$, the averaged Jaccard similarity can remain at the same value because the Jaccard sets continue to hold the same elements. This is a problem because the L-BFGS-B optimization algorithm uses gradient information to determine its search direction for $\boldsymbol{\alpha}$ and the Jaccard similarity gradients are often zero due to the reasoning given above, which results in premature termination of the L-BFGS-B optimizer. We now describe an approach to "smooth" the Jaccard similarity objective function that directly addresses these concerns.

For notational simplicity, we let $\{a_i\}_{i=1:20}$ and $\{b_i\}_{i=1:20}$ denote the actual and predicted amino acid frequencies, respectively, at a particular AHo position for a given CF. As before, t represents the cutoff separating high versus low frequency amino acids. We also introduce the following indicator function $f(a,t) \equiv \mathbb{1}\{a \geq t\}$ for any amino acid frequency a. If we further let $A = \mathcal{A}(\{a_i \mid a_i \geq t\})$ and $B = \mathcal{A}(\{b_i \mid b_i \geq t\})$ with $\mathcal{A}(\cdot)$ as defined in the

methods section, then the Jaccard similarity between sets A and B can be rewritten as:

$$J(A,B) \equiv \frac{|A \cap B|}{|A \cup B|} = \frac{\sum_{i=1}^{20} f(a_i,t) f(b_i,t)}{\sum_{i=1}^{20} \min\{1, f(a_i,t) + f(b_i,t)\}}.$$

The local flatness of the Jaccard similarity objective is due to the constant regions of $f(a_i,t)$ and the non-smooth curvature of $f(a_i,t)$ at the jump point t. It turns out that $f(a_i,t)$ can also be described as the limit of the following function:

$$f_\epsilon(a_i,t) = \frac{1}{1+e^{-\epsilon(a_i-t)}},$$

as $\epsilon \to \infty$. Thus, to obtain a "smooth" transformation of $J(A,B)$, we replace $f(a_i,t)$ with $f_\epsilon(a_i,t)$ in the above equation of $J(A,B)$ and set ϵ (i.e. the steepness parameter) to be a small number. Figure 5.6 plots the function $f_\epsilon(a_i,0.2)$ against $a_i \in [0,1]$ for various values of ϵ.

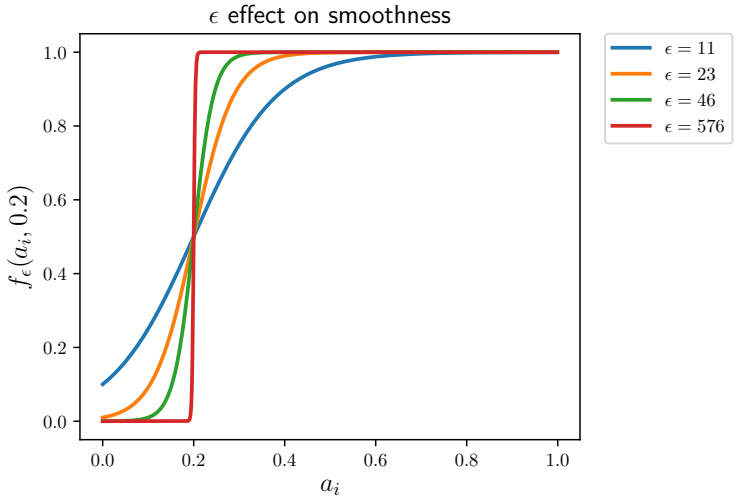


Figure 5.6: A plot of the function $f_\epsilon(a_i,0.2)$ against $a_i \in [0,1]$ for various values of ϵ. As ϵ gets larger, $f_\epsilon(a_i,0.2)$ tends to the indicator function $f(a_i,0.2)$.

Fortunately, the use of this "smooth" Jaccard similarity function allows the L-BFGS-B

optimization algorithm to converge properly. To use this "smoothed" objective function in the right manner, we were interested in finding the largest values of ϵ that still permitted proper L-BFGS-B convergence. We utilized $\epsilon = 23$ throughout all our Jaccard similarity experiments because we found that this value of ϵ satisfied our selection criteria specified previously.

Supplementary Figures and Tables

Objective Function	$\widehat{\mathbf{X}}^*$	$\widehat{\lambda}_1$	$\widehat{\lambda}_2$	$\widehat{d}$	Regularized CV
L_2 Error	$\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\}$	10^{-7}	10^{-5}	3	0.0453
Jaccard Similarity ($t = 0.2$)	$\{\widehat{\mathbf{X}}_{\text{naiveAA}}\}$	10^{-7}	10^{-7}	2	0.9316

Table 5.5: The results from fitting the regularized models using 5-fold cross-validation. We present the optimal tuning parameters selected from $\lambda_1, \lambda_2 = 10^{-7}, 5.05 \times 10^{-6}, 10^{-5}$ and $d = 1, 2, 3$ and show the associated cross-validated performance estimates. Note that the possible choices of $\mathbf{X}^*$ for the L_2 error metric include the $\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}\}$ and $\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\}$ groupings, while the $\{\widehat{\mathbf{X}}_{\text{naiveAA}}\}$ and $\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}\}$ groupings are the possible $\mathbf{X}^*$ choices for the smoothed Jaccard similarity objective.

Objective Function	Model Type	Model fitting: test	Briggs	Liao
L_2 Error	Best (Regularized)	0.0492	0.0511	0.0991
	Best (Unregularized)	0.0494	0.0512	0.100
Jaccard Similarity ($t = 0.2$)	Best (Regularized)	0.9289	0.9227	0.8516
	Best (Unregularized)	0.9289	0.9229	0.8516

Table 5.6: The unregularized and regularized model performance using either L_2 Error or Jaccard Similarity resulting from predicting on independent datasets. We provide results for the testing portion of the model fitting dataset, the Briggs validation dataset, and the Liao dataset. Note that the term "baseline" refers to predictions made using only the input sequence (i.e. model predictions with all parameter values of $\boldsymbol{\alpha}$ set to 0) and lower L_2 error and higher Jaccard Similarity is preferred.

