CONSISTENCY OF VDJ REARRANGEMENT AND SUBSTITUTION PARAMETERS ENABLES ACCURATE B CELL RECEPTOR SEQUENCE ANNOTATION

DUNCAN RALPH AND FREDERICK A. MATSEN IV

ABSTRACT. VDJ rearrangement and somatic hypermutation work together to produce antibody-coding B cell receptor (BCR) sequences for a remarkable diversity of antigens. It is now possible to sequence these BCRs in high throughput, and analysis of these sequences is bringing new insight into how antibodies develop, in particular for broadly-neutralizing antibodies against HIV and influenza. A fundamental step in such sequence analysis is to annotate each base as coming from a specific one of the V, D, or J genes, or from a non-templated insertion. Previous work has used simple parametric distributions to model transitions from state to state in a hidden Markov model (HMM) of VDJ recombination, and assumed that mutations occur via the same process across sites. However, codon frame and other effects have been observed to violate these parametric assumptions for such coding sequences, suggesting that a non-parametric approach to modeling the recombination process could be useful. In our paper, we find that indeed large modern data sets suggest a model using parameter-rich per-allele categorical distributions for HMM transition probabilities and per-allele-per-position mutation probabilities, and that using such a model for inference leads to significantly improved results. We present an accurate and efficient BCR sequence annotation software package using a novel HMM “factorization” strategy. This package, called partis [https://github.com/psathyrella/partis/], is built on a new general-purpose HMM compiler that can perform efficient inference given a simple text description of an HMM.

SUMMARY STATEMENT

The binding properties of antibodies are determined by the sequences of their corresponding B cell receptors (BCRs). These BCR sequences are created in “draft” form by VDJ recombination, which randomly selects and deletes from the ends of
V, D, and J genes, then joins them together with additional random nucleotides. If they pass initial screening and bind an antigen, these sequences then undergo an evolutionary process of mutation and selection, “revising” the BCR to improve binding to its cognate antigen. It has recently become possible to determine these antibody-determining BCR sequences resulting from this process in high throughput. Although these sequences implicitly contain a wealth of information about both antigen exposure and the process by which we learn to resist pathogens, this information can only be extracted using computer algorithms. In this paper, we employ a computational and statistical approach to learn about the VDJ recombination process. Using a large data set, we find consistent and detailed patterns in the parameters, such as amount of V gene deletion, for this process. We can then use this parameter rich model to perform more accurate per-sequence attribution of each nucleotide to either a V, D, or J gene, or a non-templated insertion.

**INTRODUCTION**

The molecular sequences of B and T cell receptors (BCRs and TCRs) determine what antigens will be recognized by these lymphocytes, with B cells recognizing antigens via immunoglobulins and T cells recognizing antigenic peptides presented by the major histocompatibility complex. Together, the molecular sequences of BCRs and TCRs encode immunological memory against a great diversity of antigens. Diversity in these receptor loci is generated first by the process of VDJ recombination, in which germline-encoded V, D, and J genes are randomly selected, the ends of the genes are deleted by some amount, and then joined together with random non-templated insertions (Figure 1). BCR sequences diversify further through the Darwinian process of somatic hypermutation and clonal selection.

In the about 35 years since the discovery of VDJ recombination by Susumu Tonegawa, sequencing assays have been widely applied to BCRs and TCRs to further our understanding of this process. Recently, high throughput sequencing
has given a remarkable perspective on the forces determining the immunoglobulin repertoire \(^{[1-4]}\). In addition to advancing basic science understanding, such sequencing is being applied to learn how antibodies develop against antigens of medical interest, such as through influenza vaccination \(^{[5]}\). There have also been spectacular advances using this technology to understand the ontogeny of HIV broadly neutralizing antibodies, with the still-elusive goal of eliciting them with an HIV vaccine \(^{[6-9]}\).

A fundamental step in the analysis of such a sequencing data set is to reconstruct the origin of each nucleotide in each sequence: whether it came from a non-templated insertion or from a germline V, D, or J gene, and if so, which one and where. Even if a complete collection of alleles for the germline V, D, and J genes were available, this problem would be challenging because deletion obscures the boundaries between insertions and germline V, D, and J gene sequences. The BCR case is made more difficult by the processes of somatic hypermutation and clonal selection: if a BCR sequence does not match a germline gene in the area of an insertion-deletion boundary, it may have come from a mutated germline position or a non-templated insertion. We will call this general problem of describing the source of each nucleotide in a BCR (or TCR) sequence the BCR (or TCR) annotation problem. Here we will focus on the more challenging BCR variant of the problem.

One approach is to leverage general-purpose tools for doing pairwise sequence search such as BLAST \(^{[10]}\) and Smith-Waterman local alignment \(^{[11]}\) for the annotation problem. Adding BCR-specific aspects to these basic algorithms has resulted in a collection of useful tools such as IgBLAST \(^{[12]}\) and the online annotation tool on the IMGT \(^{[13]}\) website.

However, BCR sequence formation is quite complex (reviewed in \(^{[3]}\)) and this complexity invites a modeling-based approach, specifically in the framework of hidden Markov models (HMMs). HMMs for sequence analysis consist of a directed graph on “hidden state” nodes with defined start and end states, with each
node potentially “emitting” a nucleotide base or amino acid residue [14, 15]. In the BCR case, the hidden states represent either (gene, nucleotide position) pairs or non-templated insertion, and the emission probabilities incorporate the probability of somatic hypermutation at that base. The HMM approach to BCR annotation has been elegantly implemented first in SODA [16], then iHMMunealign [17], and then SODA2 [18]. The transition probabilities for these HMM methods are modeled parametrically: specifically, they use the negative binomial distribution as first used in [19], and the emission probabilities come from the same mutation process across positions (even if the process is context-dependent) [17].

High throughput sequencing has become commonplace since these HMMs were designed, which offers both a challenge and an opportunity for model-based approaches. It is a challenge because millions of distinct sequences are now available from a single sample of B cells, so methods must be efficient. On the other hand, it is an opportunity because such large data sets offer the opportunity to develop and fit models with much more detail.

