scPipe: a flexible data preprocessing pipeline for single-cell RNA-sequencing data

Luyi Tian[1,2,*], Shian Su[1], Daniela Amann-Zalcenstein[1], Christine Biben[1], Shalin H. Naik[1], Matthew E. Ritchie[1,2,3,*],

1 Molecular Medicine Division, The Walter and Eliza Hall Institute of Medical Research, Parkville, 3052, Australia.
2 Department of Medical Biology, The University of Melbourne, Parkville, 3010, Australia.
3 School of Mathematics and Statistics, The University of Melbourne, Parkville, 3010, Australia.

*** tian.l@wehi.edu.au, mritchie@wehi.edu.au**

Abstract

Single-cell RNA sequencing (scRNA-seq) technology allows researchers to profile the transcriptomes of thousands of cells simultaneously. Protocols that incorporate both designed and random barcodes to label individual cells and molecules have greatly increased the throughput of scRNA-seq, but give rise to a more complex data structure. There is a need for new tools that can handle the various barcodes used by different protocols and exploit this information for quality assessment at the sample-level and provide effective visualization of these results in preparation for higher-level analyses.

We developed scPipe, an R package that allows barcode demultiplexing, read alignment and gene-level quantification and quality control of raw sequencing data generated by multiple 3 prime end sequencing protocols that include CEL-seq, MARS-seq and Drop-seq. scPipe produces a count matrix that is essential for downstream analysis along with an HTML report that summarises data quality. These results can be used as input for downstream analyses including normalization, visualization and statistical testing. scPipe performs this processing in a few simple R commands, promoting reproducible analysis of single-cell data that is compatible with the emerging suite of scRNA-seq analysis tools available in R/Bioconductor. The *scPipe* R package is available from https://github.com/LuyiTian/scPipe.

Introduction

Single-cell transcriptome profiling offers researchers a powerful method for studying gene regulation that allows transcriptional heterogeneity to be explored in a manner that is not possible using bulk RNA sequencing. The introduction of cellular barcodes, which are sequences attached to the dT-primer that are distinct for each cell, has increased the throughput and substantially reduced the cost of single-cell RNA sequencing (scRNA-seq). These barcodes allow for the demultiplexing of reads after samples are pooled together for sequencing. Apart from cellular barcodes, molecular barcodes or unique molecular identifiers (UMIs), are frequently employed to remove PCR duplicates and allow identification of unique mRNA molecules, thereby reducing technical noise. The multiple levels of barcoding used in scRNA-seq experiments create additional challenges in data processing together with new opportunities for quality control (QC). Different protocols use different barcode configurations, which means a flexible approach to data preprocessing is required. Existing methods such as *UMI-tools* [4], *umitools* (http://brwnj.github.io/umitools/) and *umis* [5] have been developed for handling random UMIs and packages such as *scater* [3] and *scran* [2] can preprocess the resulting counts by performing general QC and normalization of scRNA-seq data. *scPipe* was developed to address the lack of a comprehensive workflow for processing sequencing data from 3 prime (3') end protocols that can deal with both UMIs and sample barcodes, map reads to the genome and summarise these results into gene-level counts. Additionally this pipeline collates useful metrics for QC during preprocessing.

Approach

scPipe is an R package that takes as its input fastq data generated from all popular 3' end scRNA-seq protocols and their variants, such as CEL-seq, MARS-seq and Drop-seq. Although not specifically designed for non-UMI protocols, it can also process data generated by Smart-seq and Smart-seq2. The pipeline outputs both a gene count matrix and a variety of QC statistics that are presented in a standalone HTML report that includes QC plots and other data summaries. The *scPipe* package is written in R and C++ and uses the *Rhtslib* software (https://bioconductor.org/packages/Rhtslib) for BAM I/O.

