Foundations and applications of single-cell RNA sequencing

Thesis by
Ali Sina Booeshaghi

In Partial Fulfillment of the Requirements
for the degree of
Doctor of Philosophy in Mechanical Engineering

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2022
(Defended 27th of May 2022)

Ali Sina Booeshaghi
ORCID: 0000-0002-6442-4502

ACKNOWLEDGEMENTS

Thank you to the many organisms, across the tree of life, who have sacrificed their lives so that humankind can better understand health and disease. May we pay it forward.

ABSTRACT

Single-cell RNA-sequencing is an experimental technique for studying cellular gene expression, with a multitude of engineering challenges. These challenges transcend the boundaries of traditional academic disciplines and the field of mechanical engineering, that aims to address roadblocks in critical technologies towards engineering our environment, is central to this endeavor.

This thesis addresses three engineering challenges that must be met in order to realize the goal of bringing single-cell RNA sequencing to the clinic. The first is scalable cellular isolation and sampling. Chapter 2 describes the *poseidon* and *colosseum* instruments that enable massive scale single-cell isolation and collection. They each have novel design elements that reduce cost and enable modularity, at a similar accuracy to expensive commercial alternatives.

The second challenge is the rapid preprocessing of single-cell RNA-sequencing data. Chapter 3 describes the *kallisto* | *bustools* command-line tools that make scalable scRNAseq analysis fast and efficient. These tools implement novel algorithms for sequence read-alignment, barcode error correction, and molecular counting that helps resolve ambiguities in sequence mapping.

The third challenge is refining gene expression data to the isoform level. This refinement is crucial for understanding transcriptional regulation and the effects of alternative splicing in biological processes. Towards that end, I have extended the *kallisto* | *bustools* workflow to process full-length scRNAseq data taking advantage of expectation maximization algorithm to disambiguate sequence alignments. Chapter 4 describes how I used these tools to assemble the first ever spatially-resolved single-cell isoform atlas, and in particular one of great interest in the neuroscience community (the mouse primary motor cortex) with data generated with three RNA-sequencing assays.

PUBLISHED CONTENT AND CONTRIBUTIONS

1. Melsted, Páll, Booeshaghi, A. Sina, et al. "Modular, efficient and constant-memory single-cell RNA-seq preprocessing." *Nature biotechnology* 39.7 (2021): 813-818. https://doi.org/10.1038/s41587-021-00870-2

P.M., **A.S.B**., L. Liu and L.P. developed the algorithms for bustools and P.M., **A.S.B**. and L. Liu wrote the software. **A.S.B**. conceived of and performed the UMI and barcode calculations motivating the algorithms. F.G. implemented and performed the benchmarking procedure, and curated indices for the datasets. **A.S.B**. and E.d.V.B. designed and produced the comparisons between Cell Ranger and kallisto bustools. L. Lu investigated in detail the performance of different workflows on the "10k mouse neuron" data and produced the analysis of that dataset. **A.S.B.** designed the RNA velocity workflow and performed the RNA velocity analyses. K.M.H contributed to the development of the reproducible workflow. K.E.H. developed and investigated the effect of reference transcriptome sequences for pseudoalignment. J.G. interpreted results and helped to supervise the research. **A.S.B.** planned, organized and prepared figures. **A.S.B.**, E.d.V.B., P.M. and L.P. planned the manuscript. **A.S.B.** and L.P. wrote the manuscript.

2. Yao, Zizhen, et al. "A transcriptomic and epigenomic cell atlas of the mouse primary motor cortex." *Nature* 598.7879 (2021): 103-110. https://doi.org/10.1038/s41586-021-03500-8

A.R., A.T., B.T., C.R., C.R.V., D.B., D.M., E.L.D., E.Z.M., H.T., H.Z., J. Goldy, J.S., K.C., K.L., K. Smith, M.K., M.T., N.D., N.M.N., O.F., T.C., T.N.N. and T.P. contributed to RNA data generation. A.B., A.C.R., A.I.A., A.P.-D., C.L., H.L., J.D.L., J.K.O., J.R.E., J.R.N., M.M.B., S.-Y.N. and Y.E.L. contributed to DNA methylation (snmC-seq2) data generation. A.P.-D., B.R., J.D.L., J.K.O., M.M.B., S.P., X.H., X.W. and Y.E.L. contributed to snATAC data generation. A.M., B.R.H., C.C., C.v.V., E.A.M., F.X., H.C., H.C.B., J.C., J. Goldy, J.K., J.O., M.G., M.H., O.R.W., R.F., R.H., R.S.A., S.A.A., S.-Y.N., V.F., W.I.D. and Z.Y. contributed to data archive/infrastructure. A.R., **A.S.B.**, B.T., D.R., E.A.M., E.D.V., E.P., E.Z.M., F.X., H.L., H.R.d.B., H.Z., J.D.W., J. Goldy, J. Gillis, J.O., K. Smith, K. Street, K.V.d.B., L.P., M.C., O.F., O.P., P.V.K., Q.H., R.F., S.D., S.F., S.-Y.N., T.B., V.N., V.S., W.I.D., Y.E.L. and Z.Y. contributed to data analysis. A.R., B.R., B.T., C.L., E.A.M., E.D.V., E.Z.M., F.X., H.L., H.Z., J.D.W., J. Gillis, J.N., M.C., M.M.B., P.V.K., Q.H., R.F., S.F., T.B., Y.E.L. and Z.Y. contributed to data interpretation. **A.S.B.**, E.A.M., F.X., H.L., H.Z., J.D.W., J. Gillis, L.P., M.C., Q.H., S.F., Z.J.H. and Z.Y. contributed to writing the manuscript.

3. Network, BRAIN Initiative Cell Census. "A multimodal cell census and atlas of the mammalian primary motor cortex." *Nature* 598.7879 (2021): 86. https://doi.org/10.1038/s41586-021-03950-0

Omics data analysis: E.A., T.E.B., T.B., **A.S.B.**, M.C., D.D., S.D., J.R.E., R.F., S.F., O.F., J. Gillis, J. Goldy, Q.H., N.L.J., P.V.K., F.M.K., B.B.L., E.S.L., Y.E.L., S. Linnarsson, H.L., E.Z.M., E.A.M., S.-Y.N., V.N., L.P., O.P., E.P., A.R., D.R., H.R.d.B., K. Siletti, K. Smith, S. Somasundaram, K. Street, V.S., B.T., W.T., E.D.V., K.V.d.B., C.T.J.v.V., J.D.W., F. Xie, Z.Y., H.Z., J.Z. and J.N.

4. Bloom, Joshua S., et al. "Massively scaled-up testing for SARS-CoV-2 RNA via next-generation sequencing of pooled and barcoded nasal and saliva samples." *Nature Biomedical Engineering* 5.7 (2021): 657-665. https://doi.org/10.1038/s41551-021-00754-5

J.S.B. and V.A.A. wrote the manuscript with assistance from C.L., J.F., L.K., E.E., E.M.J., A.R.C., N.B.L., M.G. and S.Kosuri. E.M.J., A.R.C., N.B.L., M.G., S.W.S., J.S.B. and S.Kosuri designed barcodes and performed early testing and analysis of protocols and reagents. C.L., Y.Y., Y.Z., L.G., R.D. and M.J.B. provided early guidance and key automation resources. E.E., D.H., N.L. and C.K. developed the registration webapp and IT infrastructure. L.S., C.M., M.G., E.M.J., N.B.L., S.Kosuri, I.L., O.F.B., V.A.A. and J.S.B. performed and analysed experiments. **A.S.B.** and L.P. analysed misassignment of index barcodes. V.A.A., O.B.G., S.C., E.E.H., G.O. and B.J.C. collected and processed clinical samples. D.M. optimized operational protocols and D.M., S.Kay, M.L., T.L. and E.E. optimized scale up. E.E., L.K., J.F., C.L., Y.Y., Y.Z. and J.B. provided helpful insights into protocols, software, and development and optimization of our specimen collection and handling. F.Y., E.M.T., K.M.K., J.P. and M.H. developed the diversified S standard mixture, N1 primers and flu primers.

5. Booeshaghi, A., et al. "Principles of open source bioinstrumentation applied to the poseidon syringe pump system." *Scientific reports* 9.1 (2019): 1-8. https://doi.org/10.1038/s41598-019-48815-9

J.G. conceived of the project and developed the initial design for the syringe pumps. **A.S.B.** designed the syringe pump system and microscope, and implemented the poseidon software. E.V.B. helped with the design the poseidon system and oversaw hardware printing and design. **A.S.B.** and E.V.B. tested the poseidon system. J.G., **A.S.B.** and E.V.B. formulated the design principles. D.B. developed an initial version of the software. **A.S.B.**, E.V.B., J.G. and L.P. wrote the manuscript.

6. Booeshaghi, A. Sina, et al. "Markedly heterogeneous COVID-19 testing plans among US colleges and universities." *MedRxiv* (2020). https://doi.org/10.1101/2020.08.09.20171223

ASB and LP started compiling the testing database; FT filled in the majority of entries. FT drafted the initial manuscript; **ASB**, FT, ZB and LP wrote the manuscript. **ASB** performed the analysis of the testing database and made the figures. BR analyzed college endowments and their relationship to testing plans. **ASB**, FT, BR and LP contributed references.

7. Booeshaghi, A. Sina, and Lior Pachter. "Decrease in ACE2 mRNA expression in aged mouse lung." *biorxiv* (2020). https://doi.org/10.1101/2020.04.02.021451

A.S.B. analyzed the data and **A.S.B.** and L.P. wrote the manuscript.

8. Booeshaghi, A., et al. "Isoform cell-type specificity in the mouse primary motor cortex." *Nature* 598.7879 (2021): 195-199. https://doi.org/10.1038/s41586-021-03969-3

A.S.B. and L.P. conceived the study. **A.S.B.** implemented the methods and produced the results and figures. **A.S.B.** and L.P. analysed the data and wrote the manuscript. Z.Y., C.v.V., K.S., B.T. and H.Z. produced the SMART-seq and 10xv3 data.

9. Booeshaghi, A. Sina, and Lior Pachter. "Normalization of single-cell RNA-seq counts by log (x+ 1) or log (1+ x)." *Bioinformatics* 37.15 (2021): 2223-2224. https://doi.org/10.1093/bioinformatics/btab085

A.S.B. analyzed the data and **A.S.B.** and L.P. wrote the manuscript.

10. Booeshaghi, A., et al. "Reliable and accurate diagnostics from highly multiplexed sequencing assays." *Scientific reports* 10.1 (2020): 1-7. https://doi.org/10.1038/s41598-020-78942-7

A.S.B. and L.P. developed the kallisto|bustools approach to processing and analyzing HMSA data. **A.S.B.** adapted kallisto and bustools to process SwabSeq, LAMP-seq, covE-seq, and TRB-seq data. **A.S.B.** performed the analyses and collected results for the paper. N.L. developed the bcl2fastq + starcode processing approach with assistance from A.R.C., S.W.S., and J.S.B. J.G. assisted with technical aspects of the SwabSeq assay and in assessing the kallisto|bustools workflow results. L.L. created Fig. 1 and explored the

sample index structures of LAMP-seq, TRB-seq, and SwabSeq. S.K., N.L.B., A.R.C., S.W.S. and J.S.B. developed SwabSeq. **A.S.B.**, L.L., and L.P. wrote the manuscript.

11. Booeshaghi, A. Sina, and Lior Pachter. "Benchmarking of lightweight-mapping based single-cell RNA-seq pre-processing." *bioRxiv* (2021). https://doi.org/10.1101/2021.01.25.428188

A.S.B. analyzed the data and **A.S.B.** and L.P. wrote the manuscript.

12. Booeshaghi, A. Sina, et al. "Low-cost, scalable, and automated fluid sampling for fluidics applications." *HardwareX* 10 (2021): e00201. https://doi.org/10.1016/j.ohx.2021.e00201

ASB, YK, and LP designed the fraction collector. JG helped set instrument specifications. YK assembled and built the fraction collector and performed the experiments. ASB, YK, and KHM designed the GUI. KHM coded the installable GUI and YK, KHM, and ASB coded the web-browser GUI. **ASB** coded the browser-serial package. **ASB**, YK, and KHM wrote the documentation. **ASB** and YK analyzed the data and made figures. **ASB**, YK, and LP wrote the manuscript.

13. Booeshaghi, A. Sina, et al. "Depth normalization for single-cell genomics count data." *bioRxiv* (2022). https://doi.org/10.1101/2022.05.06.490859

A.S.B., and L.P. developed the project idea. A.G.M. pre-processed the datasets. **A.S.B.** performed the analysis. **A.S.B.** and A.G.M. compiled the supplementary material. **A.S.B.** drafted the paper. I.B.H. performed the overdispersion simulation. **A.S.B.**, I.B.H., A.G.M., and L.P. wrote, reviewed, and edited the paper.

Table of Contents

Acknowledgements……..... iii
Abstract …….. iv
Published Content and Contributions……..... v
Table of Contents……... ix
Chapter I: Introduction
Section 1: Overview of single-cell RNA sequencing 1
Section 2: Practical outcomes 6
Chapter II: Hardware
Section 1: poseidon syringe pump system 13
Section 2: colosseum fraction collector 34
Chapter III: Software
Section 1: *kallisto* | *bustools* sequence preprocessing 57
Section 2: count data normalization 129
Chapter IV: Analysis of the Mouse Primary Motor Cortex 133
Chapter V: Conclusion & Outlook 177

Chapter 1

INTRODUCTION

Section 1: Overview of single-cell RNA sequencing

Single-cell RNA sequencing (scRNA-seq) is a collection of technologies for quantifying the amount of RNA molecules inside individual cells using DNA sequencing as a readout. Methods to perform RNA-seq first require cell dissociation from tissue of interest, or cell membrane permeabilization, making RNA from individual cells accessible for molecular barcoding with DNA, Figure 1 (Haque et al. 2017; J. Cao et al. 2017; Rosenberg et al. 2018; Gehring et al. 2020). To ensure that all RNA molecules within a cell are tagged with the same DNA barcode, cells are isolated into individual chambers. Commonly used isolated reaction chambers are water-in-oil droplets, generated by microfluidics (Zilionis et al. 2017), or individual wells inside of microwell plates (Picelli et al. 2014). Within these reaction chambers, RNAs are tagged with DNA barcodes and then sequenced. Alternative methods, that do not isolate individual cells, take advantage of the cell's permeabilized membrane where collections of cells are grouped, split, and tagged with different DNA barcodes by diffusion transport across the cell membrane (Cheng et al. 2021). This multi-round splitting-and-pooling operation allows for greater numbers of cells to be assayed than with standard isolation procedures, but at the cost of procedural complexity. Regardless of method, distinct DNA barcodes must tag RNA molecules inside of a cell.

In order to combinatorially tag RNA molecules with distinct molecular barcodes, synthetic DNA sequences are designed that have three important properties (Tambe and Pachter 2019; Bystrykh 2012). The first is barcode uniqueness. A DNA barcode consists of a sequence of four nucleotide bases (A,T,G,C) that is built into a molecule of length (L) allowing up to 4^L possible unique barcodes. Barcodes vary in length with 16 base pairs being used in the most common scRNA-seq assay, 10x Genomics Chromium (Zheng et al. 2017). A 16 base synthetic DNA sequence allows for 4,294,967,296 possible unique barcodes where multiple copies of one barcode can be used to tag all of the molecules in one cell. In practice, not all distinct barcodes are used or even desirable. Barcode errors or short barcode lengths can result in identifier collisions making it challenging to resolve RNA sequences from different cells which happen to have been tagged with the same barcode (Hashimshony et al. 2016; X. Zhang et al. 2019).

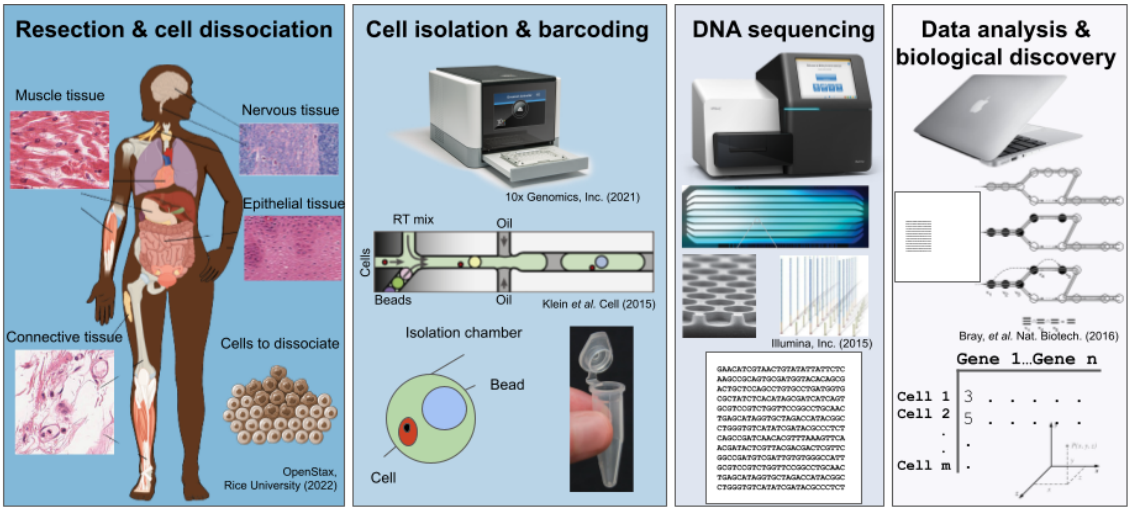


Figure 1: Overview of a single-cell RNA-sequencing. Cells are resected and dissociated prior to cell isolation and RNA barcoding. The resultant library is sequenced. The sequencing reads are aligned to a known reference and quantified to generate a *cell* by *gene* count matrix.

The second property for combinatorially tagging cellular RNA is barcode composition. DNA barcodes can be extended and composed so that different subsequences of the barcode identify different entities such as cells, molecules, or biological samples (Binan, Drobetsky, and Costantino 2019). For example, a 12 base pair barcode is appended to the aforementioned 16 base cell barcode, the former allowing multiple copies of the same RNA to be disambiguated, within one cell, and the latter allowing sets of molecules belonging to one cell to be distinct from another cell (Islam et al. 2014). Further composition is possible and enables greater "multiplexing". Samples, experimental conditions, or even CRISPR perturbations can be tagged and subsequently identified (McGinnis et al. 2019; Stoeckius et al. 2018; Dixit et al. 2016).

The third, and crucial property of synthetic DNA barcodes is the presence of a DNA-RNA binding region. Once cells have been isolated and their RNAs made accessible, either by cell isolation and cell rupture or by membrane permeabilization, RNA molecules anneal to the DNA barcodes at the DNA-RNA binding region (Figure 2). The most widely used DNA-RNA binding region is a long stretch of T bases that bind, complementarily, to the long stretch of A bases in messenger RNA's (mRNA) poly-adenylated (polyA) "tail". The polyA tail is a desirable region to target since it is added to all mRNA molecules after RNA production. Other RNAs with long stretches of polyA sequence, like long non-coding RNAs, are often also captured (Sage et al. 2020) and frequently removed from downstream analysis. Alternative methods for RNA

capture, which focus on only selected a subset of RNA, use a modified DNA-RNA binding region that is specific to a designed gene panel These methods have been developed and used for example in targeted CRISPR screens where only a handful of RNA's are known to be affected by a CRISPR perturbation (Pokhilko et al. 2021; Schraivogel et al. 2020).

A "library" of molecules comprising of RNAs that have been captured and tagged with a composite cell-specific and molecule-specific barcode are then sequenced on a standard DNA
Sequencer to produce a text file called a FASTQ file. Each entry, known as a "read", in the FASTQ file contains sequences of A,T,G, and Cs that correspond to a molecule in the library that was detected by the sequencer. In theory, every RNA molecule in each cell is tagged, captured, and sequenced so that the corresponding textual representation of the combined cellular, molecular, and RNA sequence is present in the FASTQ file. In practice, about 60% cells, and ~15% of unique RNAs from each cell are assayed per experiment with improvements being developed ("What Fraction of mRNA Transcripts Are Captured per Cell?" n.d.; M. Zhang et al. 2020; Yamawaki et al. 2021).

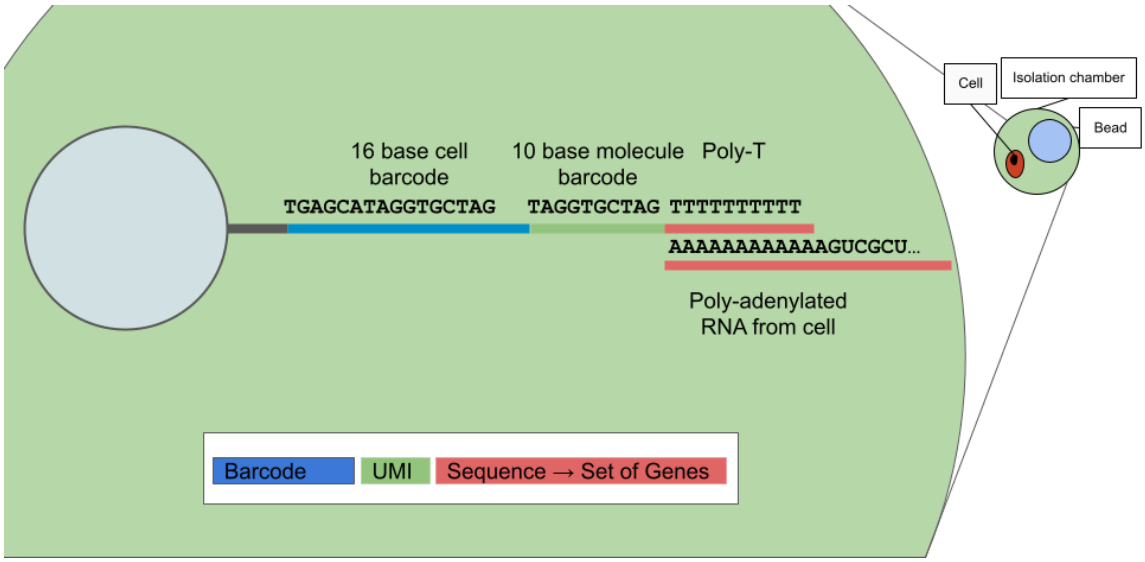


Figure 2: Cell encapsulation and RNA barcoding. Cells are encapsulated into isolation chambers and their poly-adenylated RNA are tagged with synthetic DNA barcodes that are unique to the cell and molecule.

Sequencing reads are then transformed into a matrix of sequence counts. Computing these *cell by gene* matrices requires counting the number of RNA's in the FASTQ file that

arise from different genes, per cell. This happens in three steps. The first is aligning the text sequences corresponding to RNA molecules to a known reference. Substantial effort has gone into curating references of known sequences and their corresponding genes, for hundreds of species (Cunningham et al. 2022). These references provide annotations for genomic features, such as genes or isoforms, on top of sequence information. For example the *BRCA1* gene which aids in DNA repair, is located on Chromosome 17 position 43,044,295 to 43,170,245 on the reverse strand ("BRCA Gene Mutations: Cancer Risk and Genetic Testing Fact Sheet" 2020, "Gene: BRCA1 (ENSG00000012048) - Summary - Homo_sapiens - Ensembl Genome Browser 106" n.d.).

These references are known as genomic references and contain information about all genes including introns, exons, and other genomic elements. Often a subset of this genomic reference, known as a transcript reference, is used that comprises transcript sequences many of which result in proteins. Common computational approaches for processing FASTQ files use references to find appropriate alignments for all reads. Methods like STAR, for example, use the entire genomic reference, and transcript splice junctions, whereas methods like kallisto (Bray et al. 2016) use the transcriptome reference (Dobin et al. 2013; Bray et al. 2016). The choice about which reference to use must take into consideration the experimental approach that generated the sequencing data. Most common scRNAseq assays capture cytoplasmic mRNA, or mRNA without introns, making the transcriptome a sufficient choice for read alignment. Newer assays capture nuclear RNA, or RNA containing introns, making it necessary to use a reference containing introns for read alignment (Lake et al. 2019).

After aligning reads to genes, the second step is using the cell barcode to disambiguate cells from each other. Cell barcodes can have sequence errors that are the result of improper base calling on the sequencer (Stoler and Nekrutenko 2021). Proper barcode grouping requires first correcting those errors. Some single-cell technologies such as 10x Genomics have a list of expected barcodes, known as a whitelist, which can be used to correct errors. Once barcodes have been corrected, subsequent molecular-barcode counting can then be performed on a cell-by-cell basis.

The third step is using the molecular barcode to disambiguate copies of RNA molecules within a cell, from each other. Since RNA content captured from a cell is small, PCR amplification is used to increase the copy number of each molecule. In order to distinguish between molecules that have been synthetically duplicated by PCR vs molecules that occur in high copy number in the cell, molecular barcodes known as "Unique molecular identifiers" (UMIs) are appended to the cellular barcode. Practically

speaking, UMIs are random sequences of synthetic DNA (Islam et al. 2014). Not all single-cell assays use molecular identifiers. For example, SMART-seq isolates cells into individual wells and adds only a cell barcode to the well. Since individual RNA's cannot be disambiguated with a molecular barcode to get an absolute count of molecules, statistical methods like the expectation-maximization algorithm are required to produce read counts.

The result is a matrix of positive integer counts where each row corresponds to a cell and each column corresponds to a gene. The value in the *cell* by *gene* entry corresponds to the number of unique molecular barcodes, that tagged molecules mapping to that gene, in the cell corresponding to that cell barcode. Often in scRNAseq multiple matrices are generated for different biological samples, conditions, and treatments. Statistical analysis and machine learning techniques are used to cluster cells that share similar gene expression patterns and determine sets of genes that differ between samples, conditions, and treatments. Once computed, cell type gene expression signatures can be associated with physical cells in order to map the gene expression to the cell type of origin. This map provides biologists with additional cell-type specific data that can be used to identify gene targets in malignant cells (Yeo et al. 2022) and identify cell type expression changes as a possible diagnostic tool (Ramírez-Sánchez et al. 2022). These techniques are ultimately useful for identifying shifts in cell type abundance and changes in gene expression that are biologically relevant or clinically actionable.