**Figure 1.** The VDJ recombination process, in which individual V, D, and J genes are randomly selected from a number of copies of each. These genes are then joined together via a process that deletes some randomly distributed number of nucleotides on their boundaries then joins them together with random “non-templated insertions” (N, blue). The specificity of an antibody is primarily determined by the region defined by the heavy chain recombination site, referred to as the third complementarity determining region (CDR3).
We hypothesized that large data sets would reveal reproducible fine-scale details in the probabilistic rearrangement process that could be used for improved inference. In previous foundational work, researchers have used probability distributions such as the negative binomial \cite{19} to model deletion lengths, and modeled the propensity for somatic hypermutation based on sequence context \cite{17}. However, BCR sequences are protein-coding, and thus there are constraints on insertions and deletions that come from sequence frame; these types of constraints cannot be expressed by unimodal probability distributions with few parameters. For example, the D gene is preferentially (though not exclusively) used in a specific frame post-selection \cite{2,20}, which in some cases is simply due to a stop codon being present in a specific frame \cite{3}. This suggests that a parameter-rich approach could be useful: instead of parametric distributions with a few parameters, using a large data set we could fit a per-allele categorical distribution for insertion and deletion lengths. Rather than modeling the process of somatic hypermutation with germline nucleotide context, we could simply infer the per-position per-allele mutation frequency and use this as the mutation probability.

In order to meet the challenge and opportunity of large scale data, we have built \texttt{partis}: a fast, flexible, and open source HMM framework to analyze BCR sequences. In order to process large data sets, we started by writing an efficient new HMM compiler \texttt{ham} which enables inference on an arbitrary HMM specified via a simple text file rather than having to write special-purpose computer code. We then developed an HMM “factorization” strategy, which along with extensive caching and optimizations in the overall model topology, results in an order of magnitude faster execution than previous HMM implementations for BCR/TCR sequences. This work, along with parallelization, means that the largest data sets available today are comfortably within the capabilities of \texttt{partis} running on standard hardware.

We find that HMM inference using a non-parametric approach to fitting HMM parameter distributions outperforms previous approaches. The parameters for the
FIGURE 2. Observed deletion frequencies for two V, four D, and two J alleles on the Adaptive data set. These alleles were chosen to be representative of the various shapes taken by the empirical distributions. In the complete set of plots (which are publicly available as described in the text), per-allele distributions are frequently multi-modal and appear similar between humans.
non-parametric transition and emission probabilities are fit “on the fly” for each individual data set, using a tiered aggregation strategy to scale the level of model detail to the amount of data. The accuracy of partis is further increased by using multiple sequences from the same rearrangement event (which differ due to somatic hypermutation) using pair-HMMs and their generalization to more than two emission sequences, which we will call “multi-HMMs”. We also leverage the full power of HMMs by not only calculating the best (known as Viterbi) annotations of a sequence according to that HMM, but also computing forward probabilities for single and multi-HMMs. These forward probabilities allow one to integrate out uncertainty in quantities that are not of interest, retaining more accurate estimates (with uncertainty) of those parameters that are of interest. The partis software implementation has been engineered to be extensible and maintainable: it is open source and includes continuous integration and reproducible validation using the Docker software container system [21].

Results

Empirical distributions of VDJ recombination process parameters deviate reproducibly from simple distributions. We find that the probability distributions of rearrangement parameters deviate reproducibly from any commonly used distribution. For example, our inferred deletion length distributions take on a variety of shapes (Figure 2), which vary from allele to allele but appear consistent across humans (see also Figure S1). Mutation rates vary by an order of magnitude from position to position, creating a pattern which is unique to each allele but similar across humans for that allele (Figure 3 and S2). The overall level of mutation, however, varies between humans in our data sets (as noted previously [22]). Insertion lengths, on the other hand, vary much less across alleles, and are typically unimodal (Figure 4 and S3). The full collection of plots is available on Dryad at [http://datadryad.org/review?doi=doi:10.5061/dryad.149m8](http://datadryad.org/review?doi=doi:10.5061/dryad.149m8)
**Figure 3.** Typical observed mutation frequencies for two V, two D, and two J alleles on the Adaptive data set. Mutation frequencies are highly position-dependent. While the structure of these mutation distributions appears similar between humans, the overall level of mutation varies. The first base of the conserved cysteine and tryptophan codons (i.e. the CDR3 boundaries) are indicated with black vertical dashed lines. In the complete set of plots (which are publicly available as described in the text), mutation frequencies are highly variable across sites with a pattern that is similar between humans.

In these plots, the error bars are constructed using a bootstrapping procedure: we divide unique sequences in the data into ten subsets and plot the uncertainty as the standard deviation over these subsets (see Methods).
By considering all of the parameter estimates together (Figures 5 and S4), we find that estimates are consistent between sequence subsets. That these parameter estimates are consistent between disjoint subsets of a collection of unique sequences, and to a lesser extent consistent between humans, indicates that they reflect biology rather than noise. Although this suggestion of consistency between humans is interesting, our strategy of per-human parameter fitting does not require any such assumption, and so we refrain from formalizing this suggestion of between-human consistency into a statistical statement.

Together, these observations suggest that a framework which is able to take
these detailed distributions into account will be more accurate than one using simple parametric distributions. Furthermore, the easy accessibility of many reads from a single human suggests a new strategy for such modeling: to fit parameters “on the fly” for each data set. We implemented such a system, consisting of a new HMM compiler and a new annotation framework, which we test using an independent simulation engine.

**ham.** We implemented a new HMM compiler, called ham, because none of the existing implementations were suitable for our needs (see Methods for more details). This program is able to perform classical, pair, and multi-HMM inference on any HMM within computational constraints given a text-file description of its topology. This text file description is in YAML format (http://yaml.org/), which is easy to write both by hand and by existing libraries for all popular programming languages.

Although our initial motivations lay more in usability than optimization, ham’s general purpose C++ code is slightly faster and uses somewhat less memory than code generated by the well-known HMMoC [23], even ignoring the extra time for code generation and compilation in the latter (Table 1). The ham C++ code is open source (GPL v3 license), and is available at [https://github.com/psathyrella/ham/](https://github.com/psathyrella/ham/).

**partis.** We then used ham to build a platform called partis to perform annotation of B cell receptor sequences (BCRs) using HMMs. This platform is especially effective for modern large sequencing data sets because it can infer detailed parameters.
Table 1. Efficiency of ham compared to code generated by HMMoC for the “occasionally dishonest casino” from [14]. Elapsed CPU time and memory usage (user time and maximum resident set size from the Unix time command) for the Forward probability calculation are shown for sequences of length 1 million (mean of 300 runs) and 10 million (mean of 30 runs). These estimates do not include the extra time for code generation and compilation which is necessary in HMMoC. Uncertainties are the standard error on the mean.

|          | seq length (x10^6) | user time (sec) | max memory (MB) |
|----------|--------------------|-----------------|-----------------|
| HMMoC    | 1                  | 0.57 ± 0.01     | 18.01 ± 0.01    |
| ham      | 1                  | 0.54 ± 0.01     | 12.65 ± 0.0     |
| HMMoC    | 10                 | 5.05 ± 0.07     | 168.7 ± 0.1     |
| ham      | 10                 | 4.77 ± 0.05     | 110.6 ± 0.1     |

Benchmarking on simulated data. We developed a simulation engine for B cell receptor sequences implemented in non HMM-based code separate from that used for inference (see Methods). This engine implicitly takes into account detailed dependencies between variables beyond what is possible in our inferential framework. Briefly, to create an unmuted ancestor sequence it begins by sampling a single point from the fully-joint empirical distribution over all rearrangement parameters observed in a data set. It then generates a random sequence with corresponding rearrangement parameters (e.g. VJ insertion length), and then simulates somatic hypermutation out to each leaf of a given tree using per-position mutation rates. The test data set for which we show performance comparisons may be found in the partis GitHub repository at [http://git.io/Fxuk](http://git.io/Fxuk).