The *scPipe* workflow (Figure 1A) begins with paired-end fastq data which is passed to the function `sc_trim_barcode`, which reformats the reads by moving the barcode and UMI information to the read header. There are options to perform some basic filtering in this step, including removing reads with low quality sequence in the barcode and UMI regions and filtering out low complexity reads, most of which are non-informative repetitive sequences such as polyA. These sequences can then be aligned using any popular read aligner that produces BAM formatted output. *scPipe* uses the R-based *Rsubread* [1] aligner by default. Aligned reads in the BAM file are assigned to the exons they overlap by the `sc_exon_mapping` function. This function records the mapping result, together with UMI and cell barcodes in the optional fields of the BAM file. Next, the `sc_demultiplex` function is used to demultiplex results per cell using the sample barcodes. For CEL-seq and MARS-seq, the demultiplexing is based on designed barcode sequences and allows for mismatches. For Drop-seq where the cell barcode sequences are random, the function `sc_detect_bc` can be applied to identify enriched barcodes in the data that can then be supplied to `sc_demultiplex`. The demultiplexed data are output in a csv file for each cell in which each row corresponds to a read that maps to a specific gene, and columns that include gene id, UMI sequence and mapping position. The overall barcode demultiplexing results are also recorded (Figure 1B). This information is used by the `sc_gene_counting` function to remove PCR duplicates (also known as UMI deduplication) and generate a gene count matrix. *scPipe* employs a simple yet intuitive method to correct for UMI sequencing errors by collapsing UMI sequences assigned to the same gene that have a low edit distance (1 by default). After UMI deduplication we get a gene count matrix that can be used for further analysis. QC information is collected at each step and includes the total number of mapped reads per cell, UMI deduplication statistics, per cell barcode demultiplexing statistics and UMI correction statistics and ERCC spike-in control statistics (where present).

All data are stored in an object of class `SCData`, which is inspired by the `SCESet` class in *scater* which can store gene expression and related meta data together in the one object. The `SCData` class includes additional slots for flow cytometry index data collected during cell sorting (often available in CEL-Seq experiments) and its dimension reduced form, QC information obtained during preprocessing, the type of gene id and the organism name, which can be useful in downstream functional enrichment analysis. An `SCData` object can be created by the `create_scd_by_dir` function using the data folder generated during preprocessing, or manually by specifying all the required slots. Next, alignment statistics for each cell can also be plotted using the `plotMapping` function (Figure 1C) and one can plot a pairwise scatter plot of QC metrics that includes the total molecules per cell and the number of genes detected using the `plotQC_pair` function (Figure 1D).

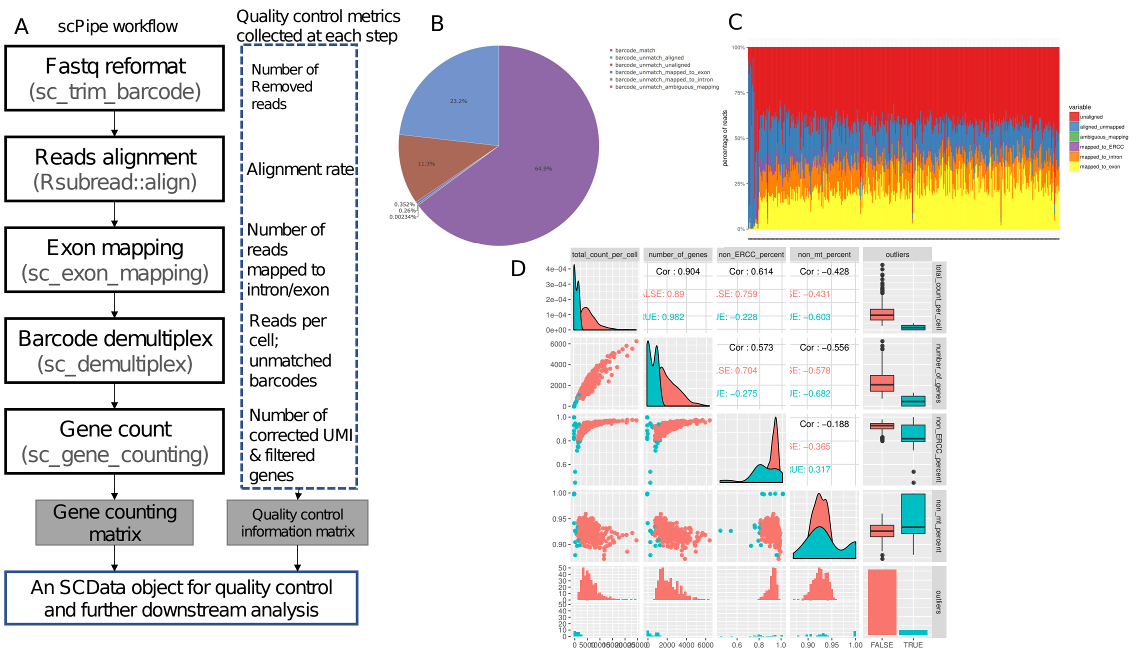
A
scPipe workflow
Quality control metrics collected at each step
Fastq reformat (sc_trim_barcode)
Reads alignment (Rsubread::align)
Exon mapping (sc_exon_mapping)
Barcode demultiplex (sc_demultiplex)
Gene count (sc_gene_counting)
Gene counting matrix
Quality control information matrix
An SCData object for quality control and further downstream analysis
Number of Removed reads
Alignment rate
Number of reads mapped to intron/exon
Reads per cell; unmatched barcodes
Number of corrected UMI & filtered genes
B
C
D
total_count_per_cell
number_of_genes
non_ERCC_percent
non_mt_percent
outliers
Cor : 0.904
Cor : 0.614
Cor : -0.428
Cor : 0.573
Cor : -0.556
Cor : -0.188