To enable statistical analysis of this matrix, such as the identification of cells that exhibit similar gene expression, matrix filtering and normalization must be performed. Normalization procedures aim to transform data in a way that obeys the assumptions of downstream analysis. Commonly used methods apply variance stabilizing transformations in an attempt to remove the gene mean-variance relationship, a relationship which is assumed technical (Ahlmann-Eltze and Huber 2021). Other methods attempt to normalize cell depth and stabilize gene stabilization (Choudhary and Satija 2022).

After normalization, unsupervised clustering schemes are performed in order to identify sets of cells that exhibit similar gene expression. Techniques like Leiden and Louvain identify communities from a K-Nearest Neighbor graph that is built from the *cell* by *cell* distance matrix (Blondel et al. 2008; Traag, Waltman, and van Eck 2019). Prior literature and expert knowledge is then required to map clusters derived from unsupervised clustering to known cell types that have been identified with orthogonal observations such as morphological, histological, or chemical methods (Y. Cao, Wang, and Peng 2020). Mapping cells to cell types requires performing statistical tests such as a t-test or

Wilcoxon rank-sum on genes between a cluster and its complement. These differential expression techniques are often employed on the normalized count matrix to find statistically significant upregulated genes that correspond to previously identified marker genes. In more complex experiments that perform scRNA-seq on multiple conditions, statistical tests can identify genes that change between conditions.

The challenges posed by scRNAseq span multiple disciplines of science and engineering. A background in mechanical engineering, fluidic dynamics, design, manufacturing, electrical engineering and controls is useful to tackle the challenges of cell encapsulation and sampling. Statistics, computation and machine learning help to analyze the large volumes of data generated by the complex assay. Ultimately it is the confluence of multiple disciplines and techniques that enable clinically relevant biological insights to be derived from massive amounts of single cells.

Section 2: Practical outcomes of single-cell RNA sequencing

A large number of single-cell protocols and analysis methods have been recently published enabling single-cell medical diagnosis and interventions (Gawel et al. 2019). Though with ever increasing amounts of data and cells to assay, new challenges arise. First, experiments are costly. A standard scRNA-seq experiment costs about $1.70 per cell with a 1.1% multiplet rate, making a standard scRNAseq experiment of 10k cells cost $17,000 (“Cost Per Cell” n.d.). The cost is primarily driven by reagent costs and costs to prepare a sequencing library followed by the hardware needed to perform cell isolation. Additionally, processes for cell isolation and library preparation are time consuming (“Pricing Overview” n.d.) and closed-source. Second, data preprocessing and analysis can be time consuming and resource intensive. For example, a standard experiment and analysis can generate terabytes of data and naïve preprocessing tools for read alignment then require days of compute time and large amounts of memory (Melsted et al. 2021). These computational challenges place additional constraints on the scale of analysis matching the scale of data production. Third, subsequent data processing and statistical methods like PCA, clustering, and differential expression often are restricted to gene-level analysis. Expression count data, where isoform counts are aggregated to the gene level, can miss isoform differences between cell types (Booeshaghi et al. 2021).

Ultimately, single-cell RNA-sequencing experiments present numerous engineering challenges. Hardware for performing scRNAseq is often closed-source, costly, and time-consuming. Additionally, the lack of transparency and modularity makes diagnosing data-quality issues associated with novel experimentation difficult. These challenges extend to software to analyze scRNAseq data. Inefficient and poorly characterized methods make evaluating data quality and generating biological insight difficult. The

interplay between methods and hardware requires a short feedback loop whereby experimental data generated with scRNAseq hardware can be analyzed with fast algorithms to understand the biological problem of interest but also to better understand and improve the device. The resultant data from the improved device can then be used to better understand and improve the methods.

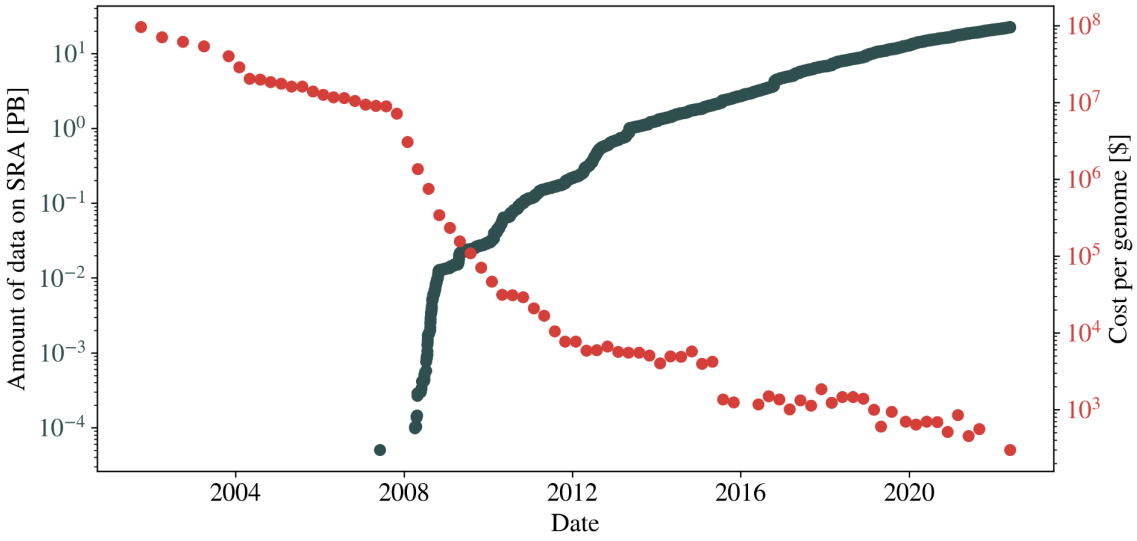


Figure 3: Cost and generation of sequencing data.

To overcome the experimental limitations imposed by cost, I developed two open source devices. The first device, described in Chapter 2.1, is a 3D-printed syringe pump system, that replaces current commercial syringe pumps, to generate single-cell isolation chambers (Booeshaghi et al. 2019). The device, named *poseidon*, can be built for under $100 and assembled in under an hour. *poseidon* can be used with open-source single-cell microfluidic chips to produce beads for RNA capture and can also be used to encapsulate single cells in droplets (Macosko et al. 2015). The second device, described in 2.2, is a 3D-printed fraction collector that enables long-time course single-cell experimentation. The device, named *colosseum*, can be built for $67 and assembled in under an hour and can be used to collect single-cell fractions from microfluidic chips without user intervention. Taken together, these devices enable scalable scRNAseq data generation.

Fast and memory-efficient computational tools are necessary to parse the increasing amounts of data (Figure 3) generated by scRNAseq. Chapter 2 describes algorithms and computational tools, as well as statistical methods to decrease the time and memory to perform scRNAseq analysis. These preprocessing algorithms transform sequencing reads into a count matrix and are implemented in a command line utility called *bustools* and are

used in conjunction with another tool called *kallisto* to form the *kallisto* | *bustools* workflow for preprocessing scRNAseq data (Melsted et al. 2021). Subsequent processing of the count matrix is required to effectively normalize the data for further statistical analysis (Booeshaghi and Pachter 2021). These preprocessing steps are fundamental for the application of statistical models to understand changes in cell-type composition and gene expression.

Chapter three discusses applications of these tools and extensions that enable the study of the mouse primary motor cortex (MOp) at isoform resolution. I use previously developed methods to analyze 286,487 cells from the MOp and I propose and validate a computational framework for the co-analysis of multiple different RNA sampling strategies and apply it to develop the first ever spatially-resolved isoform atlas of the mouse MOp.

Taken together, my thesis addresses fundamental engineering challenges for the generation, processing, and analysis of single-cell RNA sequence data that must be met in order to realize the goal of bringing single-cell RNA sequencing to the clinic.

References

Ahlmann-Eltze, Constantin, and Wolfgang Huber. 2021. “Transformation and Preprocessing of Single-Cell RNA-Seq Data.” *bioRxiv*. https://doi.org/10.1101/2021.06.24.449781.

Binan, Loïc, Elliot A. Drobetsky, and Santiago Costantino. 2019. “Exploiting Molecular Barcodes in High-Throughput Cellular Assays.” *SLAS Technology* 24 (3): 298–307.

Blondel, Vincent D., Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. “Fast Unfolding of Communities in Large Networks.” *Journal of Statistical Mechanics* 2008 (10): P10008.

Booeshaghi, A. Sina, Eduardo da Veiga Beltrame, Dylan Bannon, Jase Gehring, and Lior Pachter. 2019. “Principles of Open Source Bioinstrumentation Applied to the Poseidon Syringe Pump System.” *Scientific Reports* 9 (1): 12385.

Booeshaghi, A. Sina, and Lior Pachter. 2021. “Normalization of Single-Cell RNA-Seq Counts by Log(x + 1)* or log(1 + X).” *Bioinformatics* , March. https://doi.org/10.1093/bioinformatics/btab085.

Booeshaghi, A. Sina, Zizhen Yao, Cindy van Velthoven, Kimberly Smith, Bosiljka Tasic, Hongkui Zeng, and Lior Pachter. 2021. “Isoform Cell-Type Specificity in the Mouse Primary Motor Cortex.” *Nature* 598 (7879): 195–99.

Bray, Nicolas L., Harold Pimentel, Páll Melsted, and Lior Pachter. 2016. “Near-Optimal Probabilistic RNA-Seq Quantification.” *Nature Biotechnology* 34 (5): 525–27.

“BRCA Gene Mutations: Cancer Risk and Genetic Testing Fact Sheet.” 2020. National Cancer Institute. November 25, 2020. https://www.cancer.gov/about-cancer/causes-prevention/genetics/brca-fact-sheet.

Bystrykh, Leonid V. 2012. “Generalized DNA Barcode Design Based on Hamming Codes.” *PloS One* 7 (5): e36852.

Cao, Junyue, Jonathan S. Packer, Vijay Ramani, Darren A. Cusanovich, Chau Huynh, Riza Daza, Xiaojie Qiu, et al. 2017. “Comprehensive Single-Cell Transcriptional Profiling of a Multicellular Organism.” *Science* 357 (6352): 661–67.

Cao, Yinghao, Xiaoyue Wang, and Gongxin Peng. 2020. “SCSA: A Cell Type Annotation Tool for Single-Cell RNA-Seq Data.” *Frontiers in Genetics* 11 (May): 490.

Cheng, Junyun, Jie Liao, Xin Shao, Xiaoyan Lu, and Xiaohui Fan. 2021. “Multiplexing Methods for Simultaneous Large-Scale Transcriptomic Profiling of Samples at Single-Cell Resolution.” *Advancement of Science* 8 (17): e2101229.

Choudhary, Saket, and Rahul Satija. 2022. “Comparison and Evaluation of Statistical Error Models for scRNA-Seq.” *Genome Biology* 23 (1): 27.

“Cost Per Cell.” n.d. Accessed May 13, 2022. https://satijalab.org/costpercell/.

Cunningham, Fiona, James E. Allen, Jamie Allen, Jorge Alvarez-Jarreta, M. Ridwan Amode, Irina M. Armean, Olanrewaju Austine-Orimoloye, et al. 2022. “Ensembl 2022.” *Nucleic Acids Research* 50 (D1): D988–95.

Dixit, Atray, Oren Parnas, Biyu Li, Jenny Chen, Charles P. Fulco, Livnat Jerby-Arnon, Nemanja D. Marjanovic, et al. 2016. “Perturb-Seq: Dissecting Molecular Circuits with Scalable Single-Cell RNA Profiling of Pooled Genetic Screens.” *Cell* 167 (7): 1853–66.e17.

Dobin, Alexander, Carrie A. Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski,

Sonali Jha, Philippe Batut, Mark Chaisson, and Thomas R. Gingeras. 2013. “STAR: Ultrafast Universal RNA-Seq Aligner.” *Bioinformatics* 29 (1): 15–21.

Gawel, Danuta R., Jordi Serra-Musach, Sandra Lilja, Jesper Aagesen, Alex Arenas, Bengt Asking, Malin Bengnér, et al. 2019. “A Validated Single-Cell-Based Strategy to Identify Diagnostic and Therapeutic Targets in Complex Diseases.” *Genome Medicine* 11 (1): 47.

Gehring, Jase, Jong Hwee Park, Sisi Chen, Matthew Thomson, and Lior Pachter. 2020. “Highly Multiplexed Single-Cell RNA-Seq by DNA Oligonucleotide Tagging of Cellular Proteins.” *Nature Biotechnology* 38 (1): 35–38.

“Gene: BRCA1 (ENSG00000012048) - Summary - Homo_sapiens - Ensembl Genome Browser 106.” n.d. Accessed May 13, 2022. https://useast.ensembl.org/Homo_sapiens/Gene/Summary?g=ENSG00000012048;r=17:43044295-43170245.

Haque, Ashraful, Jessica Engel, Sarah A. Teichmann, and Tapio Lönnberg. 2017. “A Practical Guide to Single-Cell RNA-Sequencing for Biomedical Research and Clinical Applications.” *Genome Medicine* 9 (1): 75.

Hashimshony, Tamar, Naftalie Senderovich, Gal Avital, Agnes Klochendler, Yaron de Leeuw, Leon Anavy, Dave Gennert, et al. 2016. “CEL-Seq2: Sensitive Highly-Multiplexed Single-Cell RNA-Seq.” *Genome Biology* 17 (April): 77.

Islam, Saiful, Amit Zeisel, Simon Joost, Gioele La Manno, Pawel Zajac, Maria Kasper, Peter Lönnerberg, and Sten Linnarsson. 2014. “Quantitative Single-Cell RNA-Seq with Unique Molecular Identifiers.” *Nature Methods* 11 (2): 163–66.

Lake, Blue B., Song Chen, Masato Hoshi, Nongluk Plongthongkum, Diane Salamon, Amanda Knoten, Anitha Vijayan, et al. 2019. “A Single-Nucleus RNA-Sequencing Pipeline to Decipher the Molecular Anatomy and Pathophysiology of Human Kidneys.” *Nature Communications* 10 (1): 2832.

Macosko, Evan Z., Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, et al. 2015. “Highly Parallel Genome-Wide Expression Profiling of Individual Cells Using Nanoliter Droplets.” *Cell* 161 (5): 1202–14.

McGinnis, Christopher S., David M. Patterson, Juliane Winkler, Daniel N. Conrad, Marco Y. Hein, Vasudha Srivastava, Jennifer L. Hu, et al. 2019. “MULTI-Seq: Sample Multiplexing for Single-Cell RNA Sequencing Using Lipid-Tagged Indices.” *Nature Methods* 16 (7): 619–26.

Melsted, Páll, A. Sina Booeshaghi, Lauren Liu, Fan Gao, Lambda Lu, Kyung Hoi Joseph Min, Eduardo da Veiga Beltrame, Kristján Eldjárn Hjörleifsson, Jase Gehring, and Lior Pachter. 2021. “Modular, Efficient and Constant-Memory Single-Cell RNA-Seq Preprocessing.” *Nature Biotechnology* 39 (7): 813–18.

Picelli, Simone, Omid R. Faridani, Asa K. Björklund, Gösta Winberg, Sven Sagasser, and Rickard Sandberg. 2014. “Full-Length RNA-Seq from Single Cells Using Smart-seq2.” *Nature Protocols* 9 (1): 171–81.

Pokhilko, Alexandra, Adam E. Handel, Fabiola Curion, Viola Volpato, Emma S. Whiteley, Sunniva Bøstrand, Sarah E. Newey, et al. 2021. “Targeted Single-Cell RNA Sequencing of Transcription Factors Enhances the Identification of Cell Types and Trajectories.” *Genome Research* 31 (6): 1069–81.

"Pricing Overview." n.d. Accessed May 28, 2022. http://www.yerkes.emory.edu/nhp_genomics_core/rates/index.html.

Ramírez-Sánchez, Aarón D., Xiaojing Chu, Rutger Modderman, Yvonne Kooy-Winkelaar, Sibylle Koletzko, Ilma R. Korponay-Szabó, Riccardo Troncone, et al. 2022. "Single-Cell RNA Sequencing of Peripheral Blood Mononuclear Cells From Pediatric Coeliac Disease Patients Suggests Potential Pre-Seroconversion Markers." *Frontiers in Immunology* 13 (March): 843086.

Rosenberg, Alexander B., Charles M. Roco, Richard A. Muscat, Anna Kuchina, Paul Sample, Zizhen Yao, Lucas T. Graybuck, et al. 2018. "Single-Cell Profiling of the Developing Mouse Brain and Spinal Cord with Split-Pool Barcoding." *Science* 360 (6385): 176–82.

Sage, Adam P., Kevin W. Ng, Erin A. Marshall, Greg L. Stewart, Brenda C. Minatel, Katey S. S. Enfield, Spencer D. Martin, Carolyn J. Brown, Ninan Abraham, and Wan L. Lam. 2020. "Assessment of Long Non-Coding RNA Expression Reveals Novel Mediators of the Lung Tumour Immune Response." *Scientific Reports* 10 (1): 16945.

Schraivogel, Daniel, Andreas R. Gschwind, Jennifer H. Milbank, Daniel R. Leonce, Petra Jakob, Lukas Mathur, Jan O. Korbel, Christoph A. Merten, Lars Velten, and Lars M. Steinmetz. 2020. "Targeted Perturb-Seq Enables Genome-Scale Genetic Screens in Single Cells." *Nature Methods* 17 (6): 629–35.

Stoeckius, Marlon, Shiwei Zheng, Brian Houck-Loomis, Stephanie Hao, Bertrand Z. Yeung, William M. Mauck 3rd, Peter Smibert, and Rahul Satija. 2018. "Cell Hashing with Barcoded Antibodies Enables Multiplexing and Doublet Detection for Single Cell Genomics." *Genome Biology* 19 (1): 224.

Stoler, Nicholas, and Anton Nekrutenko. 2021. "Sequencing Error Profiles of Illumina Sequencing Instruments." *NAR Genomics and Bioinformatics* 3 (1): lqab019.

Tambe, Akshay, and Lior Pachter. 2019. "Barcode Identification for Single Cell Genomics." *BMC Bioinformatics* 20 (1): 32.

Traag, V. A., L. Waltman, and N. J. van Eck. 2019. "From Louvain to Leiden: Guaranteeing Well-Connected Communities." *Scientific Reports* 9 (1): 5233.

"What Fraction of mRNA Transcripts Are Captured per Cell?" n.d. 10X Genomics. Accessed May 12, 2022. https://kb.10xgenomics.com/hc/en-us/articles/360001539051-What-fraction-of-mRNA-transcripts-are-captured-per-cell-.

Yamawaki, Tracy M., Daniel R. Lu, Daniel C. Ellwanger, Dev Bhatt, Paolo Manzanillo, Vanessa Arias, Hong Zhou, et al. 2021. "Systematic Comparison of High-Throughput Single-Cell RNA-Seq Methods for Immune Cell Profiling." *BMC Genomics* 22 (1): 66.

Yeo, Alan T., Shruti Rawal, Bethany Delcuze, Anthos Christofides, Agata Atayde, Laura Strauss, Leonora Balaj, et al. 2022. "Single-Cell RNA Sequencing Reveals Evolution of Immune Landscape during Glioblastoma Progression." *Nature Immunology*, May, 1–14.

Zhang, Mingxia, Yuan Zou, Xing Xu, Xuebing Zhang, Mingxuan Gao, Jia Song, Peifeng Huang, et al. 2020. "Highly Parallel and Efficient Single Cell mRNA Sequencing with Paired Picoliter Chambers." *Nature Communications* 11 (1): 2118.

Zhang, Xiannian, Tianqi Li, Feng Liu, Yaqi Chen, Jiacheng Yao, Zeyao Li, Yanyi Huang, and Jianbin Wang. 2019. “Comparative Analysis of Droplet-Based Ultra-High-Throughput Single-Cell RNA-Seq Systems.” *Molecular Cell* 73 (1): 130–42.e5.

Zheng, Grace X. Y., Jessica M. Terry, Phillip Belgrader, Paul Ryvkin, Zachary W. Bent, Ryan Wilson, Solongo B. Ziraldo, et al. 2017. “Massively Parallel Digital Transcriptional Profiling of Single Cells.” *Nature Communications* 8 (January): 14049.

Zilionis, Rapolas, Juozas Nainys, Adrian Veres, Virginia Savova, David Zemmour, Allon M. Klein, and Linas Mazutis. 2017. “Single-Cell Barcoding and Sequencing Using Droplet Microfluidics.” *Nature Protocols* 12 (1): 44–73.

Chapter 2

HARDWARE

Section 1: Principles of open source bioinstrumentation applied to the poseidon syringe pump system

Preamble

Current open-source single-cell RNA-sequencing experiments are limited by the cost of reagents and hardware. The *poseidon* syringe pump system is a novel device for performing large-scale microfluidics experiments that overcomes these cost limitations. It uses a two-component 3D printed plastic body and stepper motor to drive a variety of syringe sizes to pump liquid into microfluidic devices. By employing off-the-shelf components, *poseidon* is substantially cheaper than commercial alternatives at the same level of flow precision. This device overcomes a roadblock in a critical technology for performing large-scale scRNAseq experiments. The low cost enables many pumps to be deployed to perform multiple cell encapsulation procedures in parallel. Additionally, *poseidon* enables further study of microfluidic geometries and material properties to overcome the stochastic nature of cell encapsulation– a task that is challenging with current closed-source devices.

Summary

The poseidon syringe pump and microscope system is an open source alternative to commercial systems. It costs less than $400 and can be assembled in under an hour using the instructions and source files available at https://pachterlab.github.io/poseidon. We describe the poseidon system and use it to illustrate design principles that can facilitate the adoption and development of open source bioinstruments. The principles are functionality, robustness, safety, simplicity, modularity, benchmarking, and documentation.

Introduction

Open source hardware projects (Gibb, n.d.) have become increasingly popular in recent years due in part to the rapid evolution of desktop 3D printers and an ecosystem of open source electronic boards like the Arduino and Raspberry Pi systems (Arduino Project 2018; Raspberry Pi Foundation 2018). These developments have spurred growing interest in laboratory instrument open source projects (Pearce 2012; Wijnen et al. 2014; André Maia Chagas 2018) including syringe pumps (Wijnen et al. 2014; UNC Greensboro n.d.),

microscopes (Andre Maia Chagas et al. 2017), fluorescence imaging devices (Nuñez et al. 2017), micro-dispensers (Forman et al. 2017) and single-cell transcriptomics technologies (Stephenson et al. 2018). While cost savings can be an important reason for development of open source hardware (Dolgin 2018), the ability to customize designs for specific applications gives open source projects a unique advantage over commercial solutions. In addition, expanding libraries of designs, software, and commonly used off-the-shelf parts can be shared and adapted across projects, meaning developers are never starting from scratch, even when designing a new instrument. For example, the RepRap project 3D printers borrowed heavily from standard software and hardware Computer Numeric Control (CNC) tools used in machining. As open source designs, electronics boards, software, and parts for 3D printers were continually published and improved, cheap and interchangeable open source hardware and software intended for 3D printing began to be repurposed for new bioinstruments such as liquid handlers (Opentrons Inc 2018), vial handlers and food dispensers (Wayland and Landgraf 2018), autosamplers (Carvalho and Murray 2018; Carvalho, Sanders, and Holloway 2018), and bioprinters (Banović and Vihar 2018; Fitzsimmons et al. 2018).

Our laboratory has a general interest in developing new methods for high-throughput single-cell applications such as Drop-seq (Macosko et al. 2015) and inDrops (Zilionis et al. 2017) which rely on precise flow rate control to operate microfluidic devices. The unpredictable landscape of single-cell genomics technology puts a high priority on flexible hardware and software that can be adapted and re-purposed as experiments evolve. The inflexible software interface and functionality offered by commercial systems, and the array of do-it-yourself electronics and instrumentation projects powered by open source hardware, inspired us to develop our own open source multi-syringe pump array and microscope system for low cost microfluidics experiments. The resulting system, which we call poseidon, is based on published open source syringe pumps (Wijnen et al. 2014) and microscope microfluidics stations (Stephenson et al. 2018) but introduces a number of innovations and adapts common 3D printer hardware and software to control the system. Requiring only off-the-shelf components and 3D printed parts, the entire poseidon system including microscope and three syringe pumps can be assembled in less than an hour for less than $400. The poseidon syringe pump array and microscope system is an open source alternative to commercial systems (Fig. 1). The pumps and microscope can be used together for microfluidics experiments, or the pumps can be connected to a computer and used independently. For scientists with tight budgets, the microscope system, which is stand-alone, is an effective solution for basic light microscopy.

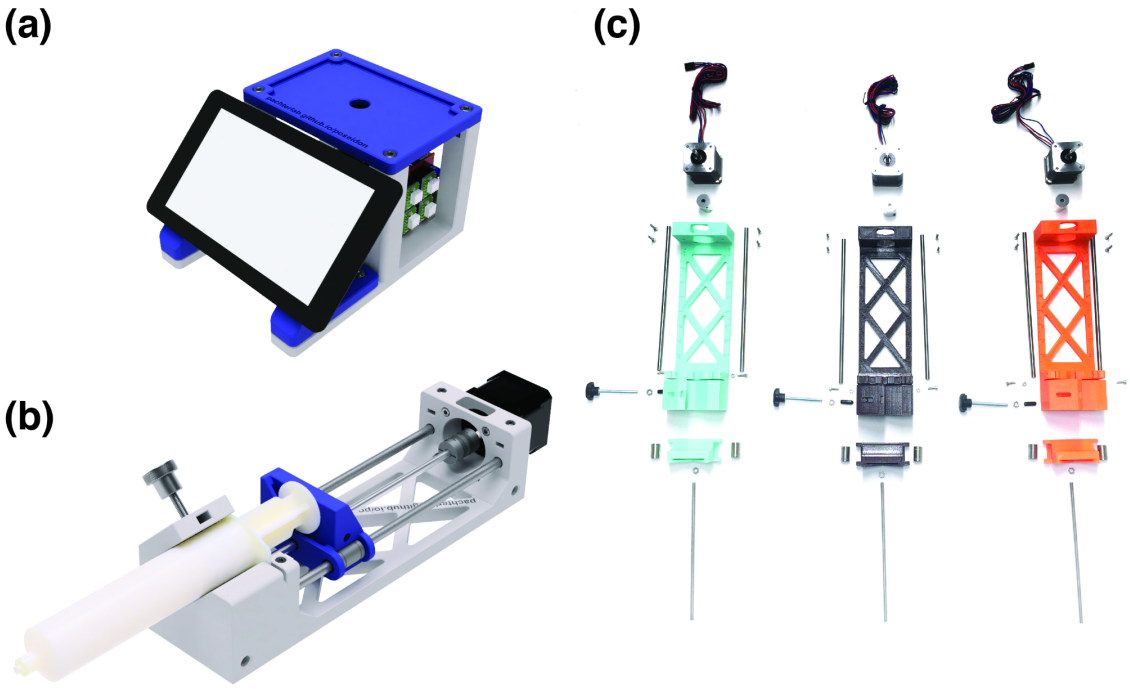


Figure 1: Model of the poseidon system. (a) CAD rendering of the microscope station and (b) a single syringe pump loaded with a 60 mL syringe. (c) Exploded view of all the components needed for assembling three pumps.

