Decombinator: a tool for fast, efficient gene assignment in T-cell receptor sequences using a finite state machine

Niclas Thomas1, James Heather2, Wilfred Ndifon3, John Shawe-Taylor4 and Benjamin Chain2,*

1CoMPLEX Department, UCL, Gower Street, London, WC1E 6BT, 2Division of Infection and Immunity, The Cruciform Building, UCL, Gower Street, London, WC1 6BT, UK, 3Department of Immunology, Weizmann Institute of Science, Rehovot 76100, Israel and 4Department of Computer Science, UCL, Gower Street, London, WC1E 6BT, UK

Associate Editor: Martin Bishop

ABSTRACT

Summary: High-throughput sequencing provides an opportunity to analyse the repertoire of antigen-specific receptors with an unprecedented breadth and depth. However, the quantity of raw data produced by this technology requires efficient ways to categorize and store the output for subsequent analysis. To this end, we have defined a simple five-item identifier that uniquely and unambiguously defines each TcR sequence. We then describe a novel application of a finite-state automaton to map Illumina short-read sequence data for individual TcRs to their respective identifier. An extension of the standard algorithm is also described, which allows for the presence of single-base pair mismatches arising from sequencing error. The software package, named Decombinator, is tested first on a set of artificial sequences and then on a set of published human TcR-β sequences. Decombinator assigned sequences at a rate more than two orders of magnitude faster than that achieved by classical pairwise alignment algorithms, and with a high degree of accuracy (>88%), even after introducing up to 1% error rates in the in silico sequences. Analysis of the published sequence dataset highlighted the strong V and J usage bias observed in the human peripheral blood repertoire, which seems to be unconnected to antigen exposure. The analysis also highlighted the enormous size of the available repertoire and the challenge of obtaining a comprehensive description for it. The Decombinator package will be a valuable tool for further in-depth analysis of the T-cell repertoire.

Availability and implementation: The Decombinator package is implemented in Python (v2.6) and is freely available at https://github.com/uclinfectionimmunity/Decombinator along with full documentation and examples of typical usage.

Contact: b.chain@ucl.ac.uk

Received on November 3, 2012; revised on December 19, 2012; accepted on January 2, 2013

1 INTRODUCTION

The power of vertebrate adaptive immunity lies in its ability to synthesize and deploy an enormously diverse repertoire of antigen-specific receptors, by a unique process of imprecise recombination of DNA segments within the lymphocyte germline. Estimates for the number of possible B- and T-cell receptors that can be generated by this mechanism are in excess of 1010 (Wang et al., 2010). Because of this diversity, global analysis of immune repertoires and responses has lagged behind the structural understanding of receptor/antigen interactions. Advances in high-throughput sequencing (HTS) now offer the possibility of probing, cataloguing and analysing immune responses with unprecedented breadth and depth (Holt and Jones, 2008; Shendure and Ji, 2008). So far, this approach has been applied to only a handful of examples. However, with the rapid increase in high-quality read length that can be achieved, and the fall in cost, the number of such datasets generated in the coming years is likely to increase rapidly. The ability to extract the maximum possible useful information from such datasets, whether to answer basic scientific questions or for more translational applications in diagnosis and disease stratification, will depend on simple and efficient bioinformatic pipelines with which to process and analyse the raw data.

An individual T-cell receptor (we focus here on T-cell receptor analysis, although a similar approach is equally applicable to B cells) is made up of a heterodimeric αβ chain (~95% of T cells) or γδ chain (5%). Each chain is made up of a variable and constant region, which is encoded by separate open reading frames and spliced together after transcription. The DNA sequence encoding the variable region is itself made up of two [variable (V) and joining (J) for α or three [V, diversity (D) and J for β] minigens. The α-chain locus contains 45 V and 50 J gene segments (not including a number of pseudogenes) (Giudicelli et al., 2005). The β-locus contains 48 V, 2D and 13 J gene segments. During T-cell development in the thymus, one V, (D) and J minigene are recombined to give a contiguous open reading frame. Additional diversity is achieved by random deletion of germline nucleotides and addition of non-template nucleotides at the VJ junction (α and γ chains) or VD and DJ junctions (β and δ chains). To unambiguously define an individual TcR chain, therefore, it is necessary to specify the V, (D) J gene which is being used, and both deletions and additions occurring at each relevant junction.

