Antigen receptor sequence reconstruction and clonality inference from scRNA-seq data

Ida Lindeman[1,2] and Michael J. T. Stubbington[1]

[1] Wellcome Sanger Institute, Hinxton, Cambridgeshire, CB10 1RQ, United Kingdom.

[2] KG Jebsen Coeliac Disease Research Centre, University of Oslo, 0372 Oslo, Norway.

Corresponding author e-mail address: mjt.stubbington@gmail.com

Running Head: Single-cell antigen receptor sequence reconstruction

Abstract

In this chapter, we describe TraCeR and BraCeR, our computational tools for reconstruction of paired full-length antigen receptor sequences and clonality inference from single-cell RNA-seq (scRNA-seq) data. In brief, TraCeR reconstructs T cell receptor (TCR) sequences from scRNA-seq data by extracting sequencing reads derived from TCRs by aligning the reads from each cell against synthetic TCR sequences. TCR-derived reads are then assembled into full-length recombined TCR sequences. BraCeR builds on the TraCeR pipeline and accounts for somatic hypermutations (SHM) and isotype switching. Here we discuss experimental design, use of the tools, and interpretation of the results.

Key words

TCR, BCR, immunoglobulin, single cell, RNA-seq, scRNA-seq, antigen receptor reconstruction, tracer, bracer

1. Introduction

T cells and B cells recognize antigens in a highly specific manner through their cell-surface T-cell receptor (TCR) or B-cell receptor (BCR). These receptors are extremely diverse heterodimers comprising a TCRα- and a TCRβ-chain (αβ T cells), a TCRγ- and a TCRδ-chain (γδ T cells) or a heavy (IgH) and a light (Igκ or Igλ) chain (B cells) encoded by genes generated through V(D)J recombination during the development of the cell in the thymus or bone marrow. High-throughput antigen receptor sequencing (Rep-seq) of a single type of chain in bulk populations has been a common strategy for characterisation of BCR- and TCR-repertoires ***(1-3)***, but lacks information about chain pairing [reviewed in ***(4,5)***]. Single-cell Rep-seq is useful to decipher paired antigen receptor repertoires, but provides very limited additional information

about the cells. While Rep-seq in combination with phenotyping primers can give information on the expression of a selected panel of genes in addition to the paired antigen receptor ***(6)***, sequencing of the entire transcriptome is a much more informative and unbiased approach [reviewed in ***(7)***].

During the last few years, single-cell RNA-sequencing (scRNA-seq) has been an extremely valuable approach for identifying and characterizing heterogeneity in cell subsets and cells in various differential states both in health and in disease [reviewed in ***(8)***]. The development of computational tools allowing researchers to reconstruct TCR- and BCR-sequences directly from scRNA-seq data thus provides a unique opportunity to gain insight into T- and B-cell immunity, cell fate, lineage evolution and antigen-specific responses by linking antigen receptor usage to the full transcriptomic identity of individual T- or B-cells.

Here we describe TraCeR ***(9)*** and BraCeR ***(10)***, our computational tools for reconstruction of paired full-length antigen receptor sequences and clonality inference from scRNA-seq data. During the last two years several other tools for TCR- and/or BCR-reconstruction from scRNA-seq data have emerged ***(11-15)***, illustrating that there is a high level of interest in approaches such as these.

TraCeR reconstructs TCR sequences from scRNA-seq data by extracting sequencing reads derived from TCRs by aligning the reads from each cell against synthetic TCR sequences representing all possible combinations of V- and J-segments. TCR-derived reads are then assembled into full-length TCR sequences. BraCeR builds on the TraCeR pipeline, accounting for somatic hypermutations (SHM) and isotype switching.

We have previously demonstrated an application for TraCeR by investigating CD4+ T-cell clonotypes in the spleen of mice as a response to a *Salmonella* infection ***(9)*** where members of each expanded T cell clone were found across various proliferation and differentiation states.

More recently, the use of TraCeR in combination with pseudotime and branching inference revealed that the progeny of a single naïve murine CD4+ T cell can be found in both T_H1 and T_{FH} compartments during an immune response to malaria ***(16)***. Patil *et al.* demonstrated clonal sharing between a population of CD4+ cytotoxic T lymphocyte (CD4-CTL) precursors and effector memory CD4-CTLs ***(17)***. Furthermore, TraCeR has been used to map regulatory T-cell (Treg) clones and memory T-cell clones across lymphoid and non-lymphoid tissues in a study focusing on identifying trajectories of tissue adaptation ***(18)***.

2. Materials

In this chapter we use the notions "[TB]raCeR" ("TraCeR" or "BraCeR") and "[tb]racer" ("tracer" or "bracer") in order to avoid unnecessary repetitions when describing both tools.

2.1 Sequencing data

The following aspects should be taken into consideration when choosing a library preparation protocol and sequencing platform for generation of data as input to [TB]raCeR.

1. Choose a library preparation protocol that generates sequencing reads from the full length of mRNA transcripts (*see* **Note 1**)
2. Paired-end (PE) reads provide the maximum reconstruction rate and accuracy compared with single-end (SE) reads.
3. Sequence your library with a minimum read length of 50 bases (*see* **Note 2**).
4. The read depth required to reconstruct TCRs or BCRs from a single cell depends on the cell type and activation state (*see* **Note 3**).
5. Make sure the reads are de-multiplexed according to the cell of origin after sequencing.
6. Perform basic quality control of the raw reads (*see* **Note 4**).

7. [TB]raCeR accepts FASTQ files (fastq or fastq.gz) as input. BraCeR also accepts assembled BCR sequences in FASTA format for clonality inference (*see* **Section 3.2.4**).

2.2 Prerequisites

2.2.1 External tool requirements for [TB]raCeR

1. Python (>=2.7.0) (*see* **Note 5**)
2. Bowtie 2 ***(22)***
3. Trinity ***(23)*** (*see* **Note 6**)
4. IgBLAST ***(24)*** (*see* **Note 7**)
5. Kallisto ***(25)*** or Salmon ***(26)*** (*see* **Note 8**)
6. Graphviz (*see* **Note 9**)

2.2.2 Additional prerequisites specific to BraCeR (*see* Note 10)

1. Python (>=3.4.0)
2. BLAST ***(27)***
3. Trim Galore!
4. PHYLIP dnapars
5. R (>= 3.1.2) and R packages for lineage reconstruction: ggplot2, Rscript, Alakazam

2.3 Installing [TB]raCeR

2.3.1 Installing [TB]raCeR from GitHub

1. Download or clone the GitHub repository (https://github.com/Teichlab/tracer or https://github.com/Teichlab/bracer) with *git clone.*

2. Install all required prerequisites.
3. Set up Python dependencies (*see* **Note 11**).
4. Install tracer/bracer module with *python setup.py install* (*see* **Note 12**).
5. Edit configuration file (*see* **Note 13**).

2.3.2 Running [TB]raCeR as a standalone Docker image

Alternatively, [TB]raCeR can be run as a standalone Docker image on DockerHub, with all of the dependencies installed and configured appropriately (*see* **Note 14**).

1. Pull the Docker container from DockerHub with *docker pull teichlab/[tb]racer*.
2. Increase the memory limit for Docker to 6-8 GB (*see* **Note 15**).
3. Run the following command, followed by any appropriate arguments, from your input data directory: *docker run -it --rm -v $PWD:/scratch -w /scratch teichlab/[tb]racer*

2.4 Testing [TB]raCeR

1. Run [*tb]racer test* with optional arguments (*see* **Note 16**).
2. Compare the output in *test_data/results/filtered_[TB]CR_summary* with the expected results in *test_data/expected_summary* (*see* **Note 17**).

3. Methods

3.1 [TB]raCeR pipeline

The [TB]raCeR pipelines consist of two main steps (**Fig 1**):

1. Reconstruction of TCR/BCR sequences from each cell (*assemble* command)
2. Creation of clonal networks (*summarise* command)

3.2 Reconstruction of TCR/BCR sequences with *Assemble*

3.2.1 Overview of pipeline

The *assemble* stage performs the steps:

1. Trimming of raw reads to remove adapter sequences and low quality sequences (BraCeR only, *see* **Note 18**).
2. Extract TCR/BCR-derived reads by alignment to a combinatorial recombinome using Bowtie 2 ***(22)*** (*see* **Note 19**).
3. Perform a second round of alignment for IgH if the reads are 50 bases or shorter to extract reads mapping mainly or solely to the CDR3 (*see* **Note 20** and **Fig 2**).
4. Assemble TCR/BCR-derived reads into contigs using Trinity.
5. Detect isotype (BraCeR only, *see* **Note 21**).
6. Determine productivity and gene usage of reconstructed sequences.
7. Collapse highly similar sequences.
8. Quantify the expression of each reconstructed sequence and filter based on expression.

3.2.2 Preparing input for [TB]raCeR

[TB]raCeR takes as input FASTQ files containing sequencing reads generated from a *single* cell. Thus, data must be demultiplexed such that reads from each cell are identified and written to separate files. Data can be paired-end (PE) or single-end (SE) with PE providing higher sensitivity sequence reconstruction.

Although it is beyond the scope of this chapter to provide instructions for quality control of scRNA-seq data, we recommend that users exclude apparently poor-quality cells from any downstream analyses. This can be done using automated methods ***(29)*** or by manual inspection

of metrics such as total number of mapped reads, percentage of reads mapping to the mitochondrial genome, rate of mapping to the transcriptome, and number of genes detected.