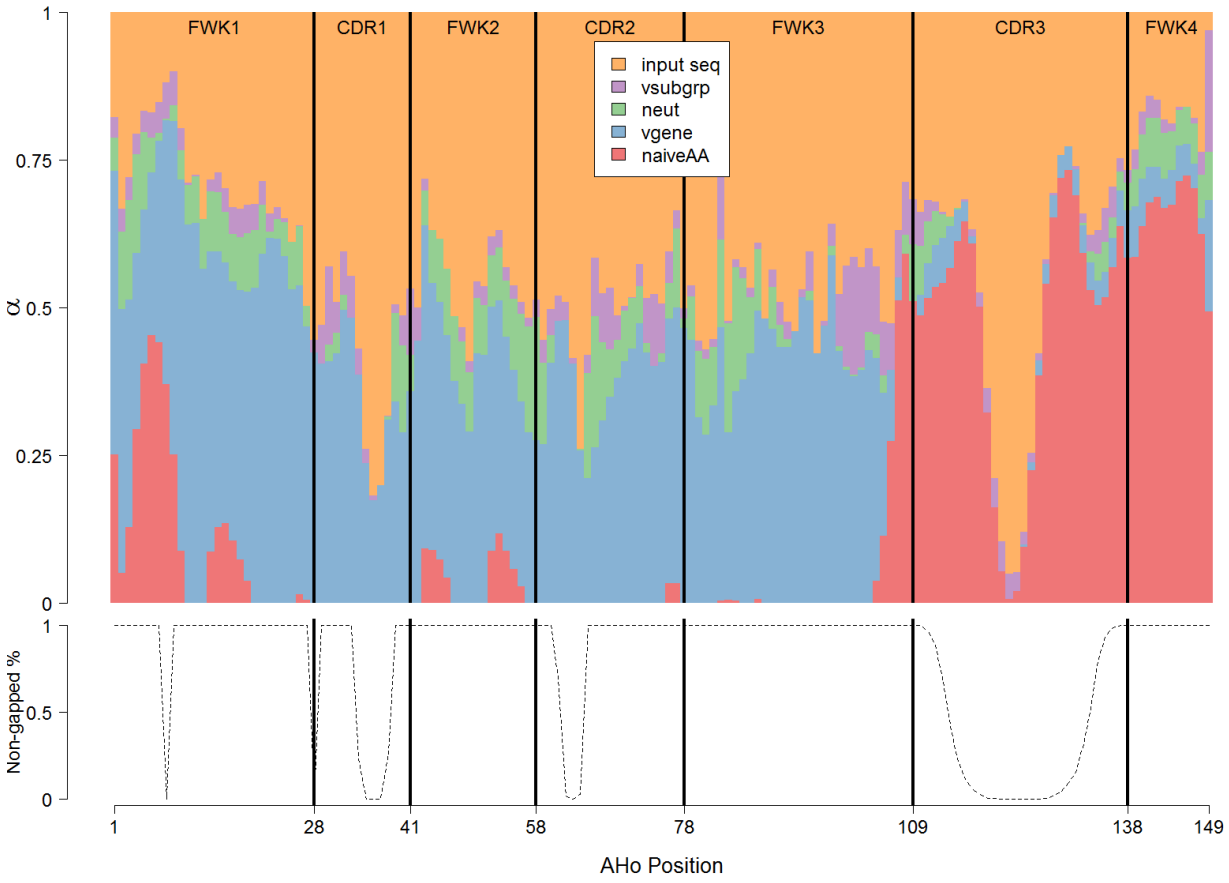


Figure 5.7: A stacked barplot of the estimated parameter values of $\boldsymbol{\alpha}$ from the best regularized L_2 model. The black vertical lines represent the boundaries between the different CDRs and FWKs. Due to the AHo antibody numbering used (Honegger and PluÈckthun, 2001), some positions are assigned to a gap character (an AHo position that does not map to a sequence position). The percentage of CFs that are not assigned to gap characters is shown in the bottom plot for each AHo position. The input sequence is heavily weighted in regions with high gap percentages because of the standard lasso penalty included in our model. The conserved Tryptophan amino acid is observed as a spike in the $\widehat{\mathbf{X}}_{\text{vgene}}$ and $\widehat{\mathbf{X}}_{\text{naiveAA}}$ profile weights following the end of CDR1 (position 43 in the AHo scheme). The conserved Cysteine amino acid that defines the beginning of CDR3 is not readily observed, presumably because this is invariant in all profiles. Generally, the input sequence has less weight in CDR3 and FWK4, which indicates that there is some conservation during affinity maturation. Beyond CDR3 and FWK4, there is a general trend that the input sequence has higher weight in the CDRs than in the FWKs, which suggests that there is a higher level of conservation in the FWKs than in the CDRs during affinity maturation. A more surprising observation is the spike in the $\widehat{\mathbf{X}}_{\text{vgene}}$, $\widehat{\mathbf{X}}_{\text{vsubgrp}}$, and $\widehat{\mathbf{X}}_{\text{neut}}$ weights at AHo position 83 near the beginning of FWK3 (the "outer" loop); this could indicate a conserved position not previously described.

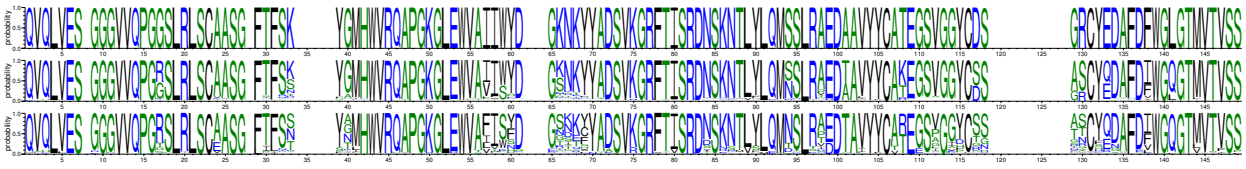


Figure 5.8: A logo plot displaying the input sequence, predicted profile, and true profile (ordered from top to bottom) for an arbitrary CF in the Briggs dataset. The logos are plotted using AHo numbers (1-149) and AHo positions undefined in the sequence are shown as empty columns. The predicted profile (middle) captures much of the amino acid composition information associated with the full profile (bottom).

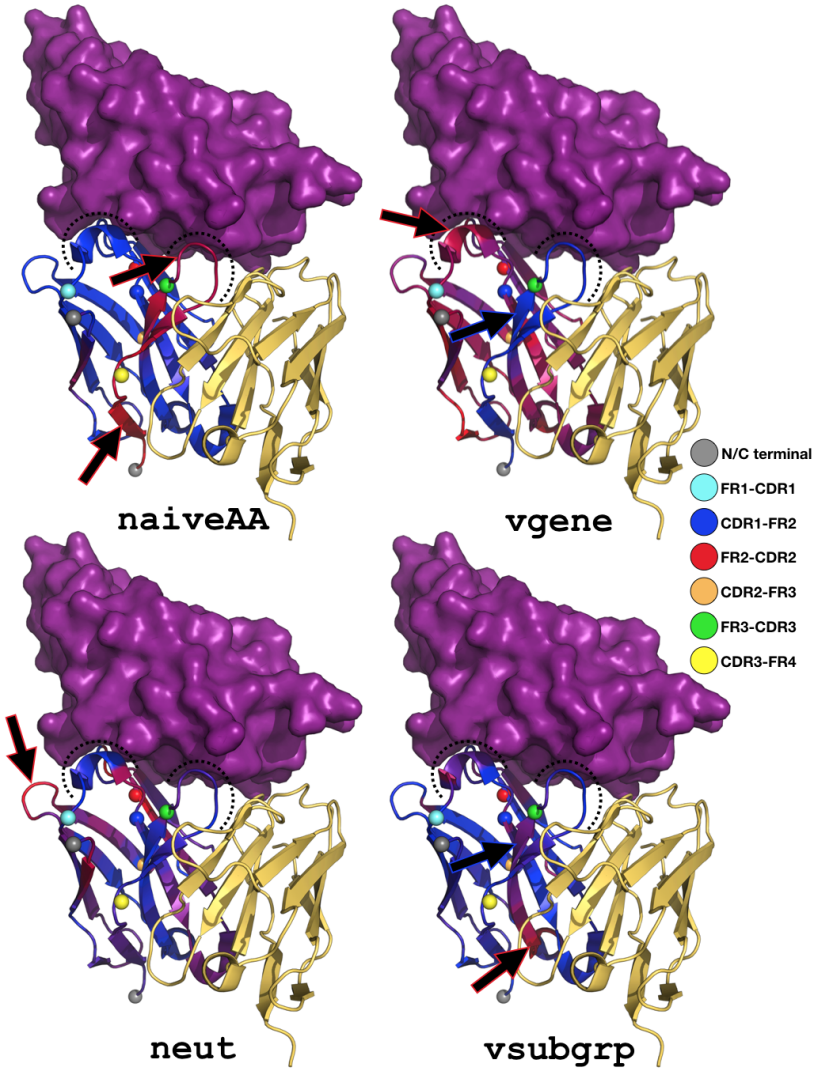


Figure 5.9: Positional profile weights $\boldsymbol{\alpha}$ mapped to an antibody protein structure (PDB: 5X8L). The antigen (PD-L1) appears as a purple surface at the top of the images, the light chain appears in yellow cartoon, and the heavy chain is displayed using a blue to red color gradient. The color gradient represents the possible values of profile weights in $\boldsymbol{\alpha}$ and goes from blue at a zero weight to red at the maximum weight for the profile. The black dashed lines mark the CDR loops; note that the CDR2 loop is hidden behind the CDR1. The colored balls represent the AHo-defined FWK/CDR boundaries. The black arrows indicate regions of high profile weight. The $\widehat{\mathbf{X}}_{\text{naiveAA}}$ profile is heavily weighted in CDR3 and FWK4. The $\widehat{\mathbf{X}}_{\text{vgene}}$ profile weighting is fairly even from FWK1 through FWK3; it spikes slightly in CDR1 and completely disappears beyond FWK3, which is expected as the V-D junction region starts past the end of FWK3. The $\widehat{\mathbf{X}}_{\text{neut}}$ profile weighting is fairly even across sites but spikes near the beginning of FWK3 (the "outer" loop). The $\widehat{\mathbf{X}}_{\text{vsubgrp}}$ profile weighting is distributed similarly to that of the $\widehat{\mathbf{X}}_{\text{vgene}}$ profile with the exception of a spike at the end of FWK3 (i.e. at the heavy and light chain interface).

