Genotyping and Copy Number Analysis of Immunoglobin Heavy Chain Variable Genes using Long Reads

by

Michael K.B. Ford

B.A.Sc Simon Fraser University, 2015

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Science

SIMON FRASER UNIVERSITY
Fall 2019

Approval

Name: **Michael K.B. Ford**

Degree: **Master of Science**

Title: **Genotyping and Copy Number Analysis of Immunoglobin Heavy Chain Variable Genes using Long Reads**

Examining Committee: **Chair:** Ramesh Krishnamurti
Professor

Maxwell Libbrecht
Senior Supervisor
Assistant Professor

Cenk Sahinalp
Supervisor
Adjunct Professor
School of Informatics, Computing and Engineering
Indiana University

Faraz Hach
Supervisor
Assistant Professor
Urological Sciences
University of British Columbia

Felix Breden
External Examiner
Professor Emeritus
Department of Biological Sciences

Date Defended: **August 21, 2019**

Abstract

One of the remaining challenges to describing an individual's genetic variation lies in the highly heterogenous and complex genomic regions which imped the use of classical reference-guided mapping and assembly approaches. Once such region is the Immunoglobulin heavy chain locus (IGH), which is critical for the development of antibodies and the immune system. Presented is ImmunoTyper, the first PacBio-based genotyping and copy-number calling tool specifically designed for IGH V genes (IGHV). ImmunoTyper's multi-stage clustering and combinatorial optimization approach is demonstrated to be the most comprehensive IGHV genotyping approach published to date, through validation using gold-standard IGH reference sequence. This preliminary work establishes the feasibility of fine-grained genotype and copy number analysis using error-prone long reads in complex multi-gene loci, and opens the door for in-depth investigation into IGHV heterogeneity using accessible and increasingly common whole genome sequence

Keywords: Convex optimization; Integer Linear Programming; Genomics; Immunoglobin Heavy Chain Variable Genes; Genotyping; Copy Number Analysis; Long Reads; Allele Assignment Problem

Acknowledgements

I would like to acknowledge Ehsan Haghshenas (School of Computing Science, Simon Fraser University) and Corey Watson (Department of Biochemistry and Molecular Genetics, University of Louisville) for their contribution and assistant to this research; as a result they are co-authors on the associated manuscript that has been prepared for journal publication. I would also like to acknowledge the support of my supervisor Faraz Hach, who has provide assistance and access to various resources that has greatly eased the progress of my graduate studies. Finally, I would like to acknowledge the assistance, guidance and training provided by my supervisor Cenk Sahinalp, without whom this research would never have happened.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

With the advent of modern, high speed bioinformatics tools and high-throughput sequencing, reconstructing a human genome has gone from being one of the big challenges in genomics to standard protocol. Despite being a routine step in modern bioinformatics pipelines, there remains parts of the genome that are difficult to reconstruct using standard techniques. One such region is the immunoglobulin heavy chain locus (IGH), whose genes encode the foundation to the structure and development of antibodies. Despite being critical to the structure and function of the adaptive immune system of vertebrates, performing genotyping and copy number analysis of IGH genes remains challenging due to the complexity of the region, which is one of the most dynamic regions of the human genome [28].

Of the four classes of coding gene segments present in the IGH region, the Variable genes class (IGHV) plays a critical role in defining epitope binding affinity, as it completely contains two, and partially contains the last of the three complementary-determining regions. However many of the IGHV alleles are highly similar (see Figure 1.1), which in combination with their short length of between 165 bp and 305 bp (mean of 291 bp) and the high number in an individual (can be greater than 50 functional genes [30, 21]), makes the problem of IGHV genotyping challenging. To further complicate the problem, the IGH region has been shown to contain many large structural variants (SVs), including segmental duplications, large insertions and deletions, and other copy number variants (CNVs) [30]. Finally, there are two non-functional orphons of IGH (on chromosome 15 and 16) which have similar sequence to IGH [16]. As a result, classical reference-based mapping approaches to IGH analysis typically perform poorly (see Figure 1.2).

To date there have been two attempts at IGHV genotyping using high throughput sequence from germline DNA-sourced materials, both focused exclusively on functional genes. For clarity, we consider a successful IGHV genotyping result to report all the IGHV genes present in a given sample, and report the allele for every copy of every IGHV gene. Work by Yu2017b created a whole genome sequencing (WGS) Illumina short read analysis pipeline for identification of IGHV and T cell receptor sequence using a reference mapping-based

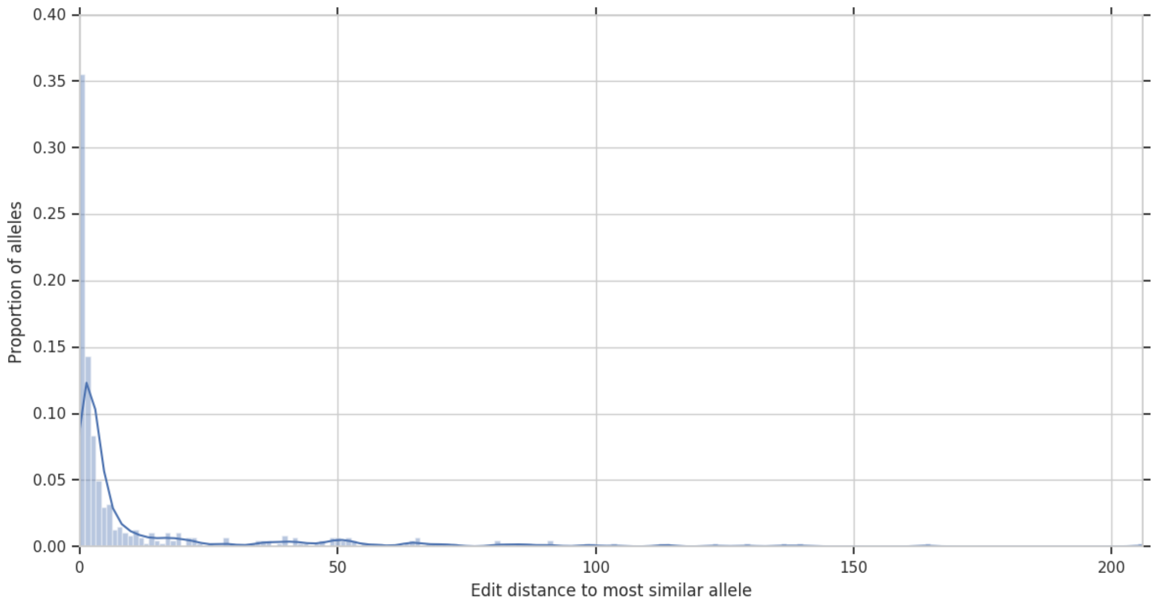

Figure 1.1: Histogram of the edit distance between each allele from the IGHV (pseudo)gene database and its most similar allele (with respect to edit distance).

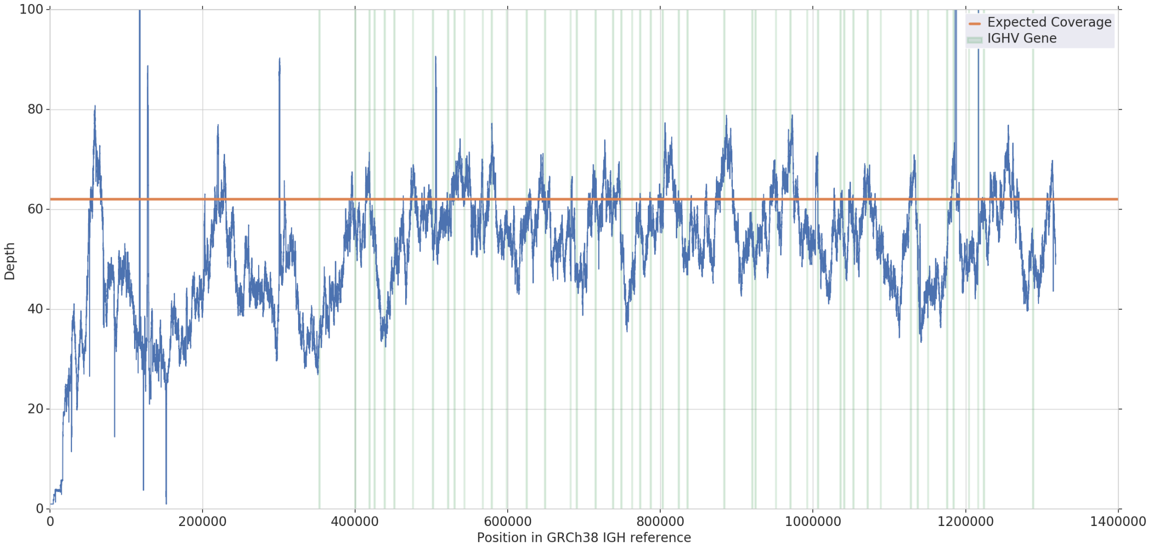

Figure 1.2: Read depth of IGH region for CHM1 WGS PacBio reads mapped to CHM1 reference using minimap2 with default parameters, demonstrating significant deviation from the expected coverage.

variant calling and frequency thresholding. While the results of their paper are initially impressive, with 8750 novel IGHV sequences having been found, there have been doubts raised regarding the accuracy of the findings by others in the field [29, 2, 12, 8]. One of the main criticisms is the reliance on a genome reference. The high degree of haplotype diversity mentioned above means that any reads that may originate from an insertion or novel sequence in the IGH region, relative to the mapping reference, will be missed from the pipeline.

The other work on IGHV genotyping using germline sequence data has been done by [20, 19], also using WGS Illumina short read data. While their initial work also relied on whole reference genome mapping, without addressing possible novel insertion sequence, their later work avoided this pitfall by mapping short reads directly to IGHV reference sequences. This method focuses on gene identification and copy number calling. However their method only calls alleles for 11 functional genes, as they identify these as only having a single copy per chromosome. Additionally, there are 7 groups of genes, each of which are a set of genes they are not able to differentiate due to high sequence similarity.

One increasingly popular approach to investigating the variations within the genes of the IGH region is through genotype and haplotype inference, using repertoire sequencing data. While the analysis of germline sequencing data is challenging, gathering sequencing data on expressed IGH sequences, typically called Adaptive Immune Receptor Repertoire sequencing (AIRR-seq), is commonplace, has established protocols and can easily be sequenced to a high depth [?]. The availability and quality of these data makes it an appealing source to infer and investigate the germline sequence, however due to the nature of IGH sequence expression this is not straightforward. An IGH mRNA sequence, as expressed by a B-cell, is not only different from the germline sequence due to VDJ recombination, but has potentially also undergone somatic hypermutation, which introduces new variants relative to the germline sequence. However despite these challenges, there have been numerous published studies and tools that have investigated the IGHV germline sequence through repertoire sequencing inference, and have been successful at identifying novel IGHV alleles and features [7, 2, 5, 23, 6, 26]. There has additionally been work done on haplotype inference through statistical learning frameworks, using the IGHJ genotype [13, 12] and/or IGHD genotype [8] as a IGHV haplotype indicator.

However it has been noted that there are challenges to performing IGHV germline analysis through repertoire inference. For example, recent work has demonstrated that inferring some IGHV variants can be nearly impossible due to the unpredictable removal of 3' bases during VDJ recombination, or be particularly hard to overcome at regions of ‘mutational hotspots’ [13]. Additionally, it has been shown that the initial reference database used can affect the reliability of inference calls for alleles that are highly similar [13].