In this study, we focus on the TcR-β chain, as HTS data are publicly available (Warren et al., 2011). As the two D region sequences are short and rather similar, it is difficult to unambiguously assign D gene usage of a specific β sequence. We, therefore, define each TcR-β sequence in terms of a unique five-part identifier. The first two parts identify the V and J chain.
regions used. The third part gives the number of 3' deletions of the V region. The fourth part gives the number of 5' deletions of the J region. The final part is the sequence found between V and J, which includes added nucleotides at the VD and DJ junctions, as well as any remaining nucleotides from the D region itself (Fig. 1).

To generate identifiers from a large number (potentially 100 million from a single experiment using Illumina HiSeq technology) of short-read sequences rapidly and efficiently, we implement the algorithm described by Aho and Corasick (Aho and Corasick, 1975). Existing algorithms to analyse TcR and Ig sequences include IMGT/HighV-QUEST (Alamyar et al., 2012) that can only analyse up to 150,000 sequences per batch, SoDA (Volpe et al., 2006), which uses dynamic programming to analyse Ig sequences, typically taking up to 25 s/sequence to perform such analysis (Volpe et al., 2006) and expectation maximization (Murugan et al., 2012) to determine the most likely distributions over recombination variables (e.g. V and J gene usage and V and J deletions). Other alternatives include classic pairwise alignment implemented in R (Ndifon et al., 2012; Pages et al., 2012), which can deal with larger batches of sequences than IMGT/HighV-Quest (Alamyar et al., 2012), but it still lacks the efficiency to deal with the prodigious increase in sequence volume now obtainable from HTS. The Aho–Corasick algorithm was originally developed for exact pattern set matching within a text string, and it has been widely used (Satya et al., 2003). It constructs a finite-state machine that resembles a trie with additional links between the various internal nodes. The trie for any specific set of query texts need only be built once in advance; thus, the preprocessing time is \( O(n) \), where \( n \) is the sum of the lengths of all keywords to search for. The algorithm then works in \( O(n + m + x) \), where \( x \) is the number of keywords found in the query text, and \( m \) is the length of the text being queried. We first implement this algorithm using a set of unique identifying short-tag sequences from each known TcR-\( \beta \) V and J sequences as the set of query texts. The algorithm is then tested, both on artificial sets of sequences created in silico and on real sets of TcR sequences (Warren et al., 2011). The standard form of the Aho–Corasick algorithm requires exact matches between query and target. To accommodate realistic estimates of sequencing error using current HTS technology, we develop an extension of the basic algorithm. In comparison with pairwise alignment, the Aho–Corasick modified algorithm increased speed of execution by over two orders of magnitude while still unambiguously assigning identifiers to \( \geq 88\% \) of sequences tested.

2 METHODS

2.1 V and J assignment using the Aho–Corasick pattern matching machine

Our approach to the problem of gene assignment in rearranged TcRs follows that described by Aho and Corasick (Aho and Corasick, 1975). We define a string as a finite, contiguous sequence containing at least one nucleotide error.

Result: classified sequence

Perform classic Aho-Corasick search using FullVKeywords

\[ \text{if no Vmatch found then} \]

Perform classic Aho-Corasick search using FirstHalfVKeywords;
Perform classic Aho-Corasick search using SecondHalfVKeywords;
counter = 0;
for \( j \) in len(FirstHalfVKeywordsFoundInSequence) do
\begin{align*}
hd &= \text{Hamming(CorrespondingFullKeyword}[j]\text{, AssumedPositionInSequence);}
\text{if } hd = 1 \text{ then }
\quad \text{counter += 1;}
\quad \text{Vmatch} = j;
\end{align*}
end
for \( k \) in len(SecondHalfVKeywordsFoundInSequence) do
\begin{align*}
hd &= \text{Hamming(CorrespondingFullKeyword}[k]\text{, AssumedPositionInSequence);}
\text{if } hd = 1 \text{ then }
\quad \text{counter += 1;}
\quad \text{Vmatch} = k;
end
if counter == 1 then
\quad assign V gene corresponding with Vmatch to sequence
end