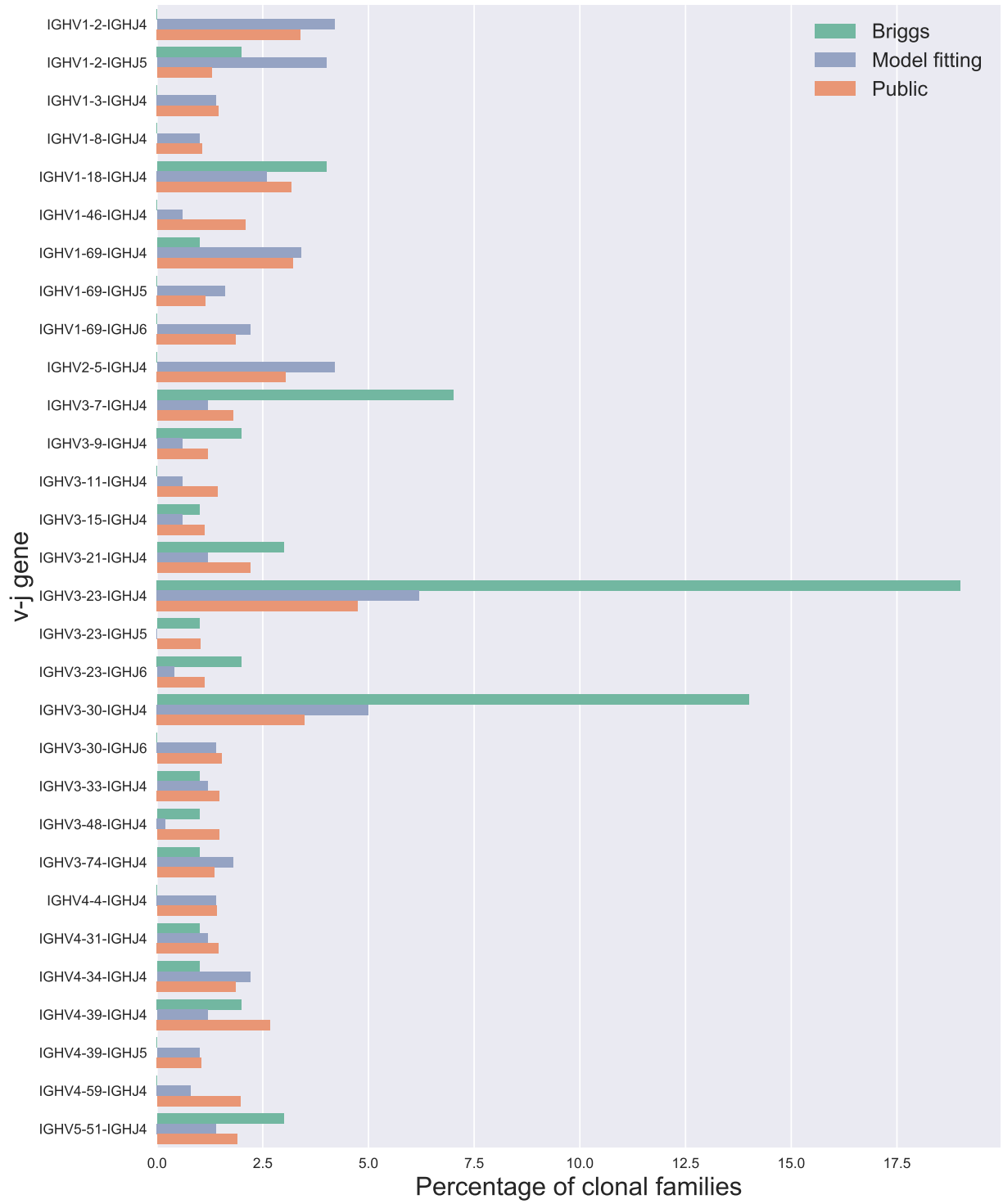


Figure 5.10: Distribution of per-clonal-family V/J gene combination usage in the different dataset partitions. Minimum frequency of 1% in either partition used as a cutoff for inclusion.

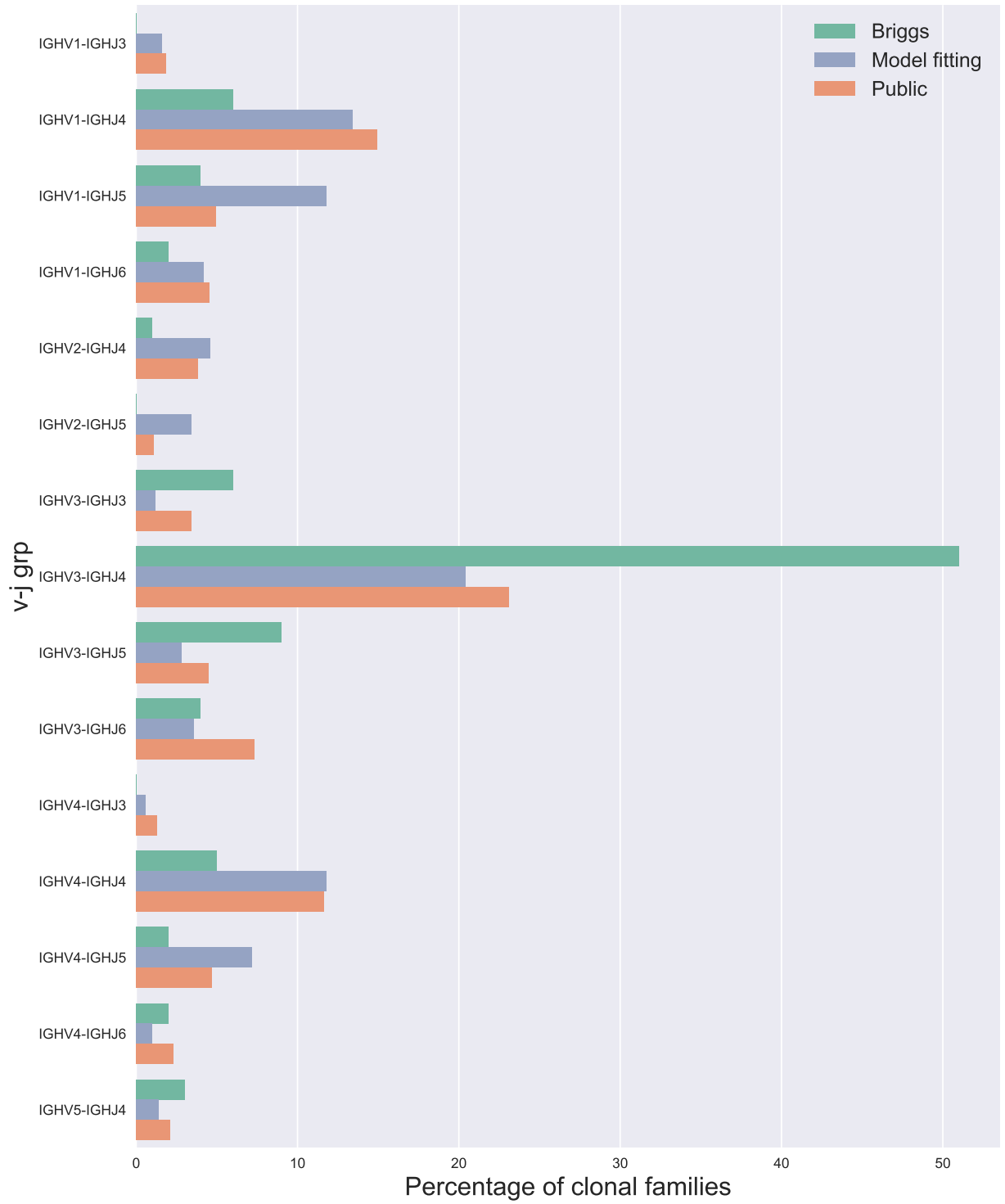


Figure 5.11: Distribution of per-clonal-family V/J subgroup combination usage in the different dataset partitions. Minimum frequency of 1% in either partition used as a cutoff for inclusion.