Another inherent challenge to IGHV inference is the effect of non-uniform expression of certain VDJ configurations. This effect can be additionally complicated by the types and ratios of B-cells that are sequenced. Fundamentally, since inferring the presence of some allele is dependent on the allele being expressed, the lack of some allele does not indicate its absence in the germline sequence. This means that while inference may result in the identification of confident true positives, true negatives are impossible to differentiate from false negatives. Additionally, since the repertoire is adaptive and dynamic, some method to account for possible temporal biases to expression ratios is necessary to confidently make claims regarding the general functional significance of the presence or absence of any given allele. The effect of expression bias is also particularly relevant to haplotype inference, whose reliance on gene usage estimates can be directly confounded by expression bias [8].

While inference techniques have made significant progress at genotyping despite the challenges, there has been little work done on the other major sources of IGH heterogeneity, namely SVs and CNVs. These variants are expected to be common, as work by Watson2013 has discovered several large scale insertions and deletions in the IGH region, each containing multiple IGHV genes. However this work was done using Sanger sequencing of BAC and fosmid clones, which is prohibitively expensive and time consuming. Haplotype inference has had some success at CNV calling, deletion detection and even phased haplotype calling [8, 12], however it is limited by gene expression bias as noted above. The work by Luo et al. includes copy number calls, but does not call alleles for genes with CNVs, thus missing a critical step in the path towards complete haplotype calling.

Another large gap in our knowledge about IGH heterogeneity are non-coding sequence variants. Non-coding sequence is already known to play a critical role in the antibody repertoire as it contains the recombination signal sequence, which is required for V(D)J recombination [11]. However limitations in methodology have inhibited investigation into possible further effects through mechanisms such as enhancers and promoters.

Identification of novel IGH and IGHV sequences, genes and alleles is an important problem, as it has been noted that the primary database for IGH gene reference sequences, hosted by the international ImMunoGeneTics information system (IMGT)[1], is incomplete[**?**], and the complexity of the IGH locus is likely to lead to high sequence heterogeneity across individuals and populations. However there is still a need for fast IGHV genotyping of known alleles using common data types that are not specific to IGH research. Such tools can be integrated into standard precision medicine pipelines, allowing for investigations such as disease association studies to be done with larger sample sizes. While the performance of IGHV genotyping tools may suffer initially depending on their degree of reliance on established IGHV reference databases, they will increase in accuracy as databases become more complete over time.

In this thesis I present *ImmunoTyper*, an IGHV genotyping and CNV calling tool that is the first to be based on long read data. In order to avoid the gene expression biases found in inference-based methods, it utilizes WGS to provide a complete picture of the IGHV germline landscape. Additionally, ImmunoTyper is the first IGH-specific tool to report non-coding sequence by providing high-quality sequence for regions flanking IGHV genes, as well as the first to provide allele and CNV calling for the vast majority of IGHV pseudogenes.

Chapter 2

Methods

2.1 Algorithmic Foundations

Our goal in this paper is selecting a set of alleles that *best* describes a set of reads from the IGHV region. The principal challenge lies in deciding what represents the *best* selection. The complexity of the problem depends on the number and heterogeneity of allele candidates. There are two key considerations that need to feature in evaluating a potential *solution*:

1. The read sequences must be similar to their matched allele as well as to each other, as much as possible.

2. The number of reads assigned to an allele must match the expected read coverage.

Both these features are quantitative and their linear combination can be used as an error function to describe the quality of an assignment of reads to alleles in the context of what we call the *Allele Assignment Problem*, which I formally define as follows.

Definition: Allele Assignment Problem (AAP) Given a set of input reads $R = \{1, ..., n\}$, and a set of candidate alleles $A = \{a_1, ..., a_m\}$ as the input, consider, for any subset of reads $s_i \subseteq R$ and an allele a_j, a function $f(s_i, a_j)$ describing the error corresponding to the assignment of s_i to allele a_j. The Allele Assignment Problem asks to partition R into non-intersecting subsets s_i and assign each subset s_i to one allele a_j such that $\sum_i f(s_i, a_j)$ is minimized. More specifically, given S, the set of all $2^m - 1$ non-empty subsets of R, consider the set of all possible assignments between each $s_i \in S$ and each $a_j \in A$ with weight $f(s_i, a_j)$. Let $x_{i,j}$ be a binary variable which takes value 1 if s_i is assigned to a_j and is 0 otherwise. The allele assignment problem thus asks to determine the values of $x_{i,j}$ that minimize the objective

$$\sum_{s_i \in S, a_j \in A} x_{i,j}\, f(s_i, a_j)$$

subject to the constraint that $\bigcup_{\forall x_{i,j}=1} s_i = R$ [1]. As such, AAP modifies the well known *many-to-one assignment* problem [22] in the following manner: (i) AAP does not have the constraint that *each* allele a_j needs to be assigned a non-empty subset s_i, nor does it have the constraint that each subset s_i is assigned to a distinct allele a_j, and (ii) the cost of assigning a read to an allele depends on the other reads assigned to the same allele. Note that any error function f that captures the features summarized above leads to a computationally difficult combinatorial optimization problem; as a result we first greedily establish some read to allele assignments through a number of distinct steps so as to reduce the size of the eventual allele assignment problem we solve.

2.2 Overview of the ImmunoTyper Approach

ImmunoTyper aims to solve the Allele Assignment Problem (AAP) through which it can identify all alleles of the IGHV genes and their respective copy numbers.[2] For that it follows a number of distinct steps as described below.

1. **IGHV-containing Read Identification and Subread Extraction**
   Reads relevant to the IGH region are identified by mapping to the GRCh38 reference. Reads originating from possible novel IGH sequence are identified by mapping the unmapped reads to the IGHV allele database. IGHV sequences are identified by mapping all extracted reads to the IGHV allele database, and subsequences containing the coding region and flanking sequence, dubbed *subreads*, are extracted.

2. **Mapping-based Clustering**
   Subreads are mapped to the IGHV allele database, and then are greedily assigned to their best mapped allele under the conditions that (i) the mapping is unambiguous and (ii) the number of assigned reads for any given allele is sufficiently close to the estimated read coverage. (Read coverage is estimated using high confidence allele mappings and the provided sequence coverage.) Subreads not meeting these criteria are passed to the next step.

3. **Allele assignment for Ambiguous Subreads**
   The set of ambiguous subreads (those which could not be assigned to a single allele unambiguously) are processed in three stages:

   (a) **Super-cluster Building**
   In order to reduce the solution space, we partition the allele assignment problem

[1] in certain applications, with the additional constraint that $(x_{i,j} = 1) \rightarrow (x_{i',j} = 0)$

[2] Note that ImmunoTyper is currently tailored for V gene analysis even though it can easily be extended to perform D or J gene analysis or could be generalized to other multi-copy genes as well.

on ambiguous subreads into smaller, independent sub-problems. This is achieved by clustering subreads based on sequence similarity, into *super-clusters*, each corresponding to a small set of alleles that share high sequence similarity.

(b) **Super-cluster Breaking**
For each super-cluster, the ILP formulation for AAP is solved independently as follows. First, candidate alleles are identified by mapping the super-cluster subreads to all IGHV alleles. Variants with respect to the consensus sequence generated from all subreads are determined. Finally an ILP formulation for AAP is solved using the commonly used Gurobi ILP package [9], to break each super-cluster into smaller clusters of subreads, each representing a single copy of an IGHV gene or pseudogene.

(c) **Allele Calling**
Each subread cluster is then assigned to an allele by mapping the consensus sequence of cluster subreads against the IGHV allele database (implicitly reducing mapping errors that would be due to read error biases).

ImmunoTyper additionally includes two independent subread filtering steps which are designed to remove subreads that were mistakenly included in the analysis due to mapping errors in the subread extraction step. (See Sections *Read coverage depth estimation* and *Unclustered Subread Merging* for details.)

Solving the Allele Assignment Problem, and ultimately IGHV genotyping in this multi-stage, optimization-based approach offers several advantages. First, by employing multiple distinct methods at different stages, we can reduce the solution space and solve the problem more efficiently. For example, the 'Mapping-based Clustering' stage prioritizes speed, but only solves allele assignments for sufficiently distinct alleles. Second, by using two different methods for allele assignment, we tailor the method to the difficulty of a given allele assignment. As a result, allele assignment for IGHV sequences that are highly similar is solved using the optimization approach in "Allele Assignment for Ambiguous Subreads", which is specifically designed to differentiate highly similar sequences by considering distinguishing variants on a nucleotide level.

2.3 Allele Database

ImmunoTyper utilizes the complete set of human IGHV gene and pseudogene alleles as provided by the The International Immunogenetics information system (IMGT:*www.imgt.org* [17]). However calls for alleles that are shorter than 200bp, redundant or poorly defined are ignored. In addition, I have modified two pseudogene sequences to avoid ambiguity in the database. See Appendix A *Filtered IMGT Alleles* for a complete record of alleles that are ignored or modified.

2.4 IGHV-containing Read Identification and Subread Extraction

ImmunoTyper takes as input a BAM file representing a PacBio WGS mapping to the GRCh38 reference, as well as the depth of coverage as a parameter. In order to extract relevant reads that contain IGHV sequences, ImmunoTyper first extracts all reads with primary and supplementary mappings to the IGH region (chr14:105586437-106880844). Second, all reads that are identified as being unmapped are also extracted.

Subread Extraction

Extracted reads from both steps above are then mapped to the IGHV allele database. This is performed using Minimap2 [18] with increased sensitivity parameters (“`-cx map-pb -k10 -w3 -N5`”) to account for any novel IGHV sequence that may not be represented in the database. Reads with no mapping are then discarded.

Non-overlapping mapping locations on every read are then identified as being IGHV sequences. A subread is extracted for every IGHV sequence, using its best mapped allele. The subread contains the IGHV sequence along with the adjacent 1000bp flanking sequence. A subread extraction is conditional on (1) The best mapping covering at least 90% of the target IGHV reference sequence. (2) Neither of the 1000bp flanking sequence being clipped by the read ends. After all the valid IGHV-containing subreads are extracted, they are oriented so as to all be on the same strand.

2.5 Mapping-based Clustering

Despite the presence of highly similar and hard to differentiate V genes [20] (see Figure 1.1 for allele sequence similarity distribution), many IGHV (pesudo)genes have sufficiently distinct sequence composition to allow for confident and unambiguous mapping results, even for error-prone long reads (see Figure 2.1). Thus ImmunoTyper identifies high-confidence assignment of subreads to alleles (again, provided that the mapping is unambiguous and the coverage of the allele by the assigned subreads is close to the estimated coverage) for good, generally leading to the identification and accurate genotyping of >50% of the IGHV (pseudo)genes in a sample, which results in a significant reduction of the computational problem (i.e. of handling the subreads that could not be confidently assigned to alleles).

