Chapter 16

Single-Cell RNA Sequencing of Human T Cells

Alexandra-Chloé Villani and Karthik Shekhar

Abstract

Understanding how populations of human T cells leverage cellular heterogeneity, plasticity, and diversity to achieve a wide range of functional flexibility, particularly during dynamic processes such as development, differentiation, and antigenic response, is a core challenge that is well suited for single-cell analysis. Hypothesis-free evaluation of cellular states and subpopulations by transcriptional profiling of single T cells can identify relationships that may be obscured by targeted approaches such as FACS sorting on cell-surface antigens, or bulk expression analysis. While this approach is relevant to all cell types, it is of particular interest in the study of T cells for which classical phenotypic criteria are now viewed as insufficient for distinguishing different T cell subtypes and transitional states, and defining the changes associated with dysfunctional T cell states in autoimmunity and tumor-related exhaustion. This unit describes a protocol to generate single-cell transcriptomic libraries of human blood $CD4^+$ and $CD8^+$ T cells, and also introduces the basic bioinformatic steps to process the resulting sequence data for further computational analysis. We show how cellular subpopulations can be identified from transcriptional data, and derive characteristic gene expression signatures that distinguish these states. We believe single-cell RNA-seq is a powerful technique to study the cellular heterogeneity in complex tissues, a paradigm that will be of great value for the immune system.

Key words Single-cell RNA sequencing, T cells, CD4, CD8, Smart-Seq2, Alignment, Clustering, Gene expression, Markers

1 Introduction

T cells initiate and orchestrate adaptive immune responses against pathogenic infections and cancers, and have crucial roles in allergy, autoimmunity, and transplant rejection. Naïve T cells are activated upon recognizing antigenic peptide-MHC molecules on antigen-presenting cells during infection. This recognition is followed by transcriptional, epigenetic, and metabolic changes inside the T cells, making the cells proliferate and release signaling molecules that regulate a new immune response [1–4]. Discoveries of new states of T cell differentiation have also showed that these states are highly plastic, and rather than being "disconnected islands," are more like neighboring territories spread out on a continuous

Enrico Lugli (ed.), *T-Cell Differentiation: Methods and Protocols*, Methods in Molecular Biology, vol. 1514,
DOI 10.1007/978-1-4939-6548-9_16,

landscape whose borders are not clearly defined. A population of T cells associated with a particular differentiated state can acquire different properties and functions classically associated with another state during a secondary immune response, demonstrating their plasticity [5–9]. Appearance of cellular flexibility may arise from truly flexible genetic programs within individual cells or, alternatively, from heterogeneous composition of states within a population. Intrinsic plasticity of genetic programs and heterogeneity in composition can confound each other in data, making it a challenge to identify clinically relevant measurements that accurately reflect the state and capability of the human immune system. Thus, single-cell experimental and analytical approaches are needed to appropriately decipher and dissect the heterogeneity of T cell populations in order to gain further understanding of their capabilities in driving immune responses.

1.1 Single-Cell Profiling Methodologies

While well-established techniques like polychromatic and imaging flow cytometry, mass cytometry, single-cell quantitative PCR (qPCR), RNA FISH (fluorescence in situ hybridization) can provide information at single-cell resolution [10, 11], such approaches can only monitor a small number of preselected candidate features at once, thus restricting the ability to examine genome-wide co-expression patterns and to interrogate cellular heterogeneity from an unbiased point of view. Furthermore, while transcriptional profiling of populations by microarrays and RNA-sequencing have proved valuable in T cell biology, it is also well-known that the average expression level of a population of cells can often be a poor representation of the states of individual cells within the population, a phenomenon known as "Simpson's paradox" [12, 13]. Indeed, measurements using single-molecule RNA FISH indicated that levels of specific transcripts could vary by several folds between presumably equivalent cells, further illustrating the value of profiling whole transcriptomes at the single-cell level [12].

Analyses of transcriptomes through massively parallel sequencing of cDNAs, derived from cellular RNA by reverse transcription (RNA-Seq), generate millions of short fragments that can be sequenced to accurately quantify expression levels, assemble new transcripts and investigate alternative RNA processing [14]. Development of RNA-seq inspired a flurry of experimental methods that consistently lowered the required starting amounts of RNA, ideally down to single-cell quantities. One of the first groups to demonstrate single-cell RNA-sequencing [15, 16] adapted a protocol initially developed for single-cell microarray studies [17], which enabled preferential amplification of 3′ ends of mRNAs, and detection of thousands of genes expressed in single mouse oocytes and early embryonic cells [15, 16]. Since this first study, several single-cell RNA-sequencing (scRNA-seq) protocols have been reported (such as Smart-Seq, Smart-Seq2, CEL-Seq, STRT, MARS-Seq,

Drop-Seq, inDrop), enabling unbiased profiling of cellular mRNA expression [18–27] and increasing information content recovered per cell. Notably, except for the Smart-Seq and Smart-Seq2 protocols, all existing scRNA-seq methods preferentially capture reads originating from either the 3′ or 5′ end of transcripts, thus limiting sequencing coverage only to the ends of the molecule. One advantage of methods that capture the 3′ or 5′ end of the transcripts is the ability to tag each molecule during reverse transcription with a DNA barcode that serves as a unique molecular identifier (UMI) that can be used to digitally count transcript numbers without PCR amplification artifacts [23]. Table 1 highlights key features of different scRNA-seq methods that have been widely used. Recent advances based on droplet microfluidics have substantially increased the cellular throughput of scRNA-seq, making it possible to profile tens of thousands of cells in a single reaction [20, 21].

Table 1
Overview of characteristics of different single cell RNA-sequencing protocols

	Poly (A) tailing	Template switching	In vitro transcription	Rolling circle amplification	5′ Selection	3′ Selection
Associated acronyms	N/A	Smart-seq, Smart-seq2	N/A	N/A	STRT	CEL-seq, MARS-seq, drop-seq, in-drop, SCRB-Seq
Full-length transcripts	Yes	Yes	Yes	Yes	No	No
Strand-specificity	No	No	No	No	Yes	Yes
Early pooling[a]	No	No	No	No	Possible	Possible
Positional bias	Weakly 3′	Weakly 3′	Weakly 3′	NO	5′ Only	3′ Only
Unique molecular identifiers (UMIs)[b]	No	No	No	No	Yes	Yes
Available commercial kits	No	Yes[c]	No	No	No	No
Key references	[15, 16]	[24–26]	[61, 62]	[63–64]	[23, 65]	[18–22]

[a]Refers to the possibility to introduce a cellular barcode identifier during first-strand synthesis
[b]UMIs are random sequences of bases used to tag each RNA molecules prior to PCR amplification [23], thereby aiding in the identification of PCR duplicates. While losses in cDNA synthesis and bias in cDNA amplification can result into severe quantitative errors when performing scRNAseq, UMIs can help in eliminating such amplification noise by enabling counting of individual molecules
[c]Clontech offers multiple generations of single-cell transcriptome SMARTer analysis kits allowing performing Smart-Seq and Smart-Seq2 protocols. Furthermore, some of these SMARTer kits are compatible with the with C1 Auto Prep integrated fluidic circuits (IFCs) from Fluidigm, which enables in an integrated workflow the capture of single cells follow by RT and whole transcriptome amplification

While still an order of magnitude lower in cellular throughput compared to single-cell protein measurement techniques like flow and mass cytometry, scRNA-seq's ability to profile transcriptome-wide information in a completely unbiased manner has already enabled many novel biological discoveries. In particular, scRNA-seq approaches have revealed previously uncharacterized subsets of cells together with endogenous marker genes specific to these subsets, and shown that cell-specific splicing and allele expression patterns can differ significantly from their apparent population averages [11, 12]. Although the functional consequences of these phenomena remain to be elucidated, it must be noted that they could not have been detected in an unbiased fashion with previous single-cell methodologies like flow cytometry and qPCR.

1.2 Smart-Seq2 Methodology

Here we describe introductory scRNA-sequencing and data analysis for single human T cells, whose libraries were prepared using the Smart-Seq2 (SS2) protocol ([25–27]; Fig. 1). SS2 is an improvement over the original Smart-Seq single-cell method [24], which was shown to generate quantitative and reproducible data from both single cells and small amounts of total RNA of 10 pg or more. With SS2, Picelli et al. further optimized the reverse transcription (RT), template switching, and pre-amplification steps of Smart-Seq to obtain an increased cDNA yield from single cells, as well as higher sensitivity, fewer technical biases and less variability [25, 26]. Importantly, their published protocol relies entirely on off-the-shelf reagents, making it more cost-effective than commercially available alternative kits.

Briefly, SS2 begins with reverse transcription of polyadenylated transcripts using an oligo-dT primer and a reverse transcriptase derived from the Moloney murine leukemia virus (MMLVRT) (Fig. 1). The reverse transcription is followed by a template switching reaction that relies on the terminal-transferase activity of the MMLVRT, wherein 2–5 untemplated nucleotides are added to the 3′ end of the nascent cDNA by the MMLVRT upon reaching the 5′ end of the mRNA [25, 26]. By the introduction of a template switch oligonucleotide primer (TSO), the MMLVRT is made to switch its template, and synthesize a complementary sequence to the TSO. As a result, every cDNA molecule derived from a full-length mRNA carries additional artificial sequences at the 5′ and the 3′ ends, which are identical to each other. This trick makes it possible to carry out PCR amplification of the cDNA using a single primer. The resulting cDNA is amplified to get enough material for subsequent experimental steps. This amplification is followed by an incubation step with Tn5 transposase to fragment the full-length double stranded cDNA and append adapters on each molecule, using the dual-index strategy developed by Illumina, Inc. Each single-cell library is then individually barcoded by PCR with index

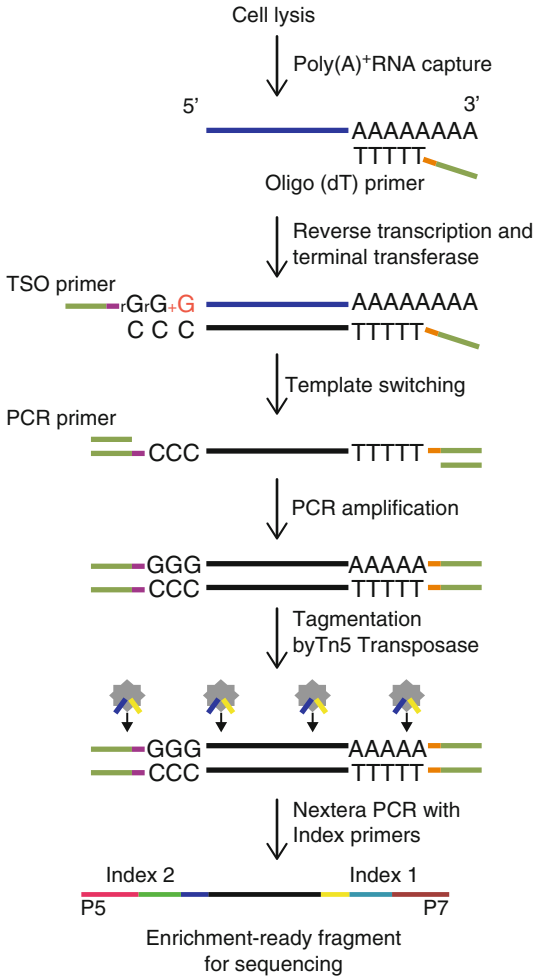


Fig. 1 Flowchart for Smart-Seq2 single cell library preparation. The diagram, modified from Picelli et al. [26], illustrates the key experimental steps in single-cell RNA-sequencing. The library preparation is performed using the Nextera XT DNA library preparation and index kits from Illumina. Figure 1 is adapted by permission from Macmillan Publishers Ltd: [NATURE PROTOCOLS] (Picelli et al. Nat Protoc 9(1): 171–181), copyright (2014)

primers. The barcoded single cells are then pooled and sequenced on an Illumina sequencer. An important difference with other methods is that every single cell is treated independently until the very end of the protocol (including library generation). While resulting in a more expensive protocol as compared to alternatives that multiplex before cDNA amplification (Table 1), SS2 has the advantage in that if interesting patterns are observed in specific single cells upon sequencing analysis, it remains possible to go back to the cDNA of those cells and generate more in-depth sequencing

data or performing further qPCR analysis selectively. Notably, Smart-Seq and Smart-Seq2 are the only protocols that can generate full-length transcript data, which provide a number of advantages compared to 5′ or 3′ tagged data (Table 1). Specifically in the context of lymphocytes, full-length information can enable the simultaneous inference of TCR/BCR clonality and gene expression state [28]. Nevertheless, current limitations of Smart-Seq/Smart-Seq2 include the lack of strand specificity, UMI (unique molecular identifiers) to quantify transcript numbers, and the inability to detect non-polyadenylated RNA (Table 1).