3.2.3 Running [TB]raCeR in *Assemble* mode with default settings

Run [TB]raCeR *assemble* with the main arguments described below. Note that the two tools have slightly different usage (see below), but many of the arguments are the same.

tracer assemble [options] <file_1> [<file_2>] <cell_name> <output_directory>

bracer assemble [options] <cell_name> <output_directory> [<file_1>] [<file_2>]

1. <*file_1*> is the FASTQ file providing #1 mates from PE sequencing or all of the reads from SE sequencing. May be left blank if running BraCeR with --*assembled_file*.
2. <*file_2*> is the FASTQ file providing #2 mates for PE reads.
3. <*cell_name*> is a name that will be used for references to the cell.
4. <*output_directory*> is the directory for output, and should be identical for cells to be summarised together.

3.2.4 Running [TB]raCeR in *Assemble* mode with optional arguments

The following optional arguments can be passed to either TraCeR or BraCeR:

1. -*s*/--*species*: Species from which the cells were derived. The default is mouse (*Mmus*) for TraCeR and human (*Hsap*) for BraCeR.
2. --*loci*: Loci to reconstruct. Default is 'A B' for TraCeR and 'H K L' for BraCeR. Include or replace with 'G D' to attempt to reconstruct TCRγ and TCRδ.
3. --*single_end*: Set this flag if your data are SE reads.
4. --*fragment_length*: The estimated average fragment length of the sequencing library. Required for SE data.

5. --*fragment_sd*: The estimated standard deviation of the average fragment. Required for SE data.
6. -*p*/--*ncores*: The number of processor cores to use. Default=1.
7. --*resource_dir*: Path to directory containing resources required for alignment. Use if you wish to use other resources contained somewhere other than the default resources directory.
8. -*c*/--*config_file*: Path to the configuration file (*see* **Note 13**). Default = *~/.[tb]racerrc*.
9. -*r*/--*resume_with_existing_files*: If this flag is set, [TB]raCeR will look for existing output files and skip already completed steps.
10. --*max_junc_len*: The maximum allowed length of junction string in a recombinant identifier (*see* **Note 22**).

The optional arguments below are specific to TraCeR:

1. -*m*/--*seq_method*: Method for generation of sequences for output and productivity assessment. Options are -*m imgt* (default) and -*m assembly* (*see* **Note 23**).
2. -*q*/--*quant_method*: Method used for expression quantification (*kallisto* or *salmon*).
3. --*small_index*: Set this flag for faster expression quantification if you have prebuilt a transcriptome index (*see* **Note 24**).
4. --*invariant_sequences:* Path to file specifying invariant sequences for particular unconventional T cell types. For an example, see the default file at *resources/Mmus/invariant_cells.json*.

The optional arguments below are specific to BraCeR:

1. --*assembled_file*: Path to a FASTA file with pre-assembled sequences for a cell. Makes BraCeR skip the alignment and assembly steps (*see* **Note 25**).
2. --*no_trimming:* Do not remove adapter sequences and low quality reads.

3. *--keep_trimmed_reads:* Keep the output files from the trimming step.

3.2.5 The sequence identifier format

In order to facilitate comparison of sequences between cells, the IgBLAST results for each TCR/BCR sequence within a cell are represented by a sequence identifier string (e.g. TRBV31_AGTCTTGACACAAGA_TRBJ2-5). This sequence identifier format consists of:

1. Most likely V gene name.
2. Junctional or CDR3 nucleotide sequence (*see* **Note 26**).
3. Most likely J gene name.

In cases where the V- or J-gene assignments are uncertain, [TB]raCeR uses a list of all possible sequence identifiers for comparisons. The sequence identifiers are not used directly by BraCeR for clonality inference, but serve as additional information in the clonotype networks.

3.2.6 Output of *Assemble*

The output of the *assemble* step is an */<output_directory>/<cell_name>* directory for each cell, containing all or most of the following subdirectories:

1. *trimmed_reads*: Contains reads trimmed by Trim Galore! if *bracer assemble* is run with *--keep_trimmed_reads*.
2. *aligned_reads:* Contains Bowtie 2 output with TCR/BCR-derived reads.
3. *Trinity_output*: Contains FASTA files with assembled contigs for each locus, as well as two log files.
4. *IgBLAST_output*: Contains IgBLAST output for the contigs from each locus.
5. *BLAST_output*: Contains BLAST output for the contigs from each locus.

6. *unfiltered_[TB]CR_seqs*: Contains files detailing all reconstructed TCR/BCR sequences (*see* **Note 27**).
7. *expression_quantification*: Contains output of Kallisto/Salmon for the entire transcriptome and the TCRs/BCRs (*see* **Note 28**).
8. *filtered_[TB]CR_seqs:* Contains the two most highly expressed recombinants for each locus.

3.3 Creation of clonotype networks and lineage trees with *Summarise*

3.3.1 Defining clonally related recombinants

Both productive and non-productively rearranged TCR sequences are considered clonally related if they share a sequence identifier, meaning that they share assignment of V- and J gene and the junctional sequence. Clonally related productively rearranged BCR sequences are identified for each locus with the Change-O toolkit ***(31)*** based on the following criteria (**Fig 3**):

1. Common V- and J-gene in the sets of potential V- and J-genes between the sequences.
2. Equal CDR3 length.
3. CDR3 nucleotide distance normalised by length < 0.2 (*see* **Note 29**).

3.3.2 Generation of clonal networks

We use custom scripts to assess the clonal groups and generate network graphs as follows.

1. Each single cell is represented by a node in the graph.
2. Reconstructed sequences are represented within nodes by horizontal lines coloured according to locus and productivity or by the sequence identifier.
3. Edges between the nodes represent clonally related TCR/BCR sequences, and are colour coded according to locus. Edges between B cells are only drawn if they share a clonally

related productive IgH and a clonally related productive Igκ or Igλ (*see* **Note 30**).

4. Edge thickness is proportional to the number of shared sequences for a locus.
5. Non-productively rearranged BCR sequences are determined to be shared within a clone group and included as edges in the graph if they have overlapping V- and J-gene assignments. If the cells only share a non-productive chain for a specific locus (Igκ or Igλ), this is shown with a dotted instead of a solid line in the clonal network.

3.3.3 Construction of immunoglobulin lineage trees

BraCeR offers a complete pipeline based on both heavy and light chains for construction of lineage trees through Change-O, Alakazam ***(34)*** and PHYLIP ***(35)***, consisting of the following:

1. Build IgBLAST reference databases using IMGT-gapped sequences (*see* **Note 31**).
2. Run IgBLAST on all sequences belonging to a clone group.
3. Parse IgBLAST output and create Change-O database.
4. Add clone number, isotype and cell name to the Change-O database for each sequence.
5. Reconstruct the germline sequences (with masked junction) in each clone group with Change-O *CreateGermlines*.
6. Concatenate productive heavy and light chain shared in each clone group (*see* **Note 32**).
7. Run the appropriate Alakazam commands through our *lineage.R* script (*see* **Note 33**).

3.3.4 Running [TB]raCeR in *Summarise* mode

Run the [TB]raCeR *summarise* command with options as described below. <*input_dir*> is the directory containing subdirectories of each cell you want to summarise (*see* **Note 34**).

[tb]racer summarise [options] <input_dir>

The following optional arguments can be passed to either TraCer or BraCeR:

1. *-c/--config_file*: Path to the configuration file (*see* **Note 13**). Default = *~/.[tb]racerrc.*
2. *-u/--use_unfiltered*: Set this option to run *summarise* with all reconstructed recombinants without filtering cases where more than two sequences are detected for a particular locus.
3. *--resource_dir:* Path to directory containing resources required for alignment. Use if you wish to use other resources contained somewhere other than the default resources directory.
4. *-s/--species:* Species of origin. Default = *Mmus* (mouse) for TraCeR and *Hsap* (human) for BraCeR (*see* **Note 35**).
5. *--loci*: Space-separated list of loci to summarise (*see* **Note 36**).
6. *-g/--graph_format*: Output format of clone networks (*see* **Note 37**).
7. *--no_networks*: Do not draw clonotype network graphs (*see* **Note 38**).

The following optional arguments are specific to TraCeR:

1. *--receptor_name*: Specify if other than "TCR" when using the *Build* module.
2. *-i/--keep_invariant*: Set this option to keep invariant cells (*see* **Note 39**).

The following optional arguments are specific to BraCeR:

1. *--IGH_networks*: Base clonality solely on IgH, allowing clone groups with different or no light chain.
2. *--dist*: Distance value (float) for clonal inference. Default=0.2 (*see* **Note 40**).
3. *--include_multiplets*: Set if you do not wish to exclude potential cell multiplets from downstream analyses.
4. *--infer_lineage*: Attempt lineage tree construction for clone groups.

3.3.5 Output of the TraCeR summarise step

The output of the TraCeR summarise step is written to *filtered_TCR<loci>_summary* or *unfiltered_TCR<loci>_summary*. The following output files are generated:

1. *TCR_summary.txt*: TCR reconstruction summary statistics file.
2. *recombinants.txt:* File listing the identifier, lengths and productivity of each reconstructed TCR for each cell.
3. *reconstructed_lengths_TCR[A|B].[pdf|txt]*: Distribution plots and underlying data displaying reconstructed VDJ region lengths for each locus.
4. *clonotype_sizes.[pdf|txt]:* Distribution of clonotype sizes shown as bar plots and underlying data.
5. *clonotype_network_[with|without]_identifiers.<graph_format>*: Clonotype networks in graphical format with recombinant identifiers or with lines representing the presence of recombinants for a locus in a cell.
6. *clonotype_network_[with|without]_identifiers.dot*: Clonotype networks described in the Graphviz DOT language.