Fig. 1. Pseudocode outlining our modification to the classic Aho–Corasick algorithm, whereby a search is first adopted using the full-length V keywords, and the modification is used if no full-length V keyword is found. The classic Aho–Corasick approach outputs all keywords found, along with their position within the sequence. An identical approach is adopted in searching for J keywords.
sequence of symbols, a *keyword* as a desired string to be found and a *target* as a string in which one searches for a keyword. The framework of Aho and Corasick allows one to search within a text string $T$ of length $m$ for the occurrence of all keywords within a pattern set $P$ of length $j$, where

$$P = \{P_1, P_2, ..., P_i\}$$  

(1)

If $n = \sum_{i=1}^{j} |P_i|$, then the algorithm works in $O(n + m + x)$ time, where $x$ is the number of keywords in $P$ that were found in $T$. This can be compared with complexity of order $O(tm)$ for Smith–Waterman pairwise alignment for each keyword of length $t$. A pattern-matching machine comprises a set of states, which are traversed by making a series of comparisons with characters of the target string. At each step, the new state reached depends on the next character seen in the target string. The pattern-matching machine’s output is derived from *goto*, *failure* and *output* functions. The goto function is based on a keyword trie, such that all keywords are contained within the trie and listed from the root of the trie, and it maps a given state and an observed input character to a new state. The failure function maps one state to another, and it is used when a given state and an observed input character are not defined for a particular state. The failure function for each state is given by the longest suffix that was found in $P$. Thus, the beauty of this approach is that it makes only one pass through the target string to find all keywords present within it. The algorithm was implemented using Acora (implemented as a C extension embedded in Python using Cython) and BioPython. This combines the speed of C with the user-friendly interface of Python. This provides a fast efficient tool capable of large-scale HTS analyses, combined with a user-friendly high-level programming language appealing to those with limited programming experience.

The keyword trie (i.e. the set of patterns to be identified) represents the set of $V$ or $J$ functional, prototypic alleles, of which there are 48Vβ, 13Jβ, 2Dβ, 45Va and 50Je in the human genome (Giudicelli *et al.*). We consider only the β chain of the TcR for the purpose of testing our software on real sequences, as it is the only one for which HTS sequence data have been published (Warren *et al.*, 2011), although our software is tested on both TcRa and TcRβ in silico sequences. In line with many other studies (Freeman *et al.*, 2009), we found that the two alternative Dβ genes are often too short (12 or 16 bp) and too similar to reliably assign in most sequences (~75% (Freeman *et al.*, 2009)) after germline deletions have occurred; therefore, these are considered to be part of the additional non-template nucleotides at the VJ junction. Rather than searching for a whole $V$ or $J$ gene, each $V$ or $J$ region was represented by a short sequence (20 bp), which we call a ‘tag’, which unambiguously identifies one, and only one, gene segment. The ‘tags’ corresponding to each $V$ or $J$ segment were found by a simple exhaustive search, where each $V$ (or $J$) region was split into a set containing all possible substrings, and then searched for in all other $V$ (or $J$) regions.

2.2 Assignment of sequences containing single mismatches

The classical Aho–Corasick algorithm requires an exact match between query and target, and a number of approaches to dealing with mismatches have been explored (Ukkonen and Wood, 1993). Exact matching for TcR region assignment leads to a significant loss of performance, as a single-base pair mismatch, which could easily arise from a sequencing error, would cause the algorithm to fail. A novel extension of this method was, therefore, developed to assign gene regions using tags that differ from the target sequence by at most one base pair.

Each tag in the set of all keyword tags for $V$ (and $J$) are subdivided into two half tags representing each full tag (or J) region. If a match is found with a particular half tag, all the full-length tags corresponding to the given half tag are aligned with the target sequence, and the Hamming distance (the number of mismatches) between full-length tag and target sequence is calculated ($V_h$ and $V_j$). If the Hamming distance between the full tag and the sequence read is >1, then the tag is discarded and no assignment made. If the Hamming distance is one for one and only one tag, the $V$ or $J$ region corresponding to that full-length tag is assigned (Fig. 2iii). Pseudocode outlining this algorithm is given in Figure 1.