The poseidon system uses a Raspberry Pi and touchscreen for the microscope and an Arduino board with a CNC shield to operate up to four pumps simultaneously. Each pump has a stepper motor that drives a lead screw, which in turn moves a sled (mounted on linear bearings) that pushes (infuses) or pulls (aspirates) the syringe plunger. The microscope camera and Arduino use USB connections to connect to the Raspberry Pi or desktop computer (Fig. 2). The system was developed using readily available tools: Autodesk Fusion 360 for CAD, Python 3 and PyQT for software, 3D printers for fabricating custom hardware pieces, and off the shelf electronics and hardware parts. (Fig. 3).

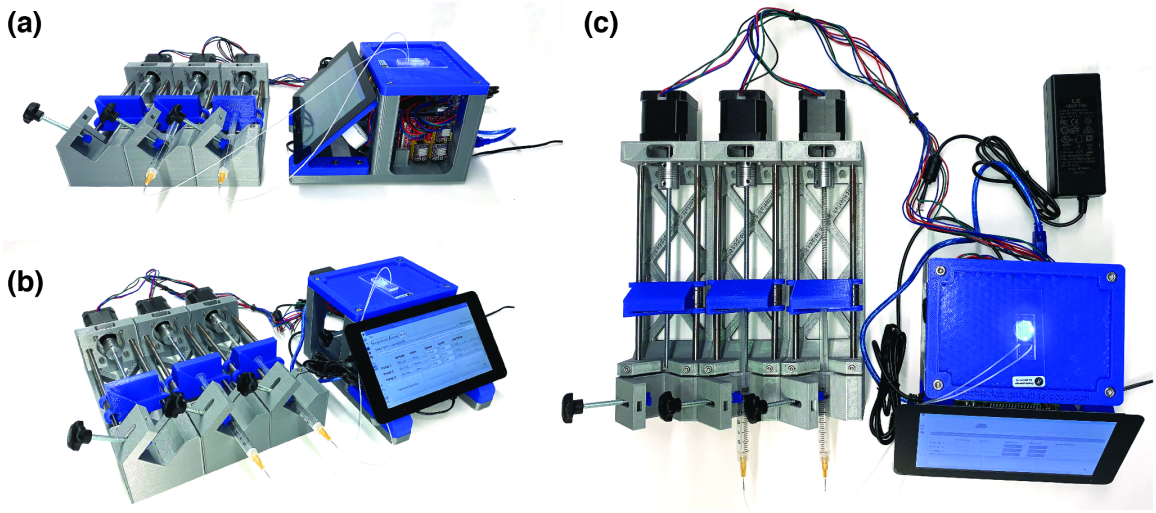


Figure 2: Using the poseidon system. Configuration of the poseidon system for running an emulsion generation microfluidics experiment where only two pumps are used. (a) Side view (b) angled view (c) top view.

The poseidon system repository is available under a BSD 2-clause license at https://github.com/pachterlab/poseidon. For reproducibility and ease of adoption we included direct links to the specific parts used for poseidon, in the GitHub repository. The following components are available:

1. 3D models and Computer Aided Design (CAD) files of the 3D printed components.
2. Pump controller and Graphical User Interface (GUI) software to control the Arduino.
3. Arduino firmware to relay commands via USB to drive the motors.

As we invested more time into poseidon, we realized that the impact of many open source bioinstruments is limited by unintentionally restrictive design decisions and inadequate documentation that discourages adoption by others. This is perhaps unsurprising as most projects are conceived and realized by non-expert developers who are themselves end users. It is with this community in mind that we present a set of guiding design principles specifically tailored to open source bioinstruments. The principles are a synthesis of our experiences designing the poseidon system from the ground up with ease of adoption as our goal. It is our intention that future developers can apply these principles from inception through testing to produce more robust, flexible systems that are more likely to

be adopted, modified, and improved by the broader community. Here we detail these design principles using the poseidon system as an illustrated example.

Results

Design principles

We strove to produce a bioinstrument that could be readily implemented and modified by others: users and designers who could improve and expand on the system. We considered that bioinstrument users generally fall into two categories: i) those who want to adopt a design and use it in a straightforward manner, and ii) those who want to tweak, improve, and adapt designs to their needs, utilizing the instrument for new use cases. While cost is one motivation for developing and using open source instruments, low cost alone cannot drive the adoption of a project for these two groups. A successful open source instrument appeals to the needs of basic and advanced users by adhering to a set of clear design principles: functionality, robustness, safety, simplicity, modularity, benchmarking, and documentation. Adhering to these principles from the beginning of the design-build-test cycle will result in improved bioinstruments ready for further development and use by others.

Functionality: Developing for an application

In engineering, a functional requirement defines a specific metric that a hardware or software system must achieve. The idea of being “good enough” attempts to capture the many design decisions that can be made during the design-build-test cycle as developers consider the tradeoffs between utility, precision, accuracy, speed, cost, and complexity that are acceptable given the application. The poseidon system needed to achieve the following functional requirements for use in microfluidic applications:

1. The syringe pumps needed to be precise enough to make monodisperse emulsions on droplet generation microfluidics chips, with flow rates on the order of 1 mL/hr.
2. The microscope needed to have sufficient magnification to examine the emulsions and view the microfluidic device during operation.
3. The hardware and control software needed to be able to run at least three pumps independently.
4. The software interface needed to be simple and allow users to easily change flow rates, select syringe type or diameter, and perform gradient pumping.
5. The software needed to operate the microscope.

These were the minimum requirements that were specified before we began developing poseidon. A similar list of specific requirements is a necessary starting point for any

bioinstrumentation project. After designing hardware that should be able to meet these objectives, we ensured the pumps operated reliably with flow rates ranging from a few hundred microliters per hour up to several hundred milliliters per minute and we selected an inexpensive USB microscope that reliably imaged our microfluidic device. The 0.3 MP microscope uses a CMOS sensor and has eight dimmable LEDs; the microscope can be readily swapped with any USB compatible microscope. Representative images are in S2.

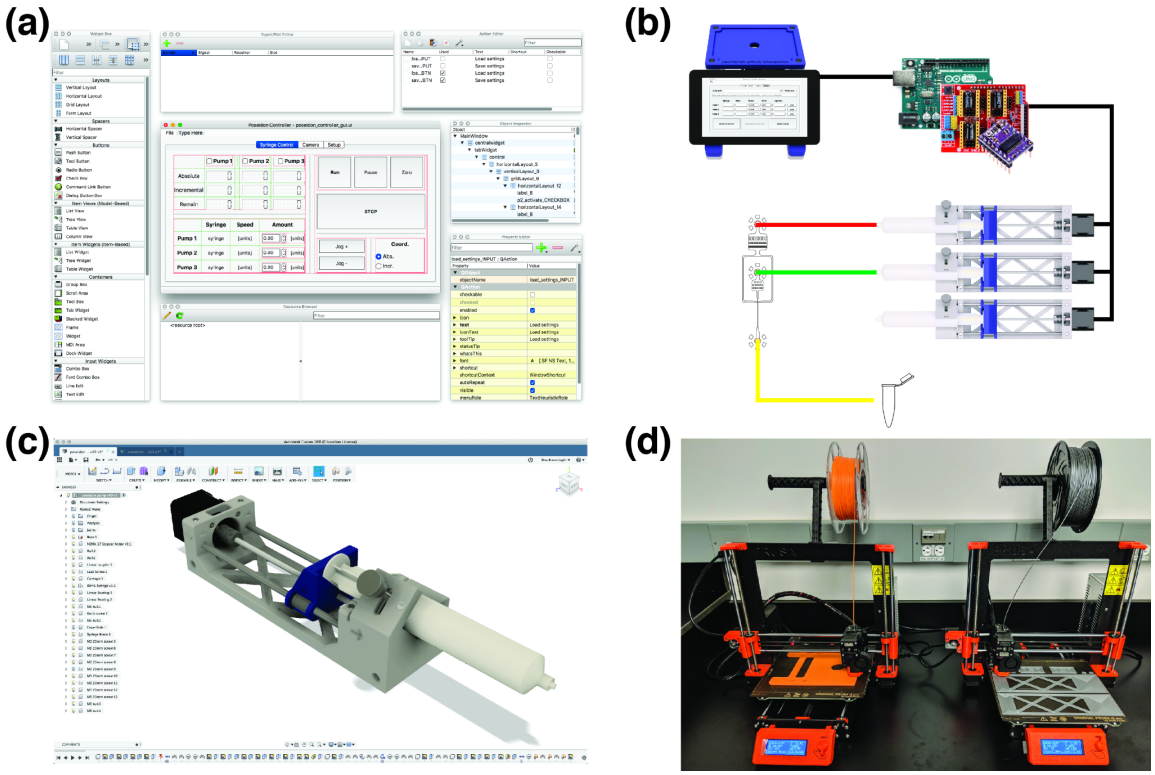


Figure 3: Overview of the tools used for developing the poseidon system. (a) The GUI was created using Qt Designer (The Qt Company n.d.), an open source drag and drop application for organizing buttons that allows users to easily change the GUI interface when adding new functionalities. (b) The GUI interfaces with a Python script that controls both the microscope and Arduino via USB. The Arduino controls the stepper motors on each pump using the CNC shield and stepper motor drivers. (c) The system's 3D printed components were designed using Fusion 360 (Autodesk n.d.), a cloud enabled CAD software that streamlines collaboration and offers free licenses for academics, hobbyists and small businesses. To modify the 3D models users can either work with Fusion 360 or any other CAD software. (d) The 3D printed components can be fabricated on any fused filament fabrication (FFF) 3D printer.

Robustness: Designing with variation in mind

Robustness encompasses not only mitigating the possibility of failure during operation but also ensuring a construction process that tolerates variability in the components. This is particularly important in biology applications where instruments must frequently work in varying physical conditions and with variable input. Ensuring robustness took considerable time, demanding attention to small details and repeated testing. For example, much open source hardware relies on 3D printed components that can introduce variability when printed on different printers. Mechanical tolerance was built into the 3D printed parts over the course of many design-build-test cycles, for example by modifying the print settings to allow for a press fit of the syringe into the pump. During testing, we discovered an unforeseen hardware issue: when there was too much sliding resistance on the carriage, the linear rods displaced and the printed plastic body bent. To stop the bending, we designed a reinforced body and secured the linear rods with set screws. This level of refinement is to be expected for any bioinstrument, and potential developers should be prepared for several design cycles to create an adoptable device.

On the software side, robustness demands testing to minimize user operation error and to ensure correct functionality. The software must be installed and tested on multiple operating systems to verify operation is as expected. In parallel, internet-capable devices such as the Raspberry Pi should be appropriately set up to avoid internet-based attacks. Once the poseidon pumps were being used for experiments in our lab and others, usability issues became apparent. For example, one version of the software configured the stepper motors to use a different microstepping than the hardware had configured, an error which the users encountered during their experiments by observing incorrect flow rates. Using the software during an experiment also revealed small usability issues that had to be corrected, such as using a drop-down menu for choosing the flow direction instead of using a + or -' sign in the displacement amount box. These improvements are relatively minor on their own, but we believe the sum total of such small modifications has an outsized impact on potential adopters testing out an unfamiliar system for the first time.

Safety: Communicating hazards to users

Safety is critically important for robust device operation and must be carefully considered in a laboratory context. When designing an open source bioinstrument one should always be aware of the health, fire, chemical, and biological hazards present in the laboratory, and other hazards that could arise during instrument construction, normal operation, and possible malfunction. The US Occupational Safety and Health Administration (OSHA) provides guidelines on hazards present in the laboratory environment (Osha n.d.) and

those arising from mechanical equipment operation (Osha n.d.). Additionally, the International Organization for Standardization (ISO) has developed comprehensive standards on machinery safety (Iso n.d.). Material Safety and Data Sheets should be used in tandem with these guidelines to design instruments that are robust to hazardous conditions, keeping the user safe.

Certain hazards can arise during the operation of the poseidon syringe pump system that are similar to those encountered when operating other equivalent devices. These include electrical shocks, clogged lines creating pressurized liquids, and material compatibility.

The poseidon syringe pump system uses 3D printed PLA plastic and standard off the shelf components which do not pose a health hazard if handled correctly. Improper handling of plastics however can pose major safety concerns. Designers should consider how their instrument operates when used under elevated temperatures exceeding the melting point of the plastic or under high stress exceeding the yield strength of the plastic. Initial tests of the poseidon syringe pump showed excessive bending of the syringe pump body which we mitigated by reinforcing the body with set screws and a thicker base. We also considered forces induced on the syringe pump due to clogging. To mitigate the possibility of catastrophic failure we set the reference voltage on the motor controller such that the motor would stall when the pressure in a 1 mL syringe reaches 4 MPa, prior to any failure occurring (“Pololu - DRV8825 Stepper Motor Driver Carrier, High Current” n.d.), and well below the ultimate tensile strength of the PLA plastic used in the current design (“MatWeb - Overview of Materials for Polylactic Acid (PLA) Biopolymer” n.d.).

The chemical properties of the materials used in designing instrument parts should be considered (Appropedia n.d.) if one is designing an instrument that could come in contact with organic solvents. We note that the PLA plastic used is compatible with most solvents (“Rosemount Analytical - Chemical Resistance Chart” n.d.). One benefit of open source 3D printable designs is that there are a number of 3D printing materials that are chemically compatible with many types of standard wet lab environmental conditions and hazards (Stratasys Inc 1989; “SLS Materials: Nylon Based Powders for Building Robust Plastic Parts” 2017, “Rosemount Analytical - Chemical Resistance Chart” n.d.).

Finally, in the development of any open source bioinstrument, after identifying safety requirements it is important that hazards and safe operating procedures be clearly communicated. We describe the safety aspects of the poseidon system on the project Github page.

Simplicity: Making it easy to source, build, and operate

Simplicity and ease-of-use are essential for the adoption of bioinstruments. Sourcing components for a design should be as easy as possible, prioritizing off-the-shelf components during development and incorporating harder to find parts only if necessary for the application at hand. An accurate and up-to-date bill of materials (BOM), with ideally more than one vendor for each part, simplifies purchasing and leads to easier adoption. For the poseidon system, we ensured that users would be able to purchase all the components from Amazon. During assembly, it is important to recognize that using specialized equipment - even soldering a circuit board - may be a barrier to adoption. While such specialized assembly processes are sometimes unavoidable, simplicity is paramount. An excellent way to assess the difficulty of assembly is to have people unfamiliar with the project perform the assembly using only the documentation available. With the poseidon system it was possible to design around most of these constraints, and we verified that assembly of a single pump by a new user following the instruction video takes less than 15 minutes, requiring only pliers and screwdrivers.

Simplicity considerations also apply to software. For example, minimizing dependency on external software libraries simplifies installation and avoids versioning issues. From a user's perspective, having a single executable file for the software is ideal. We compiled the Python scripts into single-click executable files for Mac, Windows, and Linux. After testing we realized that a flexible user interface design was critical to develop software that minimized user error. The original rigid custom GUI code did not allow us to easily resize buttons, change button layout, or add new functionalities. Using Qt Designer, a drag and drop GUI creator, we could overcome these challenges and create a basic, functional user interface that is touch-screen and click-button compatible. Additionally, the GUI can easily be adapted and modified for the needs of future adopters. The custom poseidon Arduino firmware needs to be loaded onto the Arduino Uno board following simple instructions. If users wish to use a Raspberry Pi to operate poseidon, installation requires flashing an SD card with the official version of the Raspbian OS image.

Modularity: Building independent and individual units

Because some users will want to adapt a design to new use cases, it is important to consider how easily a design can be taken apart, tweaked, and re-purposed. A modular design with independent components that can be interfaced with each other is easier to re-purpose and improve on than a tightly integrated device. When a design is not modular, developing new features may require a complete redesign. This is the problem that led us to develop poseidon in the first place – our commercial, highly integrated system was too rigid to meet our changing needs. The software could not be improved or modified, and the small integrated touchscreen interface, marketed as an advantage

("PHD ULTRA\texttrademark Syringe Pumps" n.d.), was a hindrance to routine operation. In the case of poseidon, some users might want to add additional features to the pumps such as an electronic end stop. The modularity of the Arduino board makes this change straightforward and simple to implement.

A design is also easier to modify when common and standardized parts and connectors are used. Standardization is ubiquitous in both open source hardware and software projects. Open source 3D printers use a common set of screws, rods, extruding nozzles, and electronics. Often these printers are variations of a few common, popular, and proven designs. The standardization of 3D printer parts means that they, in turn, can be readily adapted for new use cases. Almost all of the components used in the poseidon system can be found in open source desktop 3D printers.

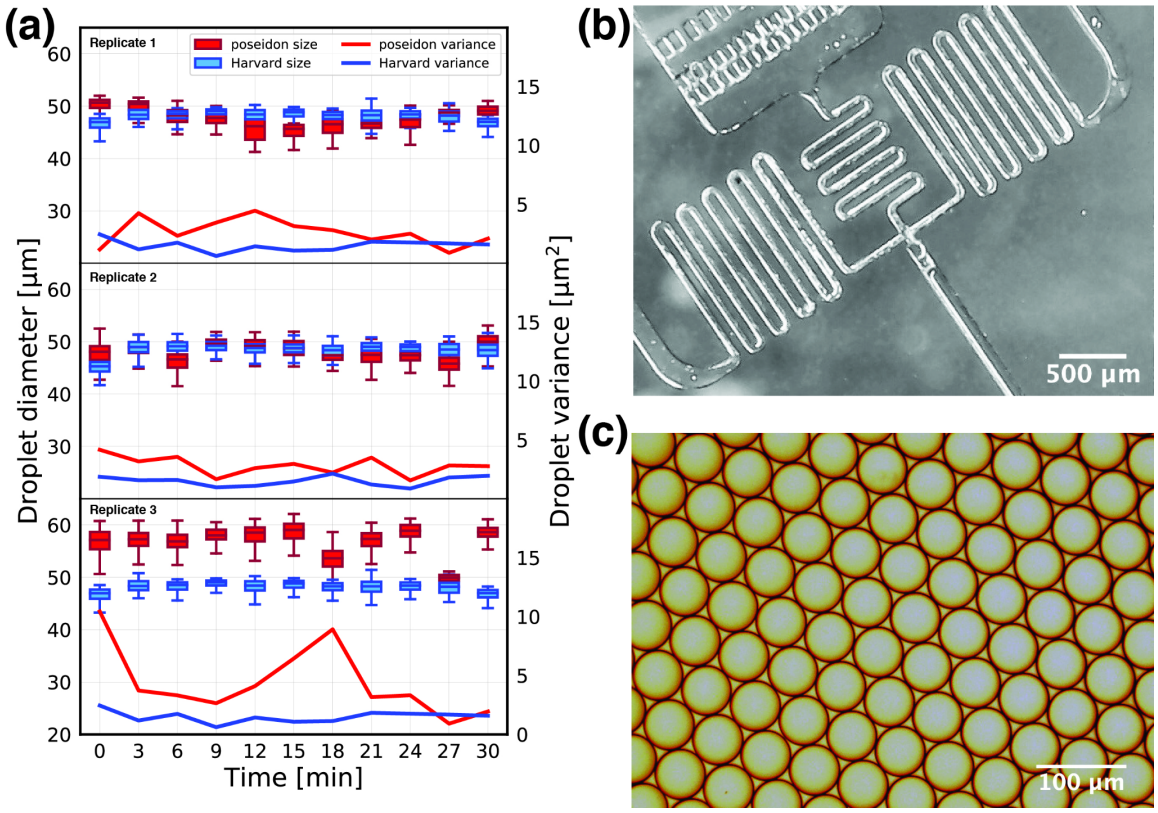


Figure 4: Benchmarking the poseidon system against the Harvard Apparatus system. Using a droplet generation chip we compared the droplet diameters between two systems. (a) A droplet size of 58 µm in diameter is expected for the given flow rates. The variance in the sizes of the droplets created with the two systems is comparable. (b) A microfluidic droplet generation chip imaged using the poseidon microscope. (c) Example

of a monodisperse emulsion produced by the poseidon system and imaged with a Motic AE31 Trinocular Inverted microscope.

Benchmarking: Validating with standard protocols
Users need to know the degree to which an instrument design is applicable to the problem at hand. Thus it is important to describe protocols where instruments have been applied and provide benchmarking results. Open source instruments may not always perform as well as commercial systems, but may still be good enough for many applications. Direct comparison with commercial instruments and clear identification of device shortcomings, or features still in development, is important to instill confidence in the device.

The poseidon system has been successfully used in generating monodisperse emulsions using the inDrops droplet generation device (Fig. 4). We achieved targeted droplet size with small variance in droplet diameter. Importantly, we directly compared poseidon with the commercial array from Harvard Apparatus (catalog numbers 2401-408 and 70-3406), demonstrating comparable variance in droplet diameter. The poseidon system can be operated at a range of flow rates (Table S1). By physically adding or removing the microstepping jumpers on the CNC shield board users can access an even wider range of flow rates, so that the same system can be used for high-precision microfluidics experiments and high flow rate applications such as protein purification.

Documentation: Describing the design completely
Clear instructions and documentation are essential to facilitate rapid and painless assembly. Videos, photographs and written descriptions are fundamental for showcasing a design and ensuring adoption. For assemblies, videos are often the most helpful documentation for users and do not take much time and effort to produce. Videos also clearly convey how much effort and time users should expect to invest in assembly. Documentation enables faster and easier understanding of design.

Users who want to modify a design will additionally benefit from understanding design decisions – both those motivated by technical considerations and those motivated by user feedback. How to implement each feature is the result of thought and iteration from the designer, but what is learned may not be readily apparent in the final designs. Documentation of lessons learned, and insight into why design features were implemented a certain way is important; sometimes modifications that seem to be an improvement will create a failure mode that is not readily apparent.

For the poseidon system, multiple build videos are available on YouTube showing the entire assembly process. In making the poseidon documentation website, we also strove

to use clearly labeled photos of the hardware with short written instructions, as this makes it easier for prospective users to grasp the design and expected time investment at a glance. The development and documentation of open source projects benefits tremendously from making use of version control repositories, which streamlines remote collaboration and development tracking. For the poseidon system, we used the online repository GitHub, which allows for version control and documentation of each change made and makes it simple to create user guides and device documentation as can be seen at https://github.com/pachterlab/poseidon.

Figure 5: Summary of the design principles for open source bioinstrumentation.

Discussion

Success and adoption of open source software demonstrates that reliable and powerful technologies can be realized through community development and improvement. We have developed a modular, highly customizable syringe pump array and USB microscope system, poseidon, with potential for broad application across the biological and chemical sciences. Syringe pumps can be used to operate microfluidic chips, control the chemical environment of a bioreactor, purify proteins, precisely add reagents to chemical reactions over time, or dispense specific amounts of fluid for any number of applications that require precise control of fluid flow. We have benchmarked the system against a commercial alternative in a demanding application: microfluidic emulsion generation. In developing the poseidon system as an open source hardware device we have illustrated seven design principles that we hope can facilitate successful development of open source hardware devices (Fig. 5.) Adhering to these principles from the outset of a project will maximize the chance of community adoption and spur further improvements.

Methods

Testing data is available at https://github.com/pachterlab/poseidon.

Acknowledgements

We thank Nicolas Bray and Kersh Theva for testing prototypes of the poseidon system and for valuable feedback. Thanks to Shannon Hateley for initial help with 3D printing and Zaid Adel Zayyad for designing the icons in Fig. 5.

References

Appropedia. n.d. "Chemical Resistance of 3D Printable Polymers: Literature Review." Accessed April 13, 2019. https://www.appropedia.org/Chemical_resistance_of_3D_printable_polymers:_literature_review.

Arduino Project. 2018. "Arduino Project." *Arduino Project*. https://www.arduino.cc/.

Autodesk. n.d. "Autodesk Fusion 360: Cloud Powered 3D CAD/CAM Software for Product Design." Accessed October 7, 2018. https://www.autodesk.com/products/fusion-360.

Banović, Luka, and Boštjan Vihar. 2018. "Development of an Extruder for Open Source 3D Bioprinting." *Journal of Open Hardware* 2 (1). https://doi.org/10.5334/joh.6.

Carvalho, Matheus C., and Rachel H. Murray. 2018. "Osmar, the Open-Source Microsyringe Autosampler." *HardwareX* 3 (April): 10–38.

Carvalho, Matheus C., Christian J. Sanders, and Ceylena Holloway. 2018. "Auto-HPGe, an Autosampler for Gamma-Ray Spectroscopy Using High-Purity Germanium (HPGe) Detectors and Heavy Shields." *HardwareX* 4 (October): e00040.

Dolgin, Elie. 2018. "How to Start a Lab When Funds Are Tight." *Nature* 559 (7713): 291–93.

Fitzsimmons, Ross E. B., Mark S. Aquilino, Jasmine Quigley, Oleg Chebotarev, Farhang Tarlan, and Craig A. Simmons. 2018. "Generating Vascular Channels within Hydrogel Constructs Using an Economical Open-Source 3D Bioprinter and Thermoreversible Gels." *Bioprinting* 9 (March): 7–18.

Forman, C. J., H. Tomes, B. Mbobo, R. J. Burman, M. Jacobs, T. Baden, and J. V. Raimondo. 2017. "Openspritzer: An Open Hardware Pressure Ejection System for Reliably Delivering Picolitre Volumes." *Scientific Reports* 7 (1): 2188.