More specifically, we identify high-confidence mappings among the subread-to-allele mappings returned by the *Subread Extraction* step by sorting them according to the number of errors in the alignment, combined with the number of bases in the reference sequence that are not included in the alignment, i.e.

$$error = NM + a_{start} + (a_{length} - a_{end})$$

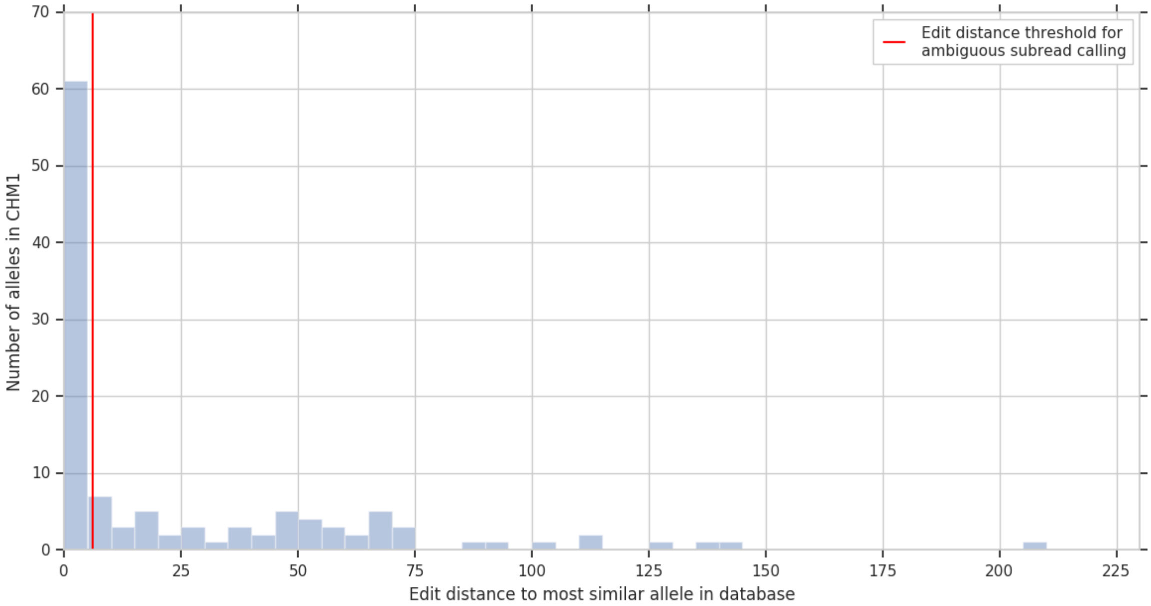

Figure 2.1: Histogram of edit distance between each CHM1 IGHV allele and its most similar IGHV allele (with respect to edit distance) from the complete database. Ambiguous mapping threshold is set to 6 (red line) as described in section 2.5.

where NM is the total number of mismatch and indel bases in the alignment, a_{start} is the start of the alignment on the reference allele, and therefore represents the number of bases in the reference allele that are not included in the alignment, and $a_{length} - a_{end}$, represents the end positing of the alignment on the reference allele subtracted from the total length of the reference allele - overall providing us the total number of reference allele bases not included in the alignment.

Subreads are then assigned to their best-mapping allele, provided that mapping is unambiguous, i.e. if the second-best mapping reference allele has at least 6 additional edit errors to the subread in comparison to the best-mapping reference allele. Subreads with an unambiguous mapping to one of the *ignored alleles* as described in Appendix A *Filtered IMGT Alleles* are discarded.

Since subreads are assigned to reference alleles based on mapping ambiguity (more specifically, a lack of mapping ambiguity) and not sequence similarity, this approach for subread clustering may still produce a valid cluster from subreads that originate from a *novel* (i.e. not in the Allele Database) IGHV (pseudo)gene, provided (i) the novel allele is sufficiently similar to an existing allele in the Database to produce acceptable mappings, and (ii) the existing allele is sufficiently distinct from all other alleles in the database so as to result in unambiguous mappings.[3] Even though I am not explicitly aiming to identify novel alleles, it is possible to generate the consensus sequence of each cluster of subreads at this stage and compare it with the reference allele they are assigned to so as to identify any difference in the sequence composition (see section *Allele Calling*), allowing for subsequent identification of any novel allele sequence.

Subreads that have ambiguous best-mapping loci are passed to the *Super-cluster Building* step.

Read coverage depth estimation

In order to confidently describe a cluster of subreads as one originating from a reference allele, it is not sufficient that the subreads have unambiguous allele assignments; the number of subreads in the cluster must also be congruent with the expected depth of coverage. In fact, depth of coverage could, in principle, be used to determine the copy number of each allele. Unfortunately it is possible that the observed depth of coverage differs from the actual sequencing depth due to natural fluctuations in sequencing coverage, or read dropout in the mapping process due to factors such as sequencing error rate and repetitive DNA in the mapping locus.

To account for any potential divergence from the actual sequencing depth, ImmunoTyper uses the results from subread mapping-based clustering to calculate a read depth statistic in order to ensure that the *expected coverage* is empirically derived from the data. To calculate the updated sequencing depth, clusters $\leq 50\%$ of the user-provided *actual* sequence depth are considered unlikely to be representative of an actual allele in the sample and are not considered, as are clusters >150% of the actual sequence depth, as these are likely to originate from alleles with multiple copies. ImmunoTyper then calculates the empirical read coverage as the median coverage of the remaining clusters.

Cluster Filtering

In order to ensure that *Mapping-based Clustering* step provides only high-confidence results, clusters are finally filtered based on the newly calculated *empirical* read coverage value. Clusters with coverage $\leq 85\%$ of this value are discarded, and their subreads are passed to the *Super-cluster Building* step. This step primarily eliminates allele assignments whose lower concordance with the expected depth are deemed lower-confidence. This step would also filter any subreads that do not contain true IGHV sequences, but were incorrectly extracted in *IGHV-containing Read Identification and Subread Extraction* due to chance sequence error. These subreads can be discarded later in the subread filtering steps during *Super-cluster Building*. After the completion of all the filtering, the remaining subread

[3] Note that if this reference allele from is also present in the dataset with subreads originating from it, its coverage will be close to an integral multiple of the overall expected coverage.

clusters and their assigned reference alleles are then called with a copy number estimated to be the integral multiple of the empirical read coverage that is closest to the size of the cluster.

2.6 Super-cluster Building

The subreads that could not be assigned in *Mapping-based Clustering* step require a more refined approach. ImmunoTyper utilizes a second clustering approach for these more difficult cases, considering both the coding region of the V genes, as well as the adjacent non-coding flanking regions - of length 1000bp.

Variants present in the non-coding flanking regions have the potential to aid subread differentiation; unfortunately distinguishing non-coding variants from sequencing errors is a major challenge. Reference-guided approaches are not possible here as there is no non-coding reference sequence/variant database available. This implies that variants must be identified through subread-to-subread comparison. Additionally, due to the high sequencing error rate of long read (PacBio) data, there is necessarily a large number of errors present in the subreads associated with the full 2000bp flanking sequence. The high error rate, combined with the lack of non-coding references and the limited utility of coding reference alleles may result in allele asssignments with a low signal-to-noise ratio. Finally, there may be thousands of subreads originating from dozens of alleles which need to be processed in this stage, implying that any method with subread-to-subread comparisons will have a large solution space.

In order to reduce the solution space of the implied problem and improve the signal-to-noise ratio, ImmunoTyper first performs a rough clustering based on a subread-to-subread sequence similarity graph as follows. Subreads are first aligned to each other (we have used Minimap2 [18] (with the "`-cx ava-pb -k14 -w3`" options to increase sensitivity). A graph is then constructed by creating a node for each subread r, and creating an edge between r and each subread r' provided the two subreads align *well* with a weight equal to the normalized error metric similar to that used in *Mapping-based Clustering.*

$$weight(r, r') = \frac{2NM + (r_{start} + (r_{length} - r_{end})) + (r'_{start} + (r'_{length} - r'_{end}))}{r_{length} + r'_{length}}$$

Here NM is the total number of mismatched and indel bases in the alignment, r_{start} is the start of the alignment on r, and therefore the length of the prefix of r not included in the alignment and $r_{length} - r_{end}$ is the length of the suffix of r not included in the alignment. Corresponding definitions apply for r'. Finally the error is normalized by the sum of the lengths of r, r'.

In order to ensure precision (and compensate for the increased sensitivity parameters used with Minimap2) we only maintain edges with weight ≤ 0.3 - the rest are deleted. Then, any node with degree 0 and its associated subread is discarded so as to eliminate subreads not sufficiently similar to others because they do not originate from the IGHV region but nevertheless were extracted in *IGHV-containing Read Identification and Subread Extraction* step due to chance sequencing or mapping errors.

The resulting *subread distance graph* can then be clustered using the Dense Subgraph Finder (DSF) tool [24]. DSF is designed to solve the 'corrupted-clique problem' as an approximation to the problem of clustering subreads originating from the same allele that have been subject to sequencing errors. It finds dense subgraphs through identification and merging of maximal cliques in the input graph. In order to ensure high precision clustering and encourage clustering that is concordant with the calculated read depth, we use the "`-min-fillin 0.95`" parameter and set the minimum cluster size at $\leq 75\%$ of the empirical read coverage using the "`-min_clust_size`" parameter. Any clusters that are smaller than the minimum are returned as single subreads and are passed to the next step.

Unclustered Subread Merging

The output of DSF is a set of dense clusters of subreads, each cluster composed of subreads with similar sequence composition. As each such cluster may include subreads that originate from more than one gene copy, I will call them super-clusters.

In addition to the super-clusters, DSF also outputs some unclustered subreads which, due to sequence error, are not sufficiently similar to other subreads to be assigned to a cluster, or were grouped into clusters smaller than the minimum size as described above. In order to assign these unclustered subreads, ImmunoTyper merges them with one of the available super-clusters. This is achieved by first constructing a representative consensus sequence for each super-cluster (using SPOA v1.1.3 [27], a SIMD-accelerated, partial-order alignment-based consensus and Multiple Sequence Alignment (MSA) tool which has been shown to be particularly effective and aligning indel-rich long reads). Unclustered subreads are then mapped to these consensus sequences (again using Minimap2 with "`-cx map-pb -k10 -w3 -N5`" options), and are added to the super-cluster with the best associated mapping. Subreads without a *good* mapping are then discarded (this second filtering step is again for eliminating subreads erroneously included in the analysis).

2.7 ILP Super-cluster Breaking

The subread super-clusters are broken into smaller clusters so that each individually represents a single allele copy - by the use of a novel ILP approach. For that we first generate a likely set of candidate alleles (described below), and then assign subreads from each super-cluster to candidate alleles using the ILP formulation (described in sections *Identifying Allele-Defining Variants* and *ILP Formulation*).

Given a super-cluster, the set of relevant candidate alleles are determined using the subread-to-allele mappings that were performed in *IGHV-containing Read Identification and Subread Extraction*. Specifically, we first generate a candidate allele pool that includes each allele that is the best-mapping allele of at least one subread in the super-cluster. In order to reduce the candidate pool, we count the number of subreads that have each allele as its best mapping; if a subread has 2 or more equally good best mapping allele, it contributes to the count of each allele by 1. We now discard any allele if (i) its (best mapping) subread count is not one of the top 10 counts among all candidate alleles, or (ii) its subread count is $\leq 50\%$ of the empirical read coverage.

Identifying Allele-Defining Variants

In order to distinguish candidate alleles from one another, we generate a set of *allele-defining* variants for each candidate allele. This is achieve by first obtaining the consensus sequence of the subreads in the super-cluster and then comparing each candidate allele with the consensus sequence.[4] We then generate the MSA (again obtained by the use of the POA method) of the candidate allele sequences and the consensus sequence. This allows us to identify a set of candidate allele-defining variants. Any of these variants that are shared among all candidate alleles are then discarded since they do not provide information for discriminating alleles; the remaining variants form our allele-defining variant set for the super-cluster.