Computational analysis of scRNA-seq data can identify groups of cells with similar expression, and characterize the key expression signatures that show variation in the data. Such information, when combined with other kinds of data (e.g., chromatin state, protein expression) can be used to derive insights into the regulatory rules that are responsible for the maintenance and plasticity of cell states. Specific goals of scRNA-seq analysis may include: (1) identifying discrete subpopulations of cells, and gene signatures that are unique to each subpopulation; (2) Identifying co-expressed gene modules and regulatory programs, and their expression levels in different subpopulations; (3) characterizing biological heterogeneity within specific discrete subpopulations, and identifying the gene modules whose variation underlie these continuous states. Notably, while there are several scRNA-sequencing protocols available [18–27], the computational workflow described below was tailored to analyze data generated from the Smart-Seq2 method.

The rest of this unit is organized as follows. We begin with a brief methodological overview of how single RNA-sequencing libraries can be prepared and sequenced using a slightly modified version of the previously published Smart-Seq2 protocol [25–27]. To illustrate the methods described in this unit, we generated single-cell libraries from 384T cells (192 $CD4^+$ and 192 $CD8^+$ T cells) FACS sorted from healthy human peripheral blood. Next, we outline the steps involved in preliminary bioinformatic analysis of single T cell RNA-sequencing data generated using the described protocol, focusing on alignment, quantification, quality control (QC), principal components analysis of expression data to determine subpopulations, differential expression analysis to determine signatures, and visualization of results. By listing these steps, which are implemented either in the Unix command line environment (i.e., alignment and expression quantification), or the R programming language (i.e., analysis of expression), we hope to introduce the reader to typical computational approaches that are widely used, and that can be applied to other datasets. We also introduce the reader to some of the commonly used software packages that are used for RNA-seq analysis, and also refer to excellent reviews that describe computational steps in more detail.

2 Materials

2.1 Generation of scRNA-Seq Libraries with Smart-Seq2

The material and equipment listed below are for generating single-cell lysate and performing the reverse transcription steps, as these are the steps that were slightly modified from the originally published method and protocols [25–27] to generate the T cell data using SS2. For the remaining reagents needed to run SS2 (including PCR amplification, quality control assessments, sequencing library generation), as well as detailed description of the SS2 protocol, please refer to Picelli et al. [26], or to Trombetta et al. [27] who describe a slightly modified version of the original protocol.

2.1.1 Lysis Buffer and Single-Cell Sorting

1. Fluorescence activated cell sorting (FACS) machine.
2. Plate centrifuge.
3. Vortex.
4. Lysis buffer: 1 % (vol/vol) 2-Mercaptoethanol in TCL buffer (Qiagen).
5. Cell sorting setup beads for green-yellow lasers (ThermoFisher Scientific).
6. 96-well PCR plates, skirted (Eppendorf).
7. MicroAmp clear adhesive film (ThermoFisher Scientific).
8. Peripheral blood mononuclear cells or T cell suspension.
9. 1× Phosphate Buffered Saline (PBS).
10. Human AB Serum (Corning).
11. Fluorescently conjugated antibodies to human antigens, such as TCRαβ, CD3 CD4, CD8, CD62L, CD45RA, and CD45RO.
12. Dry ice.

2.1.2 RNA Lysate Cleanup and First Step of Reverse Transcription Reaction

1. DynaMag-96 side-skirted magnet (ThermoFisher Scientific).
2. 8-channel and 12-channel pipettes (P20 and P200).
3. Thermal cycler.
4. RNAse decontamination solution (RNaseZap, ThermoFisher Scientific).
5. Ethanol.
6. RNase-free water.
7. RNA-SPRI beads (Agencourt RNAClean XP RNA-SPRI beads, Beckman Coulter).
8. Recombinant Ribonuclease Inhibitor (Clontech).
9. 10 μM reverse transcription DNA oligonucleotide primer (custom synthesized by Integrated DNA Technologies): AAGCAGTGGTATCAACGCAGAGTACT(30)VN.

10. dNTP mix (dATP, dCTP, dGTP, and dTTP, each at 10 mM) (ThermoFisher Scientific).
11. SuperScript II Reverse Transcriptase Kit (ThermoFisher Scientific): 5× first-strand buffer, 100 mM DTT, SuperScript II reverse transcriptase.
12. PCR 12-well tube strips and caps.
13. 50 ml conical bottom tube.
14. 1.5 ml RNAse- and DNAse-free microcentrifuge tubes.
15. Low retention tips (Rainin).

3 Experimental Procedure

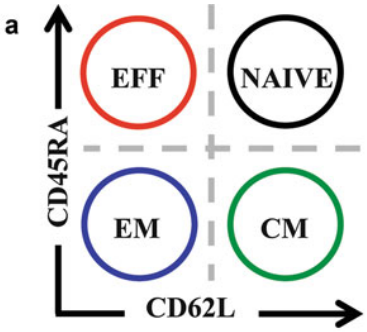
3.1 Gating Strategy and Staining of T Cell Populations

Peripheral blood mononuclear cells (PBMCs) were isolated from the blood of healthy controls individual by ficoll gradient centrifugation, followed by antibody staining, as described in Chap. 4 of this volume ("FACS analysis of memory T lymphocytes", by Enrico Lugli and colleagues). Briefly, we used antibodies against well-established surface antigens to enrich for four phenotypic subsets in each of $CD4^+$ and $CD8^+$ T cell populations—naïve cells ($CD62L^+$ $CD45RA^+$), central memory ($CD62L^+$ $CD45RA^-$), effector memory ($CD62L^-CD45RA^-$), and short-lived effector cells ($CD62L^-$ $CD45RA^+$), each at roughly 25% proportion. While these subsets are present in different abundances in circulation, our goal was to have an even representation of the four main subsets per 96-well plate. We thus opted for creating a quadrant gate of CD62L versus CD45RA (Fig. 2a) and sorted 23 cells per gated quadrant per 96-well plate (1 cell per well); one well was left empty per subset as technical control (Fig. 2b). We used this gating strategy for both CD4 (Lin^- $TCR\alpha\beta^+$ $CD3^+$ $CD4^+$ $CD8^-$) and CD8 (Lin^- $TCR\alpha\beta^+$ $CD3^+$ $CD4^-$ $CD8^+$) T cells, enabling over-sampling of rare and down-sampling of abundant cell subtypes. Figure 2b illustrates an example of plate layout. Stained cells were resuspended in 1× PBS with 2% AB human serum and kept on ice until sorted.

3.2 Preparation of Single-Cell Lysate and Cell Sorting

1. Clean the working space and pipettes with 70% EtOH and RNaseZap solutions before setting up the working plates.
2. Distribute 10 µl of lysis buffer into each well of a full-skirted-side 96-well PCR plate. Cover the plate with MicroAmp clear adhesive film and keep at room temperature until ready for single-cell isolation. Centrifuge 1 min at 300 × *g* at room temperature just before sorting to ensure the lysis buffer is at the bottom of each well of the 96-well plate.
3. In preparation to performing single-cell sorting, align sorter stream by sorting cell sorting set-up beads for green-yellow lasers. Using one of the sealed 96-well plates to be sorted in, sort 50 beads on the seal for each well of row A and H, as well

a



b

	1	2	3	4	5	6	7	8	9	10	11	12
A	$CD3^+$ $CD4^+$ $CD8^-$ **$CD62L^+$ $CD45RA^+$** (Naïve T cells): 1	1	1	1	1	1	1	1	1	1	1	1
B	Empty well	1	1	1	1	1	1	1	1	1	1	1
C	$CD3^+$ $CD4^+$ $CD8^-$ **$CD62L^+$ $CD45RA^-$** (Central memory): 1	1	1	1	1	1	1	1	1	1	1	1
D	Empty well	1	1	1	1	1	1	1	1	1	1	1
E	$CD3^+$ $CD4^+$ $CD8^-$ **$CD62L^-$ $CD45RA^+$** (Effector): 1	1	1	1	1	1	1	1	1	1	1	1
F	Empty well	1	1	1	1	1	1	1	1	1	1	1
G	$CD3^+$ $CD4^+$ $CD8^-$ **$CD62L^-$ $CD45RA^-$** (Effector Memory): 1	1	1	1	1	1	1	1	1	1	1	1
H	Empty well	1	1	1	1	1	1	1	1	1	1	1

Fig. 2 T cell sorting strategy for single-cell RNA-sequencing analysis. (**a**) Quadrant gating strategy of CD62L versus CD45RA, which together define classically established phenotypic subsets of T cells. We used this gating strategy to sort 23 cells per gate per 96-well plate for both CD4 (Lin^- $TCR\alpha\beta^+$ $CD3^+$ $CD4^+$ $CD8^-$) and CD8 (Lin^- $TCR\alpha\beta^+$ $CD3^+$ $CD4^-$ $CD8^+$) T cells. This enabled us to oversample rare subsets and down-sample abundant cell subsets. Sorted subsets included naïve cells ($CD62L^+$ $CD45RA^+$), central memory (CM: $CD62L^+$ $CD45RA^-$), effector memory (EM: $CD62L^-CD45RA^-$) and short-lived effector cells (EFF: $CD62L^-$ $CD45RA^+$). Two replicate plates were sorted for each of $CD4^+$ and $CD8^+$ cells. Our dataset altogether included 396 cells, all isolated from the same healthy donor. (**b**) Layout of a 96-well plate of $CD4^+$ T cells (one plate out of two), showing two rows per subset defined in panel **a**; 1 cell was sorted into every well containing 10 μl of lysis buffer. Wells B1, D1, F1, and H1 were left emptied as technical control

as column 1 and 2, and central wells D5, D6, D7, D8, E5, E6, E7, E8. Make sure all sorted 50 beads are well centered for every well, and adjust Z-position as needed.

4. Run at least 10,000 events to setup the gating strategy. To improve yield, plan to sort single cells on both the presence of a positive viability indicator (e.g., Calcein AM, Life Technologies) and the absence of a cell death marker. To limit the number of doublet cells being sorted into a single well, perform doublet exclusion by plotting of forward and sideward scatters areas, heights, and widths (FSC- and SSC-A/-H/-W). There are two common options used to gate for singlets using forward and sideward scatters. The first is to plot FSC-A vs. FSC-H. Events deviating from the diagonal are doublets. The second option is to perform a sequential gating. First FSC-H is plotted vs. FSC-W and then SSC-H vs. SSC-W. In both dotplots, events with a low signal width are to be gated in order to obtain singlets.

5. Perform single-cell sort of T cells, sorting a single cell into each well of the abovementioned 96-well plate containing lysis buffer, but leave 1 well empty as a technical control for every population to be studied. It is also possible to sort 1000–5000 cells into 1 well as a small population control. Adjust flow rate to a minimum, controlling events rate to be lower than 5000 events/s. Dilute the sample as needed to stay within these parameters.
6. Once sorting is complete, seal plate with MicroAmp clear adhesive film and centrifuge 1 min at 300 × *g* 4 °C. Immediately upon completing centrifugation step, flash-freeze on dry ice and transfer to −80 °C freezer until ready for lysate cleanup. This is the first stopping point of the protocol.

3.3 Perform Lysate Cleanup and Reverse Transcription of mRNA Species

We generated our single-cell data using a slightly modified version of the Smart-Seq2 protocol [25–27]. The initial steps of the protocol including lysis of single cells, lysate cleanup, and reverse transcription of mRNA species have been slightly modified and are described below. All remaining steps of the protocol can be followed using the protocol reported by Picelli et al. [26], or to Trombetta et al. [27]. The following steps can be performed inside a biosafety cabinet or a RNA workstation (if available); otherwise, they should be carefully performed on a standard benchtop that has been thoroughly cleaned.

1. Thoroughly vortex RNA-SPRI beads to ensure a uniform suspension and aliquot volume to be used in PCR strip tubes. While bringing RNA-SPRI beads to room temperature (allow 30 min), use 70% EtOH and then RNAseZap to clean the workbench and all equipment used to process RNA.
2. Once beads are warmed up to room temperature, thaw lysate plate on ice for 1 min, and then centrifuge for 1 min at 800 × *g* at room temperature.

   All following steps are done at room temperature unless mentioned otherwise.
3. Add 2.2 volume RNA-SPRI beads to each well of lysate (2.2 × ~10 µl cell lysate = 22 µl of beads to be added per well) and mix well by pipetting up and down ten times with P200 multichannel. Note that the PEG solution of the SPRI beads is viscous; it is recommended to use low-retention tips to limit the generation of bubbles and loss of material.
4. Incubate lysate and bead suspension for 10 min on bench (*see* **Note 1**).
5. Move the plate on a 96-well plate magnet (e.g., DynaMag-96 side-skirted magnet) and incubate for 5 min, while still covering the plate with a lid. After completing the incubation step, remove supernatant from each column of the plate by pipetting

up in an angle opposite to where the beads have clustered against the magnet while being careful to not aspirate the beads collected on the side of each well. If beads are aspirated, put back the supernatant in the well and wait for 2 min to enable the beads to cluster again against the magnet (*see* **Note 2**).