Gibb, Alicia. n.d. *Building Open Source Hardware: DIY Manufacturing for Hackers and Makers*. Addison-Wesley.

Iso. n.d. "Safety Standards Catalogue." Accessed May 13, 2019. https://www.iso.org/committee/54604/x/catalogue/.

Macosko, Evan Z., Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, et al. 2015. "Highly Parallel Genome-Wide Expression Profiling of Individual Cells Using Nanoliter Droplets." *Cell* 161 (5): 1202–14.

Maia Chagas, André. 2018. "Haves and Have Nots Must Find a Better Way: The Case for Open Scientific Hardware." *PLoS Biology* 16 (9): e3000014.

Maia Chagas, Andre, Lucia L. Prieto-Godino, Aristides B. Arrenberg, and Tom Baden. 2017. “The €100 Lab: A 3D-Printable Open-Source Platform for Fluorescence Microscopy, Optogenetics, and Accurate Temperature Control during Behaviour of Zebrafish, Drosophila, and Caenorhabditis Elegans.” *PLoS Biology* 15 (7): e2002702.

“MatWeb - Overview of Materials for Polylactic Acid (PLA) Biopolymer.” n.d. Accessed May 13, 2019. http://www.matweb.com/search/DataSheet.aspx?MatGUID=ab96a4c0655c4018a8785ac4031b9278&ckck=1.

Nuñez, Isaac, Tamara Matute, Roberto Herrera, Juan Keymer, Timothy Marzullo, Timothy Rudge, and Fernán Federici. 2017. “Low Cost and Open Source Multi-Fluorescence Imaging System for Teaching and Research in Biology and Bioengineering.” *PloS One* 12 (11): e0187163.

Opentrons Inc. 2018. “Opentrons Inc.” *Opentrons*. https://opentrons.com/.

Osha. n.d. “Laboratory Safety Guidance.” Accessed May 13, 2019a. https://www.osha.gov/Publications/laboratory/OSHA3404laboratory-safety-guidance.pdf.

———. n.d. “Machine Guarding Guidelines.” Accessed May 13, 2019b. https://www.osha.gov/SLTC/machineguarding/index.html.

Pearce, Joshua M. 2012. “Materials Science. Building Research Equipment with Free, Open-Source Hardware.” *Science* 337 (6100): 1303–4.

“PHD ULTRA\texttrademark Syringe Pumps.” n.d. Accessed April 8, 2019. https://www.harvardapparatus.com/pumps-liquid-handling/syringe-pumps/infuse-withdraw/standard-infuse-withdraw-phd-ultra-syringe-pumps.html.

“Pololu - DRV8825 Stepper Motor Driver Carrier, High Current.” n.d. Accessed April 8, 2019. https://www.pololu.com/product/2133/resources.

Raspberry Pi Foundation. 2018. “Raspberry Pi.” *Raspberry Pi*. https://www.raspberrypi.org.

“Rosemount Analytical - Chemical Resistance Chart.” n.d. Accessed May 13, 2019. http://instrumentationandcontrol.net/wp-content/uploads/2016/09/ROSEMOUNT-2010-Chemical-Resistance-Chart_.pdf.

“SLS Materials: Nylon Based Powders for Building Robust Plastic Parts.” 2017. GKN FORECAST 3D. June 28, 2017. https://www.forecast3d.com/materials/sls.

Stephenson, William, Laura T. Donlin, Andrew Butler, Cristina Rozo, Bernadette Bracken, Ali Rashidfarrokhi, Susan M. Goodman, et al. 2018. “Single-Cell RNA-Seq of Rheumatoid Arthritis Synovial Tissue Using Low-Cost Microfluidic Instrumentation.” *Nature Communications* 9 (1): 791.

Stratasys Inc, Stratasys Inc. 1989. “US5121329A - Apparatus and Method for Creating Three-Dimensional Objects.” Translated by Stratasys Inc Stratasys Inc. 1989. https://patents.google.com/patent/US5121329A/en.

The Qt Company. n.d. “Qt Designer Manual.” Accessed October 7, 2018. http://doc.qt.io/qt-5/qtdesigner-manual.html.

UNC Greensboro, Mitchell P. Croatt. n.d. “DIY Flow Chemistry Setup – UNC-Greensboro.” Accessed April 8, 2019. https://chem.uncg.edu/croatt/flow-chemistry/.

Wayland, Matthew T., and Matthias Landgraf. 2018. “A Cartesian Coordinate Robot for Dispensing Fruit Fly Food.” *Journal of Open Hardware* 2 (1): 3.
Wijnen, Bas, Emily J. Hunt, Gerald C. Anzalone, and Joshua M. Pearce. 2014. “Open-Source Syringe Pump Library.” *PloS One* 9 (9): e107216.
Zilionis, Rapolas, Juozas Nainys, Adrian Veres, Virginia Savova, David Zemmour, Allon M. Klein, and Linas Mazutis. 2017. “Single-Cell Barcoding and Sequencing Using Droplet Microfluidics.” *Nature Protocols* 12 (1): 44–73.

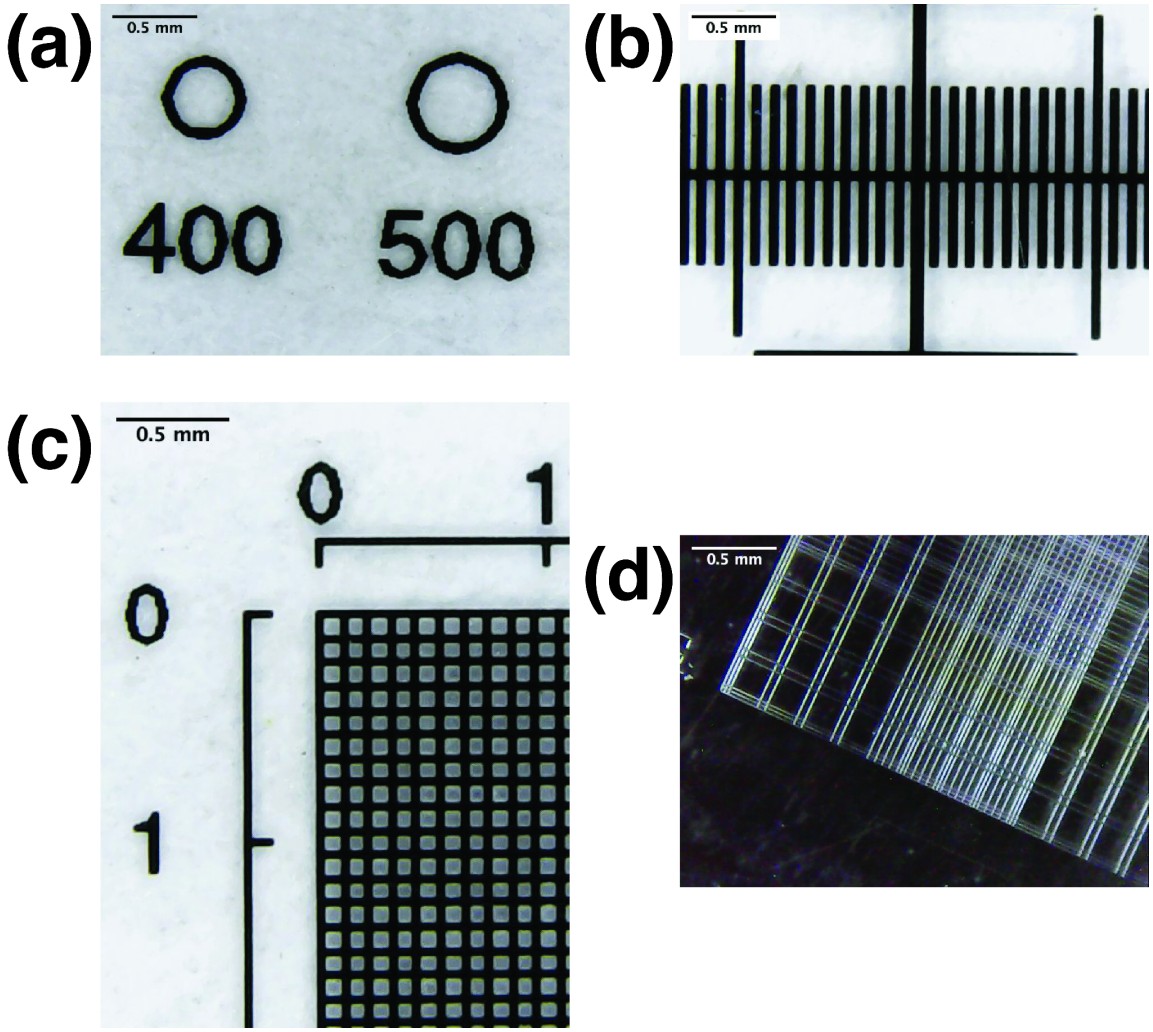
Supplementary Material

Microstep	Syringe Size: 1 mL	3	5	10	20	30	60
1		624	2,084	4,065	5,887	10,261	13,179	19,946
2	Maximum	499	1,667	3,252	4,710	8,209	10,544	15,957
4	Flow Rate	250	834	1,626	2,355	4,104	5,272	7,978
8	[mL/hr]	125	417	813	1,177	2,052	2,636	3,989
16		62	208	406	589	1,026	1,318	1,995
32		31	104	203	294	513	659	997

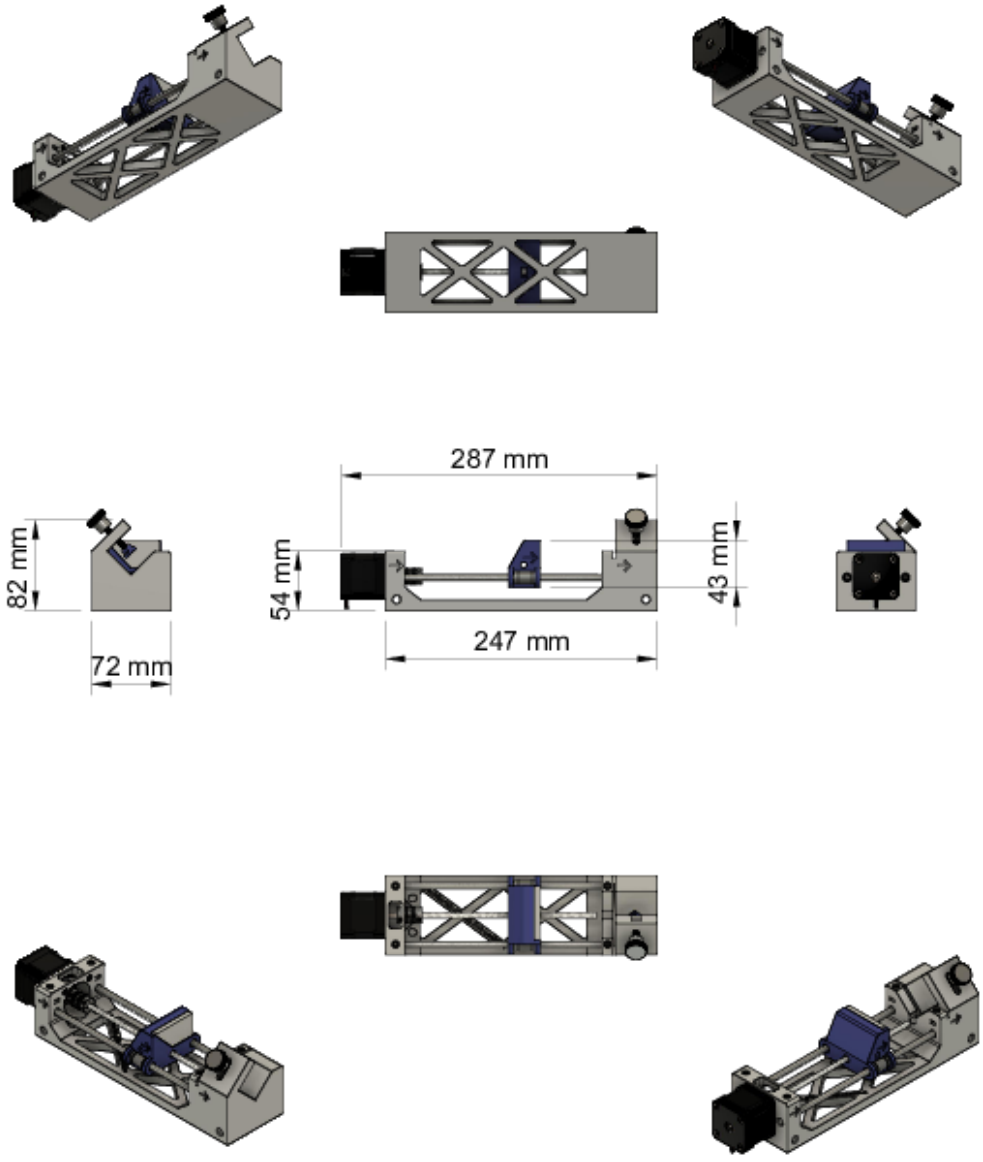
Supplementary Table 1: Maximum rates for given syringe and microstepping: Maximum flow rates [mL/hr] for a single pump for a given microstepping and Becton Dickinson (BD) syringe size. At lower flow rates, higher microstepping is desirable for a smoother flow.

Microstep	Steps per Revolution	Precision [μm]	Maximum Speed [mm/s]
1	200	4	10
2	400	2	8
4	800	1	4
8	1600	0.5	2
16	3200	0.25	1
32	6400	0.125	0.5

Supplementary Table 2: Stepper motor performance: Precision was calculated based on the steps per revolution and the pitch of the lead screw (0.8 mm/revolution). The maximum speed of the carriage was measured and an error of 5.5% ± 1.5% s.d. of the set speed was observed (n = 52).

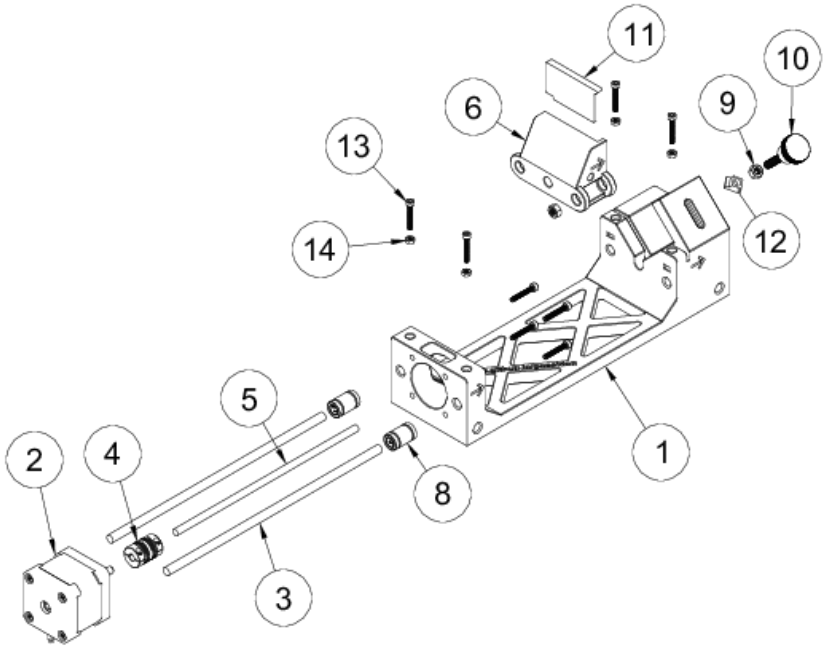


Supplementary Figure 1: \label{fig:sfig2} Representative images taken with the poseidon microscope. Images (a) - (c) are of the calibration slip supplied with the microscope system and image (d) is of the Incyto Hemocytomer DHC-B02 (Burker Turk) (http://www.incyto.com/shop/item.php?it_id=3).

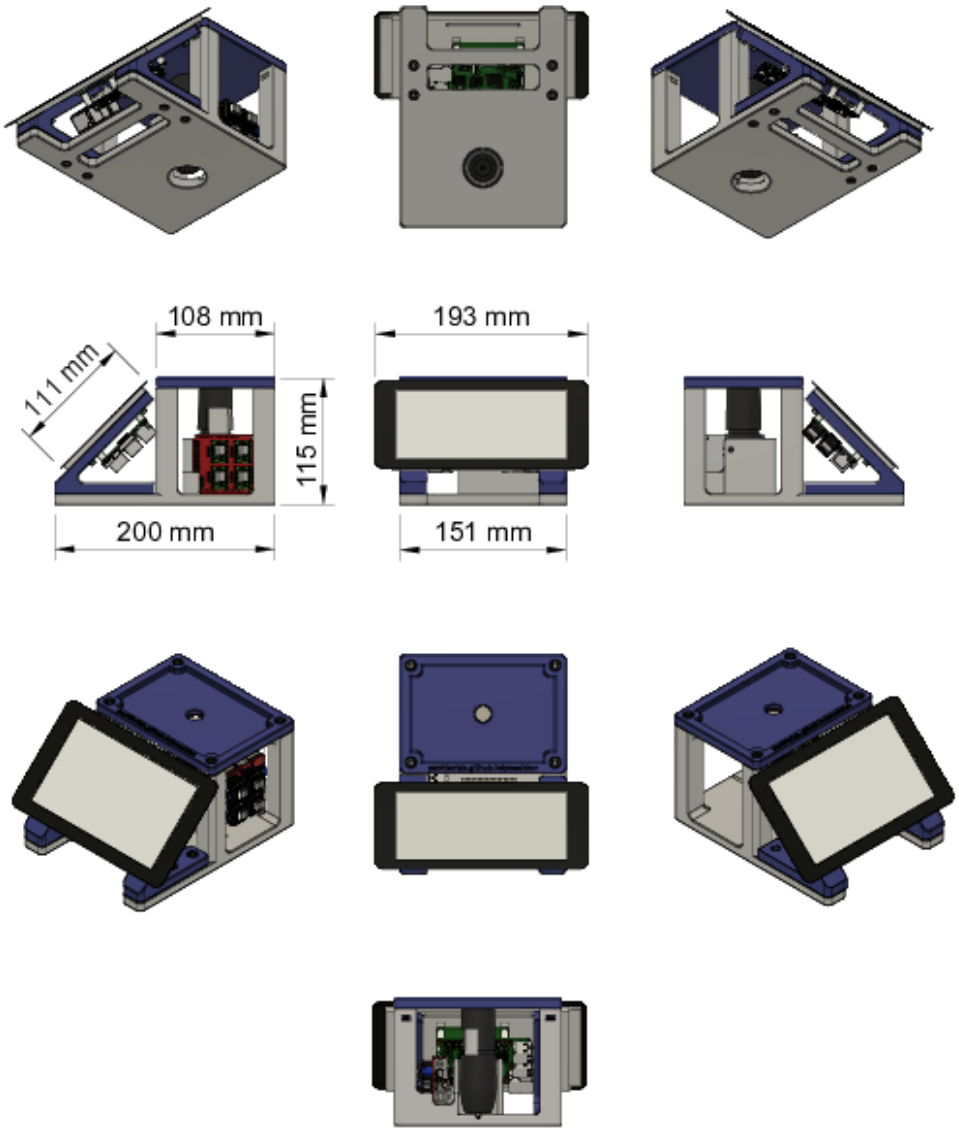


Supplementary Figure 2: CAD renderings of the poseidon syringe pump. Multiple views and major dimensions are shown.

Parts List		
Item	Qty	Part Name
1	1	3D printed pump body
2	1	NEMA 17 stepper motor
3	2	6mm steel rod, 200mm length
4	1	5mm to 5mm motor shaft coupling
5	1	M5 threaded rod, 170mm length
6	1	3D printed carriage
8	2	6mm linear bearing
9	2	M5 nut
10	1	M5 knob screw
11	1	3D printed cover slide
12	1	3D printed syringe brace
13	8	M3 screw, 20mm length
14	4	M3 nut

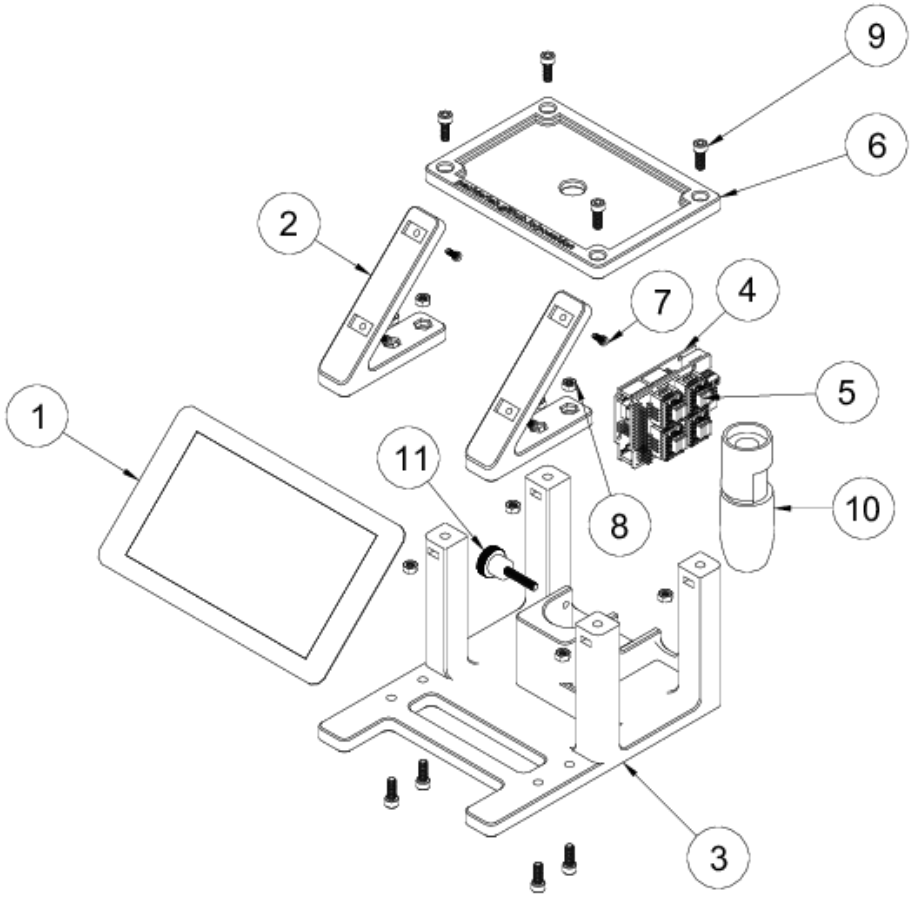


Supplementary Figure 3: Parts list and exploded view of the poseidon syringe pump.



Supplementary Figure 4: CAD renderings of the poseidon microscope. Multiple views and major dimensions are shown.

Parts List		
Item	Qty	Part Name
1	1	Raspberry Pi board and touchscreen
2	2	3D printed screen leg
3	1	3D printed microscope base
4	1	Arduino UNO
5	1	Arduino CNC shield
6	1	3D printed top stage
7	4	M3 screw, 10mm length
8	8	M5 nut
9	8	M5 screw, 14mm length
10	1	USB microscope
11	1	M5 knob screw



Supplementary Figure 5: Parts list and exploded view of the microscope.

Section 2: *colosseum* fraction collector

Preamble

Current devices for performing liquid sampling are costly and require complicated control schemes. The colosseum fraction collection is a novel bioinstrument for performing long-time series fluid sampling that performs as well as the state-of-the-art at a fraction of the cost. It uses a cam-follower mechanism that requires only one motor instead of the standard two. This lowers the device cost, overcoming a roadblock in a critical technology for performing long-time scale single-cell RNA-sequencing. By automating fluid collection in an inexpensive and scalable manner, massive amounts of scRNAseq data can be generated.

Summary

We present colosseum, a low-cost, modular, and automated fluid sampling device for scalable fluidic applications. The colosseum fraction collector uses a single motor, can be built for less than $100 using off-the-shelf and 3D-printed components, and can be assembled in less than an hour. Build Instructions and source files are available at https://doi.org/10.5281/zenodo.4677604.

Introduction

Fraction collectors that sample from a microfluidic stream [1], are preferable to manual collection that can be tedious and introduce human error [2]. Commonly used in fast protein liquid chromatography (FPLC), typical fraction collectors consist of a rotating rack loaded with containers and a distributing arm for collecting fixed volumes of fluid [3,4]. Most laboratories currently rely on commercial fraction collectors, which are expensive and difficult to customize (Supplementary Table 1). To reduce cost and facilitate custom applications, a number of open-source fraction collectors have been developed, e.g. [5,6]. These devices, while less expensive, continue to rely on complex engineering designs and parts that may be difficult to source and manufacture, thus driving costs higher, lengthening the assembly process, and complicating operation.

We have designed and built a simple, low-cost, and modular fraction collector that is easy to assemble and use. This open-source fraction collector, which we call colosseum, is based on design principles for modular, robust, open-source hardware [7], and offers advantages to commercial systems by virtue of being significantly less expensive and easily customizable.