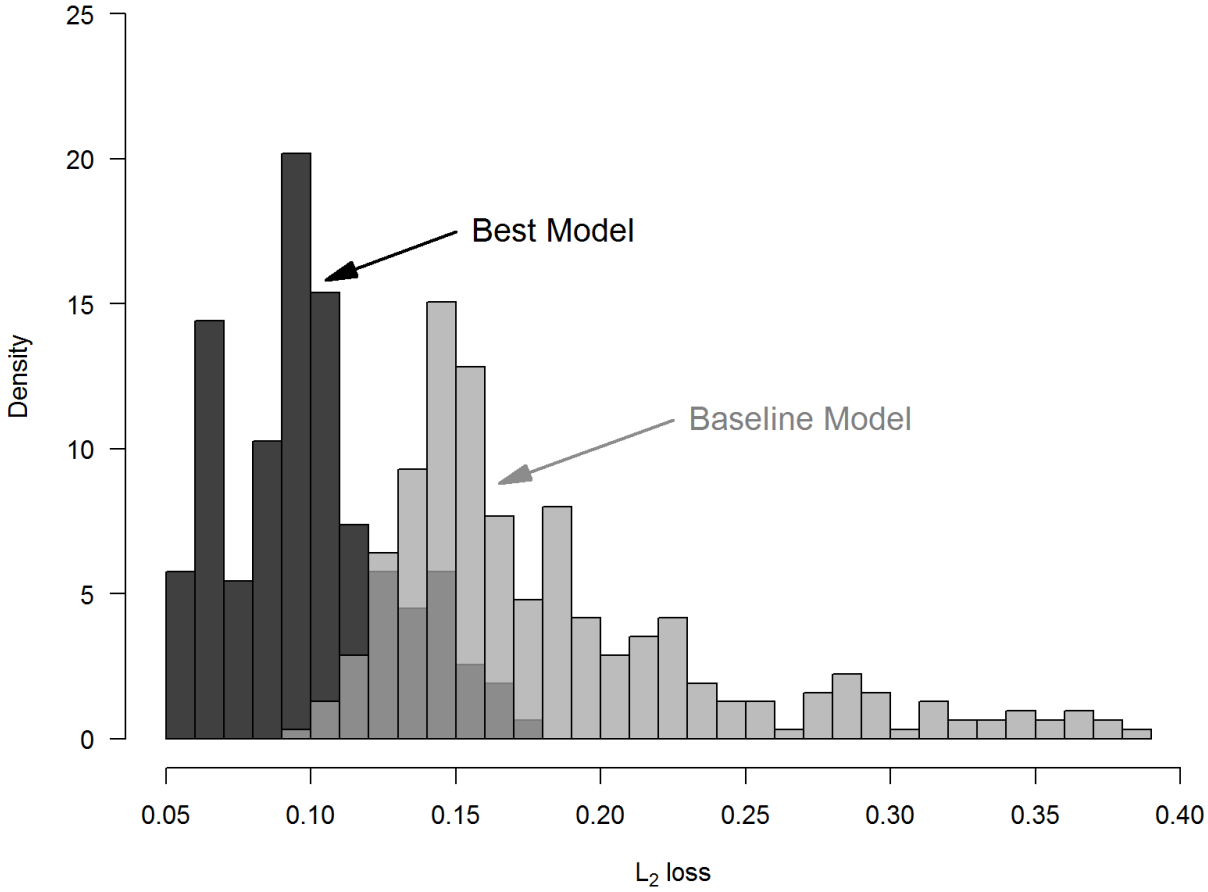


Figure 5.12: Two histograms showing L_2 loss estimates based on model predictions from the baseline model and best model for the Liao dataset (Liao et al., 2013). In this analysis, we made model predictions using each of the 312 sequences as the input sequence for our model.

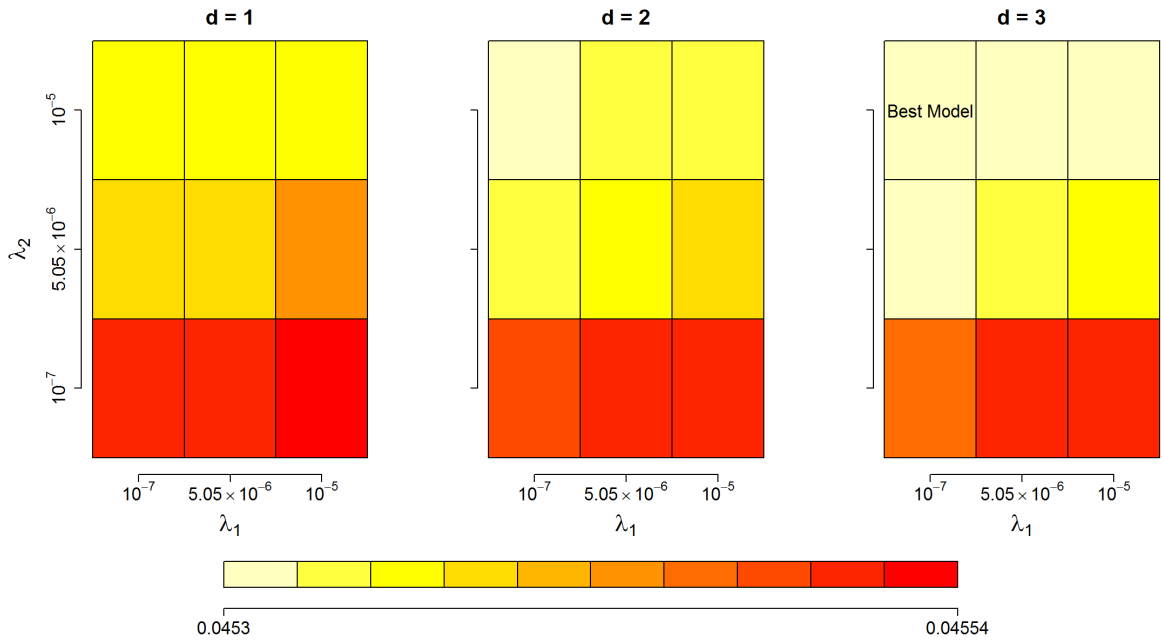


Figure 5.13: A heatmap showing 5-fold cross-validated L_2 loss results from fitting the regularized models for the $\left\{\widehat{\mathbf{X}}_{\text{naiveAA}}, \widehat{\mathbf{X}}_{\text{vgene}}, \widehat{\mathbf{X}}_{\text{neut}}, \widehat{\mathbf{X}}_{\text{vsubgrp}}\right\}$ profile grouping. We present unregularized L_2 loss estimates for tuning parameters $\lambda_1, \lambda_2 = 10^{-7}, 5.05 \times 10^{-6}, 10^{-5}$ and the order of differencing, $d = 1, 2, 3$, and mark the optimal tuning parameters found from our experiments.

Chapter 6

FUTURE DIRECTIONS

The work presented in this dissertation can be extended in several different directions. One obvious extension to the work in Chapter 3 is to generalize our post-order tree traversal algorithm to calculate higher-order moments of stochastic mapping summaries, such as the (co)skewness and (co)kurtosis. These higher-order moments could then be used to construct more complex discrepancy measures for posterior predictive model diagnostics. Similarly to the test of rate variation among sites, it is of interest to develop a posterior predictive approach to testing for rate variation among branches of the phylogeny. Posterior predictive tests of this type could serve as useful diagnostic tools to assess the appropriateness of relaxed molecular clock models (Drummond et al., 2006). The Bayesian phylo-HMM inference framework described in Chapter 4 is currently only applicable to heavy chain sequence data and modifying the procedure to operate on light chain data is one possible extension that is clearly within reach. For the regression model discussed in Chapter 5, we subsample single BCR sequences from CFs to use as model input; unfortunately, this means that our modeling analysis is conditional on a dataset that does not account for the variability associated with the subsampling process. One obvious means of fixing the above problem is to draw multiple subsamples from each CF and treat these multiple "observations" per CF within a dataset as a clustered data or weighted least squares problem.