We find that our strategy using flexible categorical distributions outperforms previous BCR annotation packages on this simulated data. Indeed, partis has higher accuracy than other methods in terms of the simple fraction of genes that
TABLE 2. Fraction of genes correctly inferred (up to allele) for all publicly-available BCR annotation methods on a simulated sample of 10,000 sequences. partis is shown both for single sequences ($k = 1$), and for a multi-HMM ($k = 5$), which performs simultaneous inference on five clonally related sequences.

| method            | V correct (%) | D correct (%) | J correct (%) |
|-------------------|---------------|---------------|---------------|
| partis ($k = 5$)  | 98            | 87            | 84            |
| partis ($k = 1$)  | 98            | 84            | 82            |
| ighutil           | 99            | 82            | 79            |
| iHMMunealign      | 90            | 66            | 75            |
| IgBLAST           | 80            | 62            | 60            |
| IMGT              | 79            | 63            | 82            |

Figure 6. Hamming distance between the inferred and true naive sequences for all available BCR annotation methods on a simulated sample of 10,000 sequences. partis is shown both for single sequences ($k = 1$), and for a multi-HMM ($k = 5$), which performs simultaneous inference on five clonally related sequences.

are correctly inferred (Figure 2), a metric used in previous studies [18, 24]. This metric, however, ignores the difference between slightly incorrect and very incorrect inferences in terms of underlying sequence (see Methods for details).

A more detailed comparison via the Hamming distance between inferred and true naive sequences (Hamming to true naive, HTTN) also shows that partis outperforms previous HMM-based implementations (Figure 6). Neither SODA [16]
Figure 7. Difference between inferred and true values for deletion lengths, insertion lengths, and mutation frequency for all available BCR annotation methods on a simulated sample of 10,000 sequences. partis is shown both for single sequences ($k = 1$) and for a multi-HMM ($k = 5$), which performs simultaneous inference on five clonally related sequences.

nor SODA2 [18] were available from the web or directly from the author for comparison. partis also gave the most accurate parameter inferences (the narrowest distributions around the correct value in Figure 7) for insertion lengths, deletion lengths, and mutation frequencies.
Additionally, we find that using multiple sequences from a single rearrangement event (differing only in somatic hypermutation) leads to more accurate inference than using one sequence at a time. ham’s ability to simultaneously emit an arbitrary number of symbols allows us to take advantage of this in a way that previous implementations could not, providing an additional boost in accuracy. This annotation with multi-HMMs is shown in the preceding Figures as “partis ($k = 5$)” because, as an example, we do inference on five sequences known to form a rearrangement group.

Although the benefits of our more detailed model decrease with fewer sequences per sample, we find that our “tiered aggregation” strategy (see Methods) performs well with smaller numbers of sequences (Figure 8). Indeed, performance does not begin to degrade appreciably until sample size decreases to around a hundred sequences.

We also find that partis is computationally efficient, making it suitable for inference on large data sets. For efficiency tests, we used samples of one and ten thousand simulated sequences. The Smith-Waterman methods in ighutil require about 0.03 seconds per sequence. The HMM-based inference, alone, of partis takes about 0.04 seconds per sequence; however partis performs a preliminary Smith-Waterman step (see Methods), and must also write the HMM input files from the Smith-Waterman and initial HMM steps. In total, then, partis takes between 0.1
and 0.2 seconds per sequence, depending on sample size. The other available HMM-based method, iHMMunealign, takes about 5 seconds per sequence on both data sets. We suspect that this gain in speed is from the HMM factorization scheme described in the Methods.

We did not attempt speed comparison with web tools, although we note that as a group they are subject to periods of high load, and return free-form text outputs which are more directed towards human readability rather than computer parse-ability.

DISCUSSION

We find substantial complexity in the details of the rearrangement process; these details are consistent between data subsets and appear consistent between humans. This agrees with previous work on codon effects upon D segment frame usage [2,20], and extends it to the V and J gene segments. This complexity suggests the use of parameter-rich categorical distributions for transition and emission probabilities in a hidden Markov model (HMM) for BCR annotation. Indeed, we find that incorporating the peculiarities and richness of modern data sets enables more accurate annotation of BCR sequences. By letting the data inform parameters such as per-position mutation frequencies, we side-step the need to perform sophisticated modeling of processes such as context-sensitive hypermutation. The multi-HMM framework enables annotation of a number of sequences in a rearrangement group, resulting in increased accuracy when combined with special insert states.

We have extended the ideas of previous authors using HMMs for BCR sequence annotation [16-18] and previous work on HMM compilers [23,24] to build partis, a system to annotate BCR sequences. We find that this system substantially outperforms previous methods when given simulated data. The partis package and its dependencies are open source, and validation can be run using a supplied Docker image.
When **partis** is presented with a data set for the first time, it infers around one to ten thousand parameters for that data set; with this in mind, we would like to be very clear concerning the potential for over-fitting. For any given application of **partis**, these parameter estimates are intermediate steps for which we are not making a claim of generality, and are instead specific to that data set. This situation is analogous to fitting the rate parameters of a mutation rate matrix when performing a maximum likelihood phylogenetic analysis. In contrast, previous methods used a collection of sequences as a training set for parameter inference and applied HMMs with those general-purpose parameters to other sequences. Thus it makes perfect sense that previous authors [18] test on simulated sequences made with a different set of parameters than the ones used for training. However, because **partis** fits parameters on the fly for each data set, it never needs to make out-of-sample predictions. This is not to say that the approach would not benefit from some form of across-sample estimation of parameters. One could use such overall parameter estimates as across-human priors via empirical Bayes estimation [25]. Such an approach will become increasingly viable as we become more familiar with the population-level distributions of the relevant parameters.