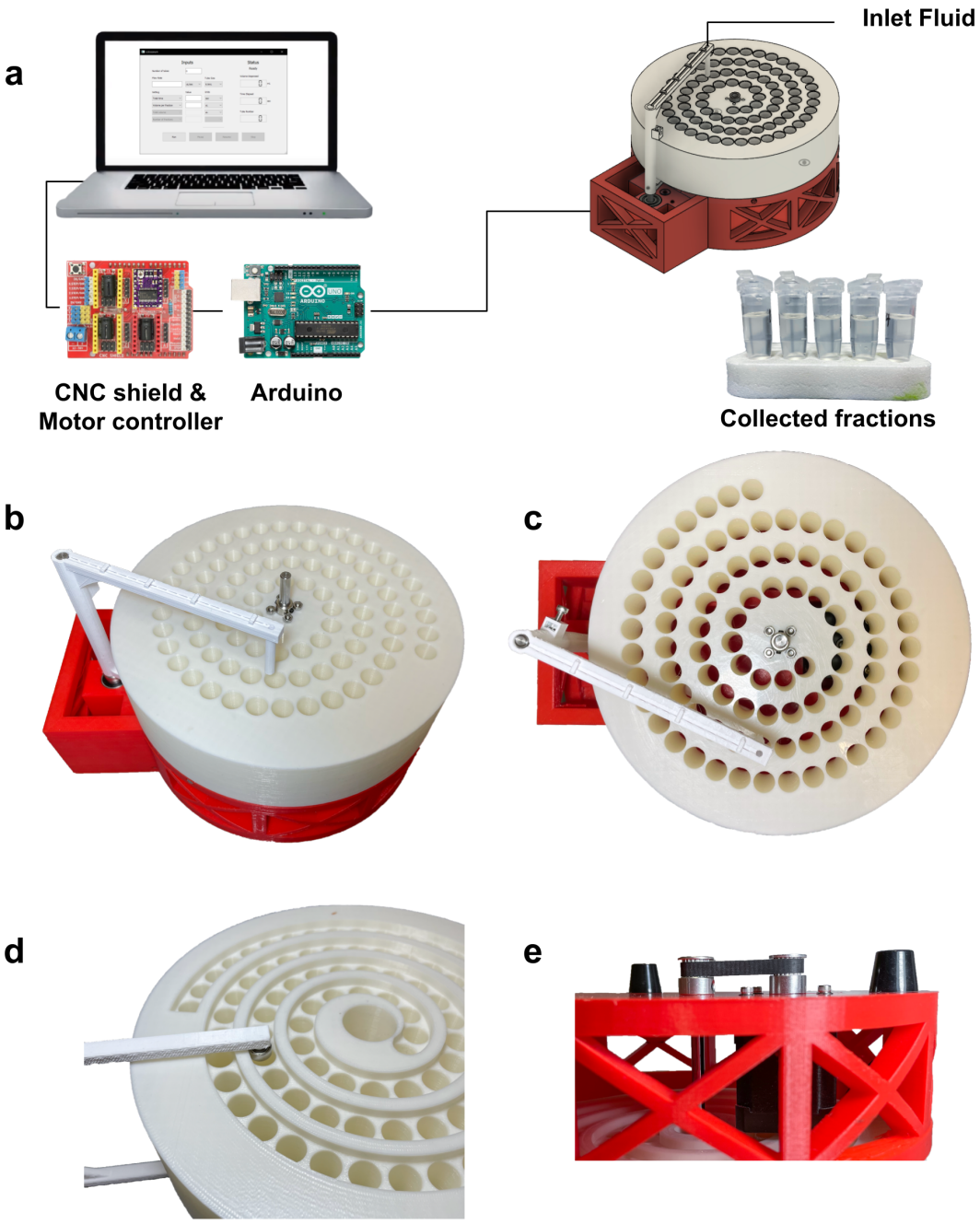


Figure 1: (a) The colosseum fraction collector (left) is controlled by a single motor. A motor controller shield (red) is connected to an Arduino Uno (blue) and drives the motor. The computer's Graphical User Interface (right) and Python backend sends motor movement instructions to the Arduino. The Arduino-motor controller then sends those instructions to the motor. A motor located in the base turns the shaft of the tube rack. Grooves in the bottom of the fraction collector constrain the dispenser arm to rotate in

tandem. (b) Angled view, (c) top view, (d) bottom view of mechanical coupling between the dispenser arm and tube rack, (e) side view of mechanical coupling of motor and tube rack.

The colosseum fraction collector can be assembled in less than an hour and costs $67.02. Unlike the micrIO [6], which is built from parts of a salvaged Illumina Genome Analyzer that costs $1500, the colosseum fraction collector uses off-the-shelf and 3D-printed parts (Supplementary Table 2). The LEGO MINDSTORM fraction collector [5] costs $500, and while it uses more commonly available components, it still requires cutting and bending of steel C-channel. Furthermore, most fraction collectors require the use of multiple axes to position a dispenser head over a reservoir. Control of such a system can require communicating with and driving up to three separate motors in tandem. The colosseum fraction collector is based on a simpler design where a mechanical coupling between the motor, the tube rack, and the dispenser arm enables rotation of the rack and position of the arm with only one motor. Designing around a single motor simplifies operation, and reduces cost, complexity, and assembly time.

Results

The colosseum fraction collector consists of four 3D-printed components, two rotary shafts, five rubber feet, one stepper motor, an Arduino, and a motor controller (Figure 1a,b). We chose the spiral tube layout (Figure 1c) instead of the rectangular tube layout of previously published fraction collectors as it enables serial fraction collection with only one motor. By coupling the dispenser arm to the tube rack with a slot-cam mechanical coupling (Figure 1d) we constrained the rotation of the tube rack and movement of the dispenser arm to rotation of a single stepper motor located in the base of the fraction collector (Figure 1e).

Hardware name	colosseum
Subject area	· Engineering and Material Science · Chemistry and Biochemistry · Biological Sciences (e.g. Microbiology and Biochemistry)
Hardware type	· Biological sample handling and preparation · Mechanical engineering and materials science

Open Source License	BSD-2
Cost of Hardware	$67.02
Source File Repository	https://doi.org/10.5281/zenodo.4677604
colosseum Project Repository	https://github.com/pachterlab/colosseum

Table 1: Specifications

The device is modular: each component can be developed, tested, and fabricated separately using mutually compatible interfaces. The tube rack fits 1.5 mL Eppendorf tubes and can easily be modified to accept tubes of varying sizes under the constraint that they follow the spiral pattern (Figure 2a). The tube rack fits 88 tubes with a packing efficiency of 60.2% relative to the optimal packing of 146 tubes on a circular disk of the same size as the tube rack (Supplementary Figure 1; [8]). In addition, the dispenser arm can be modified to accept connectors and tubing of various sizes to enable parallel dispensing.

The device is controlled by a graphical user interface (GUI) that communicates with an Arduino, CNC motor shield, stepper motor driver, and software adapted from the poseidon syringe pump (Supplementary Figure 2a,b); [7]. Experiment parameters such as flow rate, total volume, total time, volume per fraction, and number of fractions are input by the user in the GUI (Supplementary Table 3) and the Python or JavaScript back-end structures and sends Arduino-interpretable commands to the Arduino for execution. The GUI can be installed with the pip package-management tool and run with a single command on Mac, Linux, or Windows, or it can be run directly in a web browser at https://pachterlab.github.io/colosseum/.. The web-browser implementation takes advantage of the web serial specification [9] and the browser-serial API [10] to read and write from the serial port within a web browser environment.

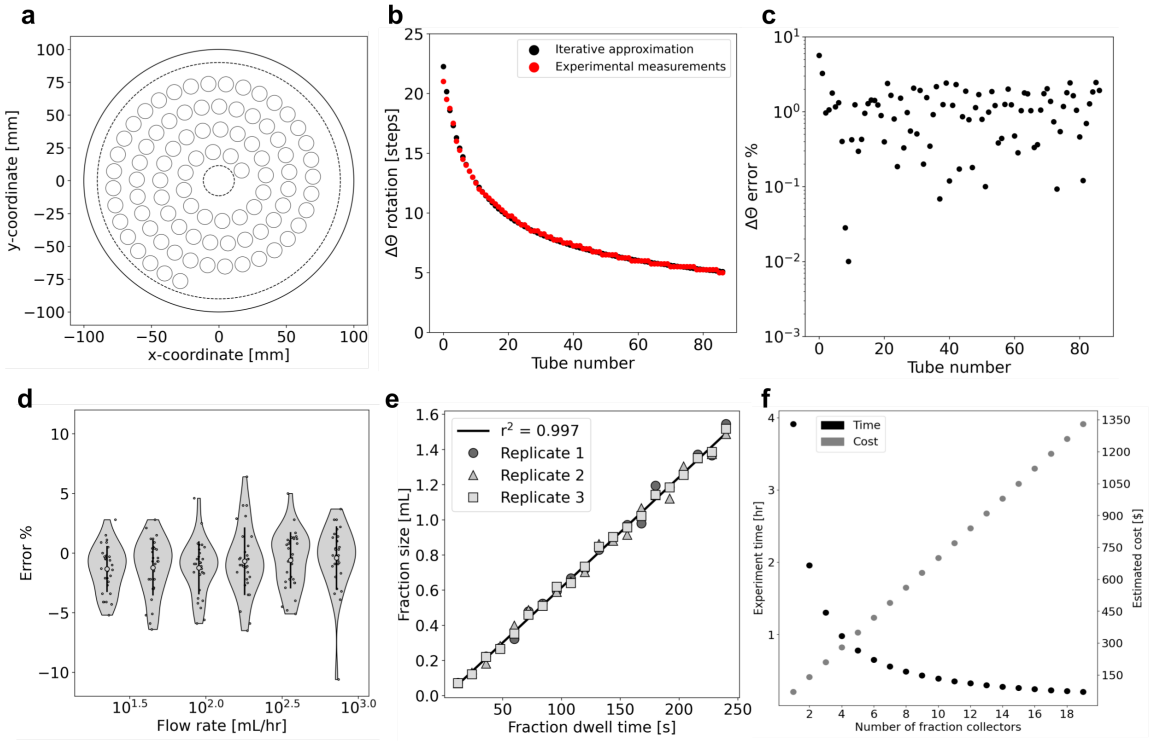


Figure 2: (a) Tube placement on the colosseum is defined by an Archimedean spiral with tubes distributed 13 mm apart along the spiral and with 17.39 mm distance between subsequent arms of the spiral. The dotted-line innermost circle corresponds to the area on the tube rack designated for the set screws on the center shaft. The larger dotted-line circle corresponds to the area available for tubes. The solid outer circle corresponds to the tube rack boundary. (b) The tubes are placed uniformly along the spiral where the arc length between any two tubes is constant, but the rotational displacement between any two tubes is nonconstant. (c) Iterative approximation to the tube locations is similar to the measured tube locations. (d) The error in the fraction size for 88 samples across a range of flow rates. (e) The fraction size increases with increasing dwell time for a constant flow rate and the Spearman correlation of the means is 0.997. (f) Multiple fraction collectors enable parallel collection which drastically decreases experimental time at a marginal increase in cost. [Code a,b,c (https://github.com/pachterlab/BKMGP_2021/blob/main/analysis/archimedian_spiral.ipynb), Code d (https://github.com/pachterlab/BKMGP_2021/blob/main/analysis/varying_flowrate.ipynb), Code e (https://github.com/pachterlab/BKMGP_2021/blob/main/analysis/constant_flowrate.ipynb), Code f (https://github.com/pachterlab/BKMGP_2021/blob/main/analysis/scalability.ipynb)]

To ensure that commands set by the GUI correctly align the dispenser arm with each collection tube, we measured and converted the angle between pairs of tubes to motor steps, and programmed this list of angular displacements into the control software (Methods). We also used a simple iterative scheme to approximate the position of equally spaced points along an Archimedean spiral and compared it to our measurements. We found high concordance in the angular displacements (Figure 2b,c). This allows us to programmatically generate arbitrary spiral motor displacements based on the distances between successive tubes and distances between successive arms of the spiral.

In order to characterize collection errors across a range of flow rates commonly used in microfluidics and FPLC [11,12], we sampled 180 fractions over six flow rates ranging from 22.5 mL/hr to 720 mL/hr. We found that the collection errors were within ±6.5%, with one sample having -10.6% error due it being the first fraction collected. These data suggest that the use of the colosseum system with the poseidon syringe pump results in accurately collected fractions (Figure 2d). Next, we sought to assess the fraction collecting performance over an increasing amount of sample volume, as is commonly performed in gradient elution series [4]. For a fixed flow rate, we collected 20 fractions with 12-second increments in collection time per tube over the course of 42 minutes in three replicates (Figure 2e). We found that the collected fractions closely followed the expected fraction amount with a Spearman correlation of 0.997, showing that the colosseum fraction collector can be used to accurately collect gradient elution series.

Discussion

We have demonstrated a low-cost, modular, and automated fraction collector that uses 3D-printed parts and off-the-shelf components, can be built in an hour, and is simple to run. We show how colosseum samples fluid accurately over a wide-range of flow rates making it useful for microfluidics experiments and FPLC. The low cost of our device could enable several instruments to run in parallel. For example a single control board can in principle run multiple fraction collectors and syringe pumps thus facilitating large-scale experiments (Figure 2f). We have also thoroughly documented the build process with instructional README's and videos (Supplementary Figure 3), and we have made all of the results described in this paper reproducible on Google Colab.

Methods

We designed the colosseum fraction collector by following basic principles of open-source hardware design [7].

Part design

The fraction collector consists of four 3D-printed parts: a base, base plate, dispenser arm, and tube rack. The base holds the base plate, dispenser arm, and tube rack in place with additional hardware. The base plate acts as a horizontal support for the main rotary shaft, with rotational bearings that support the shaft in two places. The dispenser arm consists of two connected parts: the top part of the arm holds the fluid tubing and the bottom part acts as a cam follower that follows the spiral track on the bottom of the tube rack. Collection tubes are placed in the tube rack and are organized in a spiral pattern that mirrors the pattern the dispenser arm follows during rotation. The tube rack is constrained to the shaft with a flange coupling set screw and is mechanically coupled to the motor with a timing belt so that rotation of the motor results in rotation of the tube rack and the dispenser arm.

After numerous sketch iterations, we used Fusion 360 [13] to generate a 3D model of the device and added dimensional tolerances of +3-5% to all parts to account for variance in 3D printing.

Part fabrication

STL files were generated using the 3D part models from Fusion 360. To prepare the appropriate files for 3D printing, Simplify3D [14] was used to slice the STL model and generate GCode with 10% infill and 0.2 mm layer height. Parts were printed on a Prusa i3 Mk3 3D printer [15]. GCode was loaded onto an SD card and the parts were printed using 1.75 mm diameter PLA filament with a nozzle temperature of 215 °C and a bed temperature of 60 °C. All parts were printed at 10% infill. STL files for all parts can be found in the GitHub repository (https://doi.org/10.5281/zenodo.4677604). The time to print all parts separately was approximately 73 hours, but may vary depending on the printer model used and the print settings (Supplementary Table 2). All parts required to assemble the colosseum fraction collector can be found in the bill of materials (https://doi.org/10.5281/zenodo.4677604).

Device assembly

A complete guide on how to assemble the colosseum fraction collector can be found on YouTube (https://www.youtube.com/watch?v=yG7ECh5GO0o) (Supplementary Figure 3). A step-by-step assembly guide is also available on protocols.io [16].

The assembly of the device starts with the base. Five rubber feet are screwed onto the bottom of the base to stabilize the device and to ensure that the timing pulleys on the motor and the tube rack shaft are elevated and free of obstruction. A timing belt pulley is secured to the shaft of a NEMA 17 motor and the motor is then screwed onto the floor of

the base. The tube-rack shaft is also inserted into the floor of the base along with a bearing that acts to stabilize the shaft. A timing belt pulley is secured to the shaft and couples its rotation with that of the motor. The motor and the shaft are connected by a timing belt of length 120 mm. The mounting holes in the base for the motor are designed so that the user can adjust the distance between the two timing pulleys in order to prevent slippage of the timing belt. Additionally, washers are inserted in between the base floor and the screws holding the motors so that the plastic of the base does not get worn out over time.The base plate is then screwed onto the floor of the base using M5 screws and nuts.

The dispenser arm, which is secured to a shaft with an M3 set screw, is placed into the base plate along with a bearing. A torsion spring is placed on the shaft, between the dispenser arm and the base plate to lessen slack between the dispenser cam follower and the tube rack spiral groove. The tube rack shaft is then inserted into the tube rack and secured in place with a flange coupling set screw.

The motor cables are routed through the side of the base and connected to the Arduino. The Arduino is connected to a CNC shield and DRV8825 Pololu motor controller [17]. The Arduino is also connected to a computer. This allows the user to send and receive signals to the motor via serial commands. Power is supplied to the stepper motor driver via terminals on the CNC motor shield. We supply the stepper motor driver with 12 V DC at 3.0 A. The DRV8825 has a maximum current rating of 2.2 A per phase and the bipolar NEMA 17 stepper motor has a rated current of 2.0 A per phase. The stepper motor driver limits the amount of current that can be delivered to the stepper motor via a potentiometer.

User Interface

The GUI translates the parameters set by the user into motor commands sent to the Arduino. The Arduino runs our custom firmware, pegasus https://github.com/pachterlab/pegasus, which sends command strings to the motor controller which in turn sends pulse-width-modulated signals to the motor. The GUI is written in Python using Qt, an open-source, cross-platform GUI framework (Supplementary Figure 2a). All packages related to the GUI are pip-installable and the GUI can be launched with a single command from the command line. The web-browser GUI is written in JavaScript, requires no installation, and can be run by simply navigating to a website (Supplementary Figure 2b). The GUI consists of two parts: parameter inputs and a status monitor, the latter of which displays the total volume dispensed, time elapsed, and current tube location. Upon opening the GUI, users are prompted to connect to an Arduino. To run the colosseum fraction collector, users must specify three

parameters: the flow rate, total time or total volume, and volume per fraction or number of fractions (Supplementary Table 3). The remaining parameters are calculated using the ones provided. In addition to these parameters, users must also specify the tube size to ensure that the fraction size will not be greater than the capacity of the tube. Users can operate the colosseum by pressing the run, pause, resume, and stop buttons in the GUI. All software required to run the colosseum fraction collector is freely available on Github under an open source BSD-2-Clause License.

Python 3.6 and JavaScript code is used on the back end to interpret user input from the GUI and send custom commands to the Arduino, accordingly. The Python implementation uses the pyserial package [18] to interface with the serial port and the web-browser implementation uses the browser-serial package [10]. Parameters from the GUI are translated into dwell time per tube and number of tubes to fill. The angle between each tube in the spiral was measured on Fusion 360 using the Inspect tool, saved as a csv file (Supplementary Table 4), and specified in the Python backend. These angles are then converted into the number of steps the motor must rotate. The motor stops rotating at each tube location for a specified amount of time in order to dispense the fluid into the tube. The motor then moves a set number of steps to reach the next tube. The status monitor displays the amount of total volume dispensed, how much time has elapsed since the start of the experiment and which tube the fraction is being dispensed into.

Testing and Validation

We tested the functionality of the device with numerous experiments where tap water is flown in at a set flow rate, or varying flow rate. We used the poseidon syringe pump, a 60 mL syringe, microfluidic tygon tubing and 1.5 mL Eppendorf tubes to pump fluid to the colosseum. The poseidon syringe pump was controlled with the pegasus software. For a varying number of flow rates and a set dwell time per tube for each flow rate (Supplementary Table 5), we collected 30 fractions and compared the fraction sizes to the predicted fraction size of 1 mL by weighing each tube before and after collection (Figure 2d). We used a 200x 1 mg analytical scale manufactured by Yae First Trading Co., ltd part number TEK-AB-0392 to measure the amount of collected fluid. In order to properly fit the Eppendorf tubes on the tube rack, we cut off the caps from the Eppendorf tubes before collecting fractions in them and put them back on for the final measurement making sure that the cap corresponded to the tube from which it was removed.

In follow up experiments we fixed the flow rate and linearly increased the collection time. For a fixed flow rate of 22.5 mL/hr and 20 fractions with 12-second increments in collection time per tube, we collected fractions and compared the observed fraction sizes

to the predicted fraction sizes (Figure 2e). We used pegasus to run the colosseum with varying dwell times per tube.
We estimated the cost and time for using k fraction collectors to show that these devices, when used in parallel, can reduce the experimentation time. For example, if we collect n fractions on each of k fraction collectors with a volume per fraction v and a constant flow rate f per collector then the time it takes to run this collection is t = n/k*v/f.

To test the accuracy of the measured angles between two successive tubes we used an iterative scheme to estimate the radius and angular position based of the polar form of Archimedean spiral of $r=b*\theta$ for a constant b. The radius and the arc length are used to update the angular position and then the angular position is used to update the radius.

Optimal packing was calculated with the "best known packings of equal circles in a circle" online tool [8] with the outermost disk corresponding to the diameter of the area available for tube placement and the packing disks corresponding to the distance between tubes along the arc.

Data analysis
All data analysis was performed with Python 3.7. Jupyter notebooks that run in Google Colab and all experimental data to reproduce Figure 2 can be found on our GitHub repository https://github.com/pachterlab/BKMGP_2021.

Design files

Design file name	File type	Open source license	Location of the file
colosseum_arm	CAD file	BSD-2	Available in repository
colosseum_base	CAD file	BSD-2	Available in repository
colosseum_baseplate	CAD file	BSD-2	Available in repository
colosseum_tubebed	CAD file	BSD-2	Available in repository

Bill of materials
The bill of materials can be found in the GitHub repository.

Build Instructions
Please refer to *Device Assembly* under the Methods section in **Hardware description**.

Operation instructions
Please refer to *User Interface* under the Methods section in **Hardware description**. A video guide to operating the device can also be found in this YouTube video (https://www.youtube.com/watch?v=yG7ECh5GO0o).

Validation and Characterization
Please refer to *Testing and Validation* under the Methods section in **Hardware description.**

Acknowledgments
We thank Justin Bois for naming the colosseum instrument. We also thank the Caltech Library Techlab for helping us 3D print parts. We thank Taleen Dilanyan for wet lab training and support, and Eduardo da Veiga Beltrame for assistance with 3D printing. The authors received no specific funding for this work.

Data & software availability
All data and software to reproduce the results in this manuscript can be found in Zenodo: https://doi.org/10.5281/zenodo.4677604. The project can be found in the GitHub repository: https://github.com/pachterlab/colosseum.

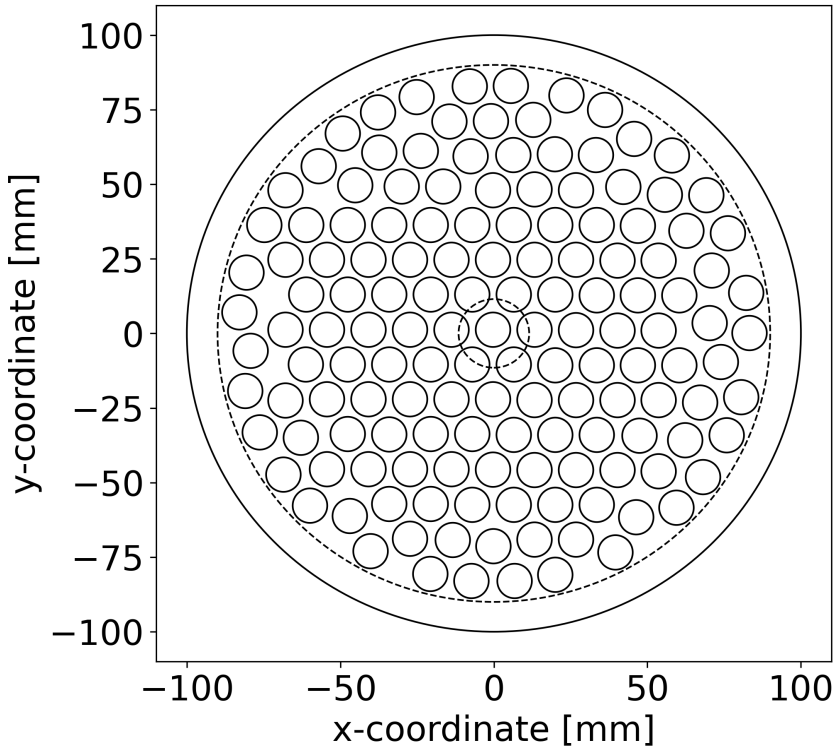
References

1. Blume C, Reale R, Held M, Millar TM, Collins JE, Davies DE, et al. Temporal monitoring of differentiated human airway epithelial cells using microfluidics. PLoS ONE. 2015;10: e0139872. doi:10.1371/journal.pone.0139872
2. Jessop-Fabre MM, Sonnenschein N. Improving reproducibility in synthetic biology. Front Bioeng Biotechnol. 2019;7: 18. doi:10.3389/fbioe.2019.00018
3. Madadlou A, O'Sullivan S, Sheehan D. Fast protein liquid chromatography. Methods Mol Biol. 2017;1485: 365–373. doi:10.1007/978-1-4939-6412-3_19
4. Polson A. A simplified fraction collector for gradient elution chromatography. Journal of Chromatography A. 1961;5: 116–120. doi:10.1016/S0021-9673(01)92828-6
5. Caputo M, Lyles JT, Salazar MS, Quave CL. LEGO MINDSTORMS Fraction Collector: A Low-Cost Tool for a Preparative High-Performance Liquid Chromatography System. Anal Chem. 2020;92: 1687–1690. doi:10.1021/acs.analchem.9b04299
6. Longwell SA, Fordyce PM. micrIO: an open-source autosampler and fraction collector for automated microfluidic input-output. Lab Chip. 2020;20: 93–106. doi:10.1039/c9lc00512a
7. Booeshaghi AS, Beltrame E da V, Bannon D, Gehring J, Pachter L. Principles of open source bioinstrumentation applied to the poseidon syringe pump system. Sci Rep. 2019;9: 12385. doi:10.1038/s41598-019-48815-9
8. Specht E. The best known packings of equal circles in a circle [Internet]. 21 Jun 2018 [cited 26 Jan 2021]. Available: http://hydra.nat.uni-magdeburg.de/packing/cci/cci.html#Applications
9. Caceres M, Grant R, Waldron R, Scheib V, Gulotta F, Hinton S, et al. Web Serial API [Internet]. 2021 [cited 5 Apr 2021]. Available: https://wicg.github.io/serial/
10. Booeshaghi S. sbooeshaghi/browser-serial: First release. Zenodo. 2021; doi:10.5281/zenodo.4671122
11. Westman E, Eriksson S, Låås T, Pernemalm PA, Sköld SE. Separation of DNA restriction fragments by ion-exchange chromatography on FPLC columns Mono P and Mono Q. Anal Biochem. 1987;166: 158–171. doi:10.1016/0003-2697(87)90558-6
12. Cheri MS, Latifi H, Sadeghi J, Moghaddam MS, Shahraki H, Hajghassem H.