BIBLIOGRAPHY

Aldous, D. (1996). Probability distributions on cladograms. In Random Discrete Structures, pages 1–18. Springer.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(3):269–342.

Ball, F. and Milne, R. K. (2005). Simple derivations of properties of counting processes associated with Markov renewal processes. Journal of Applied Probability, 42(4):1031–1043.

Birney, E., Stamatoyannopoulos, J. A., Dutta, A., Guigó, R., Gingeras, T. R., Margulies, E. H., Weng, Z., Snyder, M., Dermitzakis, E. T., Thurman, R. E., et al. (2007). Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. Nature, 447(7146):799–816.

Bloom, J. D., Raval, A., and Wilke, C. O. (2007). Thermodynamics of neutral protein evolution. Genetics, 175(1):255–266.

Boussau, B. and Gouy, M. (2006). Efficient likelihood computations with nonreversible models of evolution. Systematic Biology, 55(5):756–768.

Boyd, S. and Vandenberghe, L. (2004). Convex Optimization. Cambridge University Press.

Breden, F., Luning, E. P., Peters, B., Rubelt, F., Schramm, C., Busse, C., Vander, J. H., Christley, S., Bukhari, S., Thorogood, A., et al. (2017). Reproducibility and reuse of adaptive immune receptor repertoire data. Frontiers in Immunology, 8:1418–1418.

Briggs, A. W., Goldfless, S. J., Timberlake, S., Belmont, B. J., Clouser, C. R., Koppstein, D., Sok, D., Heiden, J. V. A., Tamminen, M. V., Kleinstein, S. H., Burton, D. R., Church, G. M., and Vigneault, F. (2017). Tumor-infiltrating immune repertoires captured by single-cell barcoding in emulsion. bioRxiv.

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. SIAM Journal on Scientific Computing, 16(5):1190–1208.

Cappé, O. and Moulines, E. (2005). Recursive computation of the score and observed information matrix in hidden Markov models. In IEEE/SP 13th Workshop on Statistical Signal Processing, 2005, pages 703–708. IEEE.

Casaz, P., Boucher, E., Wollacott, R., Pierce, B. G., Rivera, R., Sedic, M., Ozturk, S., Thomas Jr, W. D., and Wang, Y. (2014). Resolving self-association of a therapeutic antibody by formulation optimization and molecular approaches. mAbs, 6(6):1533–1539.

Chahwan, R., Edelmann, W., Scharff, M. D., and Roa, S. (2012). AIDing antibody diversity by error-prone mismatch repair. In Seminars in Immunology, volume 24, pages 293–300. Elsevier.

Clark, R. H., Latypov, R. F., De Imus, C., Carter, J., Wilson, Z., Manchulenko, K., Brown, M. E., and Ketchem, R. R. (2014). Remediating agitation-induced antibody aggregation by eradicating exposed hydrophobic motifs. mAbs, 6(6):1540–1550.

Courtois, F., Agrawal, N. J., Lauer, T. M., and Trout, B. L. (2016). Rational design of therapeutic mAbs against aggregation through protein engineering and incorporation of glycosylation motifs applied to bevacizumab. mAbs, 8(1):99–112.

Crooks, G. E., Hon, G., Chandonia, J.-M., and Brenner, S. E. (2004). WebLogo: a sequence logo generator. Genome Research, 14(6):1188–1190.

Cui, A., Di Niro, R., Vander Heiden, J. A., Briggs, A. W., Adams, K., Gilbert, T., O'Connor, K. C., Vigneault, F., Shlomchik, M. J., and Kleinstein, S. H. (2016). A model of somatic hypermutation targeting in mice based on high-throughput Ig sequencing data. The Journal of Immunology, 197(9):3566–3574.

Dimmic, M. W., Hubisz, M. J., Bustamante, C. D., and Nielsen, R. (2005). Detecting coevolving amino acid sites using Bayesian mutational mapping. Bioinformatics, 21(1):i126–i135.

Doria-Rose, N. A., Bhiman, J. N., Roark, R. S., Schramm, C. A., Gorman, J., Chuang, G.-Y., Pancera, M., Cale, E. M., Ernandes, M. J., Louder, M. K., et al. (2016). New member of the V1V2-directed CAP256-VRC26 lineage that shows increased breadth and exceptional potency. Journal of Virology, 90(1):76.

Doria-Rose, N. A., Schramm, C. A., Gorman, J., Moore, P. L., Bhiman, J. N., DeKosky, B. J., Ernandes, M. J., Georgiev, I. S., Kim, H. J., Pancera, M., et al. (2014). Developmental pathway for potent V1V2-directed HIV-neutralizing antibodies. Nature, 509(7498):55.

Drummond, A. J., Ho, S. Y. W., Phillips, M. J., and Rambaut, A. (2006). Relaxed phylogenetics and dating with confidence. PLoS Biology, 4(5):e88.

Dunbar, J. and Deane, C. M. (2015). ANARCI: antigen receptor numbering and receptor classification. Bioinformatics, 32(2):298–300.

Dunn-Walters, D. K., Dogan, A., Boursier, L., MacDonald, C. M., and Spencer, J. (1998). Base-specific sequences that bias somatic hypermutation deduced by analysis of out-of-frame human IgVH genes. The Journal of Immunology, 160(5):2360–2364.

Dutheil, J., Pupko, T., Jean-Marie, A., and Galtier, N. (2005). A model-based approach for detecting coevolving positions in a molecule. Molecular Biology and Evolution, 22(9):1919–1928.

Eddelbuettel, D. and François, R. (2011). Rcpp: Seamless R and C++ integration. Journal of Statistical Software, 40(8):1–18.

Eddelbuettel, D. and Sanderson, C. (2014). RcppArmadillo: Accelerating R with high-performance C++ linear algebra. Computational Statistics and Data Analysis, 71:1054–1063.

Ekeberg, M., Lövkvist, C., Lan, Y., Weigt, M., and Aurell, E. (2013). Improved contact prediction in proteins: using pseudolikelihoods to infer Potts models. Physical Review E, 87(1):012707.

Elhanati, Y., Sethna, Z., Marcou, Q., Callan Jr, C. G., Mora, T., and Walczak, A. M. (2015). Inferring processes underlying B-cell repertoire diversity. Philosophical Transactions of the Royal Society B: Biological Sciences, 370(1676).

Faulkner, J. R., Minin, V. N., et al. (2018). Locally adaptive smoothing with Markov random fields and shrinkage priors. Bayesian Analysis, 13(1):225–252.

Felsenstein, J. (1981). Evolutionary trees from DNA sequences: A maximum likelihood approach. Journal of Molecular Evolution, 17(6):368–376.

Felsenstein, J. (1989). PHYLIP-phylogeny inference package (version 3.2). Cladistics, 5:164–166.

Felsenstein, J. (2004). Inferring Phylogenies. Sinauer Associates, Sunderland, Massachusetts.

Felsenstein, J. (2005). PHYLIP (Phylogeny Inference Package) version 3.6. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle.

Feng, J., Shaw, D. A., Minin, V. N., Simon, N., Matsen, I., and Frederick, A. (2017). Survival analysis of DNA mutation motifs with penalized proportional hazards. arXiv preprint arXiv:1711.04057.

Flouri, T., Izquierdo-Carrasco, F., Darriba, D., Aberer, A., Nguyen, L., Minh, B., Von Haeseler, A., and Stamatakis, A. (2014). The phylogenetic likelihood library. Systematic Biology, 64(2):356–362.