2.3 Deletions and additions

Once a tag has been assigned, the number of deletions is calculated by using our knowledge of the location of the tag within that gene segment. The algorithm jumps to the end of where the full-length gene segment would finish in the sequence read and then counts back towards the tag (Fig. 3), until it finds the end of the $V$ or $J$ region, by finding three consecutive bases that match to the expected $V$ or $J$ region. Finally, the sequence found
sequences (see Section 2). We compared performance with no

tering a repertoire of sequences. We categorize each sequence

sequence in terms of an identifier, providing a simple means of clus-
tingides along with the remnants of the D gene segment used.

possible values. In a second round of

chosen with uniform probability distributions over the range of

and J genes, the number of deletions and the insertions was all

In the first round of sequences generated, the distribution of V

non-template nucleotides are added at VD and DJ junctions.

between 0 and 10 nucleotides inclusive. Finally, 0–10

We first tested Decombinator on 106

In silico

3 RESULTS

3.1 In silico sequences

We first tested Decombinator on 106 in silico generated β se-

sequences, Decombinator was

simulated sequencing error, at an error rate of 1%, a figure com-

monly quoted as the typical error rate for Illumina HTS tech-
nology (Bolotin et al., 2012; Luo et al., 2012), and at a reduced

error rate of 0.1%, which in our experience is often more typical

of Illumina HiSeq or MiSeq output (Table 1). With no simulated

sequencing error, Decombinator assigned all but one sequence

successfully in 49.7 s, whereas a standard pairwise alignment

approach implemented in R using Biostrings (Ndifon

et al., 2012; Pages et al., 2012) correctly assigned all the sequences in

19731 s (Table 2). The speed of assignment, therefore, increased

almost 400 times. On inspection, the unassigned sequence

contained two distinct J tags, the rare consequence of random,

non-template nucleotide addition. As the rate of simulated

sequencing error increased, the proportion of assigned sequences

by Decombinator decreased somewhat (Table 1), as expected.

Even so, >88% of sequences were still assigned by

Decombinator even when a sequencing error rate of 1% was

assumed, with the greater error rate contributing to an 8% in-

crease in execution time compared with sequences with no

sequencing error. We then created 105, 106 and 107 synthetic

sequences separately and used Decombinator to assign V and J

genes. Performance times were 4.1, 42.7 and 402.2 s respectively*,

confirming an approximately linear relationship between number

of sequences analysed and time taken.

Although most studies currently in the literature use β-chain
diversity as a surrogate for total-TcR diversity, we also tested the

efficiency and accuracy of Decombinator on synthetic α chains

(Table 1). We simulate VJ recombination, deletion of germline

nucleotides and addition of non-template nucleotides as
described previously. Decombinator demonstrated similar per-

formance in both accuracy and speed (Table 1), assigning

>88% of sequences at a 1% error rate in 70.0 s*. In assigning

TcRa sequences, Decombinator was >500× faster than a stand-

dard pairwise alignment approach (compare Tables 1 and 2).
As further demonstration of the efficiency of Decombinator, we developed an additional pipeline to assign V and J regions in TcR sequences, using the standard Boyer-Moore algorithm (Boyer and Moore, 1977) on full-length V and J sequences, and an implementation of pairwise alignment using the shorter unique 20 bp V keywords and 12 bp J keywords as queries instead of full-length V and J segments (Table 3). Because of such a comparison being computationally demanding, we conducted these comparisons on a random subset of 10 000 sequences from the original set of 10^6 artificial TcRβ sequences at both 0% and 1% sequencing error. At both 0% and 1% sequencing error rates, Decombinator was >1000× faster than a pipeline using the standard Boyer-Moore algorithm (Table 3), demonstrating the benefits of using a finite-state automaton (FSA) for such a task, allowing the user to search for multiple keywords at the same time, rather than in a piecewise fashion. As expected, alignment with the shorter V and J keywords to assign V and J regions in artificial TcRβ sequences offered a significant improvement to alignment with the full-length V and J gene segments, as expected, but was still >100× slower than Decombinator (Table 3).