3.3.6 Output of the BraCeR summarise step

The following output files and subdirectories may be generated (depending on options):

1. *BCR_summary.txt:* BCR reconstruction summary statistics file.
2. *changeodb.tab:* Database file describing all reconstructed sequences in single cells. Recombinants in suspected multiplets are included if run with --*include_multiplets*.
3. *filtered_multiplets_changeodb.tab:* Database file with reconstructed recombinants from suspected multiplets unless run with --*include_multiplets*.
4. *IMGT_gapped.tab:* Database file for all reconstructed sequences based on IMGT-gapped reference sequences.

5. *reconstructed_lengths_BCR[H|K|L].[pdf|txt]*: VDJ region length distribution plots with underlying data for the assembled BCRs for a locus.
6. *clonotype_sizes.[pdf|txt]*: Bar graph with underlying data visualising clonotype size distribution.
7. *clonotype_network_[with|without]_identifiers.<graph_format>:* Clonotype networks with recombinant identifier strings or lines denoting the presence of recombinants.
8. *clonotype_network_[with|without]_identifiers.dot:* Clonotype networks described in the Graphviz DOT language.
9. *lineage_trees/:* Subdirectory containing lineage trees if run with *--infer_lineage.*
10. Intermediate output files (*see* **Note 41**).

3.4 Quality control of output

A current challenge of scRNA-seq is being able to detect and filter out reads that are not in fact derived from a single cell, but rather from unintentional cell multiplets or cross-contamination due to PCR chimeras or free RNA from lysed cells ***(38)***. The number of reconstructed chains for a locus may be used to filter out multiple captures or potential contaminations because a single B- or T-cell should not have more than two recombined antigen receptor chains for a given locus. It is important to filter out such cells from the dataset as they otherwise could hinder correct clonotype inference. Furthermore, TraCeR and BraCeR are built on the assumption that each cell contains a maximum of two reconstructed sequences for each BCR/TCR locus, and BCR/TCR reconstruction from bulk samples or unintentional cell multiplets may therefore potentially give rise to some incorrectly reconstructed sequences. Filtering of suspected cell multiplets is done automatically for BraCeR, and can be employed manually for TraCeR.

3.4.1 Automatic cell multiplet detection

BraCeR identifies potential cell multiplets or cross-contamination if more than two recombined sequences are reconstructed for any one BCR locus in a cell. Such cells are then excluded from further analysis steps unless *summarise* was run with *--include_multiplets*.

3.4.2 Manual inspection of potential cell multiplets

The frequency of cell multiplets may vary from dataset to dataset, and the importance of removing potential cell multiplets versus the risk of filtering out false potential multiplets may also vary depending on the biological question and experimental setup. Filtering of potential multiplets could therefore be done with several degrees of strictness, and should be determined by the user for each individual dataset. Our general recommendations for manual inspection and removal of potential cell multiplets are (from more permissive to more restrictive filtering):

1. Run *[tb]racer summarise* with *--use_unfiltered*.
2. Create a new directory for cells to be filtered out.
3. Open the *[TB]CR_summary.txt* in the unfiltered summary folder and look at the section named "#Cells with more than two recombinants for a locus#". Take note of any cell that has more than three reconstructed sequences for any locus, and move the result folder from the assembly step for these cells to the new folder for filtered cells.
4. Open the *<cell_name>/unfiltered_[TB]CR_seqs/unfiltered_[TB]CRs.txt* file for each cell with more than two recombinants for a locus. Discard cell if all recombinants for the locus are substantially different from each other (*see* **Note 42**).
5. Look at the clonotype network using the unfiltered cells. If one cell containing multiple chains for a locus connects to two or more distinct clone groups with their own set of sequences not shared with other sub-clone-groups, the cell is likely to be a cell multiplet.
6. Depending on the desired balance of retaining potential cell multiplets versus discarding false multiplets, you could also take into account cells in which two distinct productive

recombinants have been reconstructed for more than one locus in a cell, e.g. T cells with two productive TCRα in addition to two productive TCRβ or B cells with two productive IgH and also two productive Igκ and/or Igλ. Such cells have a higher probability of being cell multiplets, although discarding them may mask true biological information.

7. Cells with two productive recombinants for a locus are expected at varying frequencies (e.g. TCRα: 30%, TCRβ: 2–10%, IgH: 2–5%, Igκ: 11% in mice) ***(40)***. If you observe significantly higher proportions in your data, you may wish to consider the likelihood that some of these represent doublets.

3.5 Interpreting TraCeR clonotype output

The clonal inference based on reconstructed TCRs is represented as graphical output with either horizontal lines indicating whether a recombinant for each locus is present (**Fig 4A**) or full recombinant identifiers (**Fig 4B**). The network without identifiers gives a good overview of the overall clonality in the cell population, whereas the network with identifiers only shows clonally expanded cells and details the identity of the shared TCR sequences.

The networks shown in **Fig 4** show edges between nodes representing cells that share one or more reconstructed sequences. Whether sharing of TCR sequences can be seen as evidence of clonality depends on how strictly you wish to define clonality. Given that detection sensitivity is not 100% (and depends on various experimental and biological parameters) all the TCR sequences present in a cell may not always be reconstructed. Clone groups must therefore be inferred based on various levels of evidence and interpreted by the user depending on the biological questions to be answered. Here we discuss a few patterns that could be observed in the clonotypes networks, exemplified by clone groups in **Fig 4A**.

1. Many small clone groups consist of cells all sharing the same productive TCRα and productive TCRβ (e.g. the green, pink, purple yellow, orange, grey and red clones).
2. Many of the clone groups also exhibit sharing of additionally reconstructed chains when these were detected (pink, turquoise, yellow and orange clones). Sharing of such additional TCR sequences within a clone group strengthens the evidence of correct clonal assignments due to the extremely small likelihood that two independent cells would undergo the same complete set of recombination events during development in the thymus.
3. It is possible for a single TCRβ to be found in combination with multiple TCRα because developing T cells first recombine the TCRβ-locus and proliferate before recombining their TCRα-loci. Examples of this can be seen as sub-clones sharing both a TCRα and TCRβ within a larger clone group of cells only sharing a TCRβ (turquoise clone). Such groups may indicate that, in these cases, the TCRβ is important in conferring antigen specificity.
4. In some cases, the only shared TCR sequence may be non-productive or a TCRα due to failed reconstruction of other chains (e.g. blue clone). We cannot be certain that the cell belongs to the clone group, but we also have no evidence to the contrary.
5. Some strange clone groups may appear as cells all sharing different single TCR chains (e.g. blue-grey clone). This is likely not a real clone group, as all of the cells have a TCRβ and TCRα reconstructed, but some of them share only a TCRα while others share only a TCRβ, but not the same TCRβ for all the cells.
6. The presence of nodes connecting two or more smaller clone groups could be an indication of unsuccessfully removed cell multiplets (not seen in **Fig 4A**).

3.6 Interpreting BraCeR clonotype output

3.6.1 Clonotype networks

As for TraCeR, the clonal inference based on reconstructed BCRs is represented as graphical output with horizontal lines indicating whether a clonally related recombinant for each locus is present (**Fig 5A**) or full recombinant identifiers (**Fig 5B**). Here we discuss a few patterns that could be observed in the BraCeR clonotype networks, exemplified by clone groups in **Fig 5A**.

1. Unless BraCeR is run with *--IGH_networks*, all the cells in each clone group are required to share at least one productive IgH and one productive Igκ or Igλ. This requirement makes the clonal assignments fairly certain.
2. If BraCeR is run with *--IGH_networks*, larger clone groups with cells sharing a clonally related IgH may consist of sub-clones sharing a specific light chain. This cannot be exemplified by **Fig 5A** as, in this instance, BraCeR was not run with *--IGH_networks*.
3. Sharing of additional reconstructed BCR sequences, either productive or non-productive, within a clone group strengthens the evidence of correct clonal assignments due to the extremely small likelihood that two independent cells would undergo the same complete set of recombination events during development in the bone marrow.
4. Some cells have additionally reconstructed chains that are not shared within the clone group (e.g. one cell in the largest clone group having two productive Igλ). This could be due to varying expression levels and hence differences in reconstruction sensitivity, technical issues such as contamination or misassemblies, or display true biological variability (*see* **Note 43**).
5. Clone groups spanning different isotypes and subtypes of main isotypes may be observed (e.g. two of the clone groups spanning IgA1 and IgG1), indicating that members of the clone have undergone class-switching.

3.6.2 Lineage trees

The lineage trees resulting from running BraCeR with --*infer_lineage* are useful to acquire more information about the similarity of the clonally related sequences within a clone group and how they may have evolved through affinity maturation. These lineage trees are built using maximum parsimony (*see* **Note 44**) with the inferred combined heavy and light chain germline sequence as outgroup (black node) and inferred intermediate sequences not observed in the sample as white nodes (**Fig 6**). The larger nodes in each lineage tree are labelled with the cell name(s) containing the sequence representing each node, and the background colour of each node corresponds to the isotype(s) of the IgH. The size of each node is proportional to the number of cells in which the sequence was reconstructed.

3.6.3 Further repertoire analysis using external tools

The output of BraCeR may be further analysed with other available tools for BCR repertoire analysis such as the Change-O suit for analysis of SHM, lineage reconstruction and repertoire diversity. BraCeR aims to follow common data standards as they are being outlined by the Adaptive Immune Receptor Repertoire (AIRR) community ***(45)*** in order to facilitate use of external tools (*see* **Note 45**). Most current tools for BCR repertoire analysis are designed for high-throughput repertoire sequencing data of bulk samples, and do not automatically deal with paired heavy and light chain sequences. Some practical guidelines for BCR sequencing repertoire analysis have been reviewed in ***(46)***.