We have also presented a robust validation framework for **partis**. For effective validation we need a set of sequences which is both representative of the data on which we will apply the method, and for which we know the true rearrangement history. Unfortunately real data, which is the most readily-available source of sequences which satisfy the first criterion, does not exist which also satisfies the second. In [17], for instance, validation was performed on two data sets which were each known to consist of a single clone with unknown true annotation, and the authors used the fraction of sequences annotated to have the same germline alleles as a proxy for accuracy. This approach cannot test whether the methods are inferring the correct allele, and more significantly, it tests on only two points in the entire, many-dimensional rearrangement space. In other words, validation was performed on a sample of two rearrangements, and gives no information as to
how each method might perform on every other possible combination of germline allele choices and deletion boundaries. We thus chose to use simulation, for which by construction one knows the correct answer, but which also requires proof that it accurately represents data (see Methods for details).

The results presented here have all used the IMGT set of germline sequences. We acknowledge the substantial discussion concerning this set’s accuracy [26][27], but IMGT is still by far the most popular resource for germline information and tools. We have thus designed partisso it can switch to any other germline database using a simple command-line flag, for instance as new information becomes available [27][28].

We emphasize that the good accuracy obtained using the multi-HMM framework in simulation assumed that we knew that all of the sequences derived from a single rearrangement event. On real data, of course, we do not know which sequences derive from the same rearrangement event, and in fact this is a challenging problem. Our next step will be to use the HMM framework presented here to cluster sequences together by rearrangement event.

While we and others have found HMMs to be useful for the BCR annotation problem, they do have certain limitations. The Markov assumption (that the current state is ignorant of all states except for the previous one) makes it difficult to propagate information across the HMM. For example, levels of mutation are correlated between different segments of the BCR [22]; thus upon traversing the V segment of a query sequence we have information concerning the overall mutation rate in the rest of the sequence. The Markov chain’s conditional independence, however, makes it impossible to propagate this information to the D and J segments in a strict HMM framework. Additionally, HMMs cannot account for palindromic insertions [29], complex strand interaction events [30][31], or the appearance of tandem D segments [2]. Conditional Random Fields (reviewed in [32]) could provide a way around some of these limitations; linear-chain conditional random fields enjoy many of the attractive computational properties of HMMs
while allowing for more complex dependencies. As a final note, we have not attempted to model insertion/deletion mutations within gene segments [33], which necessarily reduces the accuracy of inference for sequences with these mutations. Such events, though rare in the Adaptive data set [22], could be incorporated into an HMM with additional transitions between the states representing insertions and deletions within the HMM, although we have not attempted that here.

**METHODS**

**Data sets used.** As our primary data set, we used the Illumina-sequenced data generated from a single time point sample from each of three humans, as recently described in [34]. This data set is distinguished by its size (30 million total unique sequences, split between three humans and naive versus memory compartments) and the experimental design, which used replicate wells to enable template count estimation and decrease error [35]. This data will be available at [http://adaptivebiotech.com/link/publicBCellResource](http://adaptivebiotech.com/link/publicBCellResource) upon publication of [35]; we will call it the “Adaptive” data set.

We also show results on the data from [5], which used Illumina paired-end sequencing to investigate memory B cell recall after vaccination in 15 humans. This data set used unique barcodes to decrease error, a technique which was validated using a reproducibility experiment. This data can be publicly accessed in GenBank ([http://www.ncbi.nlm.nih.gov/projects/gap/cgi-bin/study.cgi?study_id=phs000656.v1.p1](http://www.ncbi.nlm.nih.gov/projects/gap/cgi-bin/study.cgi?study_id=phs000656.v1.p1)); we will call this the “Vollmers” data set.

**ham.** We considered writing specialized code to do HMM inference on B cell receptor sequences as previous authors have done [16–18]. However, for maximum flexibility and usability, we were inspired by other work [23, 24] to build an HMM “compiler” that can do HMM inference given a description of the HMM in a format that is simpler to interpret and modify than specialized inferential machinery. We were greatly inspired by the HMMoC [23] and StochHMM [24] tools, but neither of them fully satisfied our needs. We found the XML input format in HMMoC
too complicated to script, and the paradigm of auto-generated C++ code to be extremely difficult to test and debug. StochHMM, meanwhile, had a custom configuration file format, lacked a pair-HMM implementation, and possessed an extraordinarily extensive but not entirely functional feature set. However, StochHMM’s overall structure, and its basic idea of reading the HMM specification from a text file rather than generating code, were similar to what we desired, so we used it as a starting point for a complete rewrite. (As an aside, we note that the excellent HMMER tool \cite{HMMER} only implements profile HMMs, and thus is not appropriate for our needs.)

We were thus led to build a new tool, which we call ham. ham uses the intuitive YAML text format, which is easy to read and write by hand for simple cases, while being equally simple to use for more complex models via existing programming libraries (such as for python and C++). This makes it trivial to test many different HMM topologies, since such modifications only require editing a few lines in the text file.

In a simple comparison on the canonical “occasionally dishonest casino” example \cite{occasionally}, we find it slightly faster and more memory-efficient than HMMoC (Table 1). To generate these results we ran the Viterbi algorithm using both programs on sequences of length one (resp. ten) million, and averaged CPU time and memory use over 300 (resp. 30) runs. We note in passing that HMMoC’s XML for this example consists of 5961 characters, while ham’s YAML only requires 440.

ham can simultaneously emit onto an arbitrary number \((k)\) of outputs, such as for a pair-HMM \((k = 2)\). As we describe in Section \textsection we this ability is useful for BCR ancestral inference.

We have also included an optimization we call “chunk-caching” which greatly speeds repetitive dynamic programing (DP) table calculations. Any time the left-hand side of a sequence is the same as that of another sequence, the corresponding left hand side of their DP tables will be identical. We have thus included the ability
Figure 9. A few states in the internal region of an HMM. On the left is a state representing a position in a gene with a germline G, which usually emits a G, but sometimes emits other bases (i.e. mutates). If this state is near enough to the start or end of the gene, there will likely be transitions from the initial state, or to the end state (upper arrows). On the other hand, if it is in the middle of a gene the path is more likely to simply traverse the states in order (straight horizontal arrows).

to cache and recall previous DP tables in order to avoid unnecessary recalculation in such cases.

HMM architecture. We follow previous work [16–18] by representing each germline base in each V, D, and J allele as an HMM state (Figure 9). Insertions are represented as a separate class of states, and 5’ (3’) deletions as transitions to or from germline states which are not at the start (end) of the gene. All of these states can be combined to create a single HMM for the entire VDJ rearrangement process. While it can be useful to think of individually calculating the probability of each such path, in practice one traverses the dynamic programming (DP) table using the recursion relations found in [14].