Finally, we investigated the effects of the modification we introduced to the classic Aho-Corasick pattern-matching algorithm, to account for sequencing error. Performance was equivalent on sequences without error, as expected. However, once a sequencing error rate of 1% was introduced, a further 10% of sequences were lost when using an approach that did not incorporate our modification, resulting in only 74.0 and 78.5% of sequences correctly assigned for α and β sequences, respectively. Moreover, the implementation of our modification resulted only in minor reductions in efficiency (see Tables 1 and 4).

### 3.2 Analysis of published TcRβ sequences

Having tested the algorithm on simulated datasets where assignment and missassignment rates could be measured directly, we next tested performance on a set of published HTS TcRβ sequences (Warren et al., 2011). Datasets of varying depth were available for TcRβ sequences between 100 and 150 bp in length amplified from three individuals by 5' RACE, and sequenced using an Illumina GAIIx analyser. Decombinator identified 44.2% of the sequence reads from Male 1, obtained from two separate samples. A total of 1 098 894 sequences were assigned for Male 1, of which 386 643 were distinct. Counting only distinct reads (to avoid bias because of selective antigen-driven clonal expansion), the distributions of V and J region usage obtained from analysis of three different individuals are shown in Figure 5. V and J region usage was non-uniform as described previously (Freeman et al., 2009), but it showed a similar overall pattern between the three individuals.

In addition to determining V and J gene usage, the algorithm determines the distribution of the deletion of nucleotides from the 5'- and 3'-ends of V and J, respectively, and the length of insert. Figures 6–8 show the distribution of these three parameters for Male 1. The data clearly show the non-random distribution of both deletion and addition processes, and they are in broad agreement with previously published estimates (Freeman et al., 2009).

### Table 1. Performance of Decombinator on 10^6 in silico α and β sequences with 0, 0.1 and 1% sequencing error

<table>
<thead>
<tr>
<th>TcR chain</th>
<th>Sequence error (%)</th>
<th>Assigned</th>
<th>Misassigned</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>54.7</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>49.7</td>
</tr>
<tr>
<td>A</td>
<td>0.1</td>
<td>98.8</td>
<td>0.9</td>
<td>56.5</td>
</tr>
<tr>
<td>B</td>
<td>0.1</td>
<td>98.8</td>
<td>0.1</td>
<td>52.5</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>88.1</td>
<td>0.8</td>
<td>70.0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>88.0</td>
<td>1.0</td>
<td>57.1</td>
</tr>
</tbody>
</table>

A minimum of 88% of sequences are assigned even at an assumed error rate of 1%, whereas ~99% of sequences are successfully assigned at an error rate of 0.1%. Sequences that are not assigned are assumed to be junk sequences and are discarded.

### Table 2. Performance of a pipeline where classification is determined by pairwise alignment (Table 1)

<table>
<thead>
<tr>
<th>TcR chain</th>
<th>Sequence error (%)</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>39.329</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>19.731</td>
</tr>
<tr>
<td>A</td>
<td>0.1</td>
<td>34.171</td>
</tr>
<tr>
<td>B</td>
<td>0.1</td>
<td>19.559</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>34.211</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>21.305</td>
</tr>
</tbody>
</table>

### Table 3. Performance of Decombinator on 10 000 in silico TcRβ sequences in comparison with pipelines based on the Boyer-Moore algorithm (Boyer and Moore, 1977), pairwise alignment (Pages et al., 2012) using the full-length of V and J regions and pairwise alignment using only the 20 bp V keywords and 12 bp J keywords

| Time(s): Decombinator | 0.58 | 0.59 |
| Time(s): Boyer-Moore  | 603.3| 653.8|
| Time(s): Alignment    | 245.2| 230.3|
| Time(s): Alignment (keywords) | 71.6 | 108.8 |

Having established the frequency distributions of V and J usage in a typical individual (Male 1), we used these distributions to create a further artificial set of in silico TcRβ sequences in which V and J frequency was determined by the observed frequency usage in Male 1, rather than being considered uniform. The performance of Decombinator in assigning correct identifiers to this new set of in silico sequences was equivalent to that obtained using the simpler uniform model (data not shown), confirming that the non-uniform V and J distributions do not introduce any significant bias into assignment efficiency.