Each subread is now compared against the consensus sequence using the subread MSA described above, to identify the allele-defining variants it includes. Then, the candidate variants are filtered based on their subread support: if the number of subreads including a variant $\leq 0.9\cdot$ empirical read coverage, it is discarded - since it is likely a result of sequencing errors. Similarly if a variant has $\geq 2\cdot$ empirical read coverage, subread support it is discarded as well - since it is not going to be very helpful in distinguishing alleles supported by the super-cluster.

ILP Formulation

Each subread super-cluster can now be partitioned into distinct clusters, each corresponding to a single allele, using a ILP formulation defined below. Note that in order to allow for multiple copies of each candidate allele, the candidate allele set (and the associated allele-

[4] ImmunoTyper uses the POA method [15] that implements the *partial order alignment* algorithm introduced there. POA is slower than SPOA but it generates a higher quality consensus sequence of the subreads and as well as their implied MSA.

defining variants) is duplicated by the *max-copy-number* value, a user-defined parameter with a default value of 4.

Given a super-cluster C, let a_j denote the j-th candidate allele and r_i denote the i-th subread associated with C.

Variables

$$\text{Let } D_i^j = \begin{cases} 1 & \text{if } r_i \text{ has been assigned to } a_j \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Let } \delta^j = \begin{cases} 1 & \text{if } a_j \text{ is called for } C \\ 0 & \text{if } a_j \text{ is not called for } C \end{cases}$$

Constraints

$$\text{For all } r_i, \quad \sum_{a_j} D_i^j = 1 \tag{2.1}$$

$$\text{For all } a_j, \quad \sum_{r_i} D_i^j \geq \text{empirical read coverage} \cdot 0.9 \tag{2.2}$$

$$\text{For all } r_i, a_j, \delta_j \geq D_i^j \tag{2.3}$$

$$min\_num \leq \sum_{a_j} \delta^j \leq max\_num \tag{2.4}$$

here
$min\_num \approx size(C)/(\text{empirical read coverage} * 0.9)$
$max\_num \approx size(C)/(\text{empirical read coverage} * 1.1)$
where $\approx$ rounds the value to the closest integer.
(The above *interval constraint* allows each super-cluster to deviate from the empirical read coverage.)

Objective

Minimize:

$$\sum_{a_j} \alpha * code\_var\_cov(a_j) + non\_code\_var\_cov(a_j)$$

where:
$\mathit{code\_var\_cov}(a_j) =$

$$\sum_{v_k \in V_C} \begin{cases} \left| \left(\sum_{r_i \text{ if } v_k \in r_i} D_i^j \right) - \left(\delta^j * \mathit{expcov} \right) \right| & \text{if } v_k \in a_j \\ \\ \left(\sum_{r_i \text{ if } v_k \in r_i} \right) D_i^j & \text{otherwise} \end{cases}$$

$\mathit{non\_code\_var\_cov}(a_j) =$

$$\sum_{v_k} \left| \left(\sum_{r_i \text{ if } v_k \in r_i} D_i^j \right) - \left(\delta^j * \mathit{expcov} \right) \right|$$

Here, given the set of all variants V for all reads and candidate alleles, v_k denotes the k-th variant in V and $V_C \subseteq V$ denotes the set of all allele defining variants for all candidate alleles. Additionally, α is a user defined parameter with default value 1000 - optimized for simulated data; $\mathit{code\_var\_cov}(a_j)$ is the variant coverage error for allele-defining variants in a_j and $\mathit{non\_code\_var\_cov}(a_j)$ is the variant coverage error for non-coding variants for subreads assigned to a_j.

Cluster merging and re-breaking

A super-cluster may fail to be partitioned so as to be assigned to distinct alleles in the following two cases: (1) there is no *qualifying* candidate allele, or (2) the ILP infeasible. In both cases we deduce that we have a poor-quality clustering, discard the super-cluster and assign each of its subreads to its *best-mapping valid cluster* as follows. We first map the subread to the consensus sequence obtained for every cluster from 2.7 *LP Super-cluster Breaking* (using SPOA). Any such cluster with a newly mapped subread is then merged with all its sibling clusters to re-create the original subread super-cluster - additionally containing one or more newly mapped subreads. The super-cluster is then re-partitioned using a new instance of the ILP. This iterative process is repeated until no such erroneous cluster is obtained by the ILP formulation (the user may put an upper bound on the number of attempts, which is set to 3 by default).

Allele Calling

In the final step, each subread cluster, obtained by partitioning a super-cluster, is assigned to an allele by first generating its consensus sequence (using SPOA), and then mapping the consensus sequence to the allele reference database as defined in 2.3 *Allele Database* with a copy number of one.[5]

[5] Any mapper including our own lordFAST [10] or Minimap2 [18] can be used here - however we have observed that our non-standard mapping of long reads to short reference alleles works best with Blasr [3] on simulated data.

Chapter 3

Results

Due to the lack of published IGH germline sequences, our ability to validate allele calls and copy number variants is limited. As a result, I performed experiments using simulated data using both the GRCh37 and GRCh38 references, which are the only published complete IGH sequences. Since the GRCh38 IGH reference is derived from the CHM1 hydatidiform mole haploid genome [30], I were also able to perform tests with real data using publicly available WGS data for CHM1. For clarity, I will use CHM1 - instead of GRCh38 to reference this sample.

3.1 Simulated data

Simulated data experiments were set up with the goal of testing the ImmunoTyper method, without the confounding effects of unavoidable noise inherent in WGS datasets.

For generating the simulated data, I first extracted the IGHV genes and pseudogenes, along with 1kbp flanking regions, from the GRCh37 (NCBI NC_000014.8:106031614-107289051) and CHM1 (NCBI NC_000014.9:105586437-106880844) references using the NCBI GenBank annotations [4]. Next I discarded all sequences corresponding to alleles that are ignored (as described in Appendix A *Filtered IMGT Alleles*). The reads were simulated from the IGHV-containing sequences at 20x using Simlord [25] in single-pass configuration, resulting in a 15.8% mean total error rate. The resulting sets of reads were then combined and provided as input to ImmunoTyper. The option "`-no-coverage-estimation`" was used to skip the subread coverage estimation step described in section *Read coverage depth estimation*, and instead use the user provided depth parameter, in this case 20x. In addition to these simulated haploid runs, the subreads from both samples were combined to simulate a diploid sample. As can be seen in Table 3.1, ImmunoTyper demonstrates strong results in all simulated samples, with precision and recall above 94%, with the exception of 89% recall in the simulated CHM1 sample. Additionally, in all but the GRCh37 sample ImmunoTyper was able to successfully differentiate alleles that were distinguished by only a single SNP (See 3.3 *Investigation into False Positive Allele Calls*, Figures 3.5-3.7).

Table 3.1: Genotype Results for Simulated and CHM1 Real Data Samples.

Sample	Num of Occurances in Reference	Number IGHV IGHV Calls	Precision	Recall	True Pos	False Pos	False Neg
CHM1 (simulated)	117	111	94.6%	89.7%	105	6	12
GRCh37 (simulated)	112	109	97.2%	94.6%	106	3	6
CHM1 + GRCh37 (simulated)	229	227	94.3%	93.4%	214	13	15
CHM1 WGS	117	110	87.3%	82.1%	96	14	21

Table 3.2: Allele Sequence Error Reduction Results

Sample	Expected read error	Median mapping error
CHM1 (simulated)	15.8%	2.0%
GRCh37 (simulated)	15.8%	2.0%
CHM1 + GRCh37 (simulated)	15.8%	2.2%
CHM1 WGS	16.0%[a]	2.3%

[a] taken from [14].

In addition to these simulated haploid runs, the subreads from both samples were combined to create a set of 4596 reads that simulate a diploid sample. Of the input reads, 2760 were identified as ambiguous.

Results are shown in Table 3.1, where ImmunoTyper demonstrates strong results in all simulated samples, with precision and recall above 94%, with the exception of 89% recall in the simulated CHM1 sample. Note that the results in Table 3.1 are for all functional IGHV genes and non-functional IGHV pseudogenes. Additionally, in all cases except GRCh37 ImmunoTyper was able to successfully differentiate alleles that were distinguished by only a single SNP (see supplementary section *Investigation into False Positive Allele Calls*, Figures 3.5-3.7). Note that True Pos indicates the allele was called by ImmunoTyper and was present in the sample, False Pos indicates the allele was called by ImmunoTyper but was not in the sample, and False Neg indicates the allele was not called by ImmunoTyper but was present in the sample.

3.2 Sequence recovery and reference mapping.

To further evaluate the performance of ImmunoTyper in subread error reduction, consensus sequences (including coding and non-coding flanking sequences) from all clusters were mapped back to their reference sequence using minimap2 with default parameters. As shown in Table 3.2, ImmunoTyper reduces the median sequence error rate by at least 86% from the raw read error rate. Visualizations of the distribution of error reduction can be found in Figures 3.1-3.4.

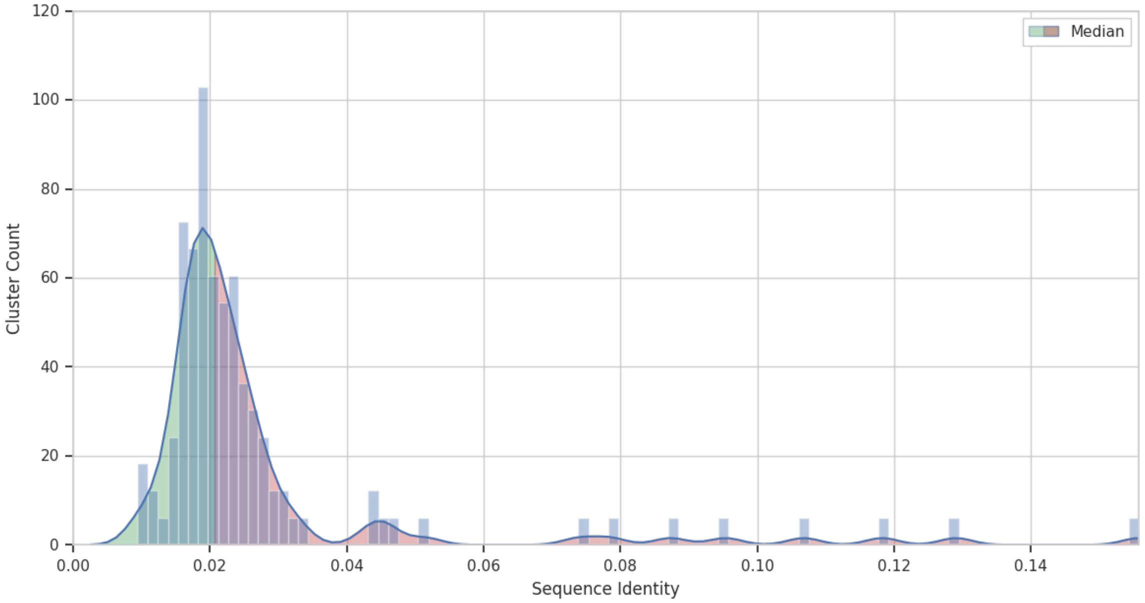

Figure 3.1: Histogram of sequence similarity between CHM1 simulated cluster consensus sequences and their best mapping location on the IGH CHM1 reference