Factorization. All inferential operations on the HMM described in the previous section* can be made faster by a process of factorization, which performs inference on a collection of smaller HMMs but returns exactly the same results as a monolithic HMM. Each of these smaller HMMs has the topology obtained by extracting it from the monolithic HMM, resulting in the topologies shown in Figure 10. In
order to motivate the procedure, we observe that 1) once a path hits a state corresponding to a given allele, it cannot transition to any state corresponding to a different allele for that segment (for example, a path in allele \( D_1 \) cannot transition to \( D_2 \)); and 2) the only external information needed to calculate the DP table on a single allele’s HMM for a given query sequence is where in the query sequence to start and where to end.

In order to describe the factorization process in more detail, we will use the notation of \[14\] referring to a single HMM: \( e_s(b) \) represents the probability of emitting \( b \) in state \( s \), and \( a_{ss'} \) is the transition probability from state \( s \) to \( s' \). We also use \( p_i(x, \pi) \) as an abbreviation for \( e_{\pi_i}(x_i)a_{\pi_{i-1}\pi_i} \): the contribution of position \( x \) to the forward probability.

One can write the total probability of a sequence \( x \) under a given HMM, i.e. the forward probability, as a sum over paths \( \pi \)

\[
P(x) = \sum_\pi P(x, \pi).
\]

(For the Viterbi algorithm we would replace the summation with \( \text{argmax} \).) As written, this corresponds to a single, monolithic HMM, and we are implicitly summing over all paths through all alleles in the V, D, and J segments. In order to factorize this monolithic probability, we first write the probability of each path in more detail as a product over each position \( i \) in the sequence of length \( L \):

\[
P(x, \pi) := \prod_{i=1}^{L} e_{\pi_i}(x_i)a_{\pi_{i-1}\pi_i} = \prod_{i=1}^{L} p_i(x, \pi).
\]

Here \( \pi_i \) is the state at position \( i \) in path \( \pi \), and \( x_i \) is the symbol at position \( i \) in the query sequence. Thus \( e_{\pi_i}(x_i) \) is the probability of emitting \( x_i \) from state \( \pi_i \), and \( a_{\pi_{i-1}\pi_i} \) is the transition probability from the state at \( i-1 \) to the state at \( i \).

We now subdivide this product for the path \( \pi \) into three factors for the V, D, and J segments; this divides the sequence correspondingly into three sections of length \( l_V, l_D, \) and \( L - l_V - l_D \). We will use \( P_R(x, \pi, l_V, l_D) \) to denote the forward
probability of the path $\pi$ through a single segment $R \in [V, D, J]$.

$$P(x, \pi) = \prod_{i=1}^{l_V} p_i(x, \pi) \prod_{j=l_V+1}^{l_V+l_D} p_j(x, \pi) \prod_{k=l_V+l_D+1}^{L} p_k(x, \pi)$$

$$= P_V(x, \pi, l_V, l_D)P_D(x, \pi, l_V, l_D)P_J(x, \pi, l_V, l_D)$$

$$= \prod_{R=V, D, J} P_R(x, \pi, l_V, l_D).$$

We then use this factorization to rearrange the order of operations in the sum over paths (1):

$$\sum_{\pi} P(x, \pi) = \sum_{\pi} \prod_{R=V, D, J} P_R(x, \pi, l_V, l_D)$$

$$= \sum_{l_V, l_D} \prod_{R=V, D, J} \sum_{g \in R} \sum_{\pi \in g} P_R(x, \pi, l_V, l_D),$$

where $q_g$ is the probability of choosing each allele $g$. Here we have isolated the computationally expensive sum over all states $\sum_{\pi \in g}$ as far to the right as possible, so that we sum only over the paths within each allele, and combine all alleles afterward. Factorization thus reduces a single HMM with a state for each germline position in each of a few hundred alleles, to a number of HMMs, each with states only for a single allele. Since at each step in the forward (and Viterbi) recursion relations we sum over all possible previous states for each current state, computing time scales roughly as the square of the number of states in each HMM. By reducing the maximum number of states per HMM by several orders of magnitude, factorization thus substantially decreases memory and computation time.

Insertions change this picture very little: we simply add them to the left side of each $D$ and $J$ HMM (they could as easily go on the right side of $V$ and $D$). The resulting HMM topologies are shown in Figure 10. In this version, we show a single insert state, with emission probabilities corresponding to the empirical insertion nucleotide distribution from data (compare 11).
Multi-HMMs. As described above, ham is able to do inference under a model which simultaneously emits an arbitrary number of symbols \( k \). When \( k = 2 \) this is typically called a pair HMM [14], and we call the generalized form a multi-HMM \((k \geq 2)\). In our setting, the \( k \) sequences resulting from such a multi-HMM model are the various sequences deriving from a single rearrangement event (which differ only according to point substitution from somatic hypermutation).

We can use this ability to perform improved inference on a collection of sequences deriving from the same rearrangement event, allowing us to integrate out much of the uncertainty due to somatic hypermutation in each single sequence. In other words, not all mutations are shared between clonally related sequences, and this provides valuable information that is only available if the HMM can emit several sequences simultaneously.

Although this approach allows us to share inferential power among various sequences for the germline regions, with a standard VDJ HMM topology (as in...
Figure 11. Our insertion state topology, with four states instead of one, which gives improved discrimination for pairs or tuples of sequences because it does not ignore mutation information within insertions. Self-transitions for insert states omitted for clarity.

Figure 10) non-templated insertions do not contribute to the likelihood in a meaningful way. In order to allow contributions from the non-templated insertions, we replace the single insert state with four states, corresponding to naïve-sequence insertions of A, C, G, and T (Figure 11). The emissions of these four states are then treated as for actual germline states (Figure 9): the A state, for example, has a large probability of emitting an A, and a complementary probability (equal to the observed mutation probability) of emitting one of the other three bases. For example, if we pass in five sequences that share identical insertions except that one sequence has a single mutated base, the HMM uses the information from the four non-mutated insertions to conclude that the difference in the fifth sequence is likely to be a mutation from the un-mutated ancestral sequence.

Parameter estimation. The parameters of our HMM consist of allele usage probabilities, transition probabilities, and emission probabilities; as described above we estimate categorical distributions for each of these. The inferred distributions of these parameters come from either a previous generation of HMM estimates, or from Smith-Waterman (as described below).
In order to convert from mutation frequencies $f$ to emission probabilities, we set the probability of emission for each non-germline base to $f/3$, and correspondingly set the probability of germline emission to $1 - f$. In the HMM implementation, it is trivial to account for different probabilities of mutating to different bases (instead of simply using $f/3$). However, we do not know of phylogenetic sequence simulation software (we use bppseqgen from Bio++ [37]) that implements both this and per-position mutation rates, and thus lacking reliable means of validation for this feature we leave it out.