Gansner, E. R. and North, S. C. (2000). An open graph visualization system and its applications to software engineering. Software: Practice and Experience, 30(11):1203–1233.

Gelman, A. and Meng, X.-L. (2004). Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives. John Wiley & Sons.

Gelman, A., Meng, X.-L., and Stern, H. (1996). Posterior predictive assessment of model fitness via realized discrepancies. Statistica Sinica, 6:733–807.

Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). Bayesian Data Analysis. Chapman and Hall/CRC.

Geoghegan, J. C., Fleming, R., Damschroder, M., Bishop, S. M., Sathish, H. A., and Esfandiary, R. (2016). Mitigation of reversible self-association and viscosity in a human IgG1 monoclonal antibody by rational, structure-guided Fv engineering. mAbs, 8(5):941–950.

Gong, L. I., Suchard, M. A., and Bloom, J. D. (2013). Stability-mediated epistasis constrains the evolution of an influenza protein. eLife, 2:e00631.

Gordon, N. J., Salmond, D. J., and Smith, A. F. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In IEE Proceedings F (Radar and Signal Processing), volume 140, pages 107–113. IET.

Guindon, S., Dufayard, J.-F., Lefort, V., Anisimova, M., Hordijk, W., and Gascuel, O. (2010). New algorithms and methods to estimate maximum-likelihood phylogenies: Assessing the performance of PhyML 3.0. Systematic Biology, 59(3):307–321.

Guindon, S., Rodrigo, A. G., Dyer, K. A., and Huelsenbeck, J. P. (2004). Modeling the site-specific variation of selection patterns along lineages. Proceedings of the National Academy of Sciences of the United States of America, 101(35):12957–12962.

Gupta, N. T., Adams, K. D., Briggs, A. W., Timberlake, S. C., Vigneault, F., and Kleinstein,

S. H. (2017). Hierarchical clustering can identify B cell clones with high confidence in Ig repertoire sequencing data. The Journal of Immunology, 198(6):2489–2499.

Hanson-Smith, V., Kolaczkowski, B., and Thornton, J. W. (2010). Robustness of ancestral sequence reconstruction to phylogenetic uncertainty. Molecular Biology and Evolution, 27(9):1988–1999.

Harding, F. A., Stickler, M. M., Razo, J., and DuBridge, R. (2010). The immunogenicity of humanized and fully human antibodies: residual immunogenicity resides in the CDR regions. mAbs, 2(3):256–265.

Hartigan, J. A. and Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1):100–108.

Hastie, T., Tibshirani, R., and Tibshirani, R. J. (2017). Extended comparisons of best subset selection, forward stepwise selection, and the lasso. arXiv preprint arXiv:1707.08692.

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

Hobolth, A. and Jensen, J. L. (2005). Statistical inference in evolutionary models of DNA sequences via the EM algorithm. Statistical Applications in Genetics and Molecular Biology, 4(1):1–22.

Hoehn, K. B., Lunter, G., and Pybus, O. G. (2017). A phylogenetic codon substitution model for antibody lineages. Genetics, 206(1):417–427.

Höhna, S., Landis, M. J., Heath, T. A., Boussau, B., Lartillot, N., Moore, B. R., Huelsenbeck, J. P., and Ronquist, F. (2016). RevBayes: Bayesian phylogenetic inference using graphical models and an interactive model-specification language. Systematic Biology, 65(4):726–736.

Holmes, I. and Rubin, G. M. (2002). An expectation maximization algorithm for training hidden substitution models. Journal of Molecular Biology, 317(5):753–764.

Honegger, A. and PluÈckthun, A. (2001). Yet another numbering scheme for immunoglobulin variable domains: an automatic modeling and analysis tool. Journal of Molecular Biology, 309(3):657–670.

Huelsenbeck, J. P., Nielsen, R., and Bollback, J. P. (2003). Stochastic mapping of morphological characters. Systematic Biology, 52(2):131–158.

Huelsenbeck, J. P. and Ronquist, F. (2001). MRBAYES: Bayesian inference of phylogenetic trees. Bioinformatics, 17(8):754–755.

Igawa, T., Tsunoda, H., Kuramochi, T., Sampei, Z., Ishii, S., and Hattori, K. (2011). Engineering the variable region of therapeutic IgG antibodies. mAbs, 3(3):243–252.

Jaccard, P. (1912). The distribution of the flora in the alpine zone. New Phytologist, 11(2):37–50.

Jones, D. T., Buchan, D. W., Cozzetto, D., and Pontil, M. (2011). PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. Bioinformatics, 28(2):184–190.

Kellis, M., Patterson, N., Endrizzi, M., Birren, B., and Lander, E. S. (2003). Sequencing and comparison of yeast species to identify genes and regulatory elements. Nature, 423(6937):241–254.

Kenney, T. and Gu, H. (2012). Hessian calculation for phylogenetic likelihood based on the pruning algorithm and its applications. Statistical Applications in Genetics and Molecular Biology, 11(4).

Kepler, T. B. (2013). Reconstructing a B-cell clonal lineage. I. statistical inference of unobserved ancestors. F1000Research, 2(103).

Kepler, T. B., Munshaw, S., Wiehe, K., Zhang, R., Yu, J.-S., Woods, C. W., Denny, T. N., Tomaras, G. D., Alam, S. M., Moody, M. A., Kelsoe, G., Liao, H.-X., and Haynes, B. F. (2014). Reconstructing a b-cell clonal lineage. ii. mutation, selection, and affinity maturation. Frontiers in Immunology, 5:170.

Kirik, U., Persson, H., Levander, F., Greiff, L., and Ohlin, M. (2017). Antibody heavy chain variable domains of different germline gene origins diversify through different paths. Frontiers in Immunology, 8.

Kuraoka, M., Schmidt, A. G., Nojima, T., Feng, F., Watanabe, A., Kitamura, D., Harrison, S. C., Kepler, T. B., and Kelsoe, G. (2016). Complex antigens drive permissive clonal selection in germinal centers. Immunity, 44(3):542–552.

Kyung, M., Gill, J., Ghosh, M., Casella, G., et al. (2010). Penalized regression, standard errors, and Bayesian lassos. Bayesian Analysis, 5(2):369–411.

Landais, E., Huang, X., Havenar-Daughton, C., Murrell, B., Price, M. A., Wickramasinghe, L., Ramos, A., Bian, C. B., Simek, M., Allen, S., et al. (2016). Broadly neutralizing antibody responses in a large longitudinal sub-saharan HIV primary infection cohort. PLoS Pathogens, 12(1):e1005369.

Landais, E., Murrell, B., Briney, B., Murrell, S., Rantalainen, K., Berndsen, Z. T., Ramos, A., Wickramasinghe, L., Smith, M. L., Eren, K., et al. (2017). HIV envelope glycoform heterogeneity and localized diversity govern the initiation and maturation of a V2 apex broadly neutralizing antibody lineage. Immunity, 47(5):990–1003.

Laserson, U., Vigneault, F., Gadala-Maria, D., Yaari, G., Uduman, M., Vander Heiden, J. A., Kelton, W., Jung, S. T., Liu, Y., Laserson, J., et al. (2014). High-resolution antibody dynamics of vaccine-induced immune responses. Proceedings of the National Academy of Sciences, 111(13):4928–4933.

Lauritzen, S. L. (1996). Graphical Models, volume 17. Clarendon Press.

Lefranc, M.-P. (2001). Nomenclature of the human immunoglobulin heavy (IGH) genes. Experimental and Clinical Immunogenetics, 18(2):100–116.

Lemey, P., Minin, V. N., Bielejec, F., Pond, S. L. K., and Suchard, M. A. (2012). A counting renaissance: combining stochastic mapping and empirical Bayes to quickly detect amino acid sites under positive selection. Bioinformatics, 28(24):3248–3256.

Li, W. and Godzik, A. (2006). Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics, 22(13):1658–1659.

Liao, H.-X., Lynch, R., Zhou, T., Gao, F., Alam, S. M., Boyd, S. D., Fire, A. Z., Roskin, K. M., Schramm, C. A., Zhang, Z., et al. (2013). Co-evolution of a broadly neutralizing HIV-1 antibody and founder virus. Nature, 496(7446):469.