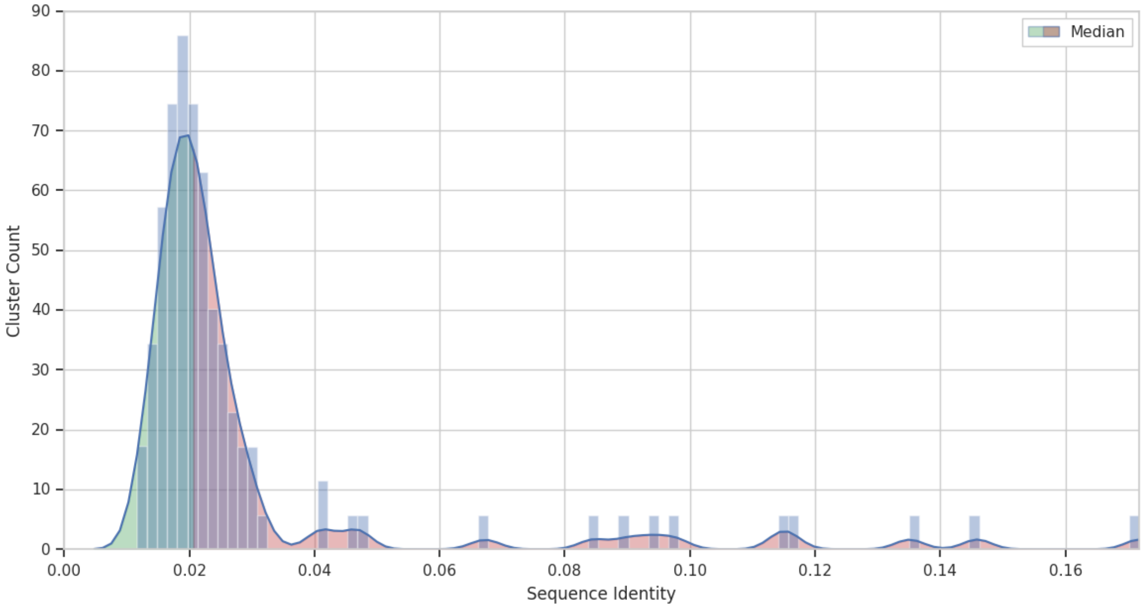

Figure 3.2: Histogram of sequence similarity between GRCh37 simulated cluster consensus sequences and their best mapping location on the IGH GRCh37 reference

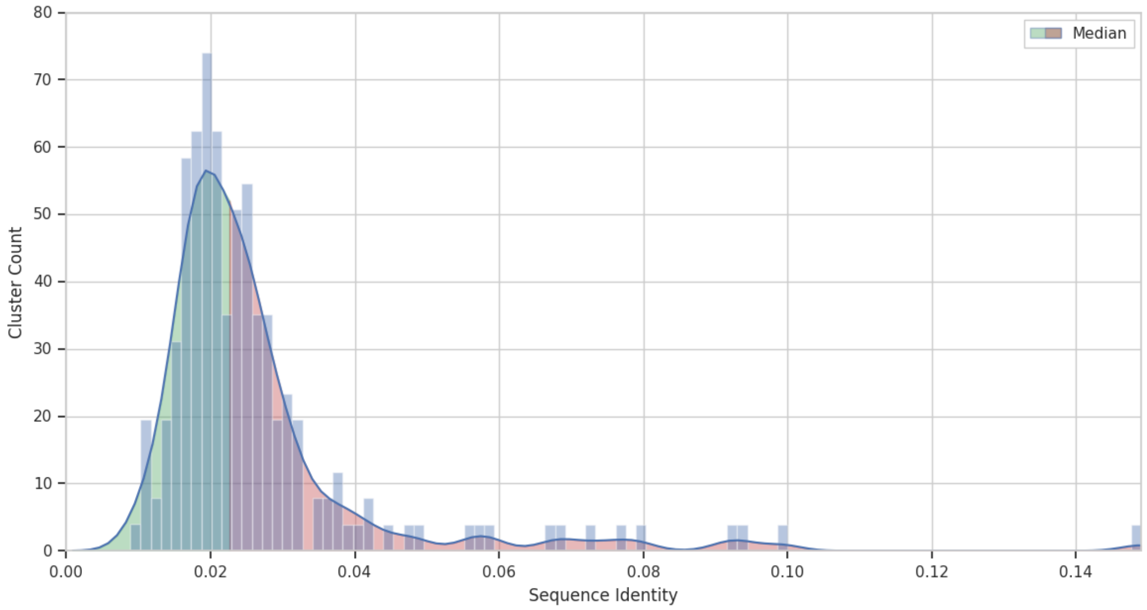

Figure 3.3: Histogram of sequence similarity between the CHM1 + GRCh37 simulated cluster consensus sequences and their best mapping location on the IGH CHM1 or IGH GRCh37 reference

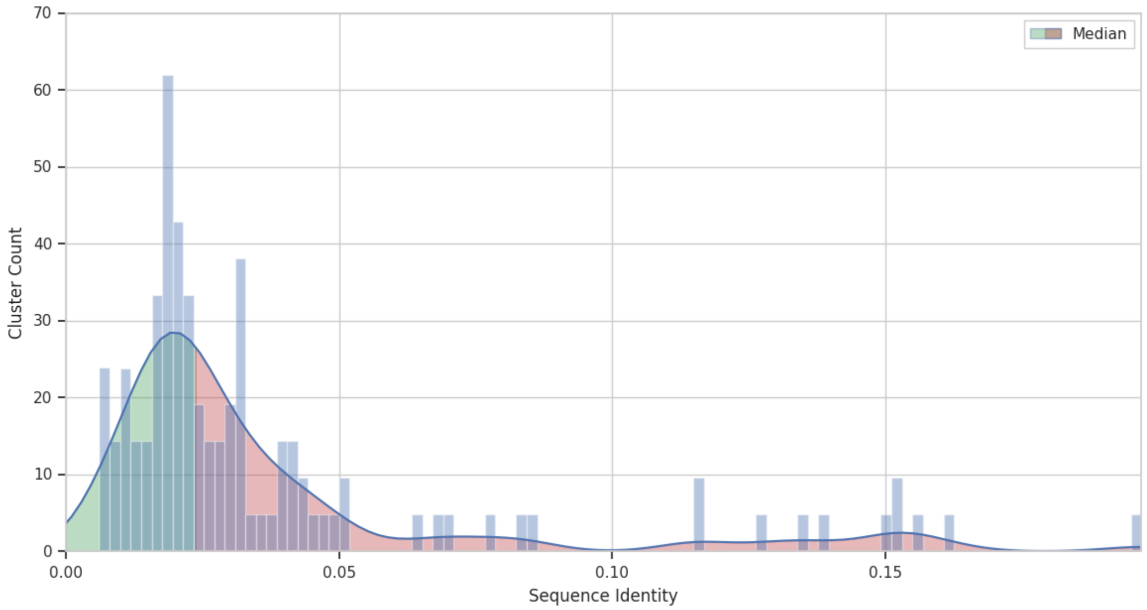

Figure 3.4: Histogram of sequence similarity between the CHM1 cluster consensus sequences and their best mapping location on the IGH CHM1 reference

3.3 Investigation into False Positive Allele Calls

In order to investigate whether sequence similarity is a major contributor to false positive allele calls, for each sample I plot the number of false positive alleles against the number of SNPs that distinguish them from their most similar allele in the sample. I also include true positives in the plot to provide context for the minimum number of variants ImmmunoTyper needs to successfully differentiate and call alleles. The plots can be found in Figures 3.5-3.8.

3.3.1 Identification of Sequence Differences Between GRCh37 and CHM1 References

The GRCh37 and CHM1 reference have significant difference in sequence and IGHV gene composition. The two references together contain 4 of the 6 known IGH insertion sequences listed in IMGT , and partially cover a 5th. [1]Clark2016Lefranc2001lefranc2001the. In Table **??** we provide the IGHV genes and pseudogenes contained in each insertion sequence, as well as list the source reference and an individual identifier.

The simulated diploid sample is the most suited to evaluate ImmunoTyper's ability to identify inserted sequence as it covers the most amount of insertions. Table **??** provides a summary of the gene and allele calls for IGHV genes and pseudogenes belonging to inserted sequence. ImmunoTyper was able to call the presence and correctly identify the alleles 12 of 14 genes and pseudogenes contained in the inserted sequences, demonstrating the ability to identify known insertion sequences in a sample. The missing allele calls were likely lost

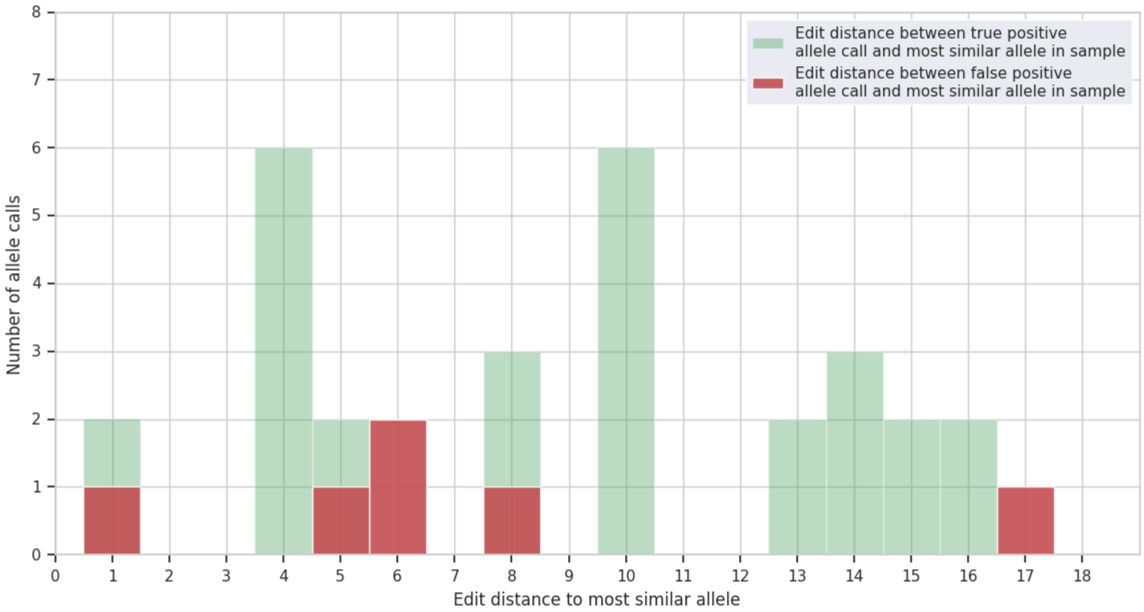

Figure 3.5: Comparing sequence similarity between TP and FP calls for the simulated CHM1 sample.