Transition probabilities are set according to empirical frequencies of inferred deletions and insertions. For example, again sub-setting by allele, the frequency with which we observe a 3’ deletion of length one gives us the transition probability from the second-to-last state in this allele to the end state (thereby bypassing the last state in the allele). This logic is repeated for all positions in each allele, and also for the 5’ end (where instead of a transition to the end state, it is a transition from the initial state). Note that we include 5’ V deletions and 3’ D deletions as a convenience (dashed lines in Figure 10) to account for varying read lengths. The insertion state self-transition probabilities are distributed geometrically with mean set to the observed mean insertion length in data. While the choice of a geometric distribution is simply a result of HMMs’ inherent lack of memory, in practice we observe that insertion lengths are not far from geometrically distributed (Figure 4).

Finally, the allele choice probabilities denoted $q_g$ above are simply set to the observed frequency of each allele.

For all parameters, we perform an initial estimation step using the Smith-Waterman based methods of [38]. These provisional parameters are used to build a set of HMMs which we then run on the same data in order to infer a more accurate set of parameters. We can feed these parameters back into the HMM and continue the process recursively (this is Viterbi training in the language of [14]), but in practice we find no significant improvement after the first iteration. In future versions we may implement full Baum-Welch parameter estimation [14]. However, given the
level of agreement which we observe for Viterbi training, we do not expect large improvement from moving to full Baum-Welch.

In order to correctly estimate parameter uncertainties we would like to know the sample size: the number of independent rearrangement events. This is not the same as the number of unique reads, because several reads can stem from a single rearrangement through somatic hypermutation and clonal expansion. In principle the sample size for each measurement should be the number of observed rearrangement events. However, we can only observe sequences rather than rearrangement events, and thus do not know precisely how many rearrangement events we have observed. Previous work has shown that a significant majority of B cell rearrangements are represented by only one sequence in healthy humans, even in a deep sample from the memory compartment [39]. In order to be conservative we posit that B-cells are members of clones with two members, and thus multiply all uncertainty measurements (e.g. in Figure 2) by a factor of $\sqrt{2}$ corresponding to this assumed two-fold reduction in sample size.

Parameter merging in small samples via tiered aggregation. As mentioned above, a more highly parameterized model is both allowed by and requires large data sets. In order to facilitate the use of partis on data sets of varying size, and also to allow accurate inference of rare clones, we have thus implemented a scheme that automatically reduces the detail of the model on smaller samples. The basic idea of this scheme, which we call tiered aggregation, is to merge together the information from increasing numbers of germline alleles, in several tiers of similarity, as the sample size decreases.

As shown above, we observe significant parameter variation between alleles, and we thus initially subset all emission and transition probabilities by allele (i.e. use different probabilities for each allele of each gene in V, D, and J). For large data sets, and for BCR repertoires with relatively homogeneous allele frequency distributions, this approach is sufficient. If we observe a particular allele only a handful of times, however, we cannot confidently estimate parameters for that allele. In
such cases we thus include information from similar alleles in the parameter estimation.

Specifically, if we observe an allele fewer than $N$ times, we take a simple average over all the observations of its allelic variants when calculating its parameters, rather than just considering that allele. If we also fail to observe $N$ occurrences of all these allelic variants together, we go on to include all alleles with the same IMGT primary version (e.g. all IGHV1 alleles). Finally, if with this criterion we still do not reach $N$ observations, we include all alleles from that segment (e.g. all $V$ alleles). Here $N$ is a tunable parameter, and in practice gives good results in simulation at about 20, which is the value used for the results in this paper.

This procedure thus has the effect of automatically compensating for decreased parameter accuracy in smaller samples by switching to a less parameter-rich model. We find this strategy to be effective in simulation (Figure 8). As expected, performance decreases with decreasing sample size, although the degradation is minimal for samples larger than a few hundred sequences. While partis still performs well on fewer than a few hundred sequences, there is little benefit to using it, as one is not taking advantage of its detailed models. Instead, one could use ighutil [38] on small data sets, as in such cases the two will perform comparably in terms of accuracy.

**Validation.** As described above, we find it necessary to validate BCR annotation methods on simulated sequences; we therefore wrote a simulator of rearrangement and somatic hypermutation. This simulator takes as input the fully-joint categorical distribution over all “rearrangement values” which are necessary to specify the rearrangement event (e.g. 3' V deletion length, which V allele, etc.). It then selects a point in this space according to the given distribution, and generates a naive sequence according to the coordinates of this point. Non-templated insertions are generated with corresponding length, and insertion bases are chosen randomly according to the empirical nucleotide distribution. This simulation
process implicitly models all correlations between “rearrangement values,” even correlations that are ignored in the HMMs.

This gives us a single sequence, corresponding to the rearrangement event’s unmutated ancestor. We then use TreeSim [40] to generate a lineage tree descending from this ancestral sequence, with a speciation rate of 1.0 and extinction rate of 0.5, conditioned on a desired number of leaves. The number of leaves is easily configurable in our simulation package either as a constant or as a random variable drawn from some distribution. For simplicity, all plots in this paper use simulation with a constant five leaves per tree; results were consistent when simulating with widely varying constant, as well as non-constant, leaves-per-tree (results not shown). For each rearrangement event we choose a tree at random, which is then rescaled according to the branch lengths observed in data in each segment (V, D, or J). This tree, together with the un-mutated ancestral sequence and the relative positional frequencies described above, are then passed to the bppseqgen mutation simulator in the Bio++ [37] package. In this step we use the empirical nucleotide model for each corresponding segment; insertions are also mutated, using parameters averaged over the three segments. This produces a final set of sequences corresponding to the descendants of a single rearrangement event.

To check that this simulator generates sequences that are close to real data, we compared output parameter distributions for this simulator to the inferred parameters from data (which are the simulator’s input). These distributions are similar (Figures S5, S6, and S7) showing that the simulator accurately recapitulates the processes generating the true data. Given this accuracy, the compatibility between true and inferred distributions in Figure 7 shows that our original parameters inferred from data are indeed close to the true parameters in data.

The validation is run within the partis directory in the main repository using the command scons validate.
We emphasize that this simulator’s only inputs are the inferred empirical parameter distributions, and that it is thus entirely independent of partis’ HMM machinery. The programs, including partis, are only presented with the simulated sequences with no further information about the simulation.