We used the results of our classification algorithm to further investigate the degree of overlap between blood samples taken...
from individuals at two different time points. Figure 9 shows the number of distinct sequences (as determined by the Decombinator identifiers) that are shared between two independent blood draws from the same individual (plots of same colour for both draws, leading diagonal), or between different individuals (non-diagonal elements). The number of common sequences suggests that the degree of overlap between samples from different individuals is an order of magnitude smaller than that between two samples from the same individual. This is confirmed by calculating the Jaccard index, $J$, an index of set overlap (Table 5), as $J$ is $O(10^{-5})$ for samples from the same individual and $O(10^{-4})$ for samples from different individuals. However, even the maximum degree of overlap between any two samples is small relative to the sample size, suggesting that only a small proportion of the available repertoire is sampled even within one single individual.

The distribution of clonotype size within the overlapping sets was further examined. We let $A$ be the sample from Male 1 at day 1, $B$ the sample from Male 1 at day 8 and $C$ to be the sample from Male 2 at day 8. Of all identifiers found in $A \cup B$, the 50 most frequent were all also found in $A \cap B$, implying that the overlapping sets are composed primarily of highly abundant TcRs, perhaps amplified in response to antigen. In contrast, none of the 50 most frequent identifiers in $A \cup C$ are found in $A \cap C$. Overlap between different individuals does not, therefore, simply reflect clonal expansion, but may reflect some constrained feature of the mechanism for generation of diversity, leading to the production of 'public' clonotypes.

### Table 4. Performance of Decombinator without any modification to the standard Aho–Corasick algorithm

<table>
<thead>
<tr>
<th>TcR chain</th>
<th>Sequence error (%)</th>
<th>Assigned</th>
<th>Misassigned</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>54.7</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>49.7</td>
</tr>
<tr>
<td>A</td>
<td>0.1</td>
<td>97.1</td>
<td>0.01</td>
<td>55.4</td>
</tr>
<tr>
<td>B</td>
<td>0.1</td>
<td>97.6</td>
<td>0.1</td>
<td>49.3</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>74.0</td>
<td>0.05</td>
<td>47.3</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>78.5</td>
<td>0.3</td>
<td>42.2</td>
</tr>
</tbody>
</table>

The implementation was tested on the same set of 10^6 in silico $\alpha$ and $\beta$ sequences with 0, 0.1 and 1% sequencing error. Without modification of the classic Aho–Corasick method, another 10% of sequences is lost (see also Table 1). This modification does not significantly affect the algorithm's efficiency.

Fig. 5. Frequency distribution of V (left) and J (right) gene usage in distinct sequences obtained from three individuals. Non-uniform usage is clearly apparent for both sets of gene segments, whereas the overall pattern of usage seems to be largely conserved across multiple individuals. The distributions of usage are based on sequences obtained from three individuals, from two separate blood draws (Warren et al., 2011), using only distinct sequences to avoid biases associated with the analysis of clonally expanded populations of cells.
features as V and J region use and the distribution of germline deletions. Moreover, in using Decombinator to process raw sequence reads, the resultant file of identifiers produced makes further downstream analyses far easier to conduct for the uninitiated, taking away the challenging step of dealing with unprocessed data and translating it to a form which is accessible to even the most inexperienced programmer.