Real-time measurement of flow rate in microfluidic devices using a cantilever-based optofluidic sensor. Analyst. 2014;139: 431–438. doi:10.1039/c3an01588b

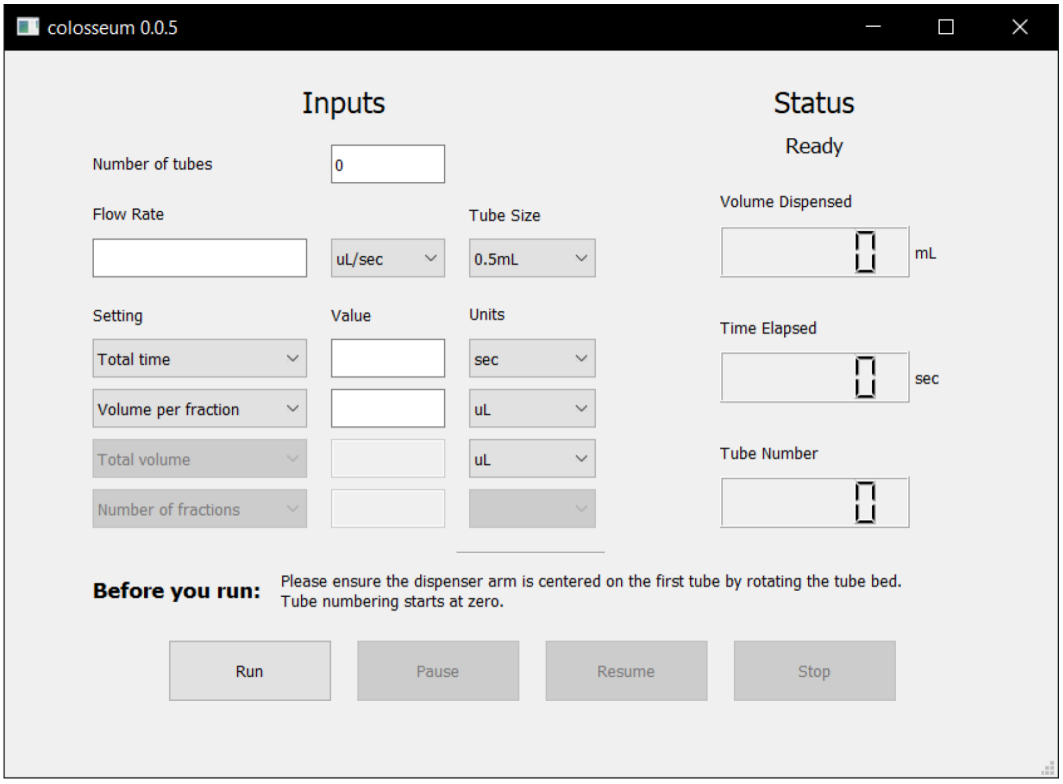
13. Song PP, Qi YM, Cai DC. Research and application of autodesk fusion360 in industrial design. IOP Conf Ser: Mater Sci Eng. 2018;359: 012037. doi:10.1088/1757-899X/359/1/012037
14. Simplify3D. Simplify3D [Internet]. [cited 16 Dec 2020]. Available: https://www.simplify3d.com/
15. Prusa Research. Prusa i3 MK3S 3D printer [Internet]. [cited 17 Dec 2020]. Available: https://www.prusa3d.com/original-prusa-i3-mk3/
16. Kil Y. Assembly Instructions for colosseum protocol guidelines [Internet]. 6 Apr 2021 [cited 6 Apr 2021]. Available: http://dx.doi.org/10.17504/protocols.io.btz3np8n
17. Pololu - DRV8825 Stepper Motor Driver Carrier, High Current [Internet]. [cited 29 Mar 2021]. Available: https://www.pololu.com/product/2133/specs
18. Liechti C. pyserial · PyPI [Internet]. 2020 [cited 6 Apr 2021]. Available: https://pypi.org/project/pyserial/

Supplementary Material

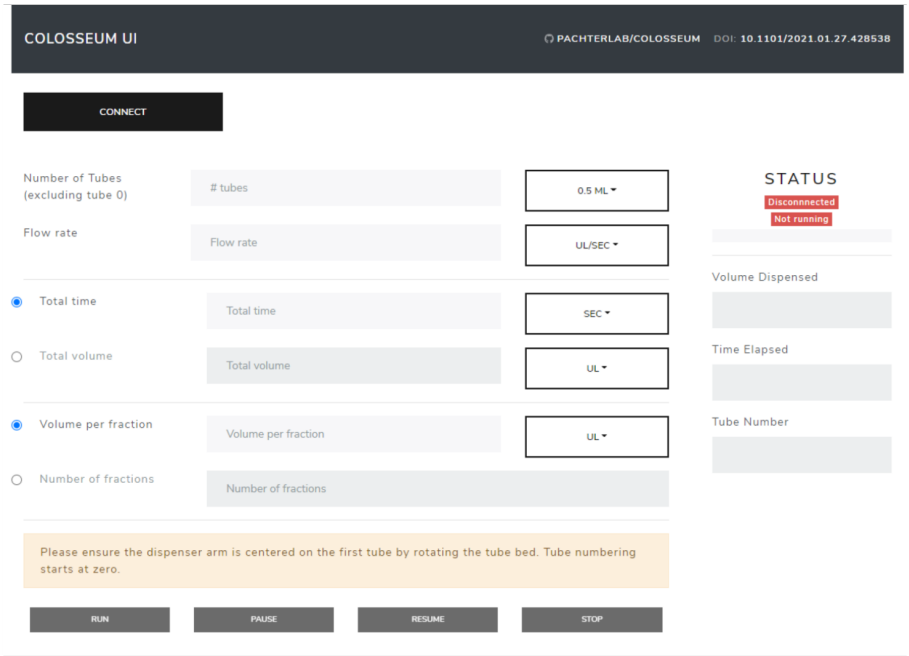


Supplementary Figure 1: Optimal packing of disks of diameter 13.5 mm (11 mm tube hole size plus 2.5 margin) in a disk of diameter 180 mm. The solid line corresponds to the outer diameter of the tube rack, the smaller dashed line corresponds to the effective area available for placing tubes, and the smallest dashed line corresponds to the empty area on the colosseum tube rack where no tubes can be placed. [Code (https://github.com/pachterlab/BKMGP_2021/blob/main/analysis/archimedian_spiral.ipynb)]

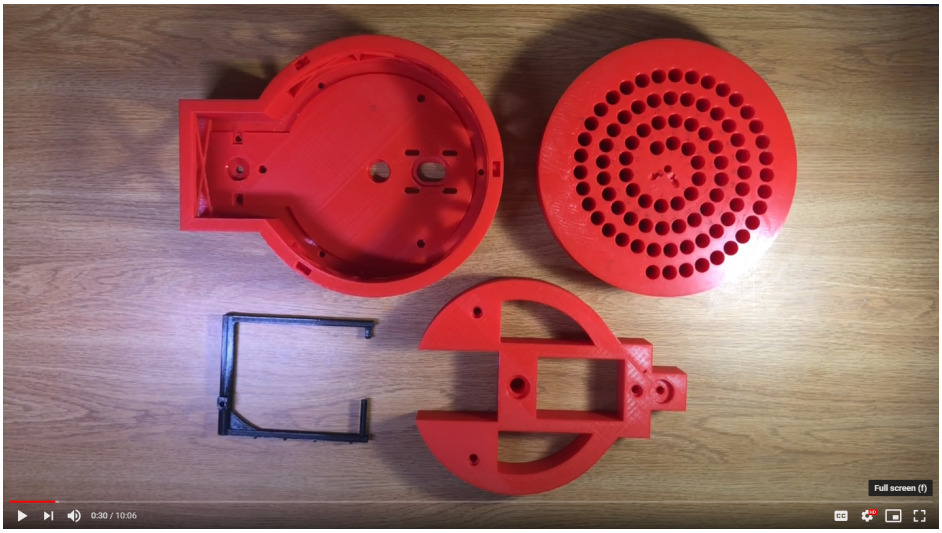
a


colosseum 0.0.5
Inputs
Status
Ready
Number of tubes
0
Flow Rate
uL/sec
Tube Size
0.5mL
Volume Dispensed
0
mL
Setting
Value
Units
Total time
sec
Volume per fraction
uL
Total volume
uL
Number of fractions
Time Elapsed
0
sec
Tube Number
0
Before you run:
Please ensure the dispenser arm is centered on the first tube by rotating the tube bed.
Tube numbering starts at zero.
Run
Pause
Resume
Stop

b


COLOSSEUM UI
PACHTERLAB/COLOSSEUM
DOI: 10.1101/2021.01.27.428538
CONNECT
Number of Tubes
(excluding tube 0)
tubes
0.5 ML
Flow rate
Flow rate
UL/SEC
Total time
Total time
SEC
Total volume
Total volume
UL
Volume per fraction
Volume per fraction
UL
Number of fractions
Number of fractions
STATUS
Disconnnected
Not running
Volume Dispensed
Time Elapsed
Tube Number
Please ensure the dispenser arm is centered on the first tube by rotating the tube bed. Tube numbering starts at zero.
RUN
PAUSE
RESUME
STOP

Supplementary Figure 2: (a) The Python-based graphical user interface (GUI) and (b) the web-based JavaScript GUI. In both GUI's left panel displays input boxes for flow rate and collection parameters and the right panel displays experiment progress.



Supplementary Figure 3: Assembly video of colosseum. This video guides the user step-by-step through the entire assembly process**.** The video is linked to in the GitHub repository https://github.com/pachterlab/colosseum.

Model	Capacity (# tubes)	Price (USD)
Cytiva Frac30 [1]	30	1,615.00
Eldex UFC [2]	135 or 160	3,971.80
Spectrum Spectra FC [3]	174	3,393.00
Buchi C-660 [4]	12, 30, or 60	13,630.11
Open-source	Customizable	<100

Supplementary Table 1: Costs and capacity of commercial fraction collectors. The costs are based on new, unused models. The capacity of each fraction collector is given by how many tubes the device can hold.

Part name	Filament weight [length]	Print time	Supports
Tube Rack	433.80 g [144.281 m]	31 h 16 min	N
Dispenser Arm	18.53 g [6.162 m]	1 h 34 min	Y
Base	271.45 g [90.285 m]	19 h 4 min	Y
Base Plate	174.70 g [58.105 m]	11 h 26 min	N
Total	898.48 g [298.833 m]	73 h 30 min	

Supplementary Table 2: Parts that require 3D printing, including, for each part, the amount of filament (weight and length) required to print, the print time, and whether support is required.

Parameter 1	Flow rate		
Parameter 2	Total time	OR	Total volume
Parameter 3	Volume per fraction	OR	Number of fractions

Supplementary Table 3: Table of input parameters for the GUI. The user must input three parameters: flow rate; total time or total volume; and volume per fraction or number of fractions. The user is limited to three parameters to avoid overconstraining the system.

Tube #	# of 1/4 steps
0	84
1	78
2	75
3	70
4	64
...	...

Supplementary Table 4: The first five rows of the angles between each tube in the tube rack. The angular distances are reported as quarter-steps of the stepper motor. [Data (https://github.com/pachterlab/BKMGP_2021/blob/main/analysis/archimedian_spiral.ipynb)]

Flow rate (mL/hr)	Dwell time (s)
720	5
360	10
180	20
90	40
45	80
22.5	160

Supplementary Table 5: Dwell time for each flow rate. To keep the expected fraction volume at 1 mL the flow rate is halved when the dwell time is doubled.

Supplementary References

1. Cytiva. Frac30 [Internet]. [cited 27 Jan 2021]. Available: https://web.archive.org/save/https://www.cytivalifesciences.com/en/us/shop/chromatography/tools-and-accessories/fraction-collectors-and-accessories/frac30-p-05647#order
2. Amazon. 1243 - UFC Universal Fraction Collector - UFC Universal Fraction Collector, Eldex [Internet]. [cited 27 Jan 2021]. Available: https://web.archive.org/save/https://www.amazon.com/1243-Universal-Fraction-Collector-Eldex/dp/B0731TY8Q8
3. Spectrum. Spectra/Chrom® CF-2 Fraction Collector [Internet]. [cited 27 Jan 2021]. Available: https://web.archive.org/save/https://www.spectrumchemical.com/OA_HTML/lab-supplies-products_SpectraChromsup-174sup-CF-2-Fraction-Collector_302400.jsp?section=23230
4. VWR. Buchi® Sepacore™ C-660 Fraction Collector [Internet]. [cited 27 Jan 2021]. Available: https://web.archive.org/save/https://us.vwr.com/store/product/4637187/buchi-sepacoretm-c-660-fraction-collector

Chapter 3

SOFTWARE

Section 1: *kallisto* | *bustools* sequencing preprocessing

Preamble

Single-cell RNA-sequencing data must be aligned and aggregated in order to generate a *cell* by *gene* matrix for downstream analysis. Current state-of-the-art tools require costly compute resources greatly limiting the scale of processing and reproducible analysis. The *kallisto* | *bustools* command-line tools implement novel counting algorithms for transforming scRNAseq data into *cell* by *gene* count matrices in a fast and memory-efficient manner. These algorithms offer tradeoffs between speed and accuracy that have been benchmarked and validated. Specifically, tradeoffs in degree of barcode error correction, and speed, as well as tradeoffs between ambiguities in sequence alignment, and speed, enable fast, memory efficient, and accurate-enough barcode counting algorithms that outperform the state-of-the–art. These tools overcome a roadblock in a critical step of scRNAseq data preprocessing, a step that precedes all downstream analysis. By developing and validating memory and time-efficient algorithms for sequencing counting, scalable and fast-turnaround scRNAseq experiments can be performed with minimal compute cost.

Summary

We describe a workflow for pre-processing of single-cell RNA-seq data that balances efficiency and accuracy. Our workflow is based on the kallisto and bustools programs, and is near-optimal in speed with a constant memory requirement providing scalability for arbitrarily large datasets. The workflow is modular, and we demonstrate its flexibility by showing how it can be used for RNA velocity analyses.

Introduction

The quantification of transcript or gene abundances in individual cells from a single-cell RNA-seq (scRNA-seq) experiment is a task referred to as pre-processing[1]. The pre-processing steps for scRNA-seq bear some resemblance to those used for bulk RNA-seq[2] and are in principle straightforward: cDNA reads originating from transcripts must be partitioned into groups according to cells of origin, aligned to reference genomes or transcriptomes to determine molecules of origin, and reads originating from PCR-duplicated molecules must be “collapsed” so they are counted only once during

quantification. The collapsing step can be facilitated with unique molecular identifiers (UMIs), which are sequences that serve as barcodes for molecules[3]. The challenges in pre-processing single-cell RNA-seq lie in the tradeoffs that must be considered in determining choices for how, and in which order, to execute the various steps. For example, in droplet-based scRNA-seq protocols, collapsing UMIs to account for PCR duplication can be performed naïvely by associating all reads that align to the same gene, with the same UMI, to a single molecule[4]. This computationally efficient procedure is based on an assumption that all reads with identical UMIs that align to the same gene arise via PCR duplication of a single molecule. Alternatively, this assumption can be relaxed, resulting in a problem formulation that is NP-complete, i.e. computationally intractable to solve optimally[5].

Another challenge in scRNA-seq pre-processing is the amount of data that must be processed. A single cell experiment can generate 10^6 - 10^{10} reads from 10^3 - 10^6 cells[6]. This is leading to bottlenecks in analysis: for example, the current standard program for pre-processing 10x Genomics Chromium scRNA-seq, the Cell Ranger software[7], requires approximately 22 hours to process 784M reads[5] using 1.5 Tb of disk space (Supplementary Table 1). For this reason, a number of new, faster workflows for scRNA-seq pre-processing based on pseudoalignment[8] have recently been developed[5,9]. However, despite improvements in running time, current workflows have memory requirements that increase with data size[5], a situation that is untenable given the pace of improvement in technology and the corresponding increase in data volume.

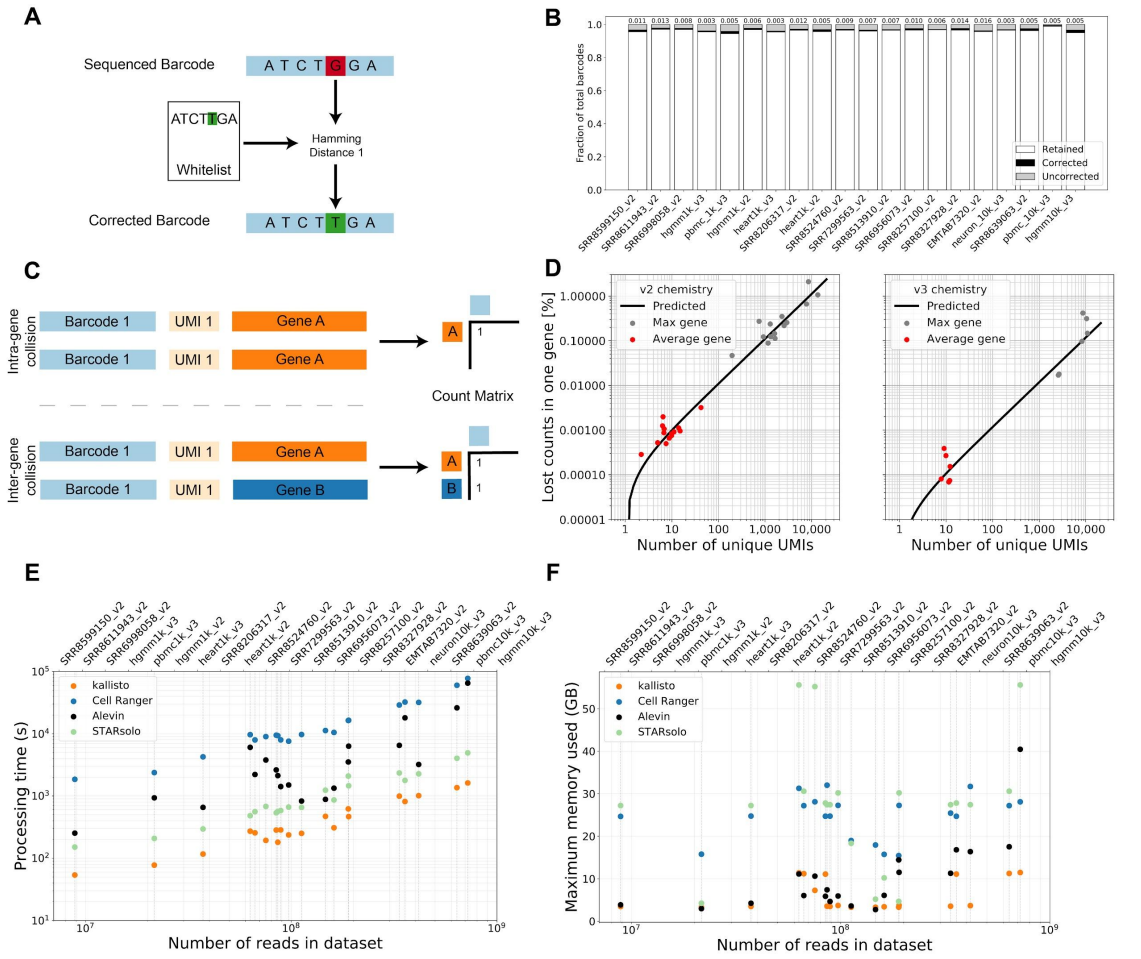


Figure 1: The kallisto bustools workflow. (A) Error correction of barcodes 1 mismatch away from barcodes in a whitelist. (B) Analysis of barcode fidelity n the benchmark panel (20 datasets) showing barcodes matching the whitelist ("retained", white), barcodes that are Hamming distance 1 away from the whitelist that were corrected (black) and uncorrected barcodes (gray). (C) UMI collapsing within genes. (D) Fraction of UMIs lost per gene across cells in the benchmark panel due to over-collapsing. The average number of intra-gene collisions resulting in lost counts due to naive collapsing for the 10 most highly expressed genes across 10xv2 datasets is 0.4% and for 10xv3 datasets 0.17%. The average loss for average expressed genes is about 0.003% (E) Running time of kallisto (orange), Cell Ranger (blue), Alevin (black), and STARsolo (green) for pre-processing the benchmark panel. (F) Memory usage of kallisto (orange), Cell Ranger (blue), Alevin (black) and STARsolo (green) for pre-processing the benchmark panel.

In recent work we introduced a format for single-cell RNA-seq data that makes possible the development of efficient workflows by virtue of decoupling the computationally demanding step of associating reads to transcripts and genes (alignment), from the other

steps required for scRNA-seq pre-processing[10]. This format, called BUS (Barcode, UMI, Set), can be produced by pseudoalignment, and rapidly manipulated by a suite of tools called BUStools (Supplementary Table 2). To illustrate the utility, efficiency, and flexibility of this approach for scRNA-seq pre-processing, we describe a Chromium pre-processing workflow based on reasoned choices for the key pre-processing steps. While we focus on Chromium, our workflow is general and can be used with other technologies. We show that our pre-processing workflow is faster and has lower memory requirements than existing methods, and we demonstrate the power of modular processing with the BUS format by developing a fast RNA velocity analysis workflow[11]. We also validate the design decisions underlying the Cell Ranger workflow. Our benchmarking and testing is comprehensive, comprising analysis of almost two dozen datasets and surpassing the scale of testing that has been performed for current workflows. Documentation and tutorials for the kallisto | bustools workflow are available at https://www.kallistobus.tools/.

Results

In designing a scRNA-seq pre-processing workflow, we began by investigating each required step: correction of barcodes, collapsing of UMIs, and assignment of reads to genes. To achieve single-cell resolution, the Chromium technology produces barcode sequences that are used to associate cDNA reads to individual cells. We began by considering the efficiency-accuracy tradeoffs involved in grouping reads with the same, or similar, barcodes to define the contents of individual cells. The Chromium barcodes arise from a "whitelist", a set of pre-defined sequences that are included with the Cell Ranger software. Grouping reads by barcode is therefore straightforward, except for the fact that barcodes may contain sequencing errors. The Cell Ranger workflow corrects all barcodes that are one base-pair change away (Hamming distance 1) from barcodes in the whitelist. An examination of a benchmark panel of 20 datasets revealed that this error correction approach can be expected to rescue, on average, 0.8% of the reads in an experiment (Figure 1a,b), a calculation based on an inferred error rate per base for each dataset (Methods, Supplementary Table 3). Thus, correction of barcodes Hamming distance 2 away from whitelist barcodes would rescue, on average, a negligible number (0.0038%) of reads (Methods). We therefore implemented a Hamming distance 1 correction method in our workflow via the **bustools correct** command.

Next, in considering how to count UMIs to generate count matrices we first investigated the extent to which "collisions" occur, i.e. cases where the same UMI occurs in reads originating from two different molecules[12](Figure 1c). While inter-gene collisions can be directly measured, intra-gene collisions cannot be distinguished from molecules that are PCR duplicates. To estimate the intra-gene collision rate we first calculated, for each cell

in the benchmark panel, the effective number of UMIs in each of the associated droplets (Supplementary Figure 1, Supplementary Note). This estimate, along with the number of inter-gene collisions and distinct UMIs observed, allowed us to estimate the extent of intra-gene collision, and therefore the counts lost due to naïve collapsing of UMIs by gene (Methods, Supplementary Note). We found that the average number of intra-gene collisions resulting in lost counts due to naive collapsing for the 10 most highly expressed genes across the 10xv2 datasets is 0.4%, and 0.17% for the 10xv3 datasets. The average percentage of lost counts per gene per cell due to naïve collapsing was less than 0.003% for v2 chemistry and 0.000048% for v3 chemistry (Figure 1d). Thus, we decided to apply naïve collapsing as it is computationally efficient and effective based on empirical evidence. This was implemented in the **bustools count** command. Notably, the recently published Alevin collapsing algorithm[5] will overestimate gene counts because reads with the same UMI pseudoaligned to the same gene are very likely to be from the same molecule even if they pseudoalign to distinct transcripts. Such situations likely result from missing or incorrect annotation[13] rather than from collisions of two distinct molecules labeled with the same UMI.

One implication of the UMI collapsing analysis is that UMI error correction is possible because UMIs with only one base-pair change away from an abundant UMI are likely to have resulted from sequencing error. To examine the benefit of such a correction we computed the expected number of UMIs that would be corrected with Hamming distance 1 correction, and found that for 10bp and 12bp UMIs only 0.5% and 0.6% of reads would be recovered, respectively, at the error rates observed in published datasets (Methods, Supplementary Figure 2). Moreover, such error correction would require identification of abundant UMIs in lieu of a whitelist, adding time and complexity to the workflow. While we believe such error correction may be warranted in the case of longer UMIs (Supplementary Figure 2), we did not include it in our workflow.