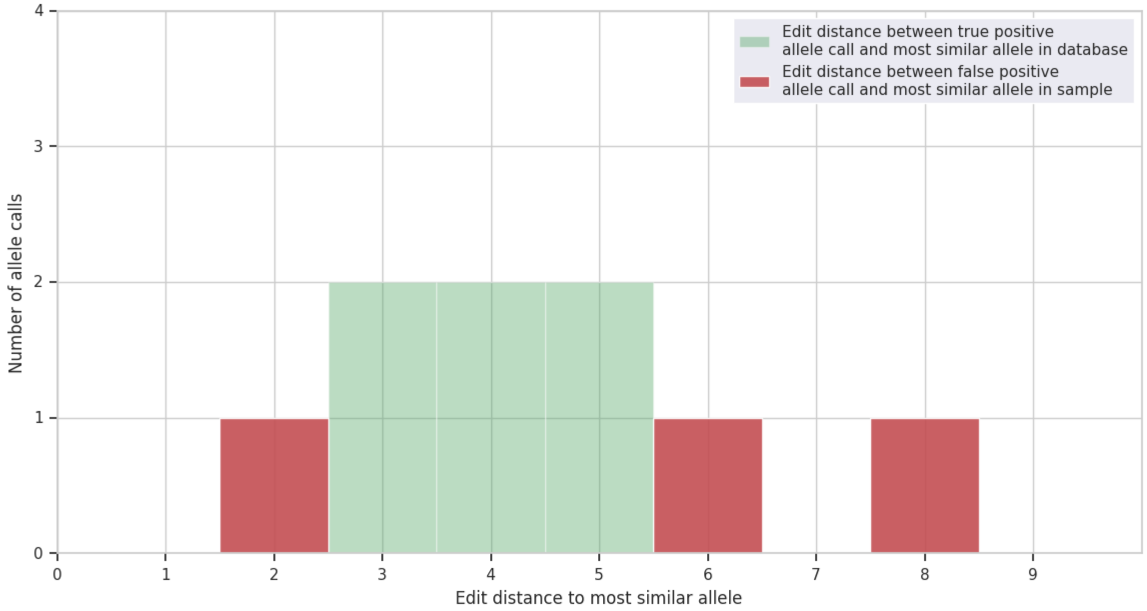

Figure 3.6: Comparing sequence similarity between TP and FP calls for the simulated GRCh37 sample.

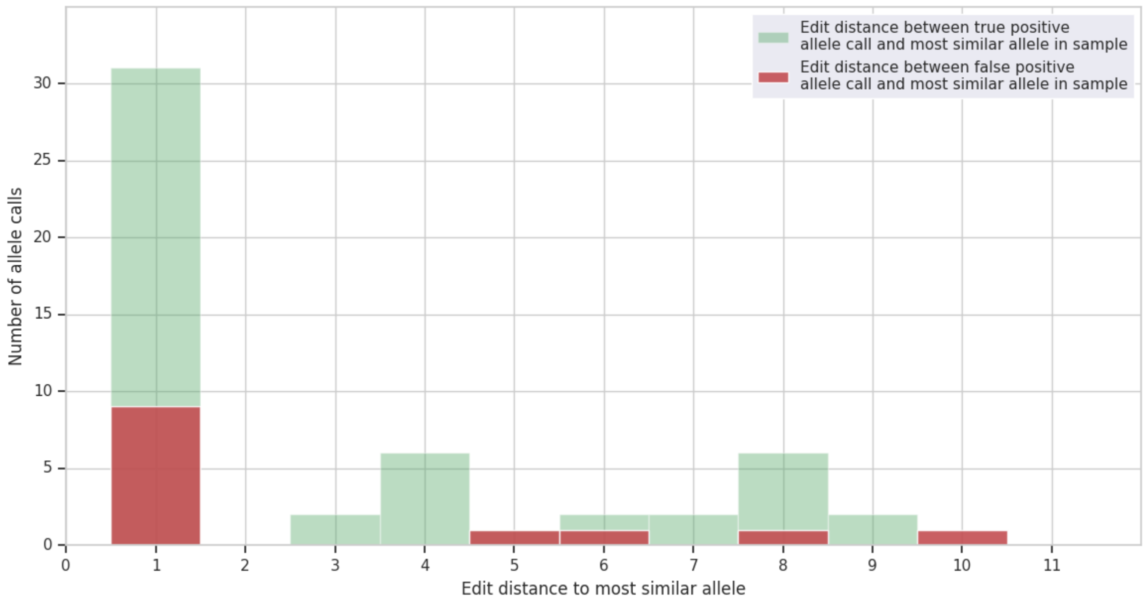

Figure 3.7: Comparing sequence similarity between TP and FP calls for the simulated CHM1+GRCh37 diploid sample.

due to high coding and flanking sequence similarity with other genes in the region (89% and 88% sequence identity for 3-69-*01 and 3-71*01; 1-8*01 and 1-69*06 respectively).

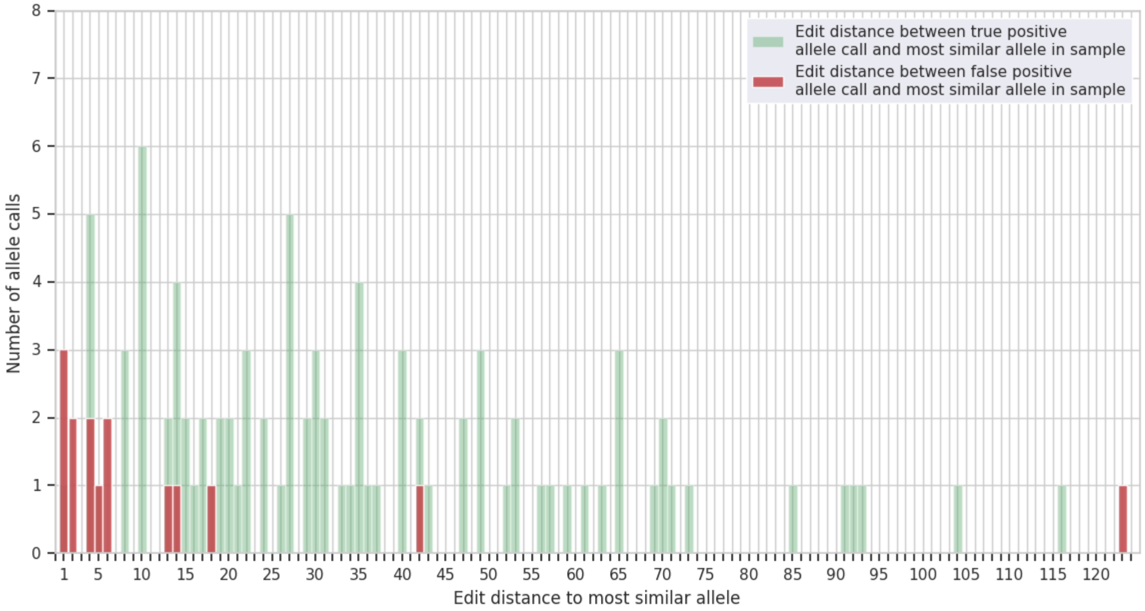

Figure 3.8: Comparing sequence similarity between TP and FP calls for the WGS CHM1 sample.

3.3.2 CNV Analysis

There are several IGHV genes in the GRCh37 and CHM1 references that are present with multiple copies. The greatest number of CNVs are present in the GRCh37 + CHM1 diploid sample, and ImmunoTyper's results for calling all CNV genes in the sample are summarized in Table 3.3. ImmunoTyper accurately calls the copies and alleles for the CNV genes in the sample in all cases except for 1-69, where the incorrect calls are likely a result of the extreme challenge of differentiating the *01 and *06 alleles as they differ by a single base pair. The 4-31 gene is included despite having a copy number of 2, because the second copy (4-30-2) is due to a duplication in the B insertion sequence in GRCh37, rather than diploidy.

Table 3.3: Calls for known CNV genes in the CHM1 + GRCh37 Sample

Gene	Num of Copies in Sample	Num of Copies Called	Correct Allele Calls	False Pos Calls	False Neg Calls
1-69	4	5	1-69-2*01, 1-69*06, 1-69*06	1-69*06, 1-69*06	1-69*01
2-70	3	3	2-70*01, 2-70D*04, 2-70*13		
3-64	3	3	3-64*02, 3-64D*06, 3-64*02		
4-31	2	2	4-30-2*01, 4-31*02		

Chapter 4

Discussion

ImmunoTyper represents a generalizable approach to multigene genotyping and copy number analysis. The results described above, while limited in sample size, provide robust validation of the methodology against publicly available genotype calls that have been produced through gold-standard approaches.

In addition to accurate genotyping results with high precision and recall, the low mapping error rates described in Section *Sequence recovery and reference mapping* demonstrate the success of our clustering approach, especially considering the high error rates of the source reads and moderate sequencing depth. However it is clear that complete IGHV genotyping using long reads is especially difficult. ImmunoTyper under-reported the number of IGHV genes present in the CHM1 WGS sample, likely due to variation in the sequencing depth or IGHV-containing subread dropout due to subreads not being identified as a result of high sequence error. Subread dropout, as well as potential noise from mistakenly including subreads from elsewhere in the genome, such as the 2 IGH orphons, are also likely explanations of the difference seen in the results of the CHM1 WGS and CHM1 simulated samples, in addition to the unavoidable shortcomings of simulating sequencing data. There also remain a few outlying cases in all samples where the allele call was incorrect and/or the sequence recovery had a high number of errors. Given the proportion of IGHV alleles which have a high degree of sequence similarity, it may be exceedingly difficult, if not impossible to achieve perfect genotyping and CNV calls using error-prone long reads, without reducing the sequence error rate through a method such as CCS reads, or increasing the sequencing depth.

In addition to identifying known IGHV alleles, ImmunoTyper also provides an opportunity to discover novel sequences, through the following features. First, the *Mapping-based clustering* step clusters reads based on ambiguity, rather than allele sequence similarity. This allows for reads originating from a novel allele to be clustered with with closest matching allele in the database. Super-clusters also account for novel alleles, as they are formed solely based on read-to-read sequence similarity, and are therefore not dependent on the known allele database. Finally, the *non_code_cov_var* error function acts as a referencefree counterbalance to *code_var_cov* error function, as it is independent of allele references and influences clustering based on read-to-read similarity, under the constraints of variant depth. As a result, the user is able to call novel alleles using the output consensus sequence for each IGHV gene. However due the challenge of calling novel alleles using long reads, especially if they differ significantly from known alleles, ImmunoTyper is focused on known allele calling.

In addition to IGH, there are other regions of the genome where ImmunoTyper could be applied with minimal modification. In particular, the immunoglobulin κ and λ light chain loci and the T cell receptor locus, are related to IGH in that they all share a similar multi-gene segment construction and undergo V(D)J recombination [11]. [19] have taken this approach by applying their tool to the T-cell beta variable locus. Extending the protocol to these similar regions is an accessible opportunity to investigate lesser-studied regions of the genome, given the current configuration of ImmunoTyper.

Fundamentally, ImmunoTyper is the first IGHV genotyping tool to use error-prone long reads, the first to integrate pseudogene calls and the first to provide data on non-coding sequence that flanks IGHV genes. While it is developed specifically for IGHV analysis, the approach and the integer linear programming (ILP) formulation for allele assignment is generalizable to any multi-gene genotyping and copy number analysis problem with known alleles.

While this initial investigation was intentionally limited to samples which have published gold-standard references, the results make us confident that ImmunoTyper represents the closest attempt at complete IGHV genotyping using WGS data to date.

Bibliography

[1] ImgtÂő, the international immunogenetics information systemÂő.

[2] Scott D Boyd, Bruno A Gaëta, Katherine J Jackson, Andrew Z Fire, 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 M Collins. Individual variation in the germline Ig gene repertoire inferred from variable region gene rearrangements. *Journal of immunology (Baltimore, Md. : 1950)*, 184(12):6986, 2010.