6. Wash beads by adding 100 μl of 80% ethanol (prepared same day with nuclease-free water) to each well. It is important that all beads be submerged by the ethanol solution. Move the plate sideways back-and-forth 3 times on the magnet, allowing the beads to move across the wells, to enable further washing of the beads. Make sure to move the plate on both directions on the magnet to ensure that both beads in columns 1 and 12 are washed. Wait for 30 s to allow the beads to cluster on the magnet before aspirating ethanol using a P200 multichannel pipette. Repeat wash two additional times.
7. Aspirate final ethanol wash with a P200 multichannel pipette and then a P20 to ensure that all residual ethanol is removed. Leave the plate on the magnet and allow beads to dry for 10 min at room temperature. Keep plate loosely covered with the lid from a fresh box of pipet tips, placed slightly ajar. Beads should look cracked within 10 min if all residual ethanol was removed properly (*see* **Note 3**).
8. Once bead pellet has dried, elute RNA from beads by resuspending dried beads in 4 μl of the following mix for every well (*see* **Note 4**):
   1 μl RNase-free H_2O.
   1 μl of 10 μM of oligo-dT RT primer.
   1 μl of dNTP mix (10 mM each).
   1 μl RNase inhibitor dilution buffer (10% RNase-Inhib, final of 4 U/μl).
9. Quickly centrifuge the plate at room temperature, letting the speed reach 200 × *g* and then immediately stop the centrifugation step. The goal is to collect all drops on the side of the wells without getting the beads to settle too much at the bottom of the well.
10. Incubate for 3 min at 72 °C to anneal oligo-dT RT primer
11. Place plate on ice immediately following incubation. Do not let the temperature of the thermal cycler go down to 4 °C with the plate still inside.
12. Remove seal, add the RT master mix and follow the steps of the original protocol [26], starting at **step 9**.

For the remaining steps of the SS2 protocol, involving reverse transcription step (*see* **Note 5**), PCR amplification (*note that for the PCR amplification, 22 cycles were performed to generate the T cell data*), quality control assessments of single-cell cDNA, generation

of sequencing library, please refer to Picelli et al. [26] for the detailed description of reagents and following experimental steps, or to Trombetta et al. [27] who also describes the protocol in great detail with slight modifications. Please note, the next safe stopping point in the protocol is after having completed the PCR amplification; after that step, the cDNA can be safely stored at −20 °C.

3.4 Single-Cell RNA-Seq Data Analysis

A major challenge in analyzing scRNA-seq data involves teasing out reliable biological signals from technical noise. While transcriptome-wide measurements of gene expression at single-cell resolution enables heterogeneity in cell-populations to be studied in a truly unbiased manner, one must also note that scRNA-seq ultimately relies on amplifying a very small amount of starting material. In addition to "amplification noise," different library preparation methods involve different sources of technical biases [29, 30]. Furthermore, as in any other quantitative assay, scRNA-seq libraries can exhibit strong variation across technical replicates prepared across different days, or by different technicians ("batch effects"). Such nonbiological variation can often confound true biological signals of interest [31].

Many of the analysis steps (e.g., alignment of reads to a reference genome/transcriptome) are identical to conventional RNA-seq analysis, but some steps (e.g., determining differentially expressed genes) are uniquely designed to account for the greater technical and biological variability captured across different samples (individual cells in this regard; see [32]. It is important to carefully design experiments whenever possible so that biological heterogeneity is not confounded by technical variation [31–34]. We believe it is critical that biological and technical replicates be included in the experiments and that all the libraries be prepared using the same protocol. When this is not possible, the computational analyst must take caution to include technical sources of variation as covariates in the analysis to correct for batch effects, and interpret biological results conservatively [30–35].

In this section, we describe the canonical steps involved in the processing of scRNA-seq data and various kinds of analyses that are routinely performed to extract biological information. We note that a variety of bioinformatic tools and software packages are available for scRNA-seq analysis (Table 2), and wherever possible, we also indicate these alternatives, and refer to some excellent papers that review existing state-of-the-art methodologies. Importantly, since this is an extremely nascent field, the steps below must be regarded as guidelines intended to be as simple as possible for a starting user. Advanced users are encouraged to explore and compare alternative strategies.

3.4.1 Alignment and Quantification of Sequencing Reads

Sequencing data generated by an Illumina sequencer are conventionally stored in the FASTQ format, in the form of files carrying the .fastq or .fastq.gz extension. Successful sample demultiplexing

Table 2
Software packages for scRNA-seq analysis

Package name (language of implementation)	Functions	References
ZIFA (Python)	Dimensionality reduction algorithm that accounts for transcript dropouts in single-cell data	[66]
Monocle (R)	Mapping transcripts on differentiation cascades, and arranging single cells along a differentiation tree (pseudotime estimation)	[67]
scLVM (Python)	Dissecting cofounding sources of variation (e.g., differentiation vs. cell-cycle) in scRNA-seq data	[68]
SCDE (R)	Testing for differential expression in scRNA-seq data where technical artifacts abound (transcript dropouts, library quality variation)	[32]
BASiCS (R)	A Bayesian framework to normalize and assess biological/technical variation in scRNa-seq data	[69]
Seurat (R)	Spatial mapping of single-cell transcriptomes based on a preexisting spatial pattern of landmark genes. R package also includes wrapper functions for analysis and visualization	[75]
Pagoda (R)	Pathway and geneset overdispersion analysis	[70]
Sincell (R)	Assessment of cell-state hierarchies	[71]
RaceID (R)	Rare cell type identification in complex populations of single cells	[72]
Scuba (Matlab)	Extracting lineage relationships from single-cell data	[73]
Scater (R)	R-based package for quality control, visualization, and preprocessing	[74]

results in a single FASTQ file (for single end sequencing), or a pair of FASTQ files (for paired end sequencing) for each sample. The two mate-pairs in paired-end reads are typically referred to as the "left read" and "right read" respectively. The first step in data analysis is to align these sequencing reads to a reference transcriptome, and obtain quantitative expression levels across various genes and transcripts for each single-cell sample (*see* **Note 6**).

The following workflow for alignment and quantification of single-cell RNA-seq reads uses publicly available software packages that can be run as commands on a standard Unix terminal, on a standard desktop/laptop or server. For researchers with minimal computational background, we note that these tools are also accessible on a drag-and-drop GUI interface on GenomeSpace [36], a Web-based portal that supports a wide-range of bioinformatics tools for integrative genomics and transcriptomics analysis, and data management (http://www.genomespace.org). Upon creating an account on the website, users can view a number of "recipes" for common bioinformatics tasks, each utilizing multiple tools. Recipes include detailed instructions and videos.

GenomeSpace is free to use, and each account is provided some cloud storage. For analysis of large datasets, the best practice use of GenomeSpace is to connect an account directly to cloud-based storage, such as Dropbox, Google Drive, or Amazon S3 buckets. We recommend the Unix-based workflow for users that have access to sufficient computing resources in their home institution.

Install Necessary Software Packages

1. The following programs/software should be installed locally or on a server:
   Programs for Read Alignment and Quantification
   (a) TopHat: https://ccb.jhu.edu/software/tophat/index.shtml.
   (b) Bowtie2: http://bowtie-bio.sourceforge.net/bowtie2/index.shtml.
   (c) Kallisto: http://pachterlab.github.io/kallisto/.

   Programs for Visualizing Alignment and Generating Quality Metrics
   (d) Samtools: http://samtools.sourceforge.net/.
   (e) Picard tools: http://broadinstitute.github.io/picard/.
   (f) Integrative Genomics Viewer (IGV): https://www.broadinstitute.org/igv/.

   Programming Environment for Performing Statistical Analysis of Gene Expression and Clustering
   (g) R: https://www.r-project.org/.
   (h) RStudio: https://www.rstudio.com/products/RStudio/.

   Aligning Reads to the Genome Using TopHat
2. TopHat is a fast splice junction mapper that aligns paired-end reads to a desired reference genome [37]. Commonly used spliced aligners are reviewed by Engström et al. [38].
   (a) Download pre-built genome indexes for a number of model organisms from the following link, https://ccb.jhu.edu/software/tophat/igenomes.shtml.

   These can also be prepared by the user by following instructions on the TopHat website.
   (b) FASTQ files for the left and right read files are required for each sample in case of paired-end sequencing data. Run TopHat as follows for each sample (*see* **Note 7**):

```
% tophat [options] <genome_index_base> [sample
ID_leftreads.fastq] [sampleID_rightreads.fastq]
```

   TopHat also allows for single-end reads as input. See https://ccb.jhu.edu/ software/tophat/manual.html for details regarding [options].

(c) Most alignment software output their results as SAM/BAM files, which are standard file formats within the genomics community (*see* **Note 8**). The output of TopHat consists of several files including accepted_hits.bam, which summarizes the alignment information in the BAM file format, that can be viewed in a human readable format using samtools:

```
% samtools view accepted_hits.bam
```

3. Most downstream programs (e.g., IGV) prefer a BAM file wherein the aligned reads are sorted according to their locations in the reference genome, and subsequently indexed for fast access. Sort BAM file using samtools:

```
%samtools sort accepted_hits.bam accepted_
hits.sort
```

The above command will produce accepted_hits.sort.bam. Next, index the file for easy retrieval by downstream programs:

```
% samtools index accepted_hits.sort.bam
```

Performing QC and Visualizing read Alignments

4. The Picard-Tools suite provides a set of Java-based command line tools that can efficiently manipulate SAM/BAM files and also provide important QC metrics regarding the quality of the alignment (e.g., mapping rate, ribosomal RNA content, and number of genes detected). For example, the following command collects statistics on the number of reads that correctly align to the reference genome:

```
% java –jar <picard-location>/CollectAlignmentSummary
Metrics.jar [options] I=accepted_hits.sort.bam
O=sampleID_align_metrics.txt
```

See http://broadinstitute.github.io/picard/command-line-overview.html for details on all the available command line tools, and their use-cases. Additional software packages for extracting QC metrics from BAM files are available (*see* **Note 9**).

5. The sequence alignment summarized in the BAM file can be graphically visualized using the Broad Institute's Integrative Genomics Viewer (IGV) program (https://www.broadinstitute.org/igv/). Once opened, IGV allows the user to select an appropriate reference genome and transcriptome annotation. The accepted_hits.sort.bam can then be loaded here and visualized as a graph of alignment counts at each location in the genome. Multiple BAM files corresponding to different samples can be loaded simultaneously and visualized as individual tracks (*see* **Note 10**).

Quantifying Transcript/Gene Abundances

6. Analysis of gene expression heterogeneity in single-cell RNA-seq data requires the quantification of gene or transcript abundances from the alignment of reads. A number of software packages are available for this purpose (*see* **Note 11**), and many review articles and protocols summarizing alternative workflows have been published [37, 38]. Here we have used Kallisto, an extremely fast and memory efficient program to quantify abundances of transcripts from RNA-seq data through "pseudoalignment" ([40]; *see* **Note 12**). Kallisto requires the user to build an index of the transcriptome from a FASTA file (*see* **Note 13**), containing the nucleotide sequences of the transcripts/gene targets that the user wishes to quantify:

```
% kallisto index -i [<index-location>/transcripts.idx]
[transcripts.fasta.gz]
```

Some transcriptome FASTA files can be downloaded from http://bio.math.berkeley.edu/kallisto/transcriptomes/ or they can be downloaded from the UCSC table browser (https://genome.ucsc.edu/cgi-bin/hgTables). Once the index is built (*see* **Note 14**), run the following command to quantify the gene/transcript abundances for each sample:

```
% kallisto quant -i [[<index-location>/transcripts.idx] -o [sampleID]
[sampleID_leftreads.fastq] [sampleID _right reads.fastq]
```

Additional options for single-end reads, multithreading and uncertainty estimates using bootstrapping are detailed in the Kallisto website: https://pachterlab.github.io/kallisto/manual.html.

7. kallisto quant produces a tab-separated text file sampleID/abundance.tsv,which summarizes the estimated numbers of RNA-Seq fragments derived from the corresponding transcripts and normalized transcript-per-million (TPM) values within the sample for each transcript/gene target listed in the transcriptome index [transcripts.idx] as separate columns (*see* **Note 15**). These count or TPM columns can be read into a programming language environment like R or python, and merged into a single tab-delimited expression matrix file prior to further exploration of the data (*see* **Note 16**).

In the following section, we explore cellular heterogeneity by analyzing the expression matrix, identify subpopulations of cells in the dataset and the gene signatures that define them. While we use the T-cell dataset as our working examples, these can be applied to other datasets. The expression data used here is available upon request to the authors by e-mail.