3.7 Building resources with *Build*

3.7.1 Introduction

The *Build* mode of [TB]raCeR creates the required resources from user-specified reference sequences. It can be used to run [TB]raCeR for species other than human or mouse, or to use a particular set of reference sequences. The *Build* mode creates synthetic reference sequences called combinatorial recombinomes, consisting of every combination of V alleles and J alleles, with a masked junctional region and leader sequence to allow mapping of TCR- or BCR-derived reads to a reference. For TCRs, the first ~260 nucleotides of the constant region gene are then appended to each synthetic sequence for the locus to allow mapping of reads running into the C region (*see* **Note 46** for BCRs). *Build* also creates databases compatible with IgBLAST and BLAST.

3.7.2 Running Build

1. Download ungapped V, D and J reference sequences from IMGT or another repository (*see* **Note 47**).
2. If you are building resources for BraCeR, also download IMGT-gapped V reference sequences from IMGT.
3. Download constant region (C) sequences from IMGT or Ensembl (*see* **Note 48**).
4. For each locus, run one of the following commands, inserting your choices and optional arguments (*see* **Note 49**):

    tracer build <species> <receptor_name> <locus_name> <N_padding> <colour> <V_seqs> <J_seqs> <C_seqs> [<D_seqs>] [options]

    bracer build <species> <locus_name> <N_padding> <colour> <V_seqs> <J_seqs> <C_seqs> [<D_seqs>] [options]

TraCeR and BraCer both expect the following main arguments:

1. <*species*>: Species (e.g. Hsap).
2. <*locus_name*>: Name of locus (e.g. H)
3. <*N_padding*>: Number of ambiguous N nucleotides between V and J (*see* **Note 50**)
4. <*colour*>: HTML colour for productive recombinants (e.g. E41A1C) or *random*.
5. <*V_seqs*>: FASTA file containing V gene sequences
6. <*J_seqs*>: FASTA file containing J gene sequences
7. <*C_seqs*>: FASTA file containing single constant region sequence
8. <*D_seqs*>: FASTA file containing D gene sequences (optional)

TraCeR and BraCeR accept the following optional arguments:

1. *-f/--force_overwrite*: Forces the program to overwrite existing resources for the species.
2. *-c/--config_file:* Path to the configuration file (*see* **Note 13**). Default = *~/.[tb]racerrc*.
3. *--resource_dir:* Path to directory containing resources required for alignment. Use if you wish to use other resources contained somewhere other than the default resources directory.
4. -o/--*output_dir:* Path to write new resource files. New resources will be written to the default resource directory if not specified.

The following optional arguments are specific for BraCeR:

1. --*C_db*: Takes FASTA file containing all C gene sequences as argument if not all C gene sequences were used to make the combinatorial recombinomes.
2. --*V*_gapped: Takes FASTA file with IMGT-gapped V reference sequences as argument (highly recommended, *see* **Note 51**).
3. --*igblast_aux*: Takes an IgBLAST auxiliary file for the species as argument, aiding correct CDR3 assignments.

4. Notes

1. We use the Smart-seq2 protocol ***(19)***, with SmartScribe (Clontech) instead of SuperScript II (Invitrogen) for reverse transcription. A comparison of several available scRNA-seq protocols is presented by Svensson *et al.* ***(20)***. Data generated from droplet-based sequencing such as 10x Genomics Chromium 3' Single Cell RNA-sequencing are not suitable for TCR/BCR reconstruction by [TB]raCeR because they do not sequence full-length mRNA. However, BCR sequences obtained by the 10x Genomics Chromium Single Cell V(D)J kits may be used with the BraCeR pipeline for clonality analysis.
2. Read lengths of 50 bases or more are recommended for optimal reconstruction sensitivity and accuracy ***(9,21)***. If you wish to use read lengths shorter than 50 bases, you will be restricted to using the Inchworm component of Trinity (*see* **Note 13**).
3. We would generally recommend to sequence T- or B-cells with a read depth of 0.25-0.50 million PE reads per cell (in total 0.5-1.0 million reads per cell). While TCR- and BCR-sequences may be successfully reconstructed with lower sequencing depths ***(9,21)***, the optimal depth depends on the cell type, activation state and on how much information about the rest of the transcriptomic profile for each cell is needed for downstream analyses.
4. Sequencing facilities often run quality control as part of their pipelines and let you know if there were any technical challenges during the run. Alternatively, you can run FastQC (http://www.bioinformatics.babraham.ac.uk/projects/fastqc/) on the raw reads.
5. Python (>=3.4.0) is required for BraCeR. TraCeR works with both Python 2 and Python 3.
6. BraCeR requires Trinity version 2.4.0 or higher. This is also the recommended version for TraCeR, although earlier versions may also be used.
7. Please note that in addition to downloading the executable files from ftp://ftp.ncbi.nih.gov/blast/executables/igblast/release/<version_number> you must

download the *internal_data* directory and put it into the same directory as the igblast executable. Run *export IGDATA=/<path_to_igblast>/igblast/1.4.0/bin* to set the *$IGDATA* environment variable to point to the location of the igblast executable.

8 . BraCeR requires Kallisto, while TraCeR may be run with either Kallisto or Salmon for quantification of the reconstructed sequences.

9 . In order to visualise clonotype networks, the drawing programs Dot and Neato need to be installed.

10. PHYLIP dnapars, R and the R packages are optional, but required in order to run BraCeR with immunoglobulin lineage reconstruction. Trim Galore! is optional, but recommended to trim adapter sequences and low quality reads before BCR reconstruction.

11. We recommend first installing numpy and biopython through your package manager or conda before setting up Python dependencies with *pip install -r requirements.txt*.

12. The resulting binaries *tracer* or *bracer* may now be run from anywhere. However, the absolute path to where [TB]raCeR was downloaded needs to be specified in the configuration file if the binary is run from outside the main [TB]raCeR directory.

13. It is important to edit the configuration file before running [TB]raCeR. An example configuration file is included in the repository (*[tb]racer.conf*). By default, this is *~/.[tb]racerrc*. If [TB]raCeR fails to find this file, it will use the *[tb]racer.conf* in the repository. [TB]raCeR automatically looks in your system's PATH for the required tools. Alternatively, you can edit the *[tool_locations]* section of the configuration file to specify the path to the executables. Please make sure to also edit the [*base_transcriptomes*] (TraCeR) or [*kallisto_transcriptomes*] (BraCeR) section to include paths to location of the transcriptome FASTA file. This must be a plain-text file, please decompress it if necessary. To activate the short read mode in TraCeR (for reads shorter than 50 bases), uncomment *inchworm_only=True* and uncomment *trinity_kmer_length*. The *trinity_kmer_length* may

be adjusted (17 is the minimum value, and 25 is default in Trinity). Trinity version 2 is required to run the short read mode.

14. You can pass all the usual arguments (except *--small_index*) to the Docker command, but without having to edit the configuration file.
15. [TB]raCeR may run out of memory during the assembly step if the memory limit for Docker is not increased to 6-8 GB. Instructions on how to do this can be found at https://docs.docker.com/docker-for-windows/#advanced for Windows and https://docs.docker.com/docker-for-mac/#advanced for Mac.
16. The TraCeR test data are derived from mouse T cells, while the data used in the BraCeR test are from human B cells. Make sure that the configuration file contains the correct resource files (mouse for TraCeR - human for BraCeR). Run *bracer test* with *--infer_lineage* to test your installation of the additionally required tools for lineage reconstruction. If you are running the test on Docker, you need to clone the GitHub repository, enter its main directory, and call *docker run -it --rm -v $PWD:/scratch -w /scratch teichlab/[tb]racer test -o test_data.*
17. Running *tracer test* should result in three cells, of which Cell 1 and Cell 2 are in a clonotype. Each cell should have a productive TCRα, a non-productive TCRα, a productive TCRβ and a non-productive TCRβ. The output of *bracer test* should also be three cells, with Cell 2 and Cell 3 belonging to a clonotype. Two of the cells should have a productive IgH and a productive Igλ, and one cell should have a productive IgH, productive Igλ and non-productive Igκ.
18. Adapter sequences and low-quality sequences are by default trimmed from the raw reads using Trim Galore! (http://www.bioinformatics.babraham.ac.uk/projects/trim_galore/) and Cutadapt ***(28)***.

19. Bowtie 2 is run with low penalties for insertion of gaps into the read or reference sequence and mapping to ambiguous N nucleotides in the reference sequence.
20. IgH CDR3 regions are variable in length and may be relatively long. BraCeR therefore runs a second round of alignment to extract CDR3-derived reads. In this step, all reads are mapped locally to the IgH reads extracted in the first alignment step in order to identify reads that partially or fully overlap the already extracted reads, thus retrieving all the reads necessary to reconstruct the full recombinant sequence. This alignment is run with high penalties for mismatches and introduction of gaps.
21. Standalone BLAST is used to determine the C-region gene for each BCR sequence.
22. This parameter is used to filter out artefacts. The default *--max_junc_len* is 50 for TraCeR and 100 for BraCeR, but may need to be set higher for γδ T cells as TCRδ chains are known to have highly variable CDR3 lengths ***(30)***.
23. The default mode of TraCeR (*-m imgt*) is to replace all but the junctional region of the reconstructed TCRs with the most likely IMGT reference sequence before assessing whether the recombinant is productively rearranged. This method may not be suitable for humans and other outbred populations with high uncharacterised genetic variability, as it ignores non-junctional sequence changes compared with the IMGT references. In these cases it may be better to run *tracer assemble* with *-m assembly* in order to base all analyses on the reconstructed sequences for humans.
24. To use the *--small-index* option, specify the location of an index built from the corresponding base_transcriptome in the configuration file under [[*salmon*|*kallisto*]*_base_indices*]. Reads will then first be quantified with the *base_index* before a small index is built from the expressed transcripts and reconstructed TCRs, and the reads are quantified with this small index.
25. If FASTQ file(s) are provided, BraCeR also quantifies the BCR sequences.