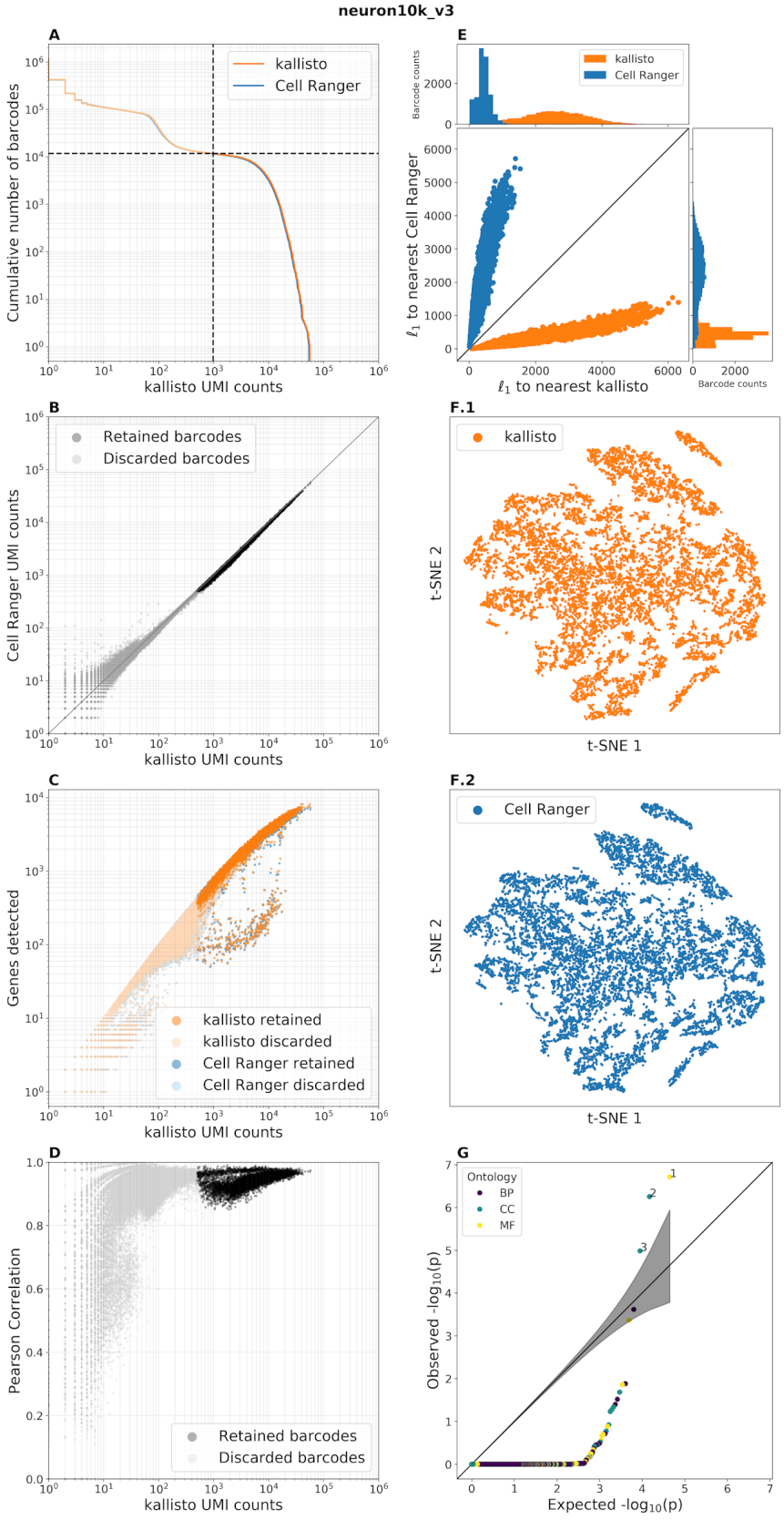

neuron10k_v3
A
kallisto
Cell Ranger
Cumulative number of barcodes
kallisto UMI counts
E
kallisto
Cell Ranger
Barcode counts
ℓ_1 to nearest Cell Ranger
ℓ_1 to nearest kallisto
Barcode counts
B
Retained barcodes
Discarded barcodes
Cell Ranger UMI counts
kallisto UMI counts
F.1
kallisto
t-SNE 2
t-SNE 1
C
kallisto retained
kallisto discarded
Cell Ranger retained
Cell Ranger discarded
Genes detected
kallisto UMI counts
F.2
Cell Ranger
t-SNE 2
t-SNE 1
D
Retained barcodes
Discarded barcodes
Pearson Correlation
kallisto UMI counts
G
Ontology
BP
CC
MF
Observed -log10(p)
Expected -log10(p)

Figure 2: 10x Genomics *Mus Musculus benchmark comparison*. Darker points\lines are retained barcodes and lighter points\lines are discarded barcodes. (A) "Knee plots" for kallisto and Cell Ranger showing, for a given UMI count (x-axis), the number of cells that contain at least that many UMI counts (y-axis). The dashed lines correspond to the Cell Ranger filtered cells. (B) Correspondence in the number of distinct UMIs per cell between the workflows. (C) Genes detected by kallisto and Cell Ranger as a function of distinct UMI counts per cell. (D) Pearson correlation between gene counts as a function of the distinct UMI counts per cell. (E) The l_1 distance between gene abundances for each kallisto cell and its nearest neighbor plotted against each kallisto cell and its corresponding Cell Ranger cell (orange) and the l_1 distance between the gene abundances for each Cell Ranger cell and its nearest neighbor plotted against each Cell Ranger cell and its corresponding kallisto cell (orange). Marginal distributions show that each kallisto cell is closest to its corresponding Cell Ranger cell and that each Cell Ranger cell is closest to its corresponding kallisto cell. (F.1) kallisto t-SNE from the first 10 principal components. (F.2) Cell Ranger t-SNE from the first 10 principal components. (G) QQ plot comparing the distribution of observed distribution of p-values of GSEA, after Bonferroni correction for multiple testing across ontologies and datasets, with the expected distribution of a uniform distribution between 0 and 1. If the observed distribution does not significantly deviate from the expected distribution, then the points should lie close to the diagonal line, $y = x$. The gray ribbon around the line is the 95% confidence interval. Here most GO terms have adjusted $p = 1$, meaning that most GO terms are very depleted of genes "differentially expressed (DE)" between the kallisto and Cell Ranger matrices. GO terms above $y = x$ are labeled. Generally, GO terms significantly enriched among "DE" genes are related to ribosomal proteins, specifically the GO terms 1, 2, 3 correspond to structural constituent of ribosome, cytosolic large ribosomal subunit, and cytosolic small ribosomal subunit. The points are colored by ontology: biological processes (BP), cellular components (CC), and molecular functions (MF).

In most scRNA-seq pre-processing workflows, assignment of cDNAs to genes utilizes genome alignment[4,14,15]. Since detailed base-pair alignment is not necessary to generate a count matrix, pseudoalignment to a reference transcriptome[8] suffices. Moreover, pseudoalignment has been shown to be highly concordant with alignment for the purposes of quantification in bulk RNA-seq[16]. To test this hypothesis we compared counts obtained by pseudoalignment using the kallisto program[8] with counts produced via Cell Ranger which is based on the STAR aligner[17]. Analysis of a *Mus Musculus* scRNA-seq dataset (Figure 2), confirms that there is a high correlation between pseudoalignment and alignment based counting (see Supplementary Figure 3 for other datasets), however in one dataset (pbmc10k_v3, Supplementary Figure 3.19) we found that pseudoalignment

produced more counts than alignment. Specifically, in the FGF23 (ENSG00000118972) gene, Cell Ranger had many fewer counts than kallisto. We hypothesized that the reason for this discrepancy was the presence of reads from unspliced transcripts crossing splice junction boundaries, and therefore being erroneously pseudoaligned to the transcriptome. To test this, we created a modified index that included a 90 base-pair overlap into the exon and the intron (one base-pair less than the length of the reads) to capture such reads and confirmed that it resolved the discrepancy (Supplementary Figure 4). We observed this problem to be rare and therefore did not deem it to be essential in a standard processing workflow. It may be that consideration of such reads will be crucial for nuclear scRNA-seq analyses[18], when the abundance of such intronic junction reads will be problematic for naïve pseudoalignment.

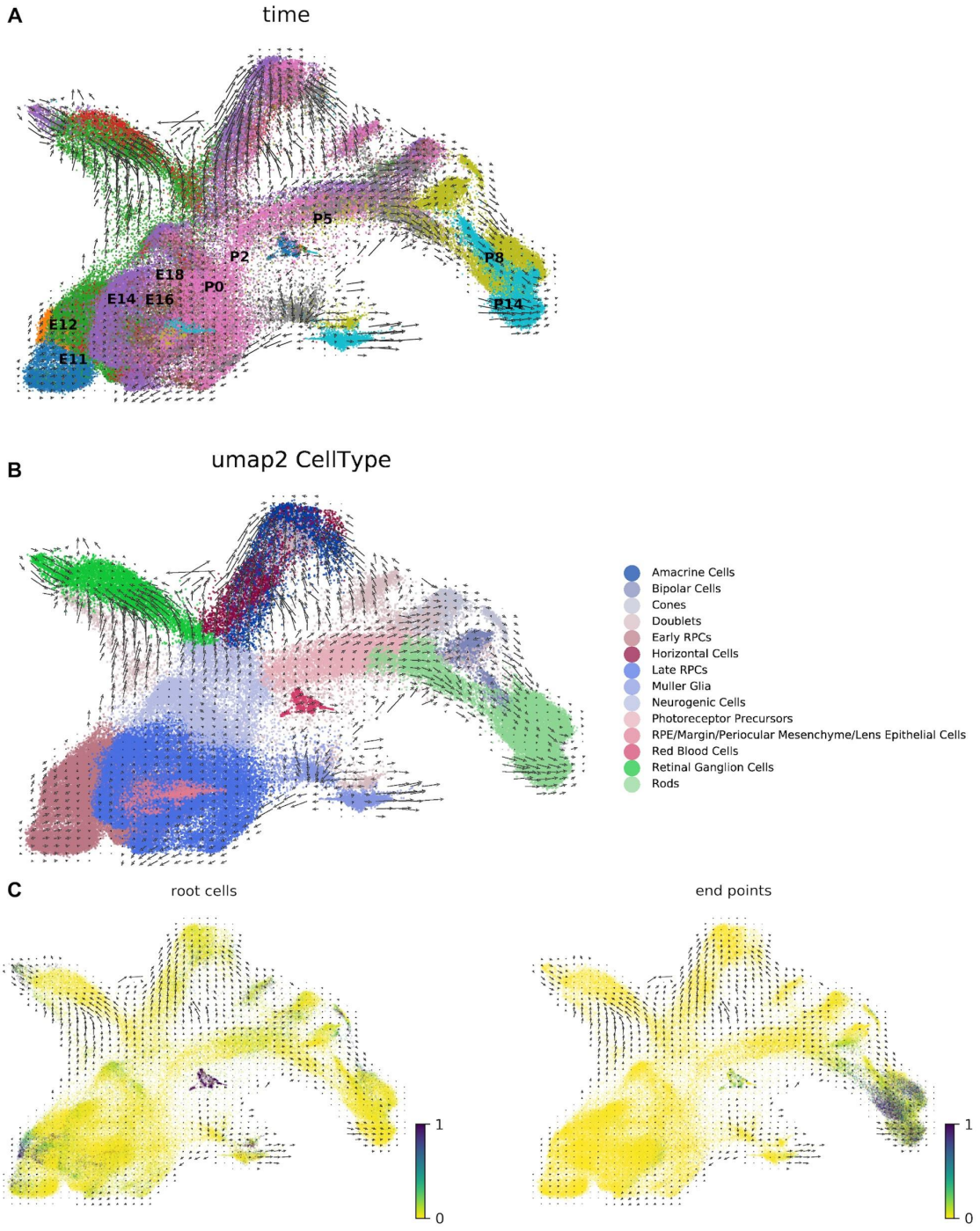


Figure 3: RNA velocity. (A) A kallisto and bustools based RNA velocity analysis of the ten stage scRNA-seq retina neurogenesis data from Clark et al. 2019[32]. (B) Clusters annotated by cell type according to Clark et al. (C) Markov diffusion process analysis highlighting source and sink cells and demonstrating that the velocity vector field is consistent with the cells' developmental trajectory.

Thus, our workflow consists of pseudoalignment of reads to a reference transcriptome to generate a BUS (barcode, UMI, set) file, and subsequent processing to correct barcode errors and produce a count matrix (Supplementary Figure 5). To ensure that memory usage is constant in the number of reads, the BUS files are sorted by barcode prior to counting using the **bustools sort** command (Supplementary Note).

While this workflow is very similar to that of Cell Ranger, it is not identical. Since Cell Ranger is widely used, we investigated the extent to which the Cell Ranger results are concordant with our workflow. We processed 20 datasets (Supplementary Table 3), chosen to contain a range of reads depths (from 8,860,361 to 721,180,737 reads per sample and 2,243 to 201,952 reads per cell) and to represent scRNA-seq from a range of tissues and species (*Arabidopsis thaliania*[19], *Caenorhabditis elegans*[20], *Danio rerio*[21], *Drosophila melanogaster*[22], *Homo sapiens*[23,24], *Mus musculus*[25–29], *Rattus norvegicus*[29]). We found a high degree of concordance with respect to quality control metrics (Figure 2a—g, Supplementary Figure S3). Crucially, in all datasets, in a joint analysis of kallisto and Cell Ranger counts, the closest cell to a kallisto cell was its associated Cell Ranger cell, i.e. the Cell Ranger cell with the same barcode sequence. Furthermore, gene count correlations between individual cells passing Cell Ranger filtering criteria were almost always above 0.90, and frequently as high as 0.99.

To assess the extent to which differences between Cell Ranger and kallisto affect biological inferences, we also compared Cell Ranger to kallisto results in a downstream analysis of the 10x Genomics E18 mouse 10k brain cells dataset. We found that pseudotime trajectory inference for neuronal precursor cells produced highly concordant results between Cell Ranger and kallisto, with the same trajectory topology and similar pseudotime values along the trajectory (Supplementary Figure 6). This result is consistent with other analysis comparisons. Projections of Cell Ranger and kallisto cells to the first two principal components (PCs) and to two dimensions of tSNE are very similar (Supplementary Figure 7.1). The results of Leiden clustering[30] are also similar between the two pre-processing workflows (Supplementary Figure 7.2). We performed differential expression (DE) analysis to identify marker genes of the clusters, and then performed gene set enrichment analysis (GSEA) on the marker genes for cell type annotation (Supplementary Figure 7.3). The marker genes and their corresponding gene sets were highly correlated between the workflows. In both Cell Ranger and kallisto results, most clusters are neuronal, and the clusters for erythrocytes (cluster 16 in both), endothelial cells (cluster 21 in kallisto, cluster 19 in Cell Ranger), and immune cells (clusters 20 and 22 in kallisto cluster 17 in Cell Ranger) can be clearly identified based on marker genes (Supplementary Figure 7.3). Correlation between the same barcodes in kallisto and Cell

Ranger with the top cluster marker genes is very high, with both the Pearson and Spearman correlation coefficient above 0.9 for the vast majority of cells (Supplementary Figure 7.4). In a separate mixed species dataset, the number and proportion of UMIs from human and mouse cells are similar between Cell Ranger and kallisto (Supplementary Figure 8). Overall, these results suggest that the Cell Ranger workflow produces results consistent with our method, not only at the level of dataset summary statistics, but also in downstream analyses.

The modularity of our approach makes possible the rapid implementation of alternative workflows. To illustrate this, we developed an RNA velocity workflow. By including intron sequences in the index for pseudoalignment we were able to identify reads originating from unspliced transcripts, and, using the **bustools capture** command, created the spliced and unspliced matrices needed for RNA velocity. Our RNA velocity workflow, which is 13 times faster than velocyto[11] analysis of the same dataset, is suitable for large datasets that were previously challenging to pre-process. To illustrate this we computed RNA velocity vectors for recently published data from the developing mouse retina[31] consisting of 113,917 cells (Figure 3). We found that six pseudotime marker genes highlighted in Clark et al. 2019[32] (Crx, Nrl, Otx2, Pax6, Rbpms, Rlbp1) displayed patterns consistent with the RNA velocity vectors, and with the pseudotime analysis of Clark et al.[32] (Supplementary Figure 9). The velocity analysis reveals new information, namely it identifies developmental states when RNA velocity is changing (Supplementary Figure 9 middle column). We verified the fidelity of our workflow by computing RNA velocity vectors on a dataset from La Manno et al. 2018[11] and comparing our results to those of the paper (Methods, Supplementary Figure 10). Furthermore, the spliced count matrix agreed with the count matrix obtained in our standard gene expression workflow (Supplementary Figure 11). Despite identifying many more unspliced counts, our resultant velocity figure was concordant with that of La Manno et al.[11].

Our scRNA-seq workflow is up to 51 times faster than Cell Ranger and up to 4.75 times faster than Alevin. It is also up to 3.5 times faster than STARsolo: a recent version of the STAR aligner adapted for scRNA-seq (Figure 1e, Supplementary Table 1). In benchmarks on the panel described in this paper, kallisto bustools running time was comparable to that of the word count (**wc**) command applied to the FASTQ files, suggesting that the kallisto bustools workflow is near-optimal in efficiency (Supplementary Figure 12). Unlike Alevin, our workflow requires a small fixed amount of constant memory that is independent of the number of reads being pre-processed (Figure 1f). The limiting memory constraint is therefore the size of the reference index, which is under 4Gb of RAM for the human transcriptome, and thus our method is

suitable for low-cost and environmentally conscious cloud computing. In cases where it is necessary to work with very large indices on small memory instances, memory needs can be reduced using an index splitting strategy (Supplementary Note), albeit with fewer resulting read counts (Supplementary Figures 13 and 14). Our speed and constant memory requirements make RNA velocity tractable for datasets of any size for the first time.

Our UMI collapsing analysis suggests that UMI sequences can be short; even just 6 base-pairs of sequence can suffice for identifying most molecules thanks to the cell barcode and gene identification for each read serving as auxiliary barcodes (Supplementary Figure 15). Furthermore, the fact that identical UMIs associated with distinct reads from the same gene are almost certainly reads from the same molecule (Figure 1d), makes it possible, in principle, to design assignment algorithms for multi-mapping reads. Reads could be assigned with an expectation-maximization algorithm which is based on estimating the copy number of each molecule in the library using a model as described in the Supplementary Note, and this is a promising direction for future work. An initial attempt at such assignment[5] appears to improve concordance between single-cell RNA-seq gene abundance estimates and those from bulk RNA-seq. The current implementation of our approach can produce transcript compatibility counts which have information about read ambiguity prior to assignment of multi-mapping reads, and can therefore be used to identify isoform-specific changes across cells and cell clusters [33].

Discussion

While we have focused on a workflow for 10x Chromium data, the bustoools commands we implemented are generic and will work with any BUS file, generated with data from any scRNA-seq technology. Distinct technologies encode barcode and UMI information differently, but the **kallisto bus** command can accept custom formatting rules. While the pre-processing steps for error correction and counting may need to be optimized for the distinguishing characteristics of different technologies, the modularity of the bustools based workflow makes such customization possible and easy.

Methods

Data Availability

A diverse set of 20 datasets was compiled for the purpose of benchmarking pre-processing workflows. Datasets produced and distributed by 10x Genomics were downloaded from the 10x Genomics data downloads page: https://support.10xgenomics.com/single-cell-gene-expression/datasets. Six v3 chemistry

datasets and two v2 chemistry datasets were downloaded and processed (Supplementary Table 3). Another 12 datasets were obtained from either the SRA or the ENA; all were produced with 10x Genomics v2 chemistry. For six of the datasets (SRR6956073, SRR6998058, SRR7299563, SRR8206317, SRR8327928, SRR8524760) the BAM files were downloaded and the Cell Ranger utility bamtofastq was run to produce fastq files for pre-processing from Cell Ranger structured BAM files. FASTQ files were downloaded directly for the datasets EMTAB7320, SRR8257100, SRR8513910, SRR8599150, SRR8611943, SRR8639063.

Details of all datasets and their accession numbers can be found in Supplementary Table 3. All genome annotations and reference transcriptomes can be found here http://dx.doi.org/10.22002/D1.1876.

Code Availability

The software versions used for the results in the paper were: Alevin v0.13.1, bustools v0.39.1, Cell Ranger v3.0.0, DropletUtils v1.6.1, kallisto v0.46.0, python 3.7, R v3.5.2, Scanpy v1.4.1, scvelo 0.1.17, Seurat v3.0, snakemake v5.3.0, STARsolo v2.7.0e, velocyto v0.17.17, wc v8.22 (GNU coreutils), and zcat v1.5 (gzip). All programs were run with default options unless otherwise specified. The code to reproduce this paper is available at https://github.com/pachterlab/MBLGLMBHGP_2021/, kallisto is available at https://github.com/pachterlab/kallisto/ and bustools is available at https://github.com/BUStools/bustools/. Documentation and tutorials for using the kallisto | bus single-cell RNA-seq workflow are available at https://www.kallistobus.tools/.

Hardware

All the benchmarks were carried out on a Supermicro server computer (2xXeon® Gold 6152 22-Core 2.1, 3.7GHz Turbo, 12 x 64GB Quad-Rank DDR4 2666MHz memory, 16 x 12TB Ultrastar He12 HUH721212ALE600, 7200 RPM, SATA 6Gb/s HDD) with CentOS7 operating system installed. The running time of all programs were evaluated using eight threads.

Transcriptome indices

Reference transcriptomes were constructed by processing datasets with Cell Ranger, downloading the constructed Cell Ranger GTF file, and then producing a transcriptome from it and the relevant genome using GFFread (http://cole-trapnell-lab.github.io/cufflinks/file_formats/#the-gffread-utility).

Inference of per-base sequencing error rate and correctable barcodes

For each dataset, the per-base error rate p was estimated by the formula

$$\hat{p} = \frac{e}{16u + e},$$

where u and e are realizations of random variables U and E, u was the number of barcodes matching the whitelist, e the number of barcodes hamming distance 1 away from a whitelist barcode, and 16 is the length of 10xv2 and 10xv3 barcodes. Letting U, E, and T be random variables representing the number of barcodes matching the whitelist, the number of barcodes hamming distance one from the whitelist, and the effective total number of barcodes respectively, the previous equation was derived by solving for p using

$$(1-p)^{16} = E\left[\frac{U}{T}\right] \text{ and } 16p(1-p)^{15} = E\left[\frac{E}{T}\right].$$

To estimate the proportion of barcodes Hamming distance 2 or greater away from a whitelist barcode, we computed

$$1 - (1-\hat{p})^{16} - 16\hat{p}(1-\hat{p})^{15}$$

for each dataset, using the estimated per-base error rate $\hat{p}$.

UMI collision estimates

A UMI associated with a read from a cell is said to have "collided" if it appears in two or more reads originating from different molecules. To estimate UMI collision rates, two types of information were used. First, reads in the same cell that originate from different genes must have originated from different molecules, and therefore the sharing of a UMI between two such reads was used as an indicator of a collision. Second, for each gene, the number of distinct UMIs associated within it was measured from the data. Based on the assumption that UMIs were sampled uniformly at random from beads, this data was used to estimate the number of intra-gene collisions (see Supplementary Note). The assumption was verified by examining the distribution of UMI counts across cells; the empirical distribution was near-uniform with the exception of a handful of UMIs (Supplementary Note Figure 2).

Comparative analysis of the benchmark panel datasets

The benchmark panel datasets (Supplementary Table 3) were processed uniformly as follows:

For each dataset a "knee plot"[34] was constructed for both Cell Ranger and kallisto by plotting, for each cell, the number of distinct UMIs in the cell vs. the number of barcodes

with at least that number of UMIs. Then, the distinct number of UMIs for kallisto and Cell Ranger were plotted against each other. Subsequently, for each cell, the number of distinct UMIs was plotted against the number of genes detected. Finally, the Pearson correlation was computed between the gene counts of kallisto and Cell Ranger for each cell.

To investigate the similarity of Cell Ranger to kallisto, the l_1 distance between each corresponding kallisto and Cell Ranger cell was computed. The distance to the nearest kallisto cell was also measured. To visualize the Cell Ranger and kallisto count matrices, t-SNE was performed on the data projected to the 10 principal components computed for each dataset using the opentSNE package (https://github.com/pavlin-policar/openTSNE/) with perplexity=30, metric="Euclidean", random_state=42, n_iter=750.

To check for systematic differences in the quantification of certain genes between Cell Ranger and kallisto, a differential expression analysis was performed on the matrices produced by the two workflows. First, the matrices were concatenated using the genes determined to be expressed in both methods. Then the counts were normalized using Seurat. DE was performed with logistic regression. Next GSEA with the R package topGO on all marker genes with adjusted p-value less than 0.05, to identify classes of genes more likely to be affected by the different workflows. Parameters of topGO are statistic = "fisher", algorithm = "weight01", and the gene universe is all genes detected in both the kallisto and the Cell Ranger matrices. For mixed species datasets (mouse and human), the gene universe used to test mouse GO terms was all mouse genes observed in both kallisto and Cell Ranger matrices, and the gene universe for human GO terms was all human genes observed in both matrices. For each dataset, the ontologies biological processes (BP), cellular components (CC), and molecular function (MF) were tested separately. According to the vignette of topGO, the test for each of the ontology with correction of network topology should be considered to be unaffected by multiple testing, as different GO terms are not independent. However, since the tests for different ontologies and datasets were independent, we applied Bonferroni correction to adjust the p-values for this. Since each single species dataset (17 in total) was tested for 3 ontologies, and each mixed species dataset (3 in total) was tested for 6 ontologies, the p-values were multiplied by 69 for 69 independent tests on ontologies performed.

Comparative analysis of the 10x Genomics E18 Mouse dataset

Analysis of the Cell Ranger and kallisto pre-processed datasets was performed in R. The DropletUtils package[35,36] was used to remove empty droplets from the kallisto gene count matrix. For Cell Ranger, the filtered matrix was used. After filtering, genes not detected in any remaining Cell Ranger or kallisto barcode were removed. Seurat was used for basic analysis. First, data was normalized by dividing the UMI count of each gene in each cell by the total UMI counts of that cell, multiplied this number by 10000. Then a

pseudocount of 1 was added, and the natural log transform was applied. Subsequently, the normalized data was scaled so the distribution of the expression of each gene would have mean of 0 and standard deviation of 1. Subsequently, 3,000 highly variable genes were selected with the vst method in Seurat. Then principal component analysis was performed on the highly variable genes in the scaled data with the R package irlba called by Seurat. The first 40 principal components were used for tSNE, which was done with the R package Rtsne called by Seurat. Clustering was performed with the Leiden algorithm [30] on the kallisto and Cell Ranger matrices. The clustering parameters were 20 nearest neighbors and resolution 1. Differential expression analysis was performed with the logistic regression method described in Ntranos et al.[33] as implemented in Seurat and applied to the normalized (unscaled) data. Spearman and Pearson correlations were computed for the top 15 cluster marker genes. Gene Set Enrichment Analysis(GSEA) was performed on cluster marker genes with adjusted $p < 0.05$ using the R package topGO[37] with the gene ontology (GO)[38,39] annotations provided by Bioconductor 3.10. SingleR[40] was used to annotate cell types based on correlation profiles with bulk RNA-seq from[41]. Then, the neuronal cell types were used for pseudotime analysis. Pseudotime analysis was done with slingshot[42] via the Docker container from dyno[43].

Species mixing

The 10x Genomics 10k 1:1 Mixture of Fresh Frozen Human (HEK293T) and Mouse (NIH3T3) Cells dataset was analyzed with kallisto and Cell Ranger for the purpose of comparing the resultant banyard plots[44]. Human and mouse genes were identified with their ENSEMBL identifiers. The total number of UMIs mapped to the human and mouse genes in each barcode was calculated with the unfiltered matrices. In Supplementary Figure 8b,c only barcodes present in both the kallisto and Cell Ranger unfiltered matrices were used.