3.4.2 Exploring Single-Cell Heterogeneity and Structure Using the Expression Matrix

Single cells resident in blood or tissues are a mixture of multiple cell types (e.g., T cells vs. B cells) and cell-states (naïve vs. activated CD8+ T cells). Bulk expression measurements provide an average readout over multiple diverse states, and thus conceal the underlying heterogeneity. In a number of situations, none of the underlying individual cell states might be represented in the average measurement, a phenomenon known as Simpson's paradox [12, 41]. Single-cell RNA-seq measurements can thus provide a more accurate picture of transcriptional heterogeneity across multiple cells, and computational techniques rooted in machine learning may be fruitfully applied to explore key genes driving heterogeneity in the data (e.g., via principal components analysis), infer cell-types and cell-states in a completely unbiased manner (e.g., clustering), and find biological processes that are reflected in genes differentially expressed in one condition vs. another (e.g., pathway analysis). Here we describe some of these techniques in the context of T cells isolated from blood. These steps can serve as a template for the user to analyze data from an alternative expression matrix, although some steps might require slight modifications (e.g., based on alternative sample naming conventions).

The following steps are implemented in R (https://www.r--project.org), a freely available, extremely versatile and popular programming language for bioinformatics analysis. Upon installing R, we recommend that users install RStudio, a free and open source integrated development environment (IDE) for R, which provides an easy to use desktop application for writing and running R scripts, and allows users to organize their code.

It is conventional for bioinformatics research groups to create a "software package" in R to accompany the publication of a method. A number of useful packages are compiled and curated as part of the "Bioconductor" project [42], and these packages can be easily installed and loaded into RStudio. The following workflow uses a number of available packages (*see* Table 2).

Set Up the Environment, and Install Necessary Packages

1. Open RStudio, and create a working directory. Copy the kallisto inferred-read counts (rows = genes, columns = cells), and separately transcripts per million (TPM) values of gene expression as tab-delimited text files Counts.txt and TPM.txt into this directory (*see* **Notes 15** and **16**). The packages ggplot2, plyr, NMF, and tsne can be installed using the install.packages() function.

```
> install.packages("ggplot2")
> install.packages("plyr")
> install.packages("NMF");
> install.packages("tsne");
```

Install the differential expression analysis package SCDE by following instructions on the website (http://hms-dbmi.github.io/scde/). Once installed in the user's repository, these packages can be loaded on to the working session as:

```
library(ggplot2)
library(plyr)
library(NMF)
library(tsne)
library(scde);
```

Explore the Data

2. Read the counts and TPM matrix into R,

```
>Counts_mat=read.table("Counts.txt", header=TRUE, sep="\t",
row.names=1, quote="");
>TPM_mat=read.table("TPM.txt", header=TRUE, sep="\t",
row.names=1, quote="");
```

Check the first five rows and three columns of the TPM matrix to make sure the data loaded correctly,

```
> TPM_mat[1:5,1:3]
               CD4.EM_S33_S308 CD8.CM_S44_S188 CD4.CM_S38_S152
C9orf152             0.00           0.00           0.00
RPS11              533.49        1954.74          21.17
ELMO2                0.00           0.00           0.00
CREB3L1              0.00              0.00        0.00
PNMA1                0.00              0.00        0.00
```

Here the rows and columns represent genes and cells. "CD4.XX" and "CD8.XX" denote $CD4^+$ and $CD8^+$ T cells, while "CD8.EM," "CD8.CM," "CD8.EF," and "CD8.NA" denote effector memory, central memory, effector, and naïve $CD8^+$ T cells (and similarly for $CD4^+$ T cells). These cells were labeled according to the quadrant gating they were sorted from (see *Gating strategy and staining of T cell populations* section). For every column, name letters preceding the first underscore symbol ("_") represents the cell type based on sorting, while the letters following it represents a unique identifier for that cell. For example Column 1 is an effector memory $CD4^+$ T cell with a label "S33_S308."

The total number of genes and cells in the dataset can be queried as the number of rows and columns in the expression matrix,

```
> dim(TPM_mat)
[1] 23686 368
```

implying 23,686 genes (rows) and 368 cells (columns). It is common practice to exclude cells that do not meet user-specified quality criteria (e.g., low mapping rates, and high 5′ or 3′ bias) that can be obtained from Picard Tools or other

programs. Here we only include libraries that have more than 3000 detected genes but no more than 10,000 detected genes.

```
genes.detected=apply(TPM_mat, 2, function(x) sum(x>0))
cells.use=(genes.detected>= 3e3) & (genes.detected<1e4)
TPM_mat1=TPM_mat[,cells.use]; Counts_mat1=Counts_mat[,cells.use]
```

Note that these cutoffs are data-dependent, both on the cell type/state and library quality (i.e., sequencing depth, alignment rate). Here we chose the minimum cutoff of number of genes detected per cell one standard deviation less than the mean value (~4500) of this quantity in our data. We also excluded a few outlier libraries that had more than 10,000 genes detected, as these were likely to be multiple cells sorted into a single well. The total number of cells in each of the eight samples ("CD8.EM," "CD4.NA," etc.) can be queried as,

```
> sample.ids=sapply(colnames(TPM_mat1), function(x) strsplit(x,"_")
[[1]][1])
> table(sample.ids)
CD4.CM CD4.EF CD4.EM CD4.NA CD8.CM CD8.EF CD8.EM CD8.NA
    45     45     46     44     43     46     45     43
```

The first command extracts the cell-type from the list of column names, which as mentioned above, have the format "CellType_SampleID." Thus individual samples in our dataset have 43–46 cells, totaling 357 cells.

3. It is good practice to explore the data, and make sanity checks before applying fancy computational techniques. A common approach is, for example, to make plots describing summary statistics of library quality (Fig. 3a–c) and expression patterns of key marker genes. A code to make simple barplots with error bars is provided in **Note 17**. The following command summarizes the average number of transcriptome mapped reads per cell, and the average number of genes detected per cell in each of the 8 samples (Fig. 3a, b),

```
> barplot_tcell(x=colSums(Counts_mat[,cells.use]), id=sample.ids,
name="Num. Mapped Reads per cell")
> barplot_tcell(x=genes.detected[cells.use], id=sample.ids,
name="Num. Genes per cell")
```

Figure 3c, which summarizes the mapping rate of reads to the genome, transcriptome and ribosomal RNA was created using the output of Picard-Tools CollectAlignmentSummary Metrics.jar introduced previously (could not be reproduced because of space constraints).

4. Another commonly asked question is: have libraries been sequenced to sufficient read-depth? A computational answer can be derived by randomly down-sampling the read counts within the counts matrix to varying degrees, and asking how

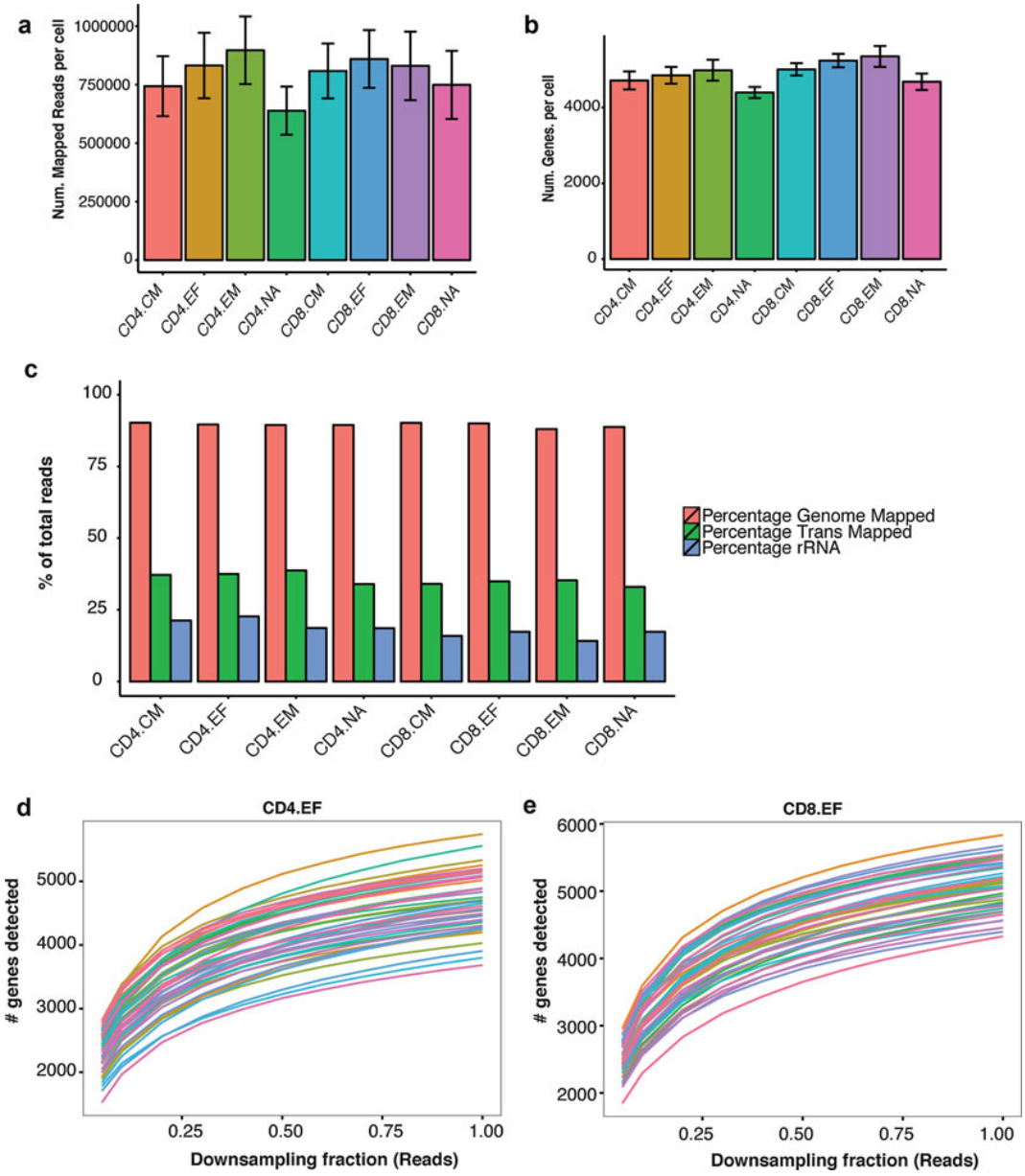


Fig. 3 Visualizing quality controls in single cell RNA-sequencing data **a** and **b**. Barplots summarizing average number of reads mapped to the transcriptome per cell (**a**), and the average number of genes detected per cell (**b**) in each of the 8 subsets. (**c**) Barplots reporting summary statistics of the percentage of the genome-mapped (*red*), percentage of transcriptome-mapped (*green*) and percentage of ribosomal-RNA (*blue*) reads for each subset profiled, as a percentage of total sequenced reads. (**d, e**) Saturation curves for $CD4^+$ effector cells ($CD4^+CD62L^-$ $CD45RA^+$) and $CD8^+$ effector cells ($CD8^+CD62L^-$ $CD45RA^+$) reveal that deeper sequencing of these libraries can detect more genes. Briefly, every single-cell library is randomly "downsampled" to a certain proportion of the initial number of mapped reads 100 times, and the average number of detected genes is recorded. Performing this exercise at 20 values of downsampling proportions (5–98 %) yields a curve for every cell. The curves show that around 80 % of genes detected at full depth are detectable at 75 % of the sequencing depth, which is around two million reads per cell. The upward slope of the curves at fraction = 98 % suggests that deeper sequencing can be beneficial

sharply the number of detected genes falls as reads are removed. Figure 3d, e show the resulting "saturation curves" for CD4 and CD8 EF cells (computational procedure for downsampling is described in the figure legend. The code not be reproduced in this note because of space constraints).

5. One can also plot the average expression of known marker genes within every group to verify whether their expression conforms to biological expectations. For example, when profiling T cells, one can look at: CD4 co-receptor (*CD4*, expressed only on $CD4^+$ T cells; Fig. 4a), CD8 co-receptor alpha chain (*CD8A*, expressed only in $CD8^+$ T cells, Fig. 4b), L-selectin (*SELL*/CD62L, a lymph node homing receptor predominantly expressed in naïve and central memory cells; Fig. 4c), and CD45 antigen/*PTPRC* (expressed in all T cells; Fig. 4d).