[3] Mark J Chaisson and Glenn Tesler. Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory. *BMC Bioinformatics*, 13(1):238, sep 2012.

[4] Karen Clark, Ilene Karsch-Mizrachi, David J Lipman, James Ostell, and Eric W Sayers. GenBank. *Nucleic acids research*, 44(D1):D67–D72, jan 2016.

[5] Martin M Corcoran, Ganesh E Phad, Néstor Vázquez Bernat, Christiane Stahl-Hennig, Noriyuki Sumida, Mats A A Persson, Marcel Martin, and Gunilla B Karlsson Hedestam. Production of individualized V gene databases reveals high levels of immunoglobulin genetic diversity. *Nature Communications*, 7, 2016.

[6] Daniel Gadala-Maria, Moriah Gidoni, Susanna Marquez, Jason Anthony Vanderheiden, Justin T Kos, Corey Watson, Kevin C O&#039;Connor, Gur Yaari, and Steven H Kleinstein. Identification of subject-specific immunoglobulin alleles from expressed repertoire sequencing data. *bioRxiv*, page 405704, jan 2018.

[7] Daniel Gadala-Maria, Gur Yaari, Mohamed Uduman, and Steven H Kleinstein. Automated analysis of high-throughput B-cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. *Proceedings of the National Academy of Sciences*, 112(8):E862, 2015.

[8] Moriah Gidoni, Omri Snir, Ayelet Peres, Pazit Polak, Ida Lindeman, Ivana Mikocziova, Vikas Kumar Sarna, Knut E A Lundin, Christopher Clouser, Francois Vigneault, Andrew M Collins, Ludvig M Sollid, and Gur Yaari. Mosaic deletion patterns of the human antibody heavy chain gene locus shown by Bayesian haplotyping. *Nature Communications*, (2019):1–14, 2019.

[9] LLC Gurobi Optimization. Gurobi Optimizer Reference Manual. 2018.

[10] Ehsan Haghshenas, S Cenk Sahinalp, and Faraz Hach. lordFAST: sensitive and Fast Alignment Search Tool for LOng noisy Read sequencing Data. *Bioinformatics*, 35(1):20–27, jul 2018.

[11] Charles Janeway. *Immunobiology : the immune system in health and disease / Charles A. Janeway, Jr. [and others].* New York : Garland Pub., 5th ed. edition, 2001.

[12] Marie J Kidd, Zhiliang Chen, Yan Wang, J Katherine, Lyndon Zhang, Scott D Boyd, Andrew Z Fire, Mark M Tanaka, Bruno A Gaëta, and Andrew M Collins. The Inference of Phased Haplotypes for the Immunoglobulin H Chain V Region Gene Loci by Analysis of VDJ Gene Rearrangements. 2012.

[13] Ufuk Kirik, Lennart Greiff, Fredrik Levander, and Mats Ohlin. Parallel antibody germline gene and haplotype analyses support the validity of immunoglobulin germline gene inference and discovery. *Molecular Immunology*, 87:12–22, 2017.

[14] David Laehnemann, Arndt Borkhardt, and Alice Carolyn McHardy. Denoising DNA deep sequencing data-high-throughput sequencing errors and their correction. *Briefings in bioinformatics*, 17(1):154–179, jan 2016.

[15] Christopher Lee, Catherine Grasso, and Mark F Sharlow. Multiple sequence alignment using partial order graphs. *Bioinformatics*, 18(3):452–464, 2002.

[16] Lefranc. *The immunoglobulin factsbook.* Academic Press, San Diego, 2001.

[17] Marie-Paule Lefranc. IMGT ®, the International ImMunoGeneTics Information System ® for Immunoinformatics. *Part B of Applied Biochemistry and Biotechnology*, 40(1):101–111, 2008.

[18] Heng Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.

[19] Shishi Luo, Jane A Yu, Heng Li, and Yun S Song. Worldwide genetic variation of the IGHV and TRBV immune receptor gene families in humans. 2(2):1–9, 2019.

[20] Shishi Luo, Jane A. Yu, and Yun S. Song. Estimating Copy Number and Allelic Variation at the Immunoglobulin Heavy Chain Locus Using Short Reads. *PLoS Computational Biology*, 12(9):1–21, 2016.

[21] Fumihiko Matsuda, Kazuo Ishii, Patrice Bourvagnet, Kei-ichi Kuma, Hidenori Hayashida, Takashi Miyata, and Tasuku Honjo. The Complete Nucleotide Sequence of the Human Immunoglobulin Heavy Chain Variable Region Locus. *The Journal of Experimental Medicine*, 188(11):2151–2162, 1998.

[22] David W Pentico. Assignment problems: A golden anniversary survey. *European Journal of Operational Research*, 176(2):774–793, 2007.

[23] Duncan K Ralph, Bjoern Peters, and Frederick A Matsen. Consistency of VDJ Rearrangement and Substitution Parameters Enables Accurate B Cell Receptor Sequence Annotation, 2016.

[24] Yana Safonova, Stefano Bonissone, Eugene Kurpilyansky, Ekaterina Starostina, Alla Lapidus, Jeremy Stinson, Laura Depalatis, Wendy Sandoval, Jennie Lill, and Pavel A. Pevzner. Ig Repertoire Constructor: A novel algorithm for antibody repertoire construction and immunoproteogenomics analysis. *Bioinformatics*, 31(12):i53–i61, 2015.

[25] Bianca K Stocker, Johannes Koster, and Sven Rahmann. SimLoRD: Simulation of Long Read Data. *Bioinformatics (Oxford, England)*, 32(17):2704–2706, sep 2016.

[26] Linnea Thörnqvist and Mats Ohlin. The functional 3âĂš-end of immunoglobulin heavy chain variable (IGHV) genes. *Molecular Immunology*, 96:61–68, 2018.

[27] Robert Vaser, Ivan Sović, Niranjan Nagarajan, and Mile Šikić. Fast and accurate de novo genome assembly from long uncorrected reads. *Genome research*, 27(5):737–746, may 2017.

[28] C T Watson and F Breden. The immunoglobulin heavy chain locus: genetic variation, missing data, and implications for human disease. *Genes And Immunity*, 13:363, may 2012.

[29] Corey T Watson, Frederick A Matsen, Katherine J L Jackson, Ali Bashir, Melissa Laird Smith, Jacob Glanville, Felix Breden, Steven H Kleinstein, Andrew M Collins, Christian E Busse, Frederick A Matsen Iv, J L Katherine, Ali Bashir, Melissa Laird Smith, Jacob Glanville, Felix Breden, Steven H Kleinstein, Andrew M Collins, and Christian E Busse. Comment on "A Database of Human Immune Receptor Alleles Recovered from Population Sequencing Data". *Journal of immunology (Baltimore, Md. : 1950)*, 198(9):3371, 2017.

[30] Corey T. Watson, Karyn M. Steinberg, John Huddleston, Rene L. Warren, Maika Malig, Jacqueline Schein, A. Jeremy Willsey, Jeffrey B. Joy, Jamie K. Scott, Tina A. Graves, Richard K. Wilson, Robert A. Holt, Evan E. Eichler, and Felix Breden. Complete haplotype sequence of the human immunoglobulin heavy-chain variable, diversity, and joining genes and characterization of allelic and copy-number variation. *American Journal of Human Genetics*, 92(4):530–546, 2013.

Appendix A

Filtered IMGT Alleles

A.1 Ignored Allele Calls

Alleles reference sequence shorter 200 bp are ignored. This includes the functional alleles:

Allele	Length
3-72*02	165
4-39*04	196

And the following non-functional alleles:

(III)-44*01	21
(III)-44D*01	21
3-62*02	106
3-76*02	155
1-12*02	154
7-56*01	154
(III)-22-2*01	30
(III)-22-2D*01	30
(III)-5-1*01	99
(III)-67-2*01	99
(II)-40-1*01	77
(II)-67-1*01	139
(II)-46-1*01	147
(II)-1-1*01	182

Additionally, the following alleles were completely removed from the database:

- 1-69D*01 was removed because it is identical in coding sequence to IGHV1-69*01
- 3-30-52*01 was removed because it differs from 3-30-2*01 by a 2bp truncation at the 3' end
- 2-70*04 was removed because it differs from 2-70D*04 by a 13bp 3' truncation

- 3-42D*01 was removed because it differs from 3-42*02 by a single bp truncation at the 5' end

Pseudogene IGHV(II)-43-1D*01 was ignored as it differs from IGHV(II)-43-1*01 by a single bp insertion, and ImmunoTyper differentiates alleles in LP Super-cluster Breaking using only SNPs. See below for a sequence comparison:

```
IGHV_II_-43-1*01        TCTGGATTCCCCAACAGAACCAGTGCTTCCTGCTGGAGCTGGATCCATCAGCCCCCAGGG 60
IGHV_II_-43-1D*01       TCTGGATTCCCCAACAGAACCAGTGCTTCCTGCTGGAGCTGGATCCATCAGCCCCCAGGG 60
                       ************************************************************

IGHV_II_-43-1*01        AAGGGA-TGGAGTGGGTCAGGTGCACAGGTCATGAAGGGAGCACAAATTCTAACCCACTC 119
IGHV_II_-43-1D*01       AAGGGACTGGAGTGGGTCAGGTGCACAGGTCATGAAGGGAGCACAAATTCTAACCCACTC 120
                       ****** *****************************************************

IGHV_II_-43-1*01        CTCAAGAGTCCAGTCACCACCTCCAGATCTATGTCCAAAAACAGCTCTTCGTATGGCTGA 179
IGHV_II_-43-1D*01       CTCAAGAGTCCAGTCACCACCTCCAGATCTATGTCCAAAAACAGCTCTTCGTATGGCTGA 180
                       ************************************************************

IGHV_II_-43-1*01        GTGACATTAGCAACAAGCACACAGCCATGT 209
IGHV_II_-43-1D*01       GTGACATTAGCAACAAGCACACAACCATGT 210
```

Alleles belonging to either of the chr15 or chr16 orphons are also ignored:

```
IGHV3-42D*01IGHV1/OR15-1*01
IGHV1/OR15-1*02
IGHV1/OR15-1*03
IGHV1/OR15-1*04
IGHV1/OR15-2*01
IGHV1/OR15-2*02
IGHV1/OR15-2*03
IGHV1/OR15-3*01
IGHV1/OR15-3*02
IGHV1/OR15-3*03
IGHV1/OR15-4*01
IGHV1/OR15-5*01
IGHV1/OR15-5*02
IGHV1/OR15-6*01
IGHV1/OR15-6*02
IGHV1/OR15-9*01
IGHV1/OR16-1*01
IGHV1/OR16-2*01
IGHV1/OR16-3*01
IGHV1/OR16-4*01
IGHV1/OR16-4*02
IGHV1/OR21-1*01
IGHV2/OR16-5*01
IGHV3/OR15-7*01
IGHV3/OR15-7*02
IGHV3/OR15-7*03
IGHV3/OR15-7*04
```

```
IGHV3/OR15-7*05
IGHV3/OR16-10*01
IGHV3/OR16-10*02
IGHV3/OR16-10*03
IGHV3/OR16-11*01
IGHV3/OR16-12*01
IGHV3/OR16-13*01
IGHV3/OR16-14*01
IGHV3/OR16-15*01
IGHV3/OR16-15*02
IGHV3/OR16-16*01
IGHV3/OR16-6*01
IGHV3/OR16-6*02
IGHV3/OR16-7*01
IGHV3/OR16-7*02
IGHV3/OR16-7*03
IGHV3/OR16-8*01
IGHV3/OR16-8*02
IGHV3/OR16-9*01
IGHV4/OR15-8*01
IGHV4/OR15-8*02
IGHV4/OR15-8*03
```

All pseudogenes that are classified as 'non-localized' are also ignored:

```
IGHV1-NL1*01
IGHV3-NL1*01
IGHV7-NL1*01
IGHV7-NL1*02
IGHV7-NL1*03
IGHV7-NL1*04
IGHV7-NL1*05
```

Alleles belonging to pseudogene IGHV(II)-20-1 are ignored due to a lack of reference sequences in the IMGT database, despite IGHV(II)-20-1*02 being listed in the CHM1 annotation.