Liu, K., Warnow, T. J., Holder, M. T., Nelesen, S. M., Yu, J., Stamatakis, A. P., and Linder, C. R. (2012). SATè-II: Very fast and accurate simultaneous estimation of multiple sequence alignments and phylogenetic trees. Systematic Biology, 61(1):90–106.

Loman, N. J., Misra, R. V., Dallman, T. J., Constantinidou, C., Gharbia, S. E., Wain, J., and Pallen, M. J. (2012). Performance comparison of benchtop high-throughput sequencing platforms. Nature biotechnology, 30(5):434–439.

Lystig, T. C. and Hughes, J. P. (2002). Exact computation of the observed information matrix for hidden Markov models. Journal of Computational and Graphical Statistics, 11(3):678–689.

Mascola, J. R. and Haynes, B. F. (2013). HIV-1 neutralizing antibodies: understanding nature's pathways. Immunological Reviews, 254(1):225–244.

McConnell, A. D., Zhang, X., Macomber, J. L., Chau, B., Sheffer, J. C., Rahmanian, S., Hare, E., Spasojevic, V., Horlick, R. A., King, D. J., et al. (2014). A general approach to antibody thermostabilization. mAbs, 6(5):1274–1282.

Meng, W., Zhang, B., Schwartz, G. W., Rosenfeld, A. M., Ren, D., Thome, J. J., Carpenter, D. J., Matsuoka, N., Lerner, H., Friedman, A. L., et al. (2017). An atlas of B-cell clonal distribution in the human body. Nature Biotechnology, 35(9):879.

Meng, X.-L. (1994). Posterior predictive p-values. The Annals of Statistics, 22(3):1142–1160.

Methot, S. and Di Noia, J. (2017). Molecular mechanisms of somatic hypermutation and class switch recombination. In Advances in Immunology, volume 133, pages 37–87. Elsevier.

Minin, V. N. and Suchard, M. A. (2008a). Counting labeled transitions in continuous-time Markov models of evolution. Journal of Mathematical Biology, 56(3):391–412.

Minin, V. N. and Suchard, M. A. (2008b). Fast, accurate and simulation-free stochastic mapping. Philosophical Transactions of the Royal Society of London B: Biological Sciences, 363(1512):3985–3995.

Mood, A. M., Graybill, F. A., and Boes, D. C. (1974). Introduction to the Theory of Statistics. International Student edition. McGraw-Hill.

Murphy, K. and Weaver, C. (2016). Janeway's Immunobiology. Garland Science/Taylor & Francis Group, LLC.

Neuts, M. F. (1995). Algorithmic Probability: A Collection of Problems. Stochastic Modeling Series. Taylor & Francis.

Nielsen, R. (2002). Mapping mutations on phylogenies. Systematic Biology, 51(5):729–739.

Nielsen, R. and Yang, Z. (1998). Likelihood models for detecting positively selected amino acid sites and applications to the HIV-1 envelope gene. Genetics, 148(3):929–936.

Page, R. D. and Holmes, E. C. (2009). Molecular Evolution: A Phylogenetic Approach. John Wiley & Sons.

Pagel, M. (1999). The maximum likelihood approach to reconstructing ancestral character states of discrete characters on phylogenies. Systematic Biology, 48(3):612–622.

Park, T. and Casella, G. (2008). The Bayesian lasso. Journal of the American Statistical Association, 103(482):681–686.

Pollard, K. S., Hubisz, M. J., Rosenbloom, K. R., and Siepel, A. (2010). Detection of nonneutral substitution rates on mammalian phylogenies. Genome Research, 20(1):110–121.

Pollard, K. S., Salama, S. R., King, B., Kern, A. D., Dreszer, T., Katzman, S., Siepel, A., Pedersen, J. S., Bejerano, G., Baertsch, R., et al. (2006a). Forces shaping the fastest evolving regions in the human genome. PLoS Genetics, 2(10):e168.

Pollard, K. S., Salama, S. R., Lambert, N., Lambot, M.-A., Coppens, S., Pedersen, J. S., Katzman, S., King, B., Onodera, C., Siepel, A., et al. (2006b). An RNA gene expressed during cortical development evolved rapidly in humans. Nature, 443(7108):167–172.

Price, M. N., Dehal, P. S., and Arkin, A. P. (2009). FastTree: computing large minimum evolution trees with profiles instead of a distance matrix. Molecular Biology and Evolution, 26(7):1641–1650.

Price, M. N., Dehal, P. S., and Arkin, A. P. (2010). FastTree 2–approximately maximum-likelihood trees for large alignments. PLoS One, 5(3):e9490.

Rabiner, L. R. (1986). An introduction to hidden Markov models. IEEE ASSP Magazine, 3(1):4–16.

Ralph, D. K. and Matsen IV, F. A. (2016a). Consistency of VDJ rearrangement and substitution parameters enables accurate B cell receptor sequence annotation. PLoS Computational Biology, 12(1).

Ralph, D. K. and Matsen IV, F. A. (2016b). Likelihood-based inference of B cell clonal families. PLoS Computational Biology, 12(10):e1005086.

Redelings, B. D. and Suchard, M. A. (2005). Joint bayesian estimation of alignment and phylogeny. Systematic Biology, 54(3):401–418.

Rogozin, I. B. and Kolchanov, N. A. (1992). Somatic hypermutagenesis in immunoglobulin genes: II. influence of neighbouring base sequences on mutagenesis. Biochimica et Biophysica Acta (BBA)-Gene Structure and Expression, 1171(1):11–18.

Rubelt, F., Busse, C. E., Bukhari, S. A. C., Bürckert, J.-P., Mariotti-Ferrandiz, E., Cowell, L. G., Watson, C. T., Marthandan, N., Faison, W. J., Hershberg, U., et al. (2017). Adaptive immune receptor repertoire community recommendations for sharing immune-repertoire sequencing data. Nature Immunology, 18(12):1274.

Rubin, D. B. (1987). The calculation of posterior distributions by data augmentation: Comment: A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The SIR algorithm. Journal of the American Statistical Association, 82(398):543–546.

Schadt, E. E., Sinsheimer, J. S., and Lange, K. (1998). Computational advances in maximum likelihood methods for molecular phylogeny. Genome Research, 8(3):222–233.

Schmidt, A. G., Xu, H., Khan, A. R., O'Donnell, T., Khurana, S., King, L. R., Manischewitz, J., Golding, H., Suphaphiphat, P., Carfi, A., et al. (2013). Preconfiguration of the antigen-binding site during affinity maturation of a broadly neutralizing influenza virus antibody. Proceedings of the National Academy of Sciences, 110(1):264–269.

Scott, S. L. (2002). Bayesian methods for hidden Markov models: Recursive computing in the 21st century. Journal of the American Statistical Association, 97(457):337–351.

Seeliger, D., Schulz, P., Litzenburger, T., Spitz, J., Hoerer, S., Blech, M., Enenkel, B., Studts, J. M., Garidel, P., and Karow, A. R. (2015). Boosting antibody developability through rational sequence optimization. mAbs, 7(3):505–515.

Sheng, Z., Schramm, C. A., Connors, M., Morris, L., Mascola, J. R., Kwong, P. D., and Shapiro, L. (2016). Effects of darwinian selection and mutability on rate of broadly neutralizing antibody evolution during HIV-1 infection. PLoS Computational Biology, 12(5):e1004940.

Sheng, Z., Schramm, C. A., Kong, R., Mullikin, J. C., Mascola, J. R., Kwong, P. D., Shapiro, L., Program, N. C. S., et al. (2017). Gene-specific substitution profiles describe the types and frequencies of amino acid changes during antibody somatic hypermutation. Frontiers in Immunology, 8.

Shugay, M., Britanova, O. V., Merzlyak, E. M., Turchaninova, M. A., Mamedov, I. Z., Tuganbaev, T. R., Bolotin, D. A., Staroverov, D. B., Putintseva, E. V., Plevova, K., et al. (2014). Towards error-free profiling of immune repertoires. Nature Methods, 11(6):653–655.