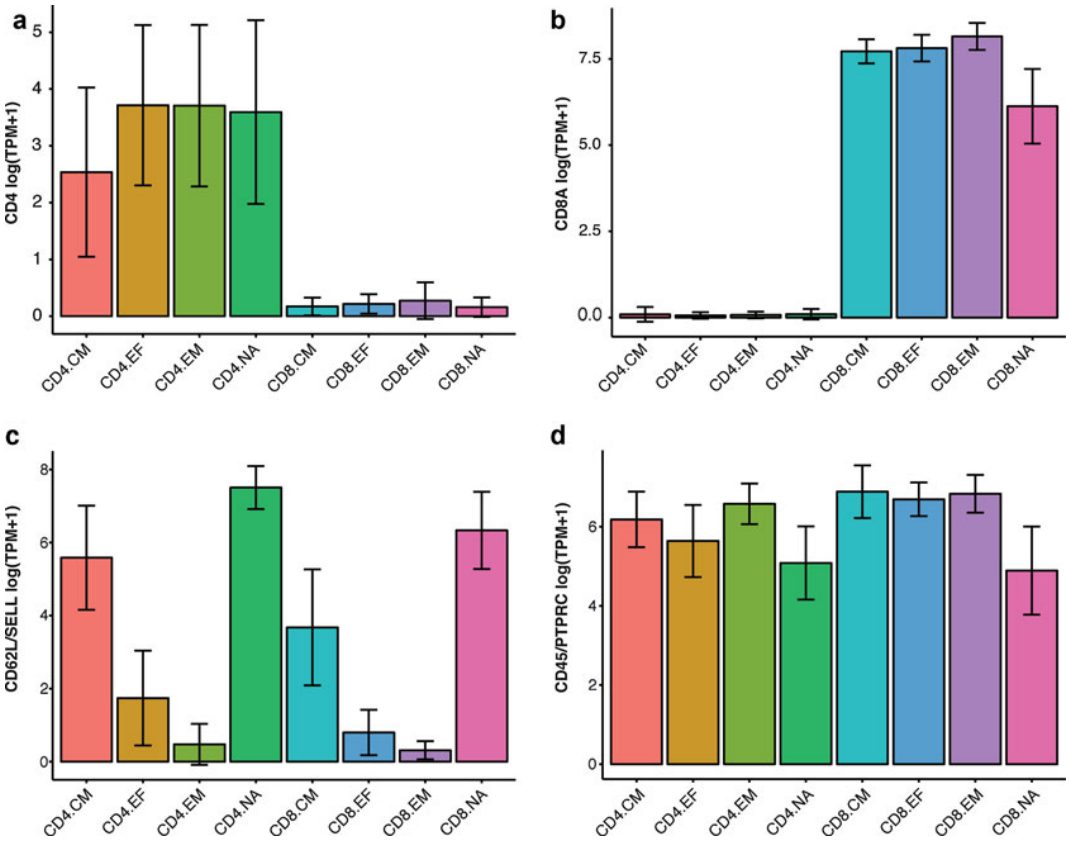


Fig. 4 Expression of known T cell markers (**a**–**d**). Illustrative barplots showing the expression (average ± SD log(TPM + 1)) of key marker genes across the 8 subsets. (**a**) *CD4*, gene encoding for CD4 co-receptor. (**b**) *CD8A*, gene encoding the alpha-subunit of the CD8 co-receptor. (**c**) *SELL*, gene encoding the lymph node homing receptor CD62L, expressed predominantly on circulating T cells. (**d**) *PTPRC*, gene encoding the CD45 antigen, expressed on all T cells

```
> logTPM_mat1=log(TPM_mat1+1)
> barplot_tcell(x=logTPM_mat1["CD4",], id=sample.ids, name="CD4
log(TPM+1)")
> barplot_tcell(x=logTPM_mat1["CD8A",], id=sample.ids, name="CD8A
log(TPM+1)")
>barplot_tcell(x=logTPM_mat1["SELL",], id=sample.ids,
name="CD62L/SELL log(TPM+1)")
> barplot_tcell(x=logTPM_mat1["PTPRC",], id=sample.ids,
name="CD45/PTPRC log(TPM+1)")
```

See Fig. 4a–d for these plots. It is common practice log transform the TPM + 1 values instead of using the raw TPMs. This causes the data points to more uniformly spread across their dynamic range, and also makes it easy to interpret differences between the transformed values as "fold changes." The addition of 1 ensures that zero TPM values remain zero in the transformed units.

Perform Principal Components Analysis (PCA)

6. PCA is a conceptually straightforward, yet an efficient and robust, approach [43] to perform unsupervised exploratory analysis on the data, and to identify structure within the single-cell data driven by the correlated expression of gene modules. It has been used successfully in a number of recent single-cell RNA-seq papers [44–47]. Here we apply PCA on the 357 cell-data while being completely blind to the cell-type labels of cells from the sorting procedure. We will then ask whether such an unbiased analysis is able to separate the different sorted populations. PCA takes as input an expression matrix, with rows as genes and columns as cells.
   (a) *Selection of variable genes:* A typical practice is to perform PCA using only those genes that show appreciable variation within the data, and avoid genes that are poorly expressed. The following R code selects genes that are expressed by at least 40% of cells in any of the 8 samples at log(TPM + 1) > 4.5, and that have a coefficient of variation across all the cells higher than 0.5 (*see* **Note 18**):

```
> pca.genes=c()
> groups=unique(sample.ids)
> for (i in groups){
cells=grep(i, colnames(TPM_mat1), value=TRUE)
include.genes=apply(logTPM_mat1, 1, function(x) ((sum(x[cells]>4.5) /
length(cells)>0.4) & sd(x)/mean(x)>0.5))
pca.genes=union(pca.genes, rownames(TPM_mat1)[include.genes])
}
pca.genes comprises 678 genes, as can be checked by the length() func-
tion:
> length(pca.genes)
[1] 678
```

(b) *PCA:* Use the log-transformed TPM matrix as input to PCA, but rescale its rows to have zero-mean and unit standard deviation. This normalization procedure, termed "z-scoring" or "standardizing," ensures that relative expression variation in all genes is treated equally irrespective of their absolute expression levels. Standardizing can be turned on in the R function prcomp, which implements PCA.

```
> pca_result=prcomp(t(logTPM_mat1[pca.genes,]), center=TRUE,
scale=TRUE)
```

Examine PCA Scores for the Presence of Cell-Subpopulations

7. Conceptually, PCA finds successive mutually independent linear combinations of genes that maximally capture correlated variation in the data. Such correlated variation might arise due to the presence of "gene modules"—groups of genes whose expression levels are highly correlated because of a common biological connection (e.g., response to T cell stimulation). Each linear combination of genes is called a "principal direction" and is ordered by the amount of variance in the data it captures (high to low). For every principal direction, the coefficient corresponding to every gene in the linear combination determines the extent to which that gene "drives" the variation captured by the principled direction, and is referred to as the gene's "loading." Genes whose loadings along a principal direction are high in magnitude and share the same sign are typically positively correlated within the data.

   Each individual cell, which corresponds to a column in the expression matrix, can then be projected on to each of these principal directions yielding a vector of "scores" for *that* cell, and so on for all the cells. The vector of scores for all the cells along the first principal direction is referred to as the first "principal component" of the data (PC1), and so on. The matrices corresponding to the loadings (genes × PCs) and scores (cells × PCs) as column vectors may be directly extracted from the output of princomp as follows:

```
pca.loadings=data.frame(pca_result$rotation)
pca.scores=data.frame(pca_result$x)
```

   With a PCA transformation in hand, a straightforward way to visualize subpopulation structures in the data is to make a 2D-scatter plot of cells based on the values of their PCs (*see* **Note 19** for scatter plot code).

```
> scatter_plot(X=pca.scores[,1:2], id=sample.ids, axis.labels=c("PC1","PC2"))
> scatter_plot(X=pca.scores[,3:4], id=sample.ids, axis.labels=c("PC3","PC4"))
```

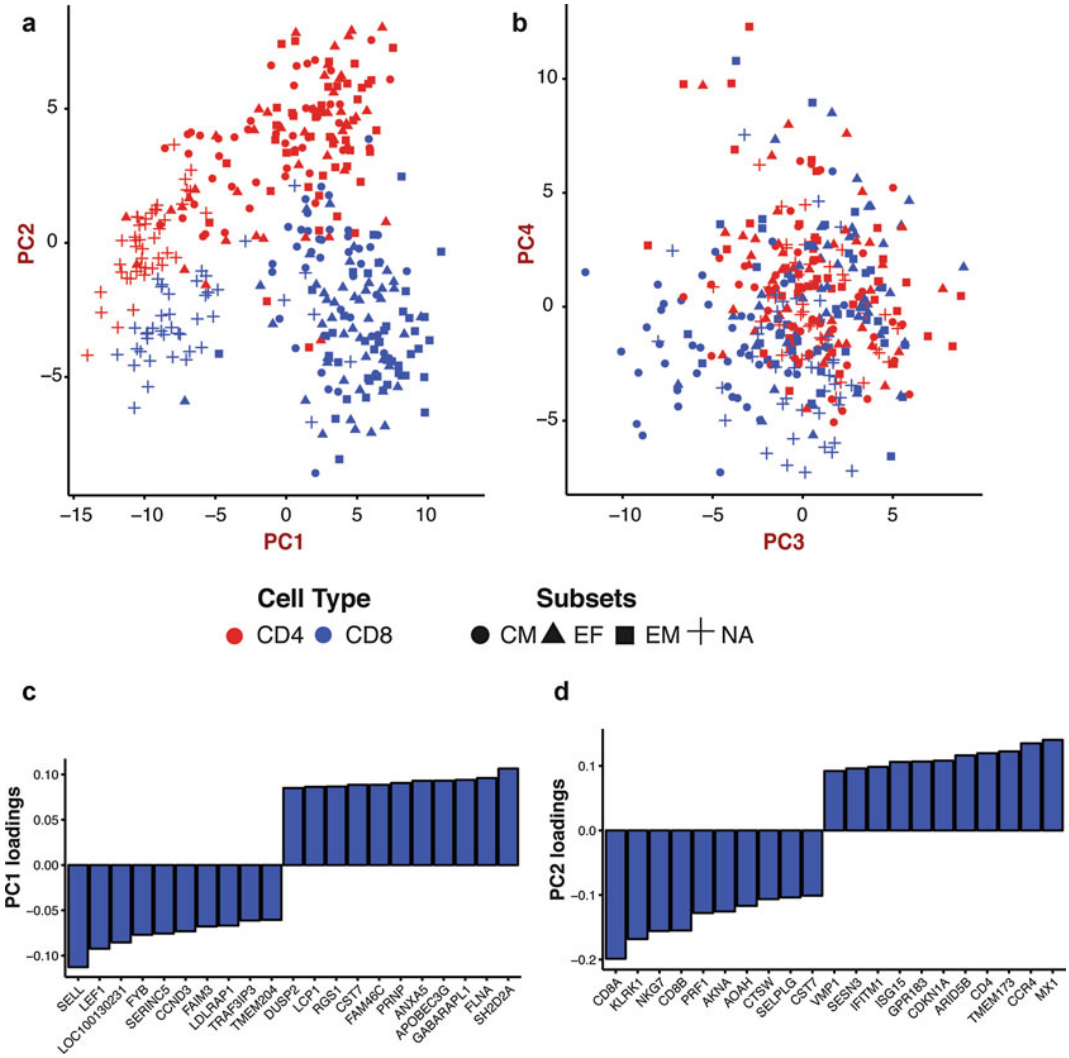


Fig. 5 Unsupervised analysis using PCA separates cell-subsets **a** and **b**. Scatter plots of PCA-scores of single-cells, along PC1-PC2 (Panel **a**) and PC3-PC4 (Panel **b**). Each point is a cell with its location plotted as value of the "score" assigned by PCA along the corresponding PC. In Panel a, PC1 (*x*-axis) separates naïve cells (+ symbol) from the remaining subsets. PC2 (*y*-axis) separates CD4$^+$ T cells (*red*) from CD8$^+$ T cells (*blue*). (**c**, **d**) Top genes (*x*-axis) driving the variation captured by PC1 and PC2, and their relative weights or loadings (*y*-axis). Genes with high positive or negative loadings are the primary contributors. The gene with the most negative loading in PC1 (**c**) is *SELL* (also called L-selectin or CD62L), the lymph node homing receptor expressed in naïve and central memory cells but not effector and effector memory cells, while genes driving PC2 (d) include known CD8$^+$ T cell related genes such as *CD8A*, *CD8B*, and *PRF1*. This mirrors the observation in Panel **a** with PC1 (*x*-axis) separating naïve T cells from the other subsets, and PC2 (*y*-axis) separating CD8$^+$ T cells from CD4$^+$ T cells

The output of these commands, Fig. 5a, b, reveals that PCA separates the different subpopulations (Note that the PCA procedure was completely blind to the subpopulation identities of the T cells). PC1 in particular appears to separate

naïve from memory and effector cells, while PC2 separates $CD4^+$ from $CD8^+$ T cells. PC3, on the other hand separates central memory $CD8^+$ T cells from the rest. These figures also suggest that the naïve cells are transcriptionally more distinct than the effector and memory cells (CM and EM).