Having defined the five-part identifier, the main aim of this study was, therefore, to develop an efficient way of mapping raw HTS sequence data to the identifier. Decombinator is an FSA based on a modified keyword trie, incorporating goto, failure and output functions that allow fast pattern matching by using information from characters that have already been matched. This approach is, therefore, completely different from previous studies using HTS to determine TcR repertoire diversity, which have relied on common methods, such as pairwise alignment (Ndifon et al., 2012) and BLAST-like alignment tool (BLAT) (Kent, 2002; Klarenbeek et al., 2010). These methods (or variants thereof) have been successful in the context of large-scale genome sequencing studies, where the objective is to assign a series of

**Fig. 6.** Distribution of germline V deletions. Non-uniform distribution of V deletions is apparent, in broad agreement with previously published data (Freeman et al., 2009). The distribution is based on sequences obtained from Male 1 from two separate blood draws, using only distinct sequences to avoid biases associated with the analysis of clonally expanded populations of cells

**Fig. 7.** Distribution of germline J deletions. As with germline V deletions, a non-uniform distribution of J deletions is apparent, in broad agreement with previously published data (Freeman et al., 2009). The distribution is based on sequences obtained from Male 1 from two separate blood draws, using only distinct sequences to avoid biases associated with the analysis of clonally expanded populations of cells

**Fig. 8.** Distribution of the number of nucleotides found between 3' V and 5' J. Sequences were obtained from Male 1 from two separate blood draws, using only distinct sequences to avoid biases associated with clonally expanded populations. The region between 3' V and 5' J includes any remnants of the D region that remains after germline deletions. The distribution is quasi-normal, centred on a mean insert length of ~12 or 13 bp

**Fig. 9.** Overlap of shared sequences between Male1, Male2 and Female from sequences obtained from the study described in (Warren et al., 2011). In each Venn diagram, the blood sample from Day 1 is shown on the left and the sample from Day 8 on the right. The numbers in each Venn diagram represent (l–r) the number of distinct sequences found in the respective sample taken on Day 1, the number of distinct sequences common to both samples and the number of distinct sequences found in the respective sample taken on Day 8. Two separate samples, even from the same individual (leading diagonal), show only partial overlap, but display a greater proportion of shared sequences than samples taken from two different individuals
Comparing two separate samples from the same individual yields Jaccard index values of $O(10^{-3})$, whereas samples from two different individuals yields Jaccard index values of $O(10^{-4})$, indicating far greater overlap of sequences observed from samples from the same individual (see Fig. 9).

Table 5. Jaccard index for set similarity between two blood draws

<table>
<thead>
<tr>
<th></th>
<th>Male1</th>
<th>Male2</th>
<th>Female</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male1</td>
<td>0.029</td>
<td>0.0002</td>
<td>0.0004</td>
</tr>
<tr>
<td>Male2</td>
<td>0.001</td>
<td>0.013</td>
<td>0.0008</td>
</tr>
<tr>
<td>Female</td>
<td>0.0007</td>
<td>0.0004</td>
<td>0.027</td>
</tr>
</tbody>
</table>

In practice, Decombinator demonstrated remarkable efficiency, improving on equivalent pairwise matching algorithms by several orders of magnitude while retaining a high degree of accuracy. Although it is possible that fine tuning the parameters of pairwise algorithms specifically in the context of TcR data could improve their efficiency somewhat, it seems unlikely that they would achieve anything like comparable speed on the current datasets. However, a weakness of the original Aho–Corasick FSA was that it required an exact match between query and target, and indeed many attempts have been made to extend the strategy to accommodate error or uncertainty in the query (Ukkonen and Wood, 1993). The major causes of mismatch are sequencing errors, whose frequency is well known (Luo et al., 2012; Volpe et al., 2006) still struggle to deal with the vast number of sequences obtained from HTS.