26. TraCeR always uses the junctional nucleotide sequence reported by IgBLAST, while BraCeR replaces this sequence with the identified CDR3 nucleotide sequence if detected.
27. The *unfiltered_[TB]CR_seqs* subdirectory contains several files. *unfiltered_[TB]CRs.txt* is a text file detailing the reconstructed TCR/BCR recombinants. *<cell_name>_[TB]CRseqs.fa* is a FASTA file listing the reconstructed TCR/BCR sequences. *<cell_name>.pkl* is a Python pickle file which is used in the *summarise* step.
28. If *assemble* is run with *--small_index,* only the quantification output using the small index will be found here.
29. The CDR3 nucleotide distance calculation is based on a human 5-mer targeting model ***(32)***, mouse 5-mer targeting model ***(33)*** or nucleotide Hamming distance for other species. Other distance threshold values can be specified with the *--dist* argument.
30. B cell clonality is based on a shared potentially clonally related IgH and light chain to increase the confidence of clonal assignment because it can be hard to determine whether very similar chains are the result of SHM or rather similar recombination events occurring by chance during B cell development. As shared light chain sequences are more likely to occur by chance as a result of similar recombination events compared to IgH, we do not show sharing of potentially clonally related light chain if the cells do not also share a clonally related IgH. If you wish to base clonality only on IgH, for example if you are studying developing B cells, *bracer summarise* may be run with *--IGH_networks*. Information about reconstructed light chains in the cells will then be included in the networks, but will not affect clonal clustering.
31. IMGT-gapped resources for mouse and human can be found within the resources directory, and may be generated through *Build* for any other species.
32. Run *summarise* with *--IGH_network* to draw lineage trees separately for each locus. This will include cells in IgH lineage trees even if they have completely different light chains.

33. In short, the *lineage.R* script loads a tab-delimited database file in the Change-O data format. Identical sequences within each clone group are then collapsed into one and annotated with the cell names, isotypes and number of cells containing the sequence. Maximum parsimony lineage trees are built through the *dnapars* method of PHYLIP with the Alakazam function *buildPhylipLineage* for each clone group. Lastly, we parse the output and modify the tree topology.
34. To remove certain cells from downstream analyses, you can move the individual result directories for each cell somewhere other than the directory used as input for the *summarise* step.
35. If you have defined new species using *Build*, you should specify the same name here.
36. TraCeR recognises TCRα (A), TCRβ (B), TCRγ (G) and TCRδ (D) and BraCeR accepts IgH (H), Igκ (K) and Igλ (L) for mouse and human. Other locus names can be specified if you have created resources with the *Build* module. By default, TraCeR will attempt to summarise TCRα and TCRβ sequences, while BraCeR will attempt to summarise IgH, Igκ and Igλ sequences. To change this, pass a space-delimited list of locus names. For example, to only look for TCRγ and TCRδ use *--loci G D*.
37. The output format needs to be one of the options detailed at http://www.graphviz.org/doc/info/output.html. In our experience the PDF format does not work when trying to summarise more than a few cells, and we recommend using the *svg* format for high resolution figures.
38. Use this option if you do not have Graphviz installed.
39. TraCeR attempts to identify invariant natural killer T (iNKT) cells ***(36)*** and mucosa-associated invariant T (MAIT) cells ***(37)*** by their characteristic TCRα gene segments. These are removed before creation of networks.

40. This distance value may need to be increased for datasets containing sequences with many SHMs, or to be decreased for datasets consisting mainly of naïve B cells. The optimal distance threshold for a dataset can be determined with the SHazaM package ***(31)***. See the developers' vignette (https://shazam.readthedocs.io/en/version-0.1.9---baseline-fixes/vignettes/DistToNearest-Vignette/) for a step-by-step explanation on how to do this.
41. Several intermediate output files may be generated when running BraCeR summarise. For a list and description of these files please see our documentation pages at https://github.com/Teichlab/bracer.
42. To check if recombinants for a locus are substantially distinct from each other we first inspect the V- and J-gene-usage of each recombinant. If two or more of the recombinants have the same V-gene- and J-gene-usage, they may represent the same rearranged TCR/BCR sequence. The recombinant with the lowest expression value is in such cases likely to be a misassembled sequence. If you wish to base your filtering on more detailed analysis, we recommend aligning the similar reconstructed sequences for the locus using a pairwise alignment tool such as the EMBOSS Matcher Tool ***(39)*** in nucleotide mode to identify the differences between them. If the difference is a single nucleotide or an insertion/deletion, especially in a homopolymer tract, the sequence with the lowest expression value is likely a misassembly due to sequencing errors and/or PCR errors.
43. True biological variability could potentially be the result of secondary rearrangements such as receptor editing of autoreactive B cells ***(41-43)***. For example, we often observe some cells with both a reconstructed productive Igκ and Igλ. Usually one of the chains is highly expressed while the other is very lowly expressed.
44. Maximum parsimony makes several assumptions that may not always hold true for immunoglobulin lineages. For the most accurate lineage tree reconstruction it could

therefore be worth also employing other phylogenetic methods more specifically designed for immunoglobulin lineage reconstruction, such as IgPhyML ***(44)***.

45. BraCeR creates output databases following the Change-O data format (described in http://changeo.readthedocs.io/en/version-0.3.12---makedb-fix/standard.html). BraCeR also adds some additional columns to the database, as described in ***(10)***.
46. Due to immunoglobulin class switching, IgH chains may be expressed in combination with one of multiple constant (C) regions, depending on isotype. To allow for alignment of reads regardless of isotype, we appended one or a few representative sequences for each isotype (IgH) or one representative Igκ or Igλ C-region sequence to the 3' end of each entry in the appropriate synthetic recombinant files for each locus.
47. It is not necessary to parse the sequence annotations of the reference sequences provided to BraCeR, as this will be done automatically.
48. To increase performance, prepare an additional file containing only a few representative C alleles for use with BraCeR, and pass this file as *C_seqs*. Provide the full C reference file with the optional argument --*C_db* for accurate isotype detection.
49. To run the Docker version of *[tb]racer build* you must specify --*resource_dir /scratch* in order to save the created resources. The newly created resource may then be used by running the Docker image from the same directory as used to build the resources, again specifying --*resource_dir /scratch*.
50. The resources distributed with [TB]raCeR were created using an N padding of 7 nucleotides for TCRβ and TCRδ, 8 nucleotides for IgH, and one nucleotide for TCRα, TCRγ, Igκ and Igλ. In our experience, changing the N padding will not have a large effect on reconstruction sensitivity and accuracy.
51. Providing IMGT-gapped V reference sequences is required for lineage reconstruction and creation of IMGT-gapped tab-delimited databases. We highly recommend providing these

sequences even if you are not running lineage reconstruction in order to accurately determine recombinant productivity and CDR3 regions.

5. References

1. Bashford-Rogers RJM, Palser AL, Huntly BJ, Rance R, Vassiliou GS, Follows GA, Kellam P (2013) Network properties derived from deep sequencing of human B-cell receptor repertoires delineate B-cell populations. Genome Research 23 (11):1874-1884. doi:10.1101/gr.154815.113

2. Weinstein JA, Jiang N, White RA, 3rd, Fisher DS, Quake SR (2009) High-throughput sequencing of the zebrafish antibody repertoire. Science 324 (5928):807-810. doi:10.1126/science.1170020

3. Wang C, Sanders CM, Yang Q, Schroeder HW, Jr., Wang E, Babrzadeh F, Gharizadeh B, Myers RM, Hudson JR, Jr., Davis RW, Han J (2010) High throughput sequencing reveals a complex pattern of dynamic interrelationships among human T cell subsets. Proceedings of the National Academy of Sciences of the United States of America 107 (4):1518-1523. doi:10.1073/pnas.0913939107

4. Rosati E, Dowds CM, Liaskou E, Henriksen EKK, Karlsen TH, Franke A (2017) Overview of methodologies for T-cell receptor repertoire analysis. BMC Biotechnology 17:61. doi:10.1186/s12896-017-0379-9

5. Bashford-Rogers RJM, Palser AL, Idris SF, Carter L, Epstein M, Callard RE, Douek DC, Vassiliou GS, Follows GA, Hubank M, Kellam P (2014) Capturing needles in haystacks: a comparison of B-cell receptor sequencing methods. BMC Immunology 15:29. doi:10.1186/s12865-014-0029-0

6. Han A, Glanville J, Hansmann L, Davis MM (2014) Linking T-cell receptor sequence to functional phenotype at the single-cell level. Nature biotechnology 32:684. doi:10.1038/nbt.2938

7. Kolodziejczyk AA, Lönnberg T (2017) Global and targeted approaches to single-cell transcriptome characterization. Briefings in Functional Genomics:elx025-elx025. doi:10.1093/bfgp/elx025

8. Stubbington MJT, Rozenblatt-Rosen O, Regev A, Teichmann SA (2017) Single-cell transcriptomics to explore the immune system in health and disease. Science 358 (6359):58-63. doi:10.1126/science.aan6828

9. Stubbington MJT, Lonnberg T, Proserpio V, Clare S, Speak AO, Dougan G, Teichmann SA (2016) T cell fate and clonality inference from single-cell transcriptomes. Nature methods 13 (4):329-332. doi:10.1038/nmeth.3800

10. Lindeman I, Emerton G, Sollid LM, Teichmann S, Stubbington MJT (2017) BraCeR: Reconstruction of B-cell receptor sequences and clonality inference from single-cell RNA-sequencing. bioRxiv. doi:10.1101/185504