RNA velocity

A human reference transcriptome FASTA file of exonic transcripts and a reference genome fasta were obtained from the UCSC Genome Browser, build Dec. 2013 GRCh38. A BED file of intronic transcripts, with an $L - 1$ (L = read length) flanking sequence added to each end, was also obtained from the UCSC Genome Browser. A unique number was appended to the end of each intronic transcript in the BED file. The genome fasta and the intronic BED file were used with **bedtools getfasta** to construct an intronic fasta file. The intronic and exonic fasta files were combined and an index was built with **kallisto index**. The reads were aligned to the index using **kallisto bus**. The barcodes in the resultant BUS file were error corrected with **bustools correct** and then sorted with **bustools sort**. To isolate the intronic counts and exonic counts for each barcode, **bustools capture** was ran twice: once using the list of intronic transcripts and once using the list of

exonic transcripts. The spliced count matrices were made by using **bustools count** on the intron-captured split.bus file, and the unspliced count matrices were made by using **bustools count** on the exon-captured split.bus file. Both matrices were loaded into an annotated data frame in a jupyter notebook for downstream analysis.

To perform the comparison to the La Manno et al. 2018 dataset (Supplementary Figures 10, 11), the data was first downloaded from the SRA (SRP129388). The cell barcodes were filtered by those in La Manno et al. 2018[11]. The data was pre-processed with the kallisto | bustools workflow. The matrices were loaded into python. The cluster labels were then transferred from La Manno et al. 2018. The velocyto notebook provided with the paper was used to reproduce the results based on the Cell Ranger\Velocyto matrices and the kallisto | bustools matrices.

For the Clark et al. 2019 RNA velocity analysis (Figure 3, Supplementary Figure 9), the data was downloaded from the SRA (GSE118614). First the cell barcodes were filtered by those in Clark et al. 2019. The cluster labels were then transferred from Clark et al. 2019 [32] and the standard velocity pipeline from scvelo was run using the kallisto spliced and unspliced matrices.

Reproducibility

Tutorials for performing multiple types of single-cell analysis are available at https://www.kallistobus.tools/tutorials in the form of Google Colab notebooks. Users can run many different types of single-cell RNA-seq workflows from standard count matrix generation to RNA velocity. Google Colab serves a jupyter notebook that runs on free cloud compute infrastructure allowing anyone with a Google account to run the notebooks. The efficiency and low memory requirements of the kallisto bustools pipeline make it possible to run on this infrastructure, and this is not possible with other single-cell RNA-seq pre-processing programs due to their higher computational requirements.

Acknowledgments

We thank Vasilis Ntranos and Valentine Svensson for helpful suggestions and comments. We thank Jeff Farrell for the *Danio rerio* gene annotation used to process SRR6956073, John Schiefelbein for the *Arabidopsis thaliana* gene annotation used to process SRR8257100, Justin Fear for the *Drosophila melanogaster* gene annotation used to process SRR8513910, and Junhyong Kim and Qin Zhu for the *Caenorhabditis elegans* gene annotation used to process SRR8611943. We thank Julien Roux for suggesting the analysis in Supplementary Figure 11. The benchmarking work was made possible, in part, thanks to support from the Beckman Institute Caltech Bioinformatics Resource Center.

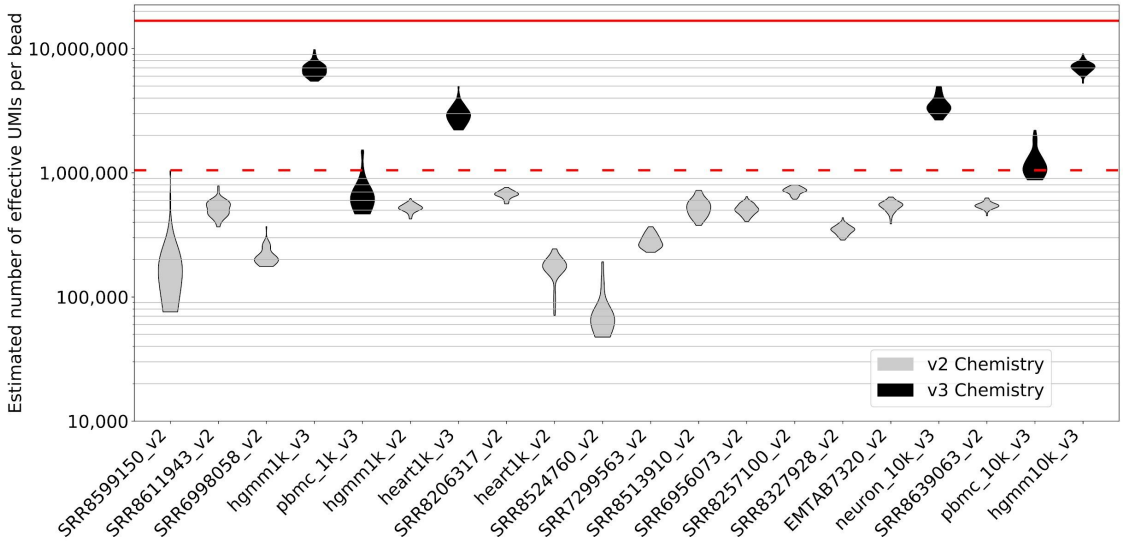
References

1. Tian, L. *et al.* scPipe: A flexible R/Bioconductor preprocessing pipeline for single-cell RNA-sequencing data. *PLoS Comput. Biol.* **14,** e1006361 (2018).
2. Conesa, A. *et al.* A survey of best practices for RNA-seq data analysis. *Genome Biol.* **17,** 13 (2016).
3. Kivioja, T. *et al.* Counting absolute numbers of molecules using unique molecular identifiers. *Nat. Methods* **9,** 72–74 (2011).
4. Parekh, S., Ziegenhain, C., Vieth, B., Enard, W. & Hellmann, I. zUMIs - A fast and flexible pipeline to process RNA sequencing data with UMIs. *Gigascience* **7,** (2018).
5. Srivastava, A., Malik, L., Smith, T., Sudbery, I. & Patro, R. Alevin efficiently estimates accurate gene abundances from dscRNA-seq data. *Genome Biol.* **20,** 65 (2019).
6. Svensson, V., Vento-Tormo, R. & Teichmann, S. A. Exponential scaling of single-cell RNA-seq in the past decade. *Nat. Protoc.* **13,** 599–604 (2018).
7. Zheng, G. X. Y. *et al.* Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.* **8,** 14049 (2017).
8. Bray, N. L., Pimentel, H., Melsted, P. & Pachter, L. Near-optimal probabilistic RNA-seq quantification. *Nat. Biotechnol.* **34,** 525–527 (2016).
9. Svensson, V. *et al.* Power analysis of single-cell RNA-sequencing experiments. *Nat. Methods* **14,** 381–387 (2017).
10. Melsted, P., Ntranos, V. & Pachter, L. The barcode, UMI, set format and BUStools. *Bioinformatics* **35,** 4472–4473 (2019).
11. La Manno, G. *et al.* RNA velocity of single cells. *Nature* **560,** 494–498 (2018).
12. Petukhov, V. *et al.* dropEst: pipeline for accurate estimation of molecular counts in droplet-based single-cell RNA-seq experiments. *Genome Biol.* **19,** 78 (2018).
13. Hayer, K. E., Pizarro, A., Lahens, N. F., Hogenesch, J. B. & Grant, G. R. Benchmark analysis of algorithms for determining and quantifying full-length mRNA splice forms from RNA-seq data. *Bioinformatics* **31,** 3938–3945 (2015).
14. Hwang, B., Lee, J. H. & Bang, D. Single-cell RNA sequencing technologies and bioinformatics pipelines. *Exp. Mol. Med.* **50,** 96 (2018).
15. Ding, J. *et al.* Systematic comparative analysis of single cell RNA-sequencing methods. *BioRxiv* (2019). doi:10.1101/632216
16. Yi, L., Liu, L., Melsted, P. & Pachter, L. A direct comparison of genome alignment and transcriptome pseudoalignment. *BioRxiv* (2018). doi:10.1101/444620
17. Dobin, A. *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29,** 15–21 (2013).
18. Habib, N. *et al.* Massively parallel single-nucleus RNA-seq with DroNc-seq. *Nat. Methods* **14,** 955–958 (2017).

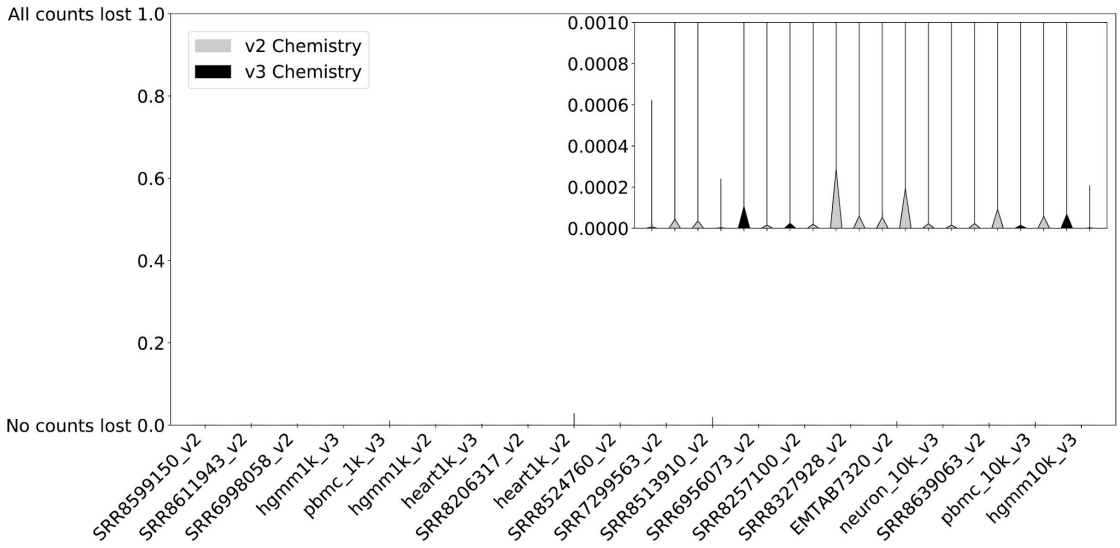
19. Ryu, K. H., Huang, L., Kang, H. M. & Schiefelbein, J. Single-Cell RNA Sequencing Resolves Molecular Relationships Among Individual Plant Cells. *Plant Physiol.* **179,** 1444–1456 (2019).
20. Packer, J. S. *et al.* A lineage-resolved molecular atlas of C. elegans embryogenesis at single-cell resolution. *Science* **365,** (2019).
21. Farrell, J. A. *et al.* Single-cell reconstruction of developmental trajectories during zebrafish embryogenesis. *Science* **360,** (2018).
22. Mahadevaraju, S., Fear, J. M. & Oliver, B. GEO Accession viewer. (2019). at <https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE125948>
23. Carosso, G. A. *et al.* Precocious neuronal differentiation and disrupted oxygen responses in Kabuki syndrome. *JCI Insight* **4,** (2019).
24. Merino, D. *et al.* Barcoding reveals complex clonal behavior in patient-derived xenografts of metastatic triple negative breast cancer. *Nat. Commun.* **10,** 766 (2019).
25. O'Koren, E. G. *et al.* Microglial Function Is Distinct in Different Anatomical Locations during Retinal Homeostasis and Degeneration. *Immunity* **50,** 723-737.e7 (2019).
26. Jin, R. M., Warunek, J. & Wohlfert, E. A. Chronic infection stunts macrophage heterogeneity and disrupts immune-mediated myogenesis. *JCI Insight* **3,** (2018).
27. Miller, B. C. *et al.* Subsets of exhausted CD8+ T cells differentially mediate tumor control and respond to checkpoint blockade. *Nat. Immunol.* **20,** 326–336 (2019).
28. Delile, J. *et al.* Single cell transcriptomics reveals spatial and temporal dynamics of gene expression in the developing mouse spinal cord. *Development* **146,** (2019).
29. Guo, L. *et al.* Resolving Cell Fate Decisions during Somatic Cell Reprogramming by Single-Cell RNA-Seq. *Mol. Cell* **73,** 815-829.e7 (2019).
30. Traag, V. A., Waltman, L. & van Eck, N. J. From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9,** 5233 (2019).
31. Clark, B. *et al.* Comprehensive analysis of retinal development at single cell resolution identifies NFI factors as essential for mitotic exit and specification of late-born cells. *BioRxiv* (2018). doi:10.1101/378950
32. Clark, B. S. *et al.* Single-Cell RNA-Seq Analysis of Retinal Development Identifies NFI Factors as Regulating Mitotic Exit and Late-Born Cell Specification. *Neuron* **102,** 1111-1126.e5 (2019).
33. Ntranos, V., Yi, L., Melsted, P. & Pachter, L. A discriminative learning approach to differential expression analysis for single-cell RNA-seq. *Nat. Methods* **16,** 163–166 (2019).
34. Soós, S. Age-sensitive bibliographic coupling reflecting the history of science: The case of the Species Problem. *Scientometrics* **98,** 23–51 (2014).
35. Lun, A. T. L. *et al.* EmptyDrops: distinguishing cells from empty droplets in droplet-based single-cell RNA sequencing data. *Genome Biol.* **20,** 63 (2019).

36. Griffiths, J. A., Richard, A. C., Bach, K., Lun, A. T. L. & Marioni, J. C. Detection and removal of barcode swapping in single-cell RNA-seq data. *Nat. Commun.* **9,** 2667 (2018).
37. Alexa, A., Rahnenführer, J. & Lengauer, T. Improved scoring of functional groups from gene expression data by decorrelating GO graph structure. *Bioinformatics* **22,** 1600–1607 (2006).
38. Ashburner, M. *et al.* Gene Ontology: tool for the unification of biology. *Nat. Genet.* **25,** 25–29 (2000).
39. The Gene Ontology Consortium. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Res.* **47,** D330–D338 (2019).
40. Aran, D. *et al.* Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. *Nat. Immunol.* **20,** 163–172 (2019).
41. Benayoun, B. A. *et al.* Remodeling of epigenome and transcriptome landscapes with aging in mice reveals widespread induction of inflammatory responses. *Genome Res.* **29,** 697–709 (2019).
42. Street, K. *et al.* Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. *BMC Genomics* **19,** 477 (2018).
43. Saelens, W., Cannoodt, R., Todorov, H. & Saeys, Y. A comparison of single-cell trajectory inference methods. *Nat. Biotechnol.* **37,** 547–554 (2019).
44. Macosko, E. Z. *et al.* Highly Parallel Genome-wide Expression Profiling of Individual Cells Using Nanoliter Droplets. *Cell* **161,** 1202–1214 (2015).

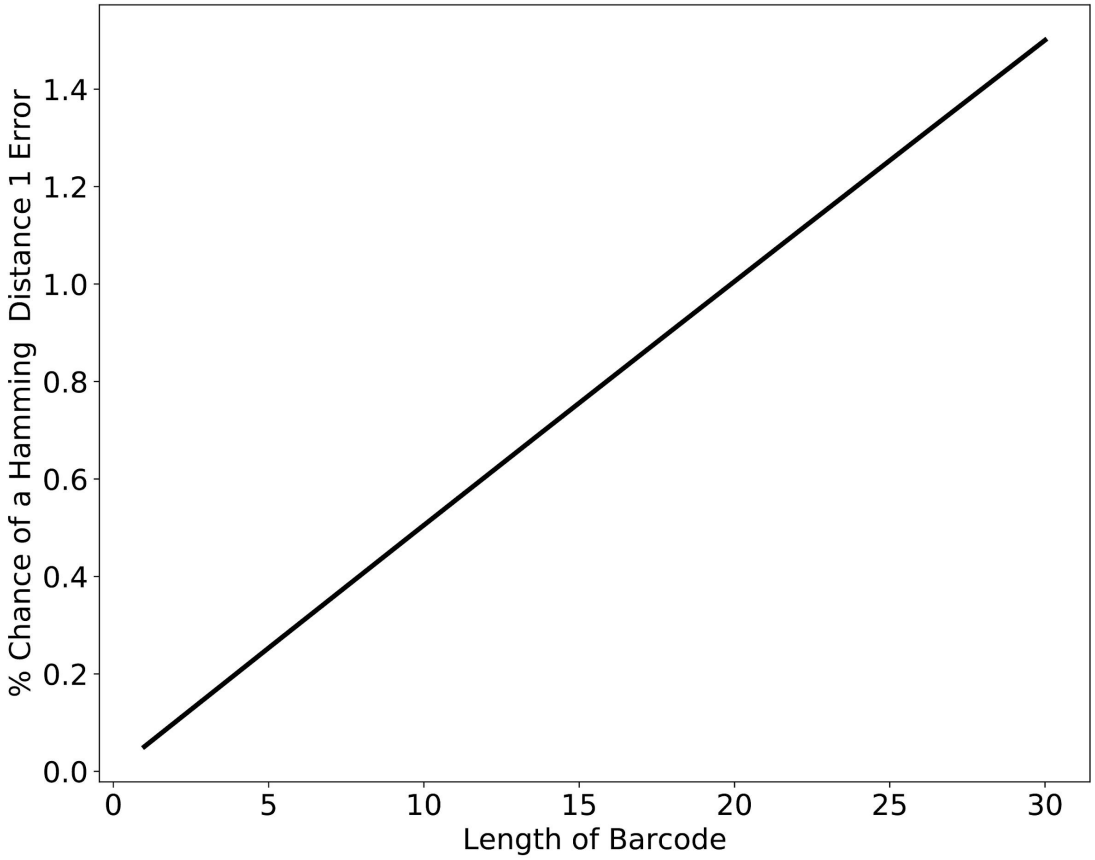
Supplementary Material



Supplementary Figure 1.1: Estimates of the effective number of UMIs per bead for each of the benchmark panel datasets, determined from observed collisions of UMIs across unique genes and assuming UMIs are sampled uniformly with replacement (see Supplementary Note for further details). The dashed red line is the theoretical maximum for the number of UMIs on a v2 chemistry bead (4^{10}=1,048,576) and the red line is the theoretical maximum for the number of UMIs on a v3 chemistry bead (4^{12}=16,777,216). The datasets are ordered by number of reads. UMI pools from 10x Chromium v2 and v3 chemistry are found to be highly complex, with the effective number of UMIs approaching the theoretical maximum in many cases. Our estimates for UMI complexity vary across experiments; this could be due to batch effects, or model misspecification. Sequencing chimeras could also affect UMI complexity estimates, specifically estimates would be increased with more chimeras. This would reduce the estimates of intra-gene collisions due to naïve collapsing.

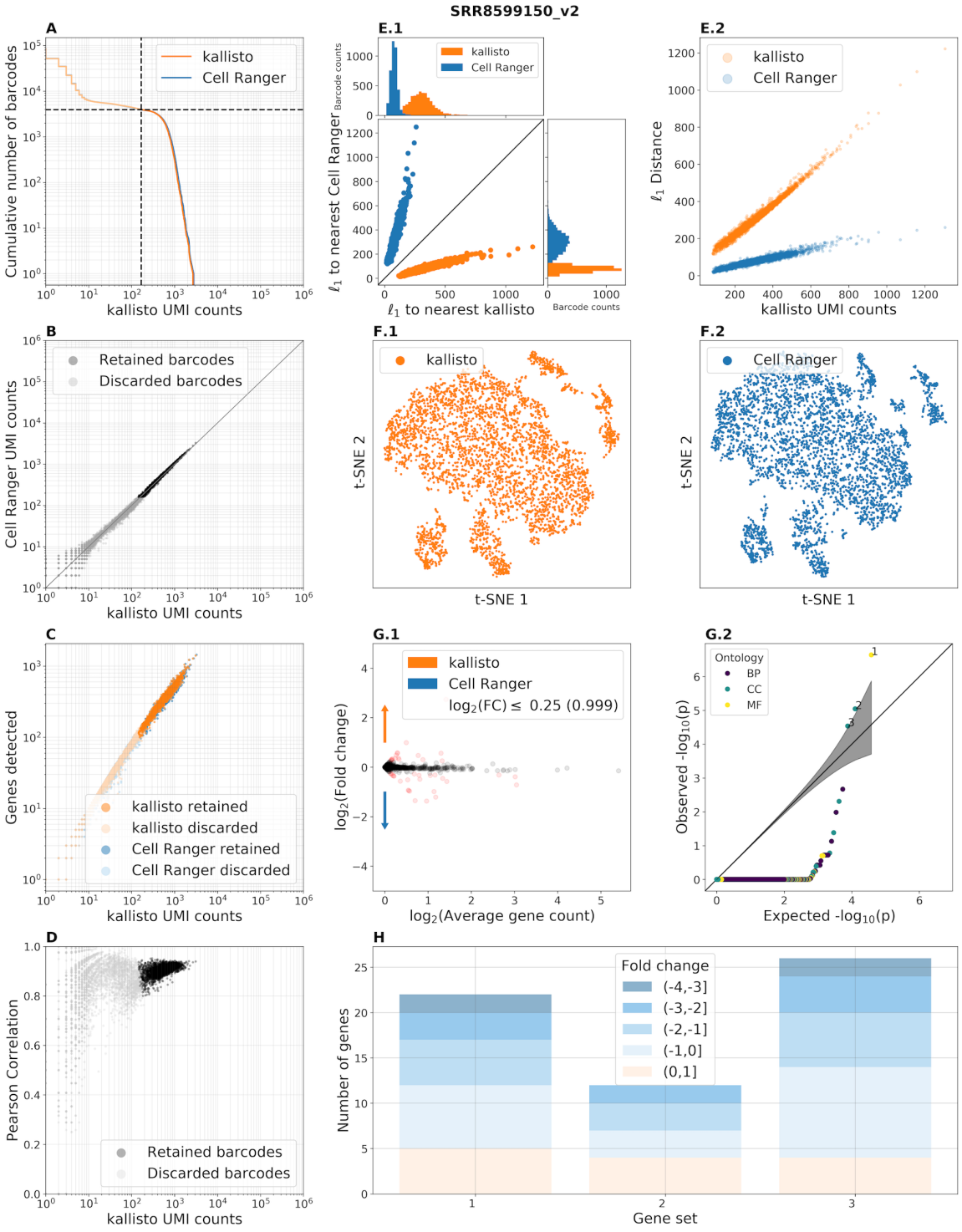


Supplementary Figure 1.2: Fraction of UMIs lost per gene across cells in the benchmark panel due to over-collapsing. The inset figure is a magnified view of the violin plot demonstrating that the fraction of counts lost due to UMI collapsing at the gene level is on the order of 0.0002.

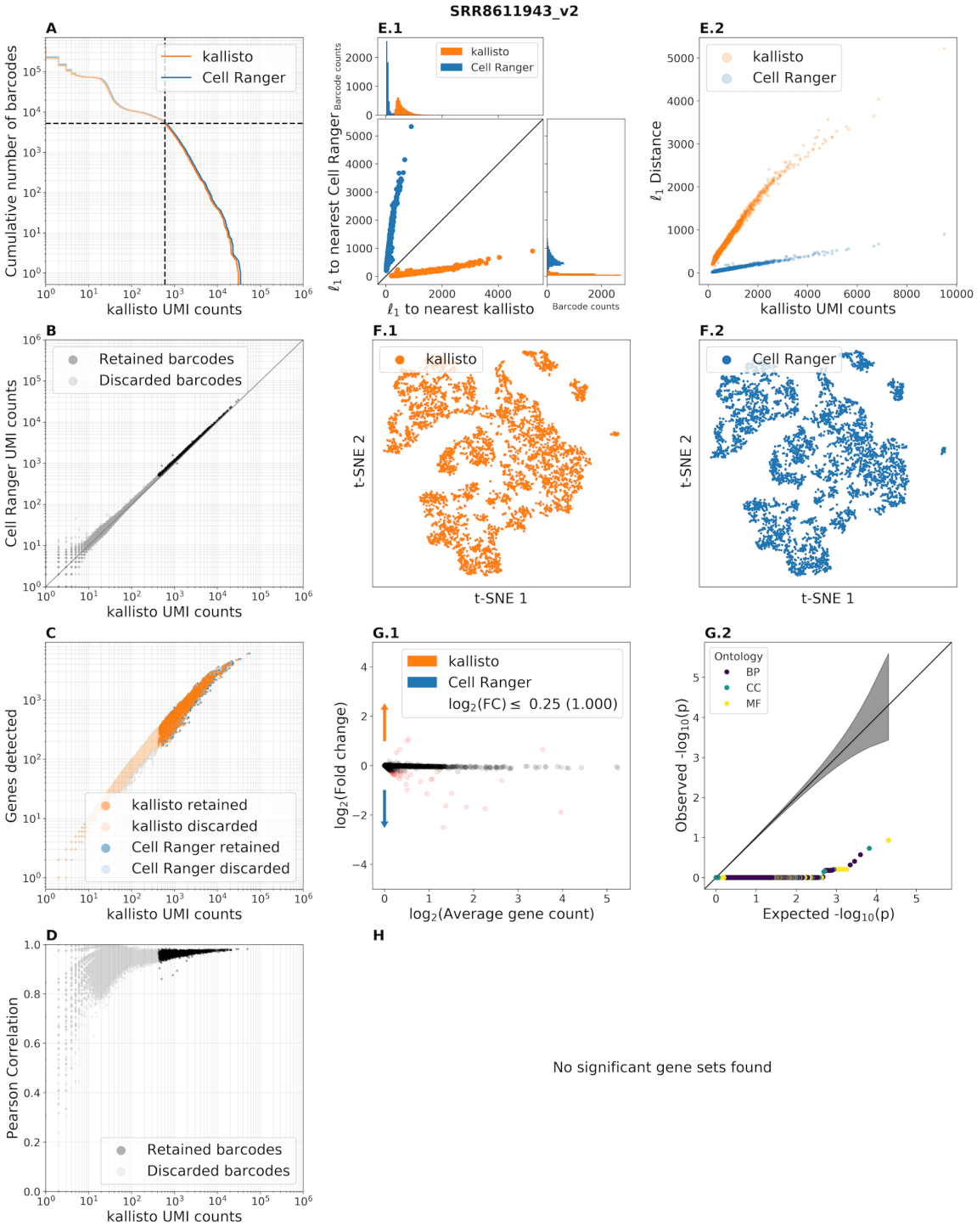


Supplementary Figure 2: The expected percentage of Barcodes (or UMIs) that will have one error and can therefore be corrected with a Hamming distance 1 correction algorithm. The y-axis displays the value of the function $f(L)$ where $f(L) = 100 \cdot L\hat{p}(1-\hat{p})^{L-1}$, and where $\hat{p}$ is the per base sequencing error probability estimated by averaging the error estimates across all the datasets in the benchmark panel (Supplementary Table 2).