Figure 1. The workflow of *scPipe* and examples of the quality control plots it generates. (**A**) The preprocessing workflow of *scPipe*. Potential quality control statistics that are collected at each step are also listed. (**B**) pie chart illustrating the overall cell barcode matching results. (**C**) stacked bar plot showing the mapping rate, separated into reads that map to exon, intron and ERCCs, (**D**) pairwise scatter plot for quality control metrics of normal and outlier cells.

scPipe implements a simple outlier-based method for discovering low quality cells and possible doublets to remove from further analysis. The method uses 5 metrics (log-transformed number of molecules detected, the number of genes detected, ERCC recovery and the percentage of reads mapping to ribosomal, or mitochondrial genes) as input. A robustified Mahalanobis Distance is calculated for each cell with outliers automatically detected based on this distance using the *mclust* package (https://cran.r-project.org/package=mclust). Outlier detection can then be reconciled with visual inspection of the QC metrics through the afore-mentioned QC plotting options to fine tune the sample-specific filtering thresholds chosen. Normalization is then possible using packages such as *scran* and *scater*. An HTML report that includes all run parameters used, statistics for QC and various types of dimension reduction plots of both the gene expression data and QC metrics can be generated using the `create_report` function. All steps in an *scPipe* analysis can be run sequentially via the `run_scPipe` command that starts with the fastq files and related annotation and ends with an `SCData` object and an HTML report.

A vignette accompanying the package provides further details on implementation and example use cases. The example dataset included in the vignette, which contains 384 cells sequenced with 100 million reads, takes 1 hour and 40 minutes to process using *scPipe* on a standard Linux server.

Discussion

With the growing popularity of scRNA-seq technology, many tools have been developed for normalization, dimensionality reduction and clustering. There are few packages designed to handle the raw data obtained from many 3' end sequencing protocols with their associated UMIs and cell-specific barcodes from beginning to end. The *scPipe* package bridges this gap between the raw fastq files with mixed barcode types and transcript sequences and the gene count matrix that is the entry point for all downstream analysis. *scPipe* outputs numerous QC metrics obtained at each preprocessing step and displays these results in an HTML report to assist end users in QC evaluations. Monitoring QC metrics over time can be particularly useful for labs that routinely process single-cell data to allow them to assess the impact changes in lab processes and protocols have on data quality. Future improvements to *scPipe* include the parallelization of certain aspects of the preprocessing and transitioning data storage to classes defined in the *SingleCellExperiment* package (https://www.bioconductor.org/packages/SingleCellExperiment/).

Funding

This work was supported by the National Health and Medical Research Council (NHMRC), Australia (Fellowship GNT1104924 to MER, Project Grant GNT1124812 to SHN and MER), a Melbourne Research Scholarship (to LT), Victorian State Government Operational Infrastructure Support and Australian Government NHMRC IRIISS.

References

1. Y. Liao, G. K. Smyth, and W. Shi. The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote. *Nucleic Acids Res*, 41:e108, 2013.

2. A. T. L. Lun, D. J. McCarthy, and J. C. Marioni. A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. *F1000Research*, 5:2122, 2016.

3. D. J. McCarthy et al. Scater: pre-processing, quality control, normalisation and visualisation of single-cell RNA-seq data in R. *Bioinformatics*, 33:1179–86, 2017.

4. T. Smith, A. Heger, and I. Sudbery. UMI-tools: modeling sequencing errors in Unique Molecular Identifiers to improve quantification accuracy. *Genome Res*, 27:491–99, 2017.

5. V. Svensson et al. Power analysis of single-cell RNA-sequencing experiments. *Nat Methods*, 14:381–87, 2017.