11. Eltahla AA, Rizzetto S, Pirozyan MR, Betz-Stablein BD, Venturi V, Kedzierska K, Lloyd AR, Bull RA, Luciani F (2016) Linking the T cell receptor to the single cell transcriptome in antigen-specific human T cells. Immunology & Cell Biology 94 (6):604-611

12. Rizzetto S, Koppstein DN, Samir J, Singh M, Reed JH, Cai CH, Lloyd AR, Eltahla AA, Goodnow CC, Luciani F (2017) B-cell receptor reconstruction from single-cell RNA-seq with VDJPuzzle. bioRxiv. doi:10.1101/181156

13. Afik S, Yates KB, Bi K, Darko S, Godec J, Gerdemann U, Swadling L, Douek DC, Klenerman P, Barnes EJ, Sharpe AH, Haining WN, Yosef N (2017) Targeted reconstruction of T cell receptor sequence from single cell RNA-seq links CDR3 length to T cell differentiation state. Nucleic acids research 45 (16):e148. doi:10.1093/nar/gkx615

14. Canzar S, Neu KE, Tang Q, Wilson PC, Khan AA (2017) BASIC: BCR assembly from single cells. Bioinformatics 33 (3):425-427. doi:10.1093/bioinformatics/btw631

15. Upadhyay AA, Kauffman RC, Wolabaugh AN, Cho A, Patel NB, Reiss SM, Havenar-Daughton C, Dawoud RA, Tharp GK, Sanz I, Pulendran B, Crotty S, Lee FE-H, Wrammert J, Bosinger SE (2018) BALDR: a computational pipeline for paired heavy and light chain immunoglobulin reconstruction in single-cell RNA-seq data. Genome Medicine 10 (1):20. doi:10.1186/s13073-018-0528-3

16. Lönnberg T, Svensson V, James KR, Fernandez-Ruiz D, Sebina I, Montandon R, Soon MSF, Fogg LG, Nair AS, Liligeto UN, Stubbington MJT, Ly L-H, Bagger FO, Zwiessele M, Lawrence ND, Souza-Fonseca-Guimaraes F, Bunn PT, Engwerda CR, Heath WR, Billker O, Stegle O, Haque A, Teichmann SA (2017) Single-cell RNA-seq and computational analysis using temporal mixture modeling resolves TH1/TFH fate bifurcation in malaria. Science Immunology 2 (9). doi:10.1126/sciimmunol.aal2192

17. Patil VS, Madrigal A, Schmiedel BJ, Clarke J, O'Rourke P, de Silva AD, Harris E, Peters B, Seumois G, Weiskopf D, Sette A, Vijayanand P (2018) Precursors of human CD4(+) cytotoxic T lymphocytes identified by single-cell transcriptome analysis. Sci Immunol 3 (19). doi:10.1126/sciimmunol.aan8664

18. Miragaia RJ, Gomes T, Chomka A, Jardine L, Riedel A, Hegazy AN, Lindeman I, Emerton G, Krausgruber T, Shields J, Haniffa M, Powrie F, Teichmann SA (2017) Single cell transcriptomics of regulatory T cells reveals trajectories of tissue adaptation. bioRxiv. doi:10.1101/217489

19. Picelli S, Faridani OR, Bjorklund AK, Winberg G, Sagasser S, Sandberg R (2014) Full-length RNA-seq from single cells using Smart-seq2. Nature protocols 9 (1):171-181. doi:10.1038/nprot.2014.006

20. Svensson V, Natarajan KN, Ly L-H, Miragaia RJ, Labalette C, Macaulay IC, Cvejic A, Teichmann SA (2017) Power analysis of single-cell RNA-sequencing experiments. Nature methods 14:381. doi:10.1038/nmeth.4220

21. Rizzetto S, Eltahla AA, Lin P, Bull R, Lloyd AR, Ho JWK, Venturi V, Luciani F (2017) Impact of sequencing depth and read length on single cell RNA sequencing data of T cells. Scientific reports 7 (1):12781. doi:10.1038/s41598-017-12989-x

22. Langmead B, Salzberg SL (2012) Fast gapped-read alignment with Bowtie 2. Nature methods 9 (4):357-359. doi:10.1038/nmeth.1923

23. Grabherr MG, Haas BJ, Yassour M, Levin JZ, Thompson DA, Amit I, Adiconis X, Fan L, Raychowdhury R, Zeng Q, Chen Z, Mauceli E, Hacohen N, Gnirke A, Rhind N, di Palma F, Birren BW, Nusbaum C, Lindblad-Toh K, Friedman N, Regev A (2011) Full-length transcriptome assembly from RNA-Seq data without a reference genome. Nature biotechnology 29 (7):644-652. doi:10.1038/nbt.1883

24. Ye J, Ma N, Madden TL, Ostell JM (2013) IgBLAST: an immunoglobulin variable domain sequence analysis tool. Nucleic acids research 41 (Web Server issue):W34-W40. doi:10.1093/nar/gkt382

25. Bray NL, Pimentel H, Melsted P, Pachter L (2016) Near-optimal probabilistic RNA-seq quantification. Nature biotechnology 34 (5):525-527. doi:10.1038/nbt.3519

26. Patro R, Duggal G, Love MI, Irizarry RA, Kingsford C (2017) Salmon provides fast and bias-aware quantification of transcript expression. Nature methods 14:417. doi:10.1038/nmeth.4197

27. Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, Bealer K, Madden TL (2009) BLAST+: architecture and applications. BMC bioinformatics 10:421. doi:10.1186/1471-2105-10-421

28. Martin M (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. EMBnetjournal 17 (1):10-12. doi:10.14806/ej.17.1.200

29. Ilicic T, Kim JK, Kolodziejczyk AA, Bagger FO, McCarthy DJ, Marioni JC, Teichmann SA (2016) Classification of low quality cells from single-cell RNA-seq data. Genome Biology 17 (1):29. doi:10.1186/s13059-016-0888-1

30. Rock EP, Sibbald PR, Davis MM, Chien YH (1994) CDR3 length in antigen-specific immune receptors. The Journal of Experimental Medicine 179 (1):323-328. doi:10.1084/jem.179.1.323

31. Gupta NT, Vander Heiden JA, Uduman M, Gadala-Maria D, Yaari G, Kleinstein SH (2015) Change-O: a toolkit for analyzing large-scale B cell immunoglobulin repertoire sequencing data. Bioinformatics 31 (20):3356-3358. doi:10.1093/bioinformatics/btv359

32. Yaari G, Vander Heiden JA, Uduman M, Gadala-Maria D, Gupta N, Stern JNH, O'Connor KC, Hafler DA, Laserson U, Vigneault F, Kleinstein SH (2013) Models of Somatic Hypermutation Targeting and Substitution Based on Synonymous Mutations from High-Throughput Immunoglobulin Sequencing Data. Frontiers in Immunology 4:358. doi:10.3389/fimmu.2013.00358

33. Cui A, Di Niro R, Vander Heiden JA, Briggs AW, Adams K, Gilbert T, O'Connor KC, Vigneault F, Shlomchik MJ, Kleinstein SH (2016) A Model of Somatic Hypermutation Targeting in Mice Based on High-Throughput Ig Sequencing Data. Journal of immunology (Baltimore, Md : 1950) 197 (9):3566-3574. doi:10.4049/jimmunol.1502263

34. Stern JN, Yaari G, Vander Heiden JA, Church G, Donahue WF, Hintzen RQ, Huttner AJ, Laman JD, Nagra RM, Nylander A, Pitt D, Ramanan S, Siddiqui BA, Vigneault F, Kleinstein SH, Hafler DA, O'Connor KC (2014) B cells populating the multiple sclerosis brain mature in the draining cervical lymph nodes. Science translational medicine 6 (248):248ra107. doi:10.1126/scitranslmed.3008879

35. Felsenstein J (1989) PHYLIP - Phylogeny Inference Package (Version 3.2). Cladistics 5:164-166. doi:citeulike-article-id:2344765

36. Brennan PJ, Brigl M, Brenner MB (2013) Invariant natural killer T cells: an innate activation scheme linked to diverse effector functions. Nature reviews Immunology 13 (2):101-117. doi:10.1038/nri3369

37. Dias J, Leeansyah E, Sandberg JK (2017) Multiple layers of heterogeneity and subset diversity in human MAIT cell responses to distinct microorganisms and to innate cytokines. Proceedings of the National Academy of Sciences 114 (27):E5434-E5443. doi:10.1073/pnas.1705759114

38. Goldstein LD, Chen Y-JJ, Dunne J, Mir A, Hubschle H, Guillory J, Yuan W, Zhang J, Stinson J, Jaiswal B, Pahuja KB, Mann I, Schaal T, Chan L, Anandakrishnan S, Lin C-w, Espinoza P, Husain S, Shapiro H, Swaminathan K, Wei S, Srinivasan M, Seshagiri S, Modrusan Z (2017) Massively parallel nanowell-based single-cell gene expression profiling. BMC Genomics 18 (1):519. doi:10.1186/s12864-017-3893-1

39. Li W, Cowley A, Uludag M, Gur T, McWilliam H, Squizzato S, Park YM, Buso N, Lopez R (2015) The EMBL-EBI bioinformatics web and programmatic tools framework. Nucleic acids research 43 (W1):W580-584. doi:10.1093/nar/gkv279

40. Brady BL, Steinel NC, Bassing CH (2010) Antigen Receptor Allelic Exclusion: An Update and Reappraisal. The Journal of Immunology 185 (7):3801-3808. doi:10.4049/jimmunol.1001158

41. Liu S, Velez M-G, Humann J, Rowland S, Conrad FJ, Halverson R, Torres RM, Pelanda R (2005) Receptor Editing Can Lead to Allelic Inclusion and Development of B Cells That Retain Antibodies Reacting with High Avidity Autoantigens. The Journal of Immunology 175 (8):5067-5076. doi:10.4049/jimmunol.175.8.5067