In practice, Decombinator demonstrated remarkable efficiency, improving on equivalent pairwise matching algorithms by several orders of magnitude while retaining a high degree of accuracy. Although it is possible that fine tuning the parameters of pairwise algorithms specifically in the context of TcR data could improve their efficiency somewhat, it seems unlikely that they would achieve anything like comparable speed on the current datasets. However, a weakness of the original Aho–Corasick FSA was that it required an exact match between query and target, and indeed many attempts have been made to extend the strategy to accommodate error or uncertainty in the query (Ukkonen and Wood, 1993). The major causes of mismatch are sequencing errors, whose frequency is well known (Luo et al., 2012). In theory, polymerase chain reaction error can introduce mismatches, although in practice the rate of polymerase chain reaction error using high-fidelity polymerases is much lower than sequencing error and can be largely ignored. To accommodate a realistic degree of sequencing error without compromising too much on efficiency, we developed a straightforward modification of the FSA, which identified sequences even if the location of the tags contained a single error. The algorithm delivered a significant increase in the proportion of sequences that could be assigned, even at error rates of 1%, which lies at the upper bound of that observed experimentally. Further modifications that could accommodate two or even more mismatches are unlikely to generate major benefits, as the chances of having two sequence errors within a typical 20 bp tag are low. However, we are exploring whether slower methods, including pairwise alignment or hidden Markov models might be useful in characterizing the small proportion of sequences which cannot be assigned by Decombinator.

The second feature that emerges immediately from analysis of the sequence data is the enormous diversity of TcRs which exist, which is reflected in the small overlaps observed between samples. The limited overlap between repeat samples from the same individual presents an obvious potential drawback, as each sample only captures a small proportion of the sequences present in that individual. Only sequences that occur at relatively high frequency are, therefore, likely to be sampled consistently by this approach. However, as the sequences at higher frequency are in fact likely to be those associated with specific immune responses, this may not prove to be a major limitation in studies focusing on events associated with antigen-specific challenges.

The overlap between samples of different individuals is even smaller, reflecting not only differences in repertoire generation (e.g. the effects of major histocompatibility complex polymorphism and different histories of antigen exposure) but also the larger possible pool of all possible TcR sequences. Given the enormous number of possible sequences, it is in fact rather surprising that there are any sequences in common at all between any two randomly chosen individuals. The sequences found are often present at rather low frequency, suggesting that they may not reflect responses to antigen. Instead, such common sequences may correspond to ‘public’ sequences (Venturi et al., 2008; Sainz-Perez et al., 2012) described previously. Like preferential V and J region usage, these sequences found at unexpectedly high frequencies within the population may reflect some specific feature of the mechanisms for generation of diversity. Further work to investigate the characteristics of these common sequences is in progress.

also worth commenting on some features of the output generated from the published set of sequences we analyse in the article (Warren et al., 2011). One striking feature is the non-uniform nature of the distribution of the different indices of the identifier. Even after restricting the analysis to distinct TcRs (i.e. counting each different identifier only once), so as to remove the potential effects of clonal expansion, both V and J usage is strikingly non-uniform. Vβ20 — 1, for example, is found at a much higher frequency, whereas Vβ15 or Vβ16 are rarely present. This pattern is, at least in part, conserved across three unrelated individuals, making it extremely unlikely that it reflects any exposure to specific antigen. The pattern observed is similar to that described in several previous studies, (Freeman et al., 2009; Warren et al., 2011), suggesting that it is not an effect of bias introduced either by experimental or computational methodology. A recent HTS study of mouse V and J usage found similar bias, which was attributed to constraints imposed by physical features of the chromosome structure (Ndifon et al., 2012). Non-uniform distributions of the number of deletions and insertions has also been observed previously (Freeman et al., 2009; Robins et al., 2009), and it presumably reflects molecular features of the recombinase machinery. Overall, learning the underlying distributions of each facet of the overall recombination process from this type of sequence data will be an important objective for future work, and it will allow us to build more realistic computational models for repertoire generation.
5 CONCLUSION

In conclusion, we describe a five-part identifier that uniquely classifies all TcR sequences, and a computational tool that maps HTS reads to this identifier efficiently and accurately. The computational tool is based on the classic approach of Aho and Corasick to pattern matching, but it crucially includes a novel modification to correct for sequencing error. These tools, and the increasing application of HTS technology to lymphocyte antigen receptor analysis, will lead to a better understanding of the rules that regulate the TcR repertoire. The Decombinator package is implemented in Python (v2.6) and is freely available at https://github.com/uclinfectionimmunity/Decombinator along with full-documentation and examples of typical usage.

Funding: This work was supported by the Engineering and Physical Sciences Research Council and the Medical Research Council, UK.

Conflict of Interest: none declared.

REFERENCES