A.1.1 Allele reference sequence modifications

Pseudogene IGHV(II)-30-41*01 has been modified by removing the 3' sequence that differentiates it from the IGHV(II)-28-1 alleles. While both the IGHV(II)-28-1*03 and IGHV(II)-28-1*01 sequences from CHM1 and GRCh37 respectively contain this 3' sequence, indicating the IGHV(II)-28-1 references should be modified, we decided that modifying a single reference sequence was more parsimonious and therefore suitable. See below for alignment of relevant alleles and sample sequences.

```
IGHV_II_-28-1*03_hg38/969-2245             CATCAACAACTATGTTTCTCAGCACACTTCTGGCTTGAGACGTCCTTGCA 1019
IGHV_II_-28-1*03_reference/1-283           ---CAACAACTATGTTTCTCAGCACACTTCTGGCTTGAGACGTCCTTGCA 1016
IGHV_II_-30-41*01_reference/1-299          ---CAACAACTATGTTTCTCAGCACACTTCTGGCTTGAGACGTCCTTGCA 1016
IGHV_II_-28-1*02_reference/1-253           ---------------------------------CTTGAGACGTCCTTGCA 986
IGHV_II_-28-1*01_reference/1-253           -------------------------------GGCTTGAGAC-TCCTTGCA 987
IGHVII-28-1*01_hg37/970-2241               CATCAACAACTATGTTTCTCAGCACACTTCTGGCTTGAGAC-TCCTTGCA 1018
                                                                            ******** ********

IGHV_II_-28-1*03_hg38/969-2245             GACCCTCTCCCTCACCTGCACTGTCTCTGGATTCCCCATCATAACCAGTG 1069
IGHV_II_-28-1*03_reference/1-283           GACCCTCTCCCTCACCTGCACTGTCTCTGGATTCCCCATCATAACCAGTG 1066
IGHV_II_-30-41*01_reference/1-299          GACCCTCTCCCTCACCTGCACTGTCTCTGGATTCCCCATCATAACCAGTG 1066
IGHV_II_-28-1*02_reference/1-253           GACCCTCTCCCTCACCTGCACTGTCTCTGGATTCCCCATCATAACCAGTG 1036
IGHV_II_-28-1*01_reference/1-253           GACCCTCTCC-TCACCTGCACTGTCTCTGGATTCCCCATCATAACCAGTG 1036
IGHVII-28-1*01_hg37/970-2241               GACCCTCTCC-TCACCTGCACTGTCTCTGGATTCCCCATCATAACCAGTG 1067
                                           ********** ***************************************

IGHV_II_-28-1*03_hg38/969-2245             TGTCCTGCTAGAATTGTATCTGCTTGCCCCTAGAAGATGGACAGGAGTGG 1119
IGHV_II_-28-1*03_reference/1-283           TGTCCTGCTAGAATTGTATCTGCTTGCCCCTAGAAGATGGACAGGAGTGG 1116
IGHV_II_-30-41*01_reference/1-299          TTTCCTGCTAGAATTGTATCTGCTTGCCCCTAGAAGATGGACAGGAGTGG 1116
IGHV_II_-28-1*02_reference/1-253           TTTCCTGCTAGAATTGTATCTGCTTGCCCCTAGAAGATGGACAGGAGTGG 1086
IGHV_II_-28-1*01_reference/1-253           TTTCCTGCTAGAATTGTATCTGCTTGCCCCTAGAAGATGGACAGGAGTGG 1086
IGHVII-28-1*01_hg37/970-2241               TTTCCTGCTAGAATTGTATCTGCTTGCCCCTAGAAGATGGACAGGAGTGG 1117
                                           * ************************************************

IGHV_II_-28-1*03_hg38/969-2245             ATCAGGTGCATGGGTTGTGAAGGGAGCACAAATTACAACCCACTGCTCAA 1169
IGHV_II_-28-1*03_reference/1-283           ATCAGGTGCATGGGTTGTGAAGGGAGCACAAATTACAACCCACTGCTCAA 1166
IGHV_II_-30-41*01_reference/1-299          ATCAGGTGCATGGGTTGTGAAGGGAGCACAAATTACAACCCACTGCTCAA 1166
IGHV_II_-28-1*02_reference/1-253           ATCAGGTGCATGGGTTGTGAAGGGAGCACAAATTACAACCCACTGCTCAA 1136
IGHV_II_-28-1*01_reference/1-253           ATCAGGTGCATGGGTTGTGAAGGGAGCACAAATTACAACCCACTGCTCAA 1136
IGHVII-28-1*01_hg37/970-2241               ATCAGGTGCATGGGTTGTGAAGGGAGCACAAATTACAACCCACTGCTCAA 1167
                                           **************************************************

IGHV_II_-28-1*03_hg38/969-2245             GAGTCCATATCCAGATCCAAGAAACAGTTCTTACAGCTGAGCTCTGTGCC 1219
IGHV_II_-28-1*03_reference/1-283           GAGTCCATATCCAGATCCAAGAAACAGTTCTTACAGCTGAGCTCTGTGCC 1216
IGHV_II_-30-41*01_reference/1-299          GAGTCCATATCCAGATCCAAGAAACAGTTCTTACAGCTGAGCTCTGTGCC 1216
IGHV_II_-28-1*02_reference/1-253           GAGTCCATATCCAGATCCAAGAAACAGTTCTTACAGCTGAGCTCTGTGCC 1186
IGHV_II_-28-1*01_reference/1-253           GAGTCCATATCCAGATCCAAGAAACAGTTCTTACAGCTGAGCTCTGTGCC 1186
IGHVII-28-1*01_hg37/970-2241               GAGTCCATATCCAGATCCAAGAAACAGTTCTTACAGCTGAGCTCTGTGCC 1217
                                           **************************************************

IGHV_II_-28-1*03_hg38/969-2245             CAGTGAACACACAACTACGCATTTTTAAGCAAAAGACGCAATGAAGGGCC 1269
IGHV_II_-28-1*03_reference/1-283           CAGTGAACACACAACTACGCATTTTTAAGCAAAAGA-------------- 1252
IGHV_II_-30-41*01_reference/1-299          CAGTGAACACACAACTACGCATTTTTAAGCAAAAGACGCAATGAAGGGCC 1266
IGHV_II_-28-1*02_reference/1-253           CAGTGAACACACAACTACGCATTTTTAAGCAAAAGA-------------- 1222
IGHV_II_-28-1*01_reference/1-253           CAGTGAACACACAACTACGCATTTTTAAGCAAAAGA-------------- 1222
IGHVII-28-1*01_hg37/970-2241               CAGTGAACACACAACTACGCATTTTTAAGCAAAAGACGCAATGAAGGGCC 1267
                                           ************************************

IGHV_II_-28-1*03_hg38/969-2245             TTCATTGT 1277
IGHV_II_-28-1*03_reference/1-283           --------
IGHV_II_-30-41*01_reference/1-299          TT------ 1268
IGHV_II_-28-1*02_reference/1-253           --------
IGHV_II_-28-1*01_reference/1-253           --------
IGHVII-28-1*01_hg37/970-2241               TTCATTGT 1275
```

Similarly, pseudogene IGHV(III)-25-1*02 has been modified by removing the 3' insertion relative to IGHV(III)-25-1*01. This was performed for the same reasons as above; the 3' insertion is present in the GRCh37 copy of IGHV(III)-25-1*01. Sequence alignment is provided below:

```
IGHV_III_-25-1*01_reference/1-295       --GAAGTTCACCGGGGGAGACAGAGGAAATAACGGTGCAGCCGGGGGCTA 1057
IGHVIII-25-1*01_hg37/1010-2306          GTGAAGTTCACCGGGGGAGACAGAGGAAATAACGGTGCAGCCGGGGGCTA 1059
IGHV_III_-25-1*02_reference/1-344       --GAAGTTCACCGGGGGAGACAGAGGAAATAACGGTGCAGCCGGGGGCTA 1057
                                          ************************************************

IGHV_III_-25-1*01_reference/1-295       TCTGAGTCTCTCCTCCAAAGACTCTGGATTCACCTTCACTGATTGCAGCA 1107
IGHVIII-25-1*01_hg37/1010-2306          TCTGAGTCTCTCCTCCAAAGACTCTGGATTCACCTTCACTGATTGCAGCA 1109
IGHV_III_-25-1*02_reference/1-344       TCTGAGTCTCTCCTGCAAAGACTCTGGATTCACCTTCACTGATTGCAGCA 1107
                                        ************** ***********************************

IGHV_III_-25-1*01_reference/1-295       TAAGCTTGGTCCAGCAAGCTCCAGGACCAGGGTTGATGTGGGCAGCAACA 1157
IGHVIII-25-1*01_hg37/1010-2306          TAAGCTTGGTCCAGCAAGCTCCAGGACCAGGGTTGATGTGGGCAGCAACA 1159
IGHV_III_-25-1*02_reference/1-344       TAAGCTTGGTCCAGCAAGCTCCAGGACCAGGGTTGATGTGGGCAGCAACA 1157
                                        **************************************************

IGHV_III_-25-1*01_reference/1-295       GGGAGAAATTGAAGAGGAAGCTCTCAGTGGTGCCCTCCATGAATACAAAG 1207
IGHVIII-25-1*01_hg37/1010-2306          GGGAGAAATTGAAGAGGAAGCTCTCAGTGGTGCCCTCCATGAATACAAAG 1209
IGHV_III_-25-1*02_reference/1-344       GGGAGAAATTGAAGAGGAAGCTCTCAGTGGTGCCCTCCATGAATACAAAG 1207
                                        **************************************************

IGHV_III_-25-1*01_reference/1-295       AATCTTCACAGTCCCCAGGACACCCTTACGTGC----------------- 1240
IGHVIII-25-1*01_hg37/1010-2306          AATCTTCACAGTCCCCAGGACACCCTTACGTGCATGGTCTCACTGATATC 1259
IGHV_III_-25-1*02_reference/1-344       AATCTTCACAGTCCCCAGGACACCCTTACGTGCATGGTCTCACTGATATC 1257
                                        *********************************

IGHV_III_-25-1*01_reference/1-295       --------------------------------------
IGHVIII-25-1*01_hg37/1010-2306          TTTACTTCTTTTATCACTTTTGTTATGTAAATCACAAT 1297
IGHV_III_-25-1*02_reference/1-344       TTTACTTCCTTTATCACTTTTGTTATGTAAAT------ 1289
```