42. Lang J, Ota T, Kelly M, Strauch P, Freed BM, Torres RM, Nemazee D, Pelanda R (2016) Receptor editing and genetic variability in human autoreactive B cells. The Journal of Experimental Medicine 213 (1):93-108. doi:10.1084/jem.20151039

43. Pelanda R (2014) Dual immunoglobulin light chain B cells: Trojan horses of autoimmunity? Current Opinion in Immunology 27:53-59. doi:10.1016/j.coi.2014.01.012

44. Hoehn KB, Lunter G, Pybus OG (2017) A Phylogenetic Codon Substitution Model for Antibody Lineages. Genetics 206 (1):417-427. doi:10.1534/genetics.116.196303

45. Breden F, Luning Prak ET, Peters B, Rubelt F, Schramm CA, Busse CE, Vander Heiden JA, Christley S, Bukhari SAC, Thorogood A, Matsen Iv FA, Wine Y, Laserson U, Klatzmann D, Douek DC, Lefranc MP, Collins AM, Bubela T, Kleinstein SH, Watson CT, Cowell LG, Scott JK, Kepler TB (2017) Reproducibility and Reuse of Adaptive Immune Receptor Repertoire Data. Front Immunol 8:1418. doi:10.3389/fimmu.2017.01418

46. Yaari G, Kleinstein SH (2015) Practical guidelines for B-cell receptor repertoire sequencing analysis. Genome Medicine 7 (1):121. doi:10.1186/s13073-015-0243-2

Figures

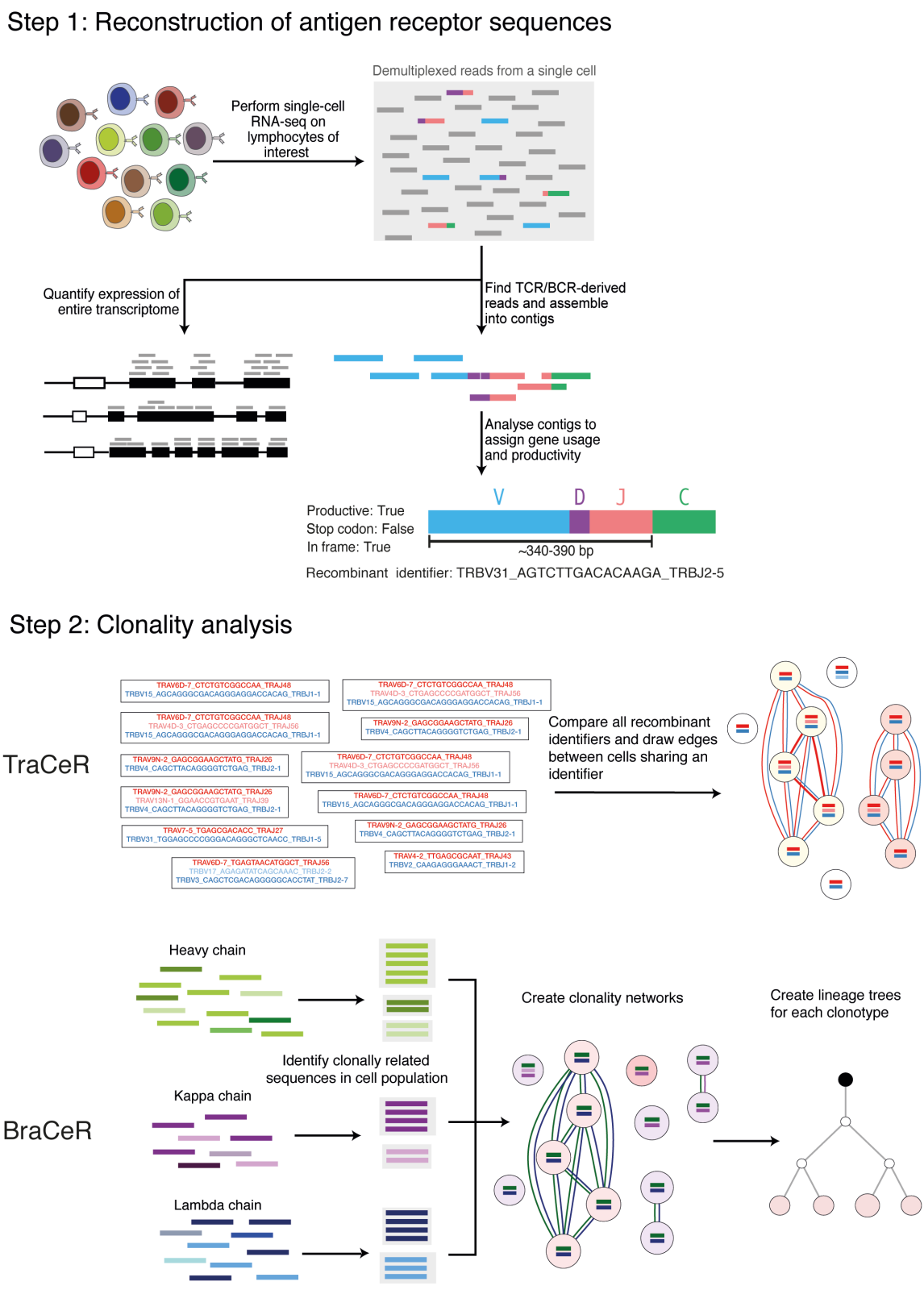

Fig 1 Overview of the [TB]raCeR pipelines.

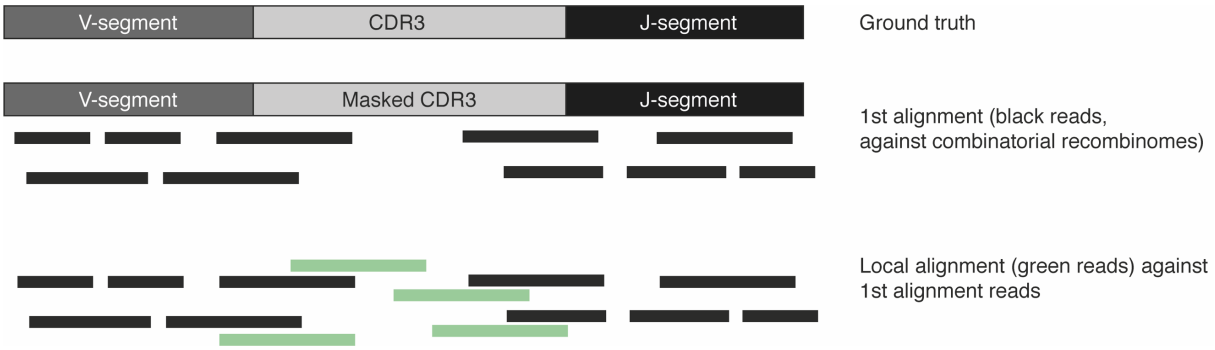

Fig 2 Illustration of the two alignment steps for IgH when reads are 50 bases or shorter.

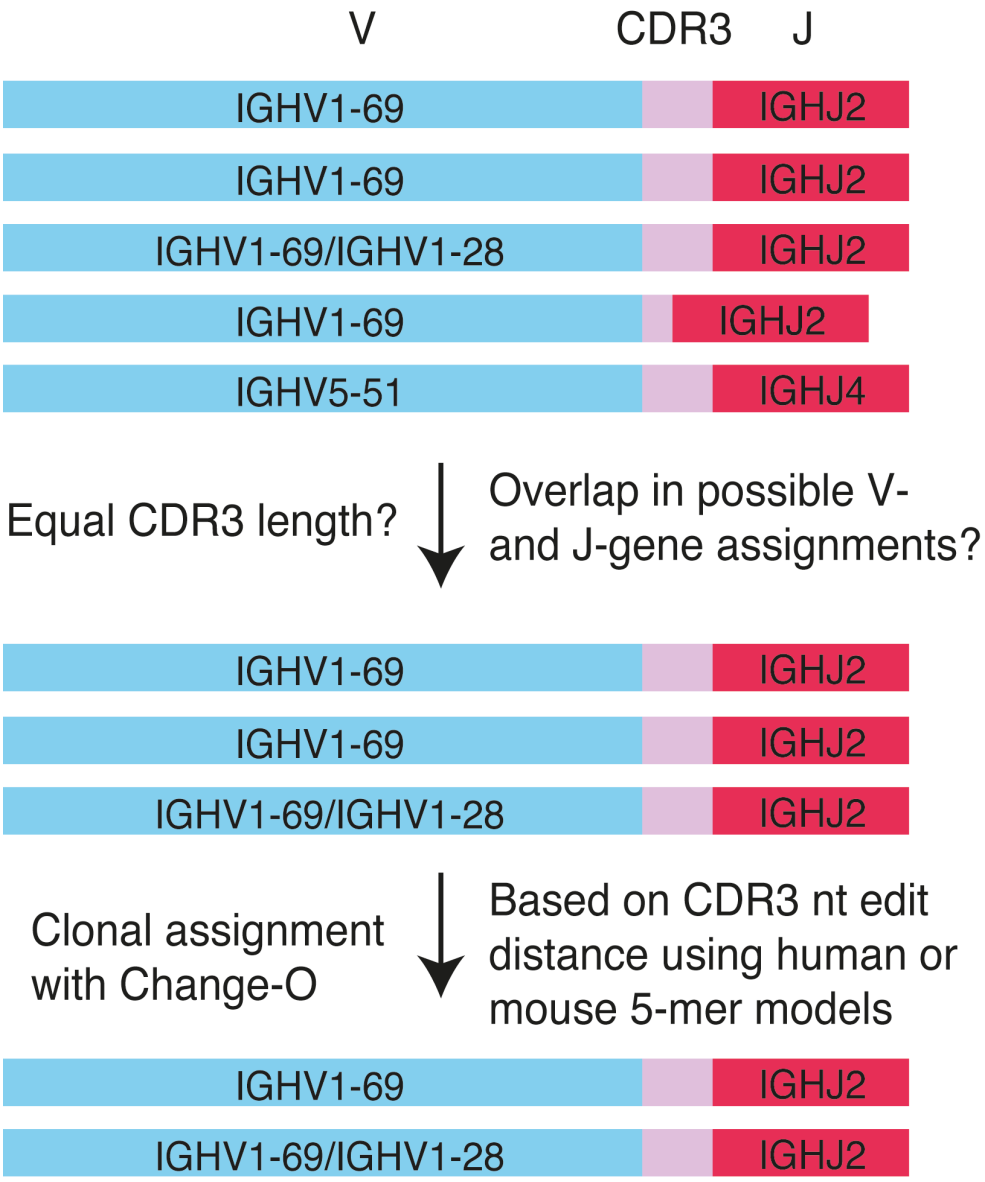

Fig 3 Identification of clonally related productive BCR sequences for each locus.