Examine PCA Loadings to Identify Key Genes That Drive Variation

8. Different principal directions (columns of the pca.loadings) are orthogonal vectors in multidimensional gene-expression space that describe prominent patterns of correlated variation. Genes with large positive values in a given loadings vector are positively correlated with each other along the direction of the vector, as are the genes with large negative values; however, these two groups of genes are anti-correlated with each other along that direction. It is usually very instructive to look at some of the top genes (positive and negative) driving individual principal directions using bar plots (*see* **Note 20** for code):

```
barplot_loadings(X=pca.loadings,pc.use=1,num.genes=10)
barplot_loadings(X=pca.loadings,pc.use=2,num.genes=10)
```

Figure 5c, d shows that a number of known T cell related genes feature among the top genes driving PC1 and PC2. The gene with the most negative loading in PC1 is *SELL* (also called L-selectin or CD62L), the lymph node homing receptor expressed in naïve and central memory cells but not effector and effector memory cells (Fig. 5c). Also featured is *LEF1*, a transcription factor that is essential for early T cell development [48]. This reflects the separation of naïve cells from the other cells along the PC1 axis in Fig. 5a. Genes driving PC2 (Fig. 5d) include known $CD8^+$ T cell related genes *CD8A*, *CD8B*, *PRF1* (perforin), *NKG7* (negative loadings), and $CD4^+$ T cell enriched genes *CD4*, *CCR4*, and *MX1* (positive loadings), which mirrors the separation of these cells along the PC2 axis in Fig. 5a [49].

We emphasize that these patterns were captured in a completely unsupervised manner by PCA from the list of highly variable genes, and were in no way "selected" based on their known biological roles.

Visualize PCA Results on a 2D Map Using t-Distributed Stochastic Neighbor Embedding (t-SNE)

9. Important modes of biological variation might manifest across multiple PCs, and thus incorporating multiple PCs into the analysis might be important to separate phenotypically/functionally distinct subpopulations of cells. A powerful way to visualize information contained in multiple PCs is to use the t-SNE algorithm [50], which performs nonlinear dimensionality reduction and generates a 2D scatter plot of cells, captur-

ing the essence of the variation contained across multiple PCs. The algorithm, implemented in the R-package tsne, needs as input a user-defined number of principal component values for all the cells.

There is no general answer to how many principal components one must consider, and describing tests for statistical significance of PCs is outside the scope of this protocol (see [51] for a review of this topic). Here we choose to keep the top 10 principal components in the data:

```
set.seed(10)
tsne.proj=data.frame(tsne(pca.scores[,1:10]),whiten=FALSE)
colnames(tsne.proj)=paste0("tsne_",c(1:2))
scatter_plot(X=tsne.proj, id=sample.ids, axis.la-
bels=c("tSNE_1","tSNE_2"))
```

Figure 6a shows the t-SNE plot. Compared to the PCA scatter, the distinction between multiple subsets now becomes much sharper now that we are incorporating more information (a total of 10 principal components in the visualization). Also note that while the $CD4^+$ and $CD8^+$ cells cluster separately, as do the naïve cells from the other subsets, the EF, CM and EM cells are spread within a single large cluster for both $CD4^+$ and $CD8^+$ cells, highlighting that the expression of very few genes contribute towards distinguishing these subsets.

Additionally one can also explore the presence of batch effects by coloring individual points on the PCA/t-SNE plots by their batch ID. Figure 6b shows that the cells cluster by their type ($CD8^+$ vs. $CD8^+$) rather than their plate ID, suggesting that batch effects are absent/weak in our data. Alternatively one can also color points by their library quality metrics (e.g., number of genes detected and mapping rate) to explore whether technical effects drive the dominant modes of variation in the data. A number of computational strategies have been published to correct for batch effects (Table 2) [34, 52].

Typical downstream steps involve partitioning the cells into different discrete "subsets" using methods such as hierarchical, k-means or density based clustering, which we do not describe here. Many such methods, however, are available as R packages [53–55].

Finding Differentially Expressed Genes

10. *Finding differentially expressed (DE) genes:* An important task in gene expression analysis is to find which genes are differentially expressed between two subsets of cells. Finding a set of DE genes might provide important clues regarding biological pathways that might be active in one subset compared to another, for example. Finding a set of reliable DE genes is a statistical exercise, and methods must take into account the

important sources of technical variability in the data ("noise"). One of the most important sources of uncertainty in scRNA-seq data is the low capture rates of transcripts during the reverse transcription step (*see* **Note 5**), which can lead to the "dropout" of genes in some libraries, even though they have non-zero expression in the biological sample [31]. Another source of technical noise is the differences in the quality of libraries that some cells produce because of upstream conditions (e.g., apoptosis in some cells, or RNA degradation). In any ordinary setting, these can confound true biological differences.

We use the R-package Single Cell Differential Expression (SCDE, Table 2) for performing differential expression analysis, which adopts a Bayesian approach to account for technical sources of variation typical in scRNA-seq [31]. SCDE may be invoked as follows for finding differentially expressed genes between $CD4^+$ naive cells and $CD4^+$ effector memory cells (*see* **Note 21**):

```
markers.CD4=scde_test(C=Counts_mat1, grp1="CD4.EM", grp2="CD4.NA",
min.fold=3)
nrow(markers.CD4)
[1] 733
```

The last output suggests that SCDE was able to detect 733 genes differentially expressed between naïve and effector memory T cells at an average log-fold change higher than 3 between the subsets. The heatmaps in Fig. 6 depict these results vividly, and were generated using the following commands:

```
CD4.NA.cells=sample.ids == "CD4.NA"
CD4.EM.cells=which(sample.ids == "CD4.EM")
A=t(scale(t(logTPM_mat1[rownames(markers.CD4[order(markers.CD4$mle),]),
c(CD4.NA.cells,CD4.EM.cells)])))
aheatmap(A,color="-RdYlBu2:10", Rowv=NA, Colv=NA)
```

We repeated the above steps to find differentially expressed genes between naïve and effector memory CD8 positive T cells, which yielded 761 differentially expressed genes (Fig. 6c, d heatmap). Interestingly, 151 genes were common between the two sets, a result that was highly statistically significant ($p < 10^{-42}$, hypergeometric test).

4 Conclusions

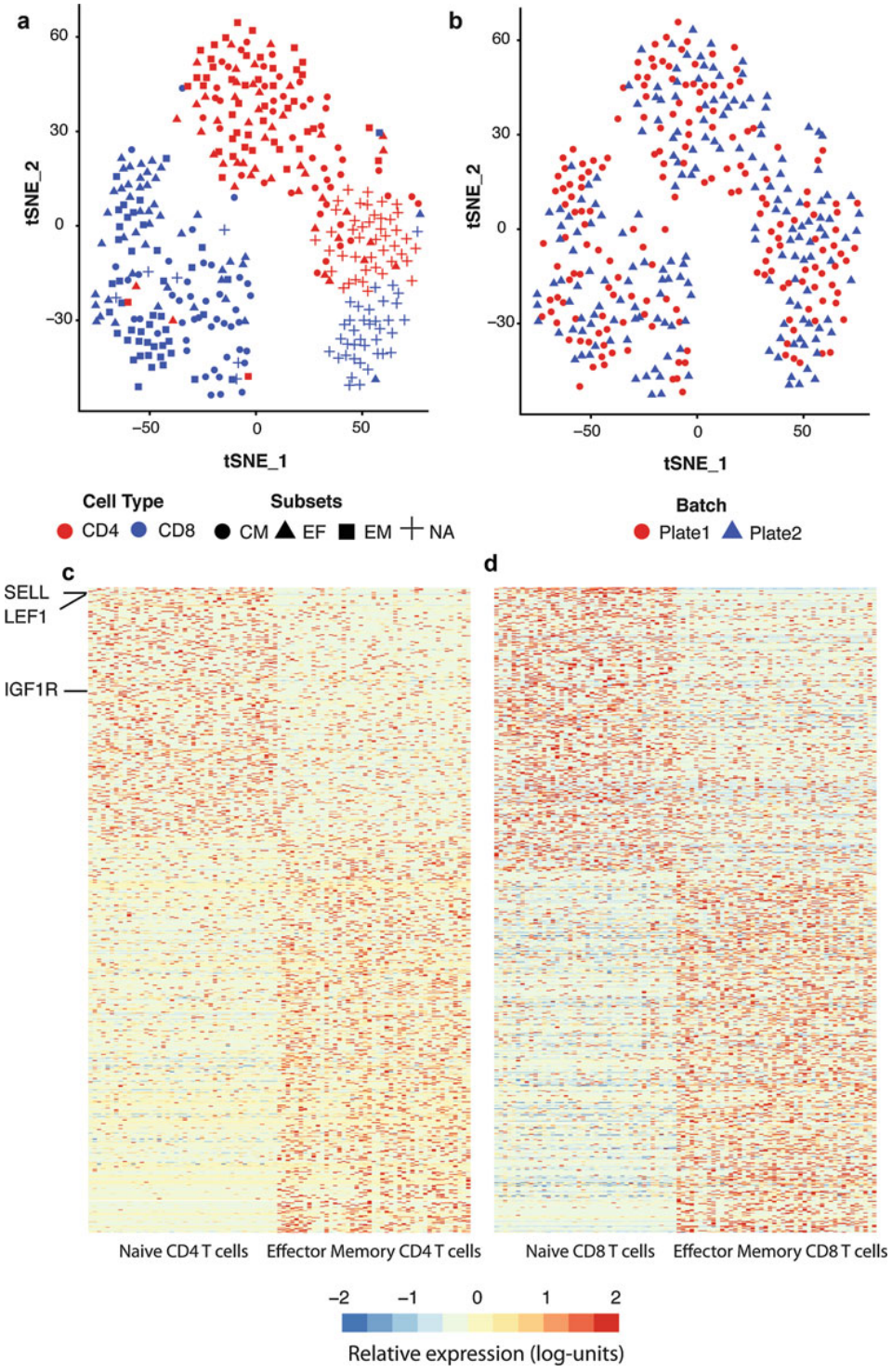
Our aim in this chapter is to introduce the uninitiated T cell biologist to key experimental and computational steps in single cell RNA-sequencing analysis. Through our working example, we show that profiling of human T cells using scRNA-seq enables hypothesis-free evaluation of cellular expression states using

off-the-shelf computational methods. A key challenge for the future is to use this powerful technique to explore the causes and consequences of T cell heterogeneity in a number of contexts, ranging from function exhaustion in diseased states to the interplay between clonal dynamics and expression state during immune responses.

5 Notes

1. If this procedure is performed on a standard bench top (i.e., not in an RNA fume hood), keep the plate covered (e.g., using the lid from a fresh box of pipet tips, placed slightly ajar) to prevent dust and debris from falling into samples during procedure, thus limiting contamination.
2. The magnet described draws the beads to alternating sides of each column of the plate. It is therefore recommended to use a P200 eight-channel pipette to remove the supernatant.
3. Let the beads dry on the magnet to limit dried beads from leaving the wells and cross-contaminate adjacent wells. Furthermore, as beads are electrostatics, recovery will be improved if beads are dried on the magnet. Noteworthy, allowing beads to dry for too long (beyond 10 min) may impair recovery of RNA. Immediately proceed with protocol when visible cracks appear in the pellet of beads. If all residual ethanol was promptly removed, beads should crack within 10 min.
4. Once the beads are dried, it is best to deposit the 4 μl of master mix on the bead pellet while the plate is still on the magnet to limit bead loss, which could otherwise easily escape the well once dried if not on the magnet because of their electrostatic properties. Then carefully move the plate away from the magnet and resuspend the 4 μl with P20 multichannel and low retention tips. The solution should be homogeneous after resuspension; it is important to pipette slowly to limit the generation of bubbles.