ACKNOWLEDGEMENTS

The authors would like to thank Trevor Bedford, Lily Blair, Robert Bradley, Connor McCoy, Vladimir Minin, Steve Quake, Harlan Robins, and Christopher Vollmers for helpful discussions and for sharing data. This research was funded by National Institutes of Health grant R01 GM113246-01, and in part by a 2013 emerging opportunity award from the University of Washington Center for AIDS Research (CFAR), an NIH funded program under award number P30AI027757 which is supported by the following NIH Institutes and Centers (NIAID, NCI, NIMH, NIDA, NICHD, NHLBI, NIA, NIGMS, NIDDK).

REFERENCES

1. Scott D. Boyd, Eleanor L. Marshall, Jason D. Merker, Jay M. Maniar, Lyndon N. Zhang, Bita Sahaf, Carol D. Jones, Birgitte B. Simen, Bozena Hanczaruk, Khoa D. Nguyen, Kari C. Nadeau, Michael Egholm, David B. Miklos, James L. Zehnder, and Andrew Z. Fire. Measurement and clinical monitoring of human lymphocyte clonality by massively parallel V-D-J pyrosequencing. Sci. Transl. Med., 1(12):12ra23–12ra23, December 2009.
2. Kevin Larimore, Michael W McCormick, Harlan S Robins, and Philip D Greenberg. Shaping of human germline IgH repertoires revealed by deep sequencing. J. Immunol., 189(6):3221–3230, 3 August 2012.
3. Katherine J L Jackson, Marie J Kidd, Yan Wang, and Andrew M Collins. The shape of the lymphocyte receptor repertoire: Lessons from the B cell receptor. Front. Immunol., 4:263, 2 September 2013.
4. George Georgiou, Gregory C Ippolito, John Beausang, Christian E Busse, Hedda Wardemann, and Stephen R Quake. The promise and challenge of high-throughput sequencing of the antibody repertoire. Nat. Biotechnol., 19 January 2014.
5. Christopher Vollmers, Rene V Sit, Joshua A Weinstein, Cornelia L Dekker, and Stephen R Quake. Genetic measurement of memory B-cell recall using antibody repertoire sequencing. Proc. Natl. Acad. Sci. U. S. A., 110(33):13463–13468, 13 August 2013.
6. Devin Sok, Uri Laserson, Jonathan Laserson, Yi Liu, Francois Vigneault, Jean-Philippe Julien, Bryan Briney, Alejandra Ramos, Karen F Saye, Khoa Le, Alison Mahan, Shenshen Wang, Mehran Kardar, Gur Yaari, Laura M Walker, Birgitte B Simen, Elizabeth P St John, Po-Ying Chan-Hui, Kristine Swiderek, Steven H Kleinste, Stephen H Kleinstein, Galit Alter, Michael S Seaman, Arup K Chakraborty, Daphne Koller, Ian A Wilson, George M Church, Dennis R Burton, and Pascal Poignard. The effects of somatic hypermutation on neutralization and binding in the PGT121 family of broadly neutralizing HIV antibodies. *PLoS Pathog.*, 9(11):e1003754, 21 November 2013.

7. Jiang Zhu, Gilad Ofek, Yongping Yang, Baoshan Zhang, Mark K Louder, Gabriel Lu, Krisha McKe, Marie Pancera, Jeff Skinner, Zhenhai Zhang, Robert Parks, Joshua Eudailey, Krissey E Lloyd, Julie Blinn, S Munir Alam, Barton F Haynes, Melissa Simek, Dennis R Burton, Wayne C Koff, NISC Comparative Sequencing Program, James C Mullikin, John R Mascola, Lawrence Shapiro, and Peter D Kwong. Mining the antibodyome for HIV-1-neutralizing antibodies with next-generation sequencing and phylogenetic pairing of heavy/light chains. *Proc. Natl. Acad. Sci. U. S. A.*, 110(16):6470–6475, 16 April 2013.

8. Feng Gao, Mattia Bonsignori, Hua-Xin Liao, Amit Kumar, Shi-Mao Xia, Xiaohi Lu, Fangping Cai, Kwan-Ki Hwang, Hongshuo Song, Tongqing Zhou, Rebecca M Lynch, S Munir Alam, M Anthony Moody, Guido Ferrari, Mark Berrong, Garnett Kelsoe, George M Shaw, Beatrice H Hahn, David C Montefiori, Gift Kamanga, Myron S Cohen, Peter Hraber, Peter D Kwong, Bette T Korber, John R Mascola, and Barton F Haynes. Cooperation of B cell lineages in induction of HIV-1-broadly neutralizing antibodies. *Cell*, 23 July 2014.

9. Thomas B Kepler, Hua-Xin Liao, S Munir Alam, Rekha Bhaskarabhatla, Ruijun Zhang, Chandri Yandava, Shelley Stewart, Kara Anast, Garnett Kelsoe, Robert Parks, Krissey E Lloyd, Christina Stolarchuk, Jamie Pritchett, Erika Solomon, Emma Friberg, Lynn Morris, Salim S Abdool Karim, Myron S Cohen, Emmanuel Walter, M Anthony Moody, Xueling Wu, Han R Altiae-Tran, Ivelin S Georgiev, Peter D Kwong, Scott D Boyd, Andrew Z Fire, John R Mascola, and Barton F Haynes. Immunoglobulin gene insertions and deletions in the affinity maturation of HIV-1 broadly reactive neutralizing antibodies. *Cell Host Microbe*, 16(3):304–313, 10 September 2014.

10. S F Altschul, W Gish, W Miller, E W Myers, and D J Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215(3):403–410, 5 October 1990.

11. T F Smith and M S Waterman. Identification of common molecular subsequences. *J. Mol. Biol.*, 147(1):195–197, 25 March 1981.

12. Jian Ye, Ning Ma, Thomas L Madden, and James M Ostell. IgBLAST: an immunoglobulin variable domain sequence analysis tool. *Nucleic Acids Res.*, 41(Web Server issue):W34–40, July 2013.
13. Marie-Paule Lefranc, Véronique Giudicelli, Chantal Ginestoux, Joumana Jabado-Michaloud, Géraldine Folch, Fatena BellaHCene, Yan Wu, Elodie Gemrot, Xavier Brochet, Jérôme Lane, Laetitia Regnier, François Ehrenmann, Gérard Lefranc, and Patrice Duroux. IMGT, the international ImMunoGeneTics information system. *Nucleic Acids Res.*, 37(suppl 1):D1006–D1012, 1 January 2009.

14. R Durbin, S R Eddy, A Krogh, and G Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

15. Sean R Eddy. What is a hidden Markov model? *Nat. Biotechnol.*, 22(10):1315–1316, October 2004.

16. Joseph M Volpe, Lindsay G Cowell, and Thomas B Kepler. SoDA: implementation of a 3D alignment algorithm for inference of antigen receptor recombinations. *Bioinformatics*, 22(4):438–444, 15 December 2005.