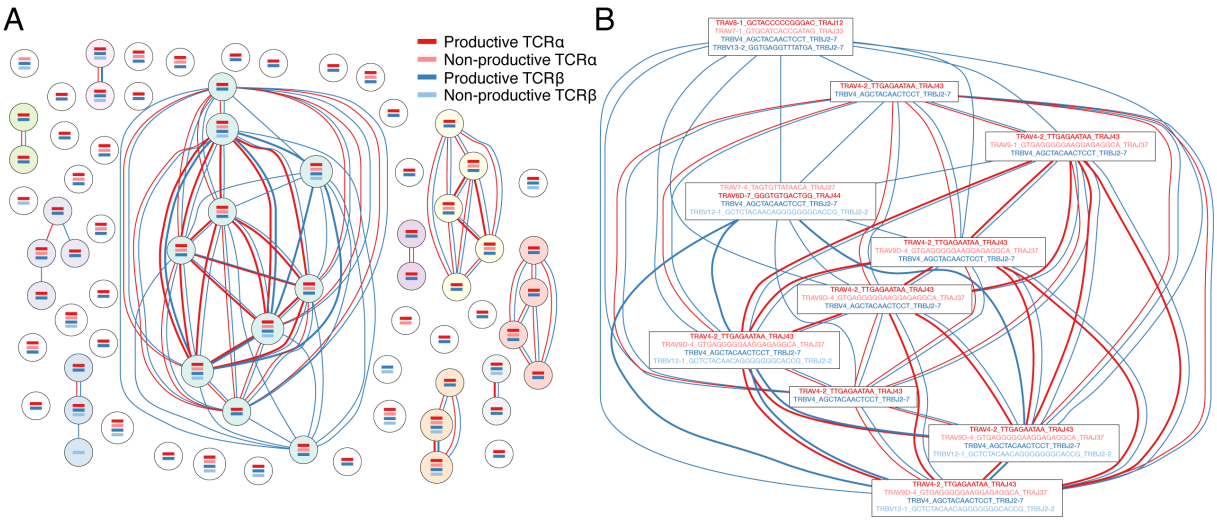

Fig 4 Example of a clonotype network from a mouse 14 days after infection with *Salmonella*. Each node represents a T cell (**A**). Horizontal bars represent reconstructed TCR sequences, with dark colours being productive and light colours non-productive. Edges between nodes indicate sharing of one or more TCRs for each locus with edge thickness being proportional to the number of sequences shared between two nodes. The clone groups have different node background colours for visualisation purposes. An example of a network with identifiers showing only one of the clone groups from the same mouse is shown in (**B**). Figures are adapted from Stubbington 2016 ***(9)***.

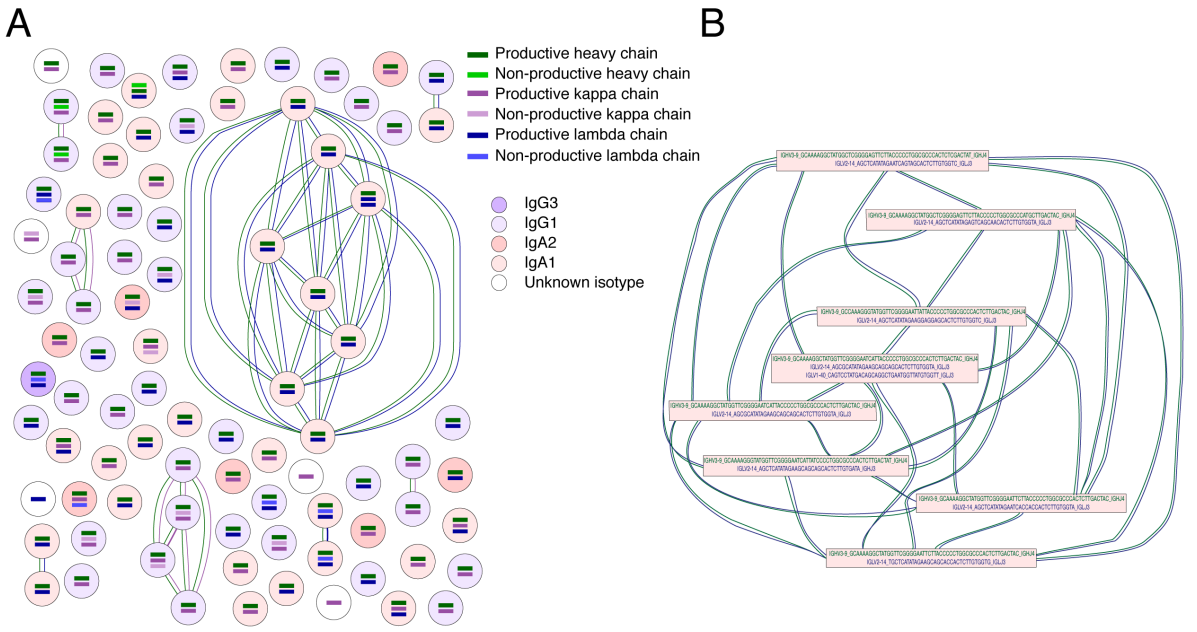

Fig 5 Example of a clonotype network from a human donor (**A**). Each node represents a plasmablast. Horizontal bars indicate reconstructed BCR sequences in each cell, with dark colours denoting productive and light colours non-productive chains. Sharing of one or more clonally related BCRs for each locus is visualised as edges between the nodes, with edge thickness being proportional to the number of shared sequences. The background colour of each node indicates the isotype of the cell. An example of a network with identifiers for one of the clone groups is shown in (**B**). Figure 5A is reproduced from Lindeman 2017 ***(10)***, and the figures are based on raw scRNA-seq data published in ***(14)***.

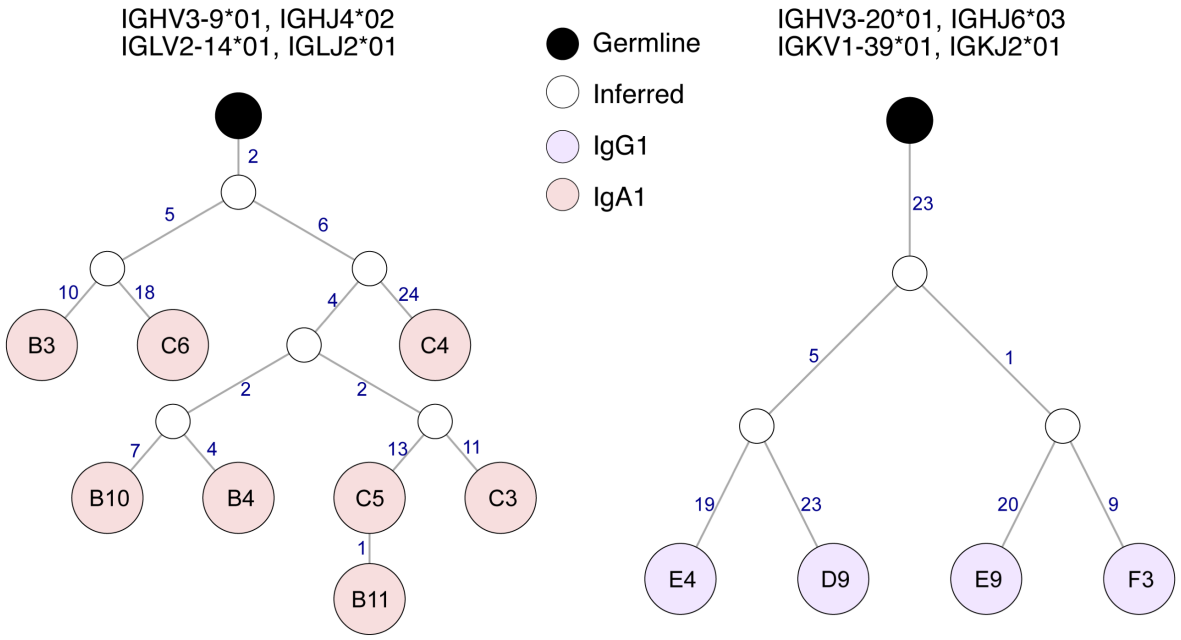

Fig 6 Example of lineage trees constructed for the two largest clone groups in **Fig 5**. Each node represents the combined productive IgH and light chain sequence of a plasmablast. Edges between nodes indicate the edit distance between the sequences. The background colour of each node indicates the isotype of the IgH. The figure is adapted from Lindeman 2017 ***(10)***.