Fig. 6 Unsupervised clustering analysis of different T cell subsets. (**a**) tSNE scatter plot of 180 $CD4^+$ (*red*) and 177 $CD8^+$ (*blue*) T cells, which was generated using the top 10 principal components in the data. The subsets are highlighted by different symbols. (**b**) The data was generated through 2 experimental batches (2 batches of 96 $CD4^+$ and 96 $CD8^+$ cells each). The mixing of cells from the two batches (*red circles* and *blue squares*) in the tSNE plot in panel **b** suggests that no significant batch effects are observed across the 180 $CD4^+$ and 177 $CD8^+$ T cells analyzed. (**c**, **d**) Heatmaps were generated to illustrate the output from using SCDE software package. Heatmap in panel (**c**) highlights the 733 genes differentially expressed between naïve and effector memory $CD4^+$ T cells at an average log-fold change higher than 3 between the subsets. (**d**) Heatmap of the 761 differentially expressed genes expressed between naïve and effector memory $CD8^+$ T cells


a
b
60
30
0
−30
tSNE_2
−50
0
50
tSNE_1
Cell Type
CD4
CD8
Subsets
CM
EF
EM
NA
Batch
Plate1
Plate2
c
d
SELL
LEF1
IGF1R
Naive CD4 T cells
Effector Memory CD4 T cells
Naive CD8 T cells
Effector Memory CD8 T cells
−2
−1
0
1
2
Relative expression (log-units)

5. For the reverse transcription (RT) step of the SS2 protocol, both Picelli et al. [25, 26], and Trombetta et al. [27] recommend using SuperScript II Reverse Transcriptase (ThermoFisher Scientific), while Satija et al. [75] reports using instead Maxima Reverse Transcriptase (ThermoFisher Scientific), which is significantly cheaper. As described above, the RT step can be inefficient with low capture rates of transcripts during the RT step, resulting in an important source of uncertainty in scRNA-seq data and leading to the "dropout" of genes in some libraries. If a low number of transcripts are detected, it is possible to troubleshoot the experimental system being used by trying different reverse transcription enzymes. Noteworthy, when using switching RT enzyme, it is important to adjust the optimal reaction temperature following the manufacturer's recommendation (e.g., SuperScript II Reverse Transcriptase optimal reaction temperature is 42 °C, while Maxima Reverse Transcriptase's is between 50 and 55 °C). In the case of T cells, both the SuperScript II Reverse Transcriptase and the Maxima Reverse Transcriptase work well at optimal reaction temperature. The data included in the figures of this protocol were generated using SuperScript II Reverse Transcriptase. Additionally, as originally reported by Picelli et al. [25, 26], trehalose and betaine may be added to the RT mix to limit the formation of RNA secondary structure (e.g., hairpins or loops) during the RT reaction, which might cause the enzyme terminating chain elongation owing to steric hindrance.
6. Reference transcriptomes are usually available as a FASTA file of transcript sequences, or as a tabular GTF file which indicates the positions of the transcripts along a reference genome. Reference genomes currently exist for many model and nonmodel organisms, and are available as FASTA files (*see* **Note 13**). In cases where a reference genome is available but not a reference transcriptome, it is possible to assemble it ab initio using reads mapped to a reference genome [56]. In cases where the genome of an organism has not been sequenced, it is possible to assemble a transcriptome de novo from bulk RNA-seq data [57].
7. As a general rule, it is implicit that the names of files in the commands specified here must include the full path to their location on the server or be available in the user's current working directory.
8. SAM is an abbreviation for "Sequence Alignment Map" format. It is one of many standard formats in which commonly used genomics software output their results. It is a human readable tab-delimited text format, each line of which corresponds to the alignment of a single read. The alignment line

has 11 mandatory fields for important alignment information such as mapping position and mapping quality score, and a number of optional fields. The BAM format is a compressed version of the SAM format that is not directly human readable but can be efficiently read using programs like samtools (https://samtools.github.io), a suite of widely used tools for reading and manipulating SAM/BAM files. Full details of the SAM/BAM format are specified here, https://samtools.github.io/hts-specs/SAMv1.pdf.

9. RNA-SeQC [58] is a single java program built on the Picard API, which computes a series of quality control metrics for RNA-seq data using the BAM files as input, and outputs a series of HTML reports and tab delimited text files summarizing QC metrics. It can be downloaded from, https://www.broadinstitute.org/cancer/cga/rna-seqc. The RSeQC package [59] provides a number of useful scripts written in the python language that use the alignment BAM files as input to perform various individual tasks like calculating sequence quality, GC bias, and coverage uniformity. It can be downloaded from http://rseqc.sourceforge.net.
10. To visualize multiple sample alignments using minimal computer memory, convert the BAM files to the tiled data file (TDF) format, which summarizes the pileup of read counts along the genome. Instructions to convert BAM files to TDF format are provided in the IGV documentation, https://www.broadinstitute.org/igv/.
11. Alternative commonly used tools to perform transcript/gene quantification from reads aligned to a reference transcriptome include RNA-seq by Expectation Maximization (RSEM) [60] and Cufflinks [37].
12. Kallisto is a recently published program, and is based on the idea of "pseudoalignment," wherein every read is queried for its compatibility with targets along the transcriptome without the need for an exact alignment. As demonstrated in Bray et al. [40] "pseudoalignment" not only computationally outperforms existing tools, but is also more robust to sequencing errors compared to techniques that involve exact alignment.
13. FASTA files use a standard text-based format for representing nucleotide sequences or protein sequences. The format also allows for sequence names and comments (meta-data) to precede the actual sequences as "headers." FASTA is a simple format, which makes it easy for most bioinformatics tools and programming languages to manipulate and parse sequences. More information can be found at https://en.wikipedia.org/wiki/FASTA_format.

14. Here we used the Hg19 transcriptome (UCSC) reference annotation. The metadata files are available on request to the authors of the manuscript.

15. Because of differences in overall sequencing depth between samples and the difference in transcript lengths, the raw read counts of gene *i* in cell *j* C*ij* do not necessarily reflect the abundance of transcripts of gene *i* in cell *j*. It is common practice therefore to normalize the read counts, and multiple approaches exist. Here we use transcripts-per-million TPM*ij*, which is the expected number of transcripts of gene *i* in cell *j* per million transcripts, given the relative length of gene *i* and the abundances of all the other genes in cell *j*. As the name suggests, the TPM values for all the genes within a given cell sum to 10^6. Kallisto automatically provides a column of calculated TPM values in the abundance.tsv file.

16. In R, these can be accomplished by repetitively using the read.table command, which reads individual tsv files as a vector, and applying the cbind command on these vectors to bind them into a matrix or a R data frame. The resulting matrix should have the genes as rows and individual single-cell samples as columns.

17. Barplot for visualizing library properties and genes. The following code can be copied and run in the console window in RStudio. This will make the function barplot_tcell available to use. Note that sentences preceded by "#" are treated as comments and not executed by R,

```
barplot_tcell=function(x=NULL, id=NULL, name=NULL){
   #Inputs:
        # x=numeric vector
   # id=factor, indicating sample ids with the same length and order as x
   # name=string, y-axis label
   #Outputs:
        # Bar plot
   df=data.frame(Value=as.numeric(x), id=as.character(id))
   means.sem=ddply(df, c("id"), summarise, mean=mean(Value),
sem=sd(Value))
   means.sem<- transform(means.sem, lower=mean-sem, upper=mean+sem)
   print(ggplot(means.sem, aes(x=id, y=mean, fill=factor(id))) +
geom_bar(stat="identity", color="black") +
      geom_errorbar(aes(ymax=upper,ymin=lower), position=position_
dodge(0.9), data=means.sem, width=0.3) +
      theme(axis.text.x=element_text(angle=45, hjust=1))+
xlab("Sample")+ylab(name)+theme_bw())
}
```

18. The Coefficient of Variation (CV) is a measure of variability of the data and is defined as the ratio of the standard deviation by the mean.

19. Scatter plot for visualizing PCA projections of cells. The following code can be copied and run in the R terminal. This will make the function scatter_plot available to use. Note that sentences preceded by "#" are treated as comments and not executed by R.

```
scatter_plot=function(X=NULL, id=NULL, axis.labels=c("PC1","PC2")){
#Inputs:
# X - Any matrix with dimensions (cells X features. Here the features
are the PCA scores or the tSNE coordinates
# id -=factor, indicating sample ids with the same length and order
as the number of rows in X
# Axis labels=x- and y-axis labels for the plot
#Output:
# Barplot
df=as.data.frame(X)
colnames(df)=c("x","y")
df$sample=substr(id,5,6)
df$type=substr(id,1,3)
p=ggplot(df, aes(x,y))+geom_point(aes(shape=sample,
color=factor(type)), size=4)+scale_color_manual(values=c("red",
"blue"), name="Cell Type")+ylab(axis.labels[2])+xlab(axis.labels[1])
p<- p+theme(axis.title.x=element_text(face="bold", colour="#990000",
size=16), axis.text.x=element_text(angle=0, vjust=0.5, size=12))
p<- p+theme(axis.title.y=element_text(face="bold", colour="#990000",
size=16), axis.text.y=element_text(angle=0, vjust=0.5, size=12))
p<- p+theme(plot.title=element_text(size=12, face="bold"))+theme_
bw()
print(p)
}
```

20. Barplot for visualizing genes that drive PCs. The following code can be copied and run in the R terminal, and will make the function barplot_loadings available to use. Note that sentences preceded by "#" are treated as comments and not executed by R.

```
barplot_loadings=function(X=pca.loadings,pc.use=1, num.genes=8){
#Inputs :
# X=matrix (genes by PCs) of PCA loadings output by prcomp
# pc.use=The PC for which loadings are desired
# num.genes=The number of top+ve and -ve genes to plot
#Output :
# Barplot
ord=order(X[,pc.use]); vals=X[ord,pc.use]
genes.select=c(1:num.genes, (nrow(X)-num.genes):nrow(X))
vals=vals[genes.select];
names(vals)=rownames(X)[ord][genes.select];
df=data.frame(loadings=vals, genes=factor(names(vals), lev-
els=names(vals)))
print(ggplot(df, aes(x=genes, y=loadings))+geom_bar(position="iden-
tity", stat="identity", color="black",fill="blue") + +theme_bw() +
theme(axis.text.x=element_text(angle=45, hjust=1))+xlab("Gene")+ylab
(paste0("PC",pc.use," loadings")))
}
```

21. Invoking SCDE for differential expression analysis. The following code can be copied and run in the R terminal, and will make the wrapper function scde_test available to use. The following command can be copied and run in the R terminal, and will make the function scatter_plot available to the user:

```
scde_test=function(C=NULL, grp1=NULL, grp2=NULL, min.fold=3){
#Inputs:
#C - Counts matrix (genes x cells)
#grp1, grp2 - a character string indicating the sample names (e.g.,
CD4.NA, CD4.EM) that can be used to extract relevant cells.
#min.fold - minimum log-fold change between the two samples for a
marker to be considered for the differential expression test (Default
value 3)
#Output:
# Table showing differentially expressed genes, their inferred log-fold
changes and statistical score.
cells.1=grep(grp1,colnames(C),value=TRUE);
cells.2=grep(grp2,colnames(C),value=TRUE);
sg=factor(c(rep(grp1,length(cells.1)), rep(grp2, length(cells.2))),
levels=c(grp1,grp2))
names(sg)=c(cells.1, cells.2);
err.model=scde.error.models(counts=round(C[,c(cells.1,cells.2)]),
groups=sg, n.cores=8, verbose=1);
valid.cells=err.model$corr.a>0
err.model=err.model[valid.cells,];
prior<- scde.expression.prior(models=err.model,
counts=round(C[,valid.cells]), show.plot=F)
diff.exp=scde.expression.difference(err.model,
round(C[,valid.cells]), prior, groups=sg[valid.cells],
n.randomizations=100, n.cores=8, verbose=1)
diff.exp<- subset(diff.exp, ((lb>0 & ub>0) | (lb<0 & ub<0)) &
abs(mle)>min.fold)
return(diff.exp[,c("mle","Z")])}
```

Acknowledgments

K. S. and A. C. V. thank Brian Haas, Dr. Timothy Tickle, Dr. Petter Brodin, Dr. Sara Garamszegi, Chris Rodman, and Noga Rogel for helpful comments and feedback on the content and organization of the manuscript. K. S. and A. C. V. thank Dr. Aviv Regev and Dr. Nir Hacohen for feedback and support.