17. Bruno A Gaëta, Harald R Malming, Katherine J L Jackson, Michael E Bain, Patrick Wilson, and Andrew M Collins. iHMMune-align: hidden Markov model-based alignment and identification of germline genes in rearranged immunoglobulin gene sequences. *Bioinformatics*, 23(13):1580–1587, 26 April 2007.

18. Supriya Munshaw and Thomas B Kepler. SoDA2: a hidden Markov model approach for identification of immunoglobulin rearrangements. *Bioinformatics*, 26(7):867–872, 1 April 2010.

19. Katherine J L Jackson, Bruno Gaeta, William Sewell, and Andrew M Collins. Exonuclease activity and P nucleotide addition in the generation of the expressed immunoglobulin repertoire. *BMC Immunol.*, 5:19, 2 September 2004.

20. Jennifer Benichou, Jacob Glanville, Eline T Luning Prak, Roy Azran, Tracy C Kuo, Jaume Pons, Cindy Desmarais, Lea Tsaban, and Yoram Louzoun. The restricted DH gene reading frame usage in the expressed human antibody repertoire is selected based upon its amino acid content. *J. Immunol.*, 190(11):5567–5577, 1 June 2013.

21. Carl Boettiger. An introduction to Docker for reproducible research, with examples from the R environment. 2 October 2014.

22. Connor O McCoy, Trevor Bedford, Vladimir N Minin, Philip Bradley, Harlan Robins, and Frederick A Matsen, IV. Quantifying evolutionary constraints on B cell affinity maturation. 12 March 2014. Submitted to Proc. Roy. Soc. B; http://arxiv.org/abs/1403.3066.

23. Gerton Lunter. HMMoC – a compiler for hidden Markov models. *Bioinformatics*, 23(18):2485–2487, 2007.

24. Korf I Lott PC1. StochHMM: a flexible hidden Markov model tool and C++ library. *Bioinformatics*, 30(11):1625–6, 2014.
25. Herbert Robbins. An empirical Bayes approach to statistics. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California, 1956.

26. C E H Lee, B Gaeta, H R Malming, M E Bain, W A Sewell, and A M Collins. Reconsidering the human immunoglobulin heavy-chain locus. *Immunogenetics*, 57(12):917–925, 2006.

27. Daniel Gadala-Maria, Gur Yaari, Mohamed Uduman, and Steven H Kleinste. Automated analysis of high-throughput B-cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. *Proc. Natl. Acad. Sci. U. S. A.*, 9 February 2015.

28. C T Watson, K M Steinberg, T A Graves, R L Warren, M Malig, J Schein, R K Wilson, R A Holt, E E Eichler, and F Breden. Sequencing of the human IG light chain loci from a hydatidiform mole BAC library reveals locus-specific signatures of genetic diversity. *Genes Immun.*, 23 October 2014.

29. Ravit Saada, Moran Weinberger, Gitit Shahaf, and Ramit Mehr. Models for antigen receptor gene rearrangement: CDR3 length. *Immunol. Cell Biol.*, 324(1-2):26–37, 15 May 2007.

30. T B Kepler, M Borrero, B Rugerio, S K McCray, and S H Clarke. Interdependence of N nucleotide addition and recombination site choice in V(D)J rearrangement. *The Journal of Immunology*, 157(10):4451–4457, 15 November 1996.

31. Katherine J L Jackson, Bruno A Gaeta, and Andrew M Collins. Identifying highly mutated IGHD genes in the junctions of rearranged human immunoglobulin heavy chain genes. *J. Immunol. Methods*, 324(1-2):26–37, 15 May 2007.

32. Charles Sutton and Andrew McCallum. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2011.

33. Thomas B Kepler, Supriya Munshaw, Kevin Wiehe, Ruijun Zhang, Jae-Sung Yu, Christopher W Woods, Thomas N Denny, Georgia D Tomaras, S Munir Alam, M Anthony Moody, Garnett Kelsoe, Hua-Xin Liao, and Barton F Haynes. Reconstructing a B-cell clonal lineage. II. mutation, selection, and affinity maturation. *Front. Immunol.*, 5:170, 22 April 2014.

34. William S. DeWitt, Paul Lindau, Thomas M. Snyder, Ryan O. Emerson, Anna M. Sherwood, Marissa Vignali, Christopher S. Carlson, Philip D. Greenberg, Natalie Duerkopp, and Harlan Robins. A public immunosequencing database of memory and naïve B cell receptors, 2015.

35. William DeWitt, Paul Lindau, Thomas Snyder, Marissa Vignali, Ryan Emerson, and Harlan Robins. Replicate immunosequencing as a robust probe of B cell repertoire diversity. 1 October 2014.

36. S R Eddy. Profile hidden Markov models. *Bioinformatics*, 14(9):755–763, 27 January 1998.

37. Dutheil J and Boussau B. Non-homogeneous models of sequence evolution in the Bio++ suite of libraries and programs. *BMC Evol Biol.*, 8(1):255, September 2008.
38. Connor O. McCoy. ighutil. [https://github.com/cmccoy/ighutil](https://github.com/cmccoy/ighutil), 2015. [http://dx.doi.org/10.5281/zenodo.14760](http://dx.doi.org/10.5281/zenodo.14760)

39. Rachael J M Bashford-Rogers, Anne L Palser, Brian J Huntly, Richard Rance, George S Vassiliou, George A Follows, and Paul Kellam. Network properties derived from deep sequencing of human B-cell receptor repertoires delineate B-cell populations. *Genome Res.*, 23(11):1874–1884, November 2013.

40. Stadler T. Simulating trees with a fixed number of extant species. *Syst Biol.*, 60(5):676–84, October 2011.
Figure S1. Typical observed deletion frequencies for two V, four D, and two J alleles on the Vollmers data set.
**Figure S2.** Typical observed mutation frequencies for two V, two D, and two J alleles on the Vollmers data set. The first base of the conserved cysteine and tryptophan codons (i.e. the CDR3 boundaries) are indicated with black vertical dashed lines. The large uncertainties at the 5’ end of V and 3’ end of J reflect that our reads very rarely extend into these regions.
Figure S3. Typical observed insertion lengths at the VD and DJ boundaries for two D and two J alleles.
Figure S4. The across-subset mean and variance of inferred parameter values for each human in the Vollmers data set across 10 disjoint subsets of the data.
**Figure S5.** Deletions and insertions for data and simulation for three different humans.
Figure S6. Mutation rates for data and simulation for three different humans over the full reads, and for the V, D, and J segments individually.
Figure S7. Per-position mutation rates for data and simulation for typical alleles in the V, D, and J segments.