Siepel, A., Bejerano, G., Pedersen, J. S., Hinrichs, A. S., Hou, M., Rosenbloom, K., Clawson, H., Spieth, J., Hillier, L. W., Richards, S., et al. (2005). Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes. Genome Research, 15(8):1034–1050.

Siepel, A. and Haussler, D. (2005). Phylogenetic hidden Markov models. In Statistical Methods in Molecular Evolution, pages 325–351. Springer.

Siepel, A., Pollard, K. S., and Haussler, D. (2006). New methods for detecting lineage-specific selection. In Research in Computational Molecular Biology: 10th Annual International Conference, RECOMB 2006, Venice, Italy, April 2-5, 2006, Proceedings, volume 3909, pages 190–205. Springer.

Simonich, C. A., Doepker, L., Ralph, D., Williams, J. A., Dhar, A., Yaffe, Z., Gentles, L., Small, C. T., Oliver, B., Vigdorovich, V., et al. (2018). Rapid development of an infant-derived HIV-1 broadly neutralizing antibody lineage. BioRxiv, page 416032.

Skare, Ø., Bølviken, E., and Holden, L. (2003). Improved sampling-importance resampling and reduced bias importance sampling. Scandinavian Journal of Statistics, 30(4):719–737.

Smith, A. F. and Gelfand, A. E. (1992). Bayesian statistics without tears: a sampling–resampling perspective. The American Statistician, 46(2):84–88.

Sompayrac, L. (2015). How the Immune System Works. The How it Works Series. Wiley.

Stamatakis, A. (2014). RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. Bioinformatics, 30(9):1312–1313.

Stamatatos, L., Pancera, M., and McGuire, A. T. (2017). Germline-targeting immunogens. Immunological Reviews, 275(1):203–216.

Stern, J. N., Yaari, G., Vander Heiden, J. A., Church, G., Donahue, W. F., Hintzen, R. Q., Huttner, A. J., Laman, J. D., Nagra, R. M., Nylander, A., et al. (2014). B cells populating the multiple sclerosis brain mature in the draining cervical lymph nodes. Science Translational Medicine, 6(248):248ra107–248ra107.

Tas, J. M., Mesin, L., Pasqual, G., Targ, S., Jacobsen, J. T., Mano, Y. M., Chen, C. S., Weill, J.-C., Reynaud, C.-A., Browne, E. P., Meyer-Hermann, M., and Victora, G. D. (2016). Visualizing antibody affinity maturation in germinal centers. Science.

Tavaré, S. (1986). Some probabilistic and statistical problems in the analysis of DNA sequences. Lectures on Mathematics in the Life Sciences, 17:57–86.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), 58(1):267–288.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005). Sparsity and smoothness via the fused lasso. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(1):91–108.

Tibshirani, R. J. et al. (2014). Adaptive piecewise polynomial estimation via trend filtering. The Annals of Statistics, 42(1):285–323.

Tomlinson, I. M., Cox, J., Gherardi, E., Lesk, A., and Chothia, C. (1995). The structural repertoire of the human v kappa domain. The EMBO journal, 14(18):4628–4638.

Tsioris, K., Gupta, N. T., Ogunniyi, A. O., Zimnisky, R. M., Qian, F., Yao, Y., Wang, X., Stern, J. N., Chari, R., Briggs, A. W., et al. (2015). Neutralizing antibodies against west nile virus identified directly from human B cells by single-cell analysis and next generation sequencing. Integrative Biology, 7(12):1587–1597.

Turchaninova, M. A., Davydov, A., Britanova, O. V., Shugay, M., Bikos, V., Egorov, E. S., Kirgizova, V. I., Merzlyak, E. M., Staroverov, D. B., Bolotin, D. A., Mamedov, I. Z., Izraelson, M., Logacheva, M. D., Kladova, O., Plevova, K., Pospisilova, S., and Chudakov, D. M. (2016). High-quality full-length immunoglobulin profiling with unique molecular barcoding. Nat. Protoc., 11(9):1599–1616.

van der Vaart, A. (1998). Asymptotic Statistics. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.

Vander Heiden, J. A., Stathopoulos, P., Zhou, J. Q., Chen, L., Gilbert, T. J., Bolen, C. R., Barohn, R. J., Dimachkie, M. M., Ciafaloni, E., Broering, T. J., et al. (2017). Dysregulation of B cell repertoire formation in myasthenia gravis patients revealed through deep sequencing. The Journal of Immunology, 198(4):1460–1473.

Vander Heiden, J. A., Yaari, G., Uduman, M., Stern, J. N., O'Connor, K. C., Hafler, D. A., Vigneault, F., and Kleinstein, S. H. (2014). pRESTO: a toolkit for processing

high-throughput sequencing raw reads of lymphocyte receptor repertoires. Bioinformatics, 30(13):1930–1932.

Victora, G. D. and Nussenzweig, M. C. (2012). Germinal centers. Annual Review of Immunology, 30(1):429–457.

Wang, N., Smith, W. F., Miller, B. R., Aivazian, D., Lugovskoy, A. A., Reff, M. E., Glaser, S. M., Croner, L. J., and Demarest, S. J. (2009). Conserved amino acid networks involved in antibody variable domain interactions. Proteins: Structure, Function, and Bioinformatics, 76(1):99–114.

Watson, C. T., Glanville, J., and Marasco, W. A. (2017). The individual and population genetics of antibody immunity. Trends Immunol., 38(7):459–470.

West, A. P., Diskin, R., Nussenzweig, M. C., and Bjorkman, P. J. (2012). Structural basis for germ-line gene usage of a potent class of antibodies targeting the CD4-binding site of HIV-1 gp120. Proceedings of the National Academy of Sciences, 109(30):E2083–E2090.

Wu, X., Zhang, Z., Schramm, C. A., Joyce, M. G., Do Kwon, Y., Zhou, T., Sheng, Z., Zhang, B., O'Dell, S., McKee, K., et al. (2015). Maturation and diversity of the VRC01-antibody lineage over 15 years of chronic HIV-1 infection. Cell, 161(3):470–485.

Wu, X., Zhou, T., Zhu, J., Zhang, B., Georgiev, I., Wang, C., Chen, X., Longo, N. S., Louder, M., McKee, K., et al. (2011). Focused evolution of HIV-1 neutralizing antibodies revealed by structures and deep sequencing. Science, 333(6049):1593–1602.

Yaari, G., Vander Heiden, J., Uduman, M., Gadala-Maria, D., Gupta, N., Stern, J., O'Connor, K., Hafler, D., Laserson, U., Vigneault, F., and Kleinstein, S. (2013). Models of somatic hypermutation targeting and substitution based on synonymous mutations from high-throughput immunoglobulin sequencing data. Frontiers in Immunology, 4:358.

Yang, Z. (1994). Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: Approximate methods. Journal of Molecular Evolution, 39(3):306–314.

Yang, Z. (1996). Among-site rate variation and its impact on phylogenetic analyses. Trends in Ecology & Evolution, 11(9):367–372.

Yang, Z. (2006). Computational Molecular Evolution, volume 284 of Oxford Series in Ecology and Evolution. Oxford University Press.

Zhai, W., Slatkin, M., and Nielsen, R. (2007). Exploring variation in the d(N)/d(S) ratio among sites and lineages using mutational mappings: applications to the influenza virus. Journal of Molecular Evolution, 65(3):340–348.

Zheng, Q. (2001). On the dispersion index of a Markovian molecular clock. Mathematical Biosciences, 172(2):115–128.

Zhou, T., Zhu, J., Wu, X., Moquin, S., Zhang, B., Acharya, P., Georgiev, I. S., Altae-Tran, H. R., Chuang, G.-Y., Joyce, M. G., et al. (2013). Multidonor analysis reveals structural elements, genetic determinants, and maturation pathway for HIV-1 neutralization by VRC01-class antibodies. Immunity, 39(2):245–258.