References

1. Lever M, Maini PK, van der Merwe PA et al (2014) Phenotypic models of T cell activation. Nat Rev Immnunol 14(9):619–629. doi:10.1038/nri3728

2. Yui MA, Rothenberg EV (2014) Developmental gene networks: a triathlon on the course to T cell identity. Nat Rev Immnunol 14(8):529–545. doi:10.1038/nri3702

3. Tscharke DC, Croft NP, Doherty PC et al (2015) Sizing up the key determinants of the CD8+ T cell response. Nat Rev Immnunol 15(11):705–716. doi:10.1038/nri3905
4. Farber DL, Yudanin NA, Restifo NP (2014) Human memory T cells: generation, compartmentalization and homeostasis. Nat Rev Immnunol 14(1):24–35. doi:10.1038/nri3567
5. Geginat J, Paroni M, Maglie S et al (2014) Plasticity of human CD4 T cell subsets. Front Immunol 5:630. doi:10.3389/fimmu.2014.00630
6. Zhu J, Paul WE (2010) Heterogeneity and plasticity of T helper cells. Cell Res 20(1):4–12. doi:10.1038/cr.2009.138
7. O'Shea JJ, Paul WE (2010) Mechanisms underlying lineage commitment and plasticity of helper CD4+ T cells. Science 327(5969):1098–1102. doi:10.1126/science.1178334
8. O'Garra A, Vieira P (2007) T(H)1 cells control themselves by producing interleukin-10. Nat Rev Immunol 7(6):425–428
9. Zhou X, Bailey-Bucktrout S, Jeker LT et al (2009) Plasticity of CD4(+) FoxP3(+) T cells. Curr Opin Immunol 21(3):281–285. doi:10.1016/j.coi.2009.05.007
10. Chattopadhyay PK, Gierahn TM, Roederer M et al (2014) Single-cell technologies for monitoring immune systems. Nat Immunol 15(2):128–135. doi:10.1038/ni.2796
11. Stegle O, Teichmann SA, Marioni JC (2015) Computational and analytical challenges in single-cell transcriptomics. Nat Rev Genet 16(3):133–145. doi:10.1038/nrg3833
12. Raj A, Peskin CS, Tranchina D et al (2006) Stochastic mRNA synthesis in mammalian cells. PLoS Biol 4(10), e309
13. Wu AR, Neff NF, Kalisky T et al (2014) Quantitative assessment of single-cell RNA-sequencing methods. Nat Methods 11(1):41–46. doi:10.1038/nmeth.2694
14. Ozsolak F, Milos PM (2011) RNA sequencing: advances, challenges and opportunities. Nat Rev Genet 12(2):87–98. doi:10.1038/nrg2934
15. Tang F, Barbacioru C, Bao S et al (2010) Tracing the derivation of embryonic stem cells from the inner cell mass by single-cell RNA-Seq analysis. Cell Stem Cell 6(5):468–478. doi:10.1016/j.stem.2010.03.015
16. Tang F, Barbacioru C, Wang Y et al (2009) mRNA-seq whole-transcriptome analysis of a single cell. Nat Methods 6(5):377–382. doi:10.1038/nmeth.1315
17. Kurimoto K, Yabuta Y, Ohinata Y et al (2006) An improved single-cell cDNA amplification method for efficient high-density oligonucleotide microarray analysis. Nucleic Acids Res 34(5), e42
18. Hashimshony T, Wagner F, Sher N et al (2012) CEL-seq: single-cell RNA-seq by multiplexed linear amplification. Cell 2(3):666–673. doi:10.1016/j.celrep.2012.08.003
19. Jaitin DA, Kenigsberg E, Keren-Shaul H et al (2014) Massively parallel single-cell RNA-seq for marker-free decomposition of tissues into cell types. Science 343(6172):776–779. doi:10.1126/science.1247651
20. Macosko EZ, Basu A, Satija R et al (2015) Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. Cell 161(5):1202–1214. doi:10.1016/j.cell.2015.05.002
21. Klein AM, Mazutis L, Akartuna I et al (2015) Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. Cell 161(5):1187–1201. doi:10.1016/j.cell.2015.04.044
22. Soumillon M, Cacchiarelli D, Semrau S et al (2014) Characterization of directed differentiation by high-throughput single-cell RNA-Seq. BioRxiv. doi: http://dx.doi.org/10.1101/003236
23. Islam S, Zeisel A, Joost S et al (2014) Quantitative single-cell RNA-seq with unique molecular identifiers. Nat Methods 11(2):163–166. doi:10.1038/nmeth.2772
24. Ramsköld D, Luo S, Wang YC et al (2012) Full-length mRNA-Seq from single-cell levels of RNA and individual circulating tumor cells. Nat Biotechnol 30(8):777–782
25. Picelli S, Björklund ÅK, Faridani OR et al (2013) Smart-seq2 for sensitive full-length transcriptome profiling in single cells. Nat Methods 10(11):1096–1098. doi:10.1038/nmeth.2639
26. Picelli S, Faridani OR, Björklund AK et al (2014) Full-length RNA-seq from single cells using Smart-seq2. Nat Protoc 9(1):171–181. doi:10.1038/nprot.2014.006
27. Trombetta JJ, Gennert D, Lu D et al (2014) Preparation of single-cell RNA-seq libraries for next generation sequencing. Curr Protoc Mol Biol 107:4.22.1–4.22.17. doi:10.1002/0471142727.mb0422s107
28. Stubbington MJT, Lönnberg T, Proserpio V et al (2015) Simultaneously inferring T cell fate and clonality from single cell transcriptomes. BioRxiv. doi: http://dx.doi.org/10.1101/025676
29. Grün D, Kester L, van Oudenaarden A (2014) Validation of noise models for single-cell transcriptomics. Nat Methods 11(6):637–640. doi:10.1038/nmeth.2930
30. Grün D, van Oudenaarden A (2015) Design and analysis of single-cell sequencing experiments. Cell 163(4):799–810. doi:10.1016/j.cell.2015.10.039

31. Hicks SC, Teng M, Irizarry RA (2015) On the widespread and critical impact of systematic bias and batch effects in single-cell RNA-Seq data. BioRxiv. doi: http://dx.doi.org/10.1101/025528
32. Kharchenko PV, Silberstein L, Scadden DT (2014) Bayesian approach to single-cell differential expression analysis. Nat Methods 11(7):740–742. doi:10.1038/nmeth.2967
33. Luo H, Li J, Chia BKH et al (2014) The importance of study design for detecting differentially abundant features in high-throughput experiments. Genome Biol 15(12):527. doi:10.1186/s13059-014-0527-7.Johnson
34. Evan W, Li C et al (2007) Adjusting batch effects in microarray expression data using empirical Bayes methods. Biostatistics 8(1):118–127. doi:10.1093/biostatistics/kxj037
35. Leek JT, Scharpf RB, Bravo HC et al (2010) Tackling the widespread and critical impact of batch effects in high-throughput data. Nat Rev Genet 11(10):733–739. doi:10.1038/nrg2825
36. Qu K, Garamszegi S, Wu F, et al. (2016) Integrative genomic analysis by interoperation of bioinformatics tools in GenomeSpace. Nat Methods. doi: 10.1038/nmeth.3732
37. Trapnell C, Roberts A, Goff L et al (2012) Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. Nat Protoc 7(3):562–578. doi:10.1038/nprot.2012.016
38. Engström PG, Steijger T, Sipos B et al (2013) Systematic evaluation of spliced alignment programs for RNA-seq data. Nat Methods 10(12):1185–1191. doi:10.1038/nmeth.2722
39. Mortazavi A, Williams BA, McCue K et al (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. Nat Methods 5(7):621–628. doi:10.1038/nmeth.1226
40. Bray N, Pimentel H, Melsted P et al (2015) Near-optimal RNA-seq quantification. BioRxiv 1505:02710
41. Trapnell C (2015) Defining cell types and states with single-cell genomics. Genome Res 25(10):1491–1498. doi:10.1101/gr.190595.115
42. Gentleman RC, Carey VJ, Bates DM et al (2004) Bioconductor: open software development for computational biology and bioinformatics. Genome Biol 5(10):R80
43. Smith LI (2002) A tutorial on principal components analysis. Cornell Univ 51:52
44. Shalek AK, Satija R, Adiconis X et al (2013) Single-cell transcriptomics reveals bimodality in expression and splicing in immune cells. Nature 498(7453):236–240. doi:10.1038/nature12172
45. Shalek AK, Satija R, Shuga J et al (2014) Single-cell RNA-seq reveals dynamic paracrine control of cellular variation. Nature 510(7505):363–369. doi:10.1038/nature13437
46. Pollen AA, Nowakowski TJ, Shuga J et al (2014) Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. Nat Biotechnol 32(10):1053–1058. doi:10.1038/nbt.2967
47. Usoskin D, Furlan A, Islam S et al (2015) Unbiased classification of sensory neuron types by large-scale single-cell RNA sequencing. Nat Neurosci 18(1):145–153. doi:10.1038/nn.3881
48. Steinke FC, Yu S, Zhou X et al (2014) TCF-1 and LEF-1 act upstream of Th-POK to promote the CD4(+) T cell fate and interact with Runx3 to silence CD4 in CD8(+) T cells. Nat Immunol 15(7):646–656. doi:10.1038/ni.2897
49. Mingueneau M, Kreslavsky T, Gray D et al (2013) The transcriptional landscape of αβ T cell differentiation. Nat Immunol 14(6):619–632. doi:10.1038/ni.2590
50. Van der Maaten L, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9:2579–2605
51. Peres-Neto PR, Jackson DA, Somers KM (2005) How many principal components? Stopping rules for determining the number of non-trivial axes revisited. Comput Stat Data Anal 49(4):974–997. doi:10.1016/j.csda.2004.06.015
52. Risso D, Ngai J, Speed TP et al (2014) Normalization of RNA-seq data using factor analysis of control genes or samples. Nat Biotechnol 32(9):896–902. doi:10.1038/nbt.2931
53. Suzuki R, Shimodaira H (2013) Hierarchical clustering with P-values via multiscale bootstrap resampling. R package
54. Bodenhofer U, Kothmeier A, Hochreiter S (2011) APCluster: an R package for affinity propagation clustering. Bioinformatics 27(17):2463–2464. doi:10.1093/bioinformatics/btr406
55. Fraley C, Raftery AE (2006) MCLUST version 3: an R package for normal mixture modeling and model-based clustering. Washington Univ Seattle Dept of Statistics
56. Trapnell C, Williams BA, Pertea G et al (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nat Biotechnol 28(5):511–515. doi:10.1038/nbt.1621

57. Grabherr MG, Haas BJ, Yassour M et al (2011) Full-length transcriptome assembly from RNA-seq data without a reference genome. Nat Biotechnol 29(7):644–652. doi:10.1038/nbt.1883
58. DeLuca DS, Levin JZ, Sivachenko A et al (2012) RNA-SeQC: RNA-seq metrics for quality control and process optimization. Bioinformatics 28(11):1530–1532. doi:10.1093/bioinformatics/bts196
59. Wang L, Wang S, Li W (2012) RSeQC: quality control of RNA-seq experiments. Bioinformatics 28(16):2184–2185. doi:10.1093/bioinformatics/bts356
60. Li B, Dewey CN (2011) RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. BMC Bioinformatics 12:323. doi:10.1186/1471-2105-12-323
61. Eberwine J, Yeh H, Miyashiro K et al (1992) Analysis of gene expression in single live neurons. Proc Natl Acad Sci U S A 89(7):3010–3014
62. Morris J, Singh JM, Eberwine JH (2011) Transcriptome analysis of single cells. J Vis Exp 50:e2634. doi:10.3791/2634
63. Pan X, Durrett RE, Zhu H et al (2013) Two methods for full-length RNA sequencing for low quantities of cells and single cells. Proc Natl Acad Sci U S A 110(2):594–599. doi:10.1073/pnas.1217322109
64. Kang Y, Norris MH, Zarzycki-Siek J et al (2011) Transcript amplification from single bacterium for transcriptome analysis. Genome Res 21(6):925–935. doi:10.1101/gr.116103.110
65. Islam S, Kjallquist U, Moliner A et al (2012) Highly multiplexed and strand-specific single-cell RNA 5′ end sequencing. Nat Protoc 7(5):813–828. doi:10.1038/nprot.2012.022
66. Pierson E, Yau C (2015) ZIFA: dimensionality reduction for zero-inflated single-cell gene expression analysis. Genome Biol 16:241. doi:10.1186/s13059-015-0805-z
67. Trapnell C, Cacchiarelli D, Grimsby J et al (2014) The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nat Biotechnol 32(4):381–386. doi:10.1038/nbt.2859
68. Buettner F, Natarajan KN, Casale FP et al (2015) Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. Nat Biotechnol 33(2):155–160. doi:10.1038/nbt.3102.Vallejos
69. Vallejos CA, Marioni JC, Richardson S (2015) BASiCS: Bayesian analysis of single-cell sequencing data. PLoS Comput Biol 11(6):e1004333. doi:10.1371/journal.pcbi.1004333
70. Fan J, Salathia N, Liu R et al (2016) Characterizing transcriptional heterogeneity through pathway and gene set overdispersion analysis. Nat Methods. doi: 10.1038/nmeth.3734
71. Juliá M, Telenti A, Rausell A (2015) Sincell: an R/bioconductor package for statistical assessment of cell-state hierarchies from single-cell RNA-seq. Bioinformatics 31(20):3380–3382. doi:10.1093/bioinformatics/btv368
72. Grün D, Lyubimova A, Kester L et al (2015) Single-cell messenger RNA sequencing reveals rare intestinal cell types. Nature 525(7568):251–255. doi:10.1038/nature14966
73. Marco E, Karp RL, Guo G et al (2014) Bifurcation analysis of single-cell gene expression data reveals epigenetic landscape. Proc Natl Acad Sci U S A 111(52):E5643–E5650. doi:10.1073/pnas.1408993111
74. Scater. https://github.com/davismcc/scater
75. Satija R, Farrell JA, Gennert D et al (2015) Spatial reconstruction of single-cell gene expression data. Nat Biotechnol 33(5):495–502. doi:10.1038/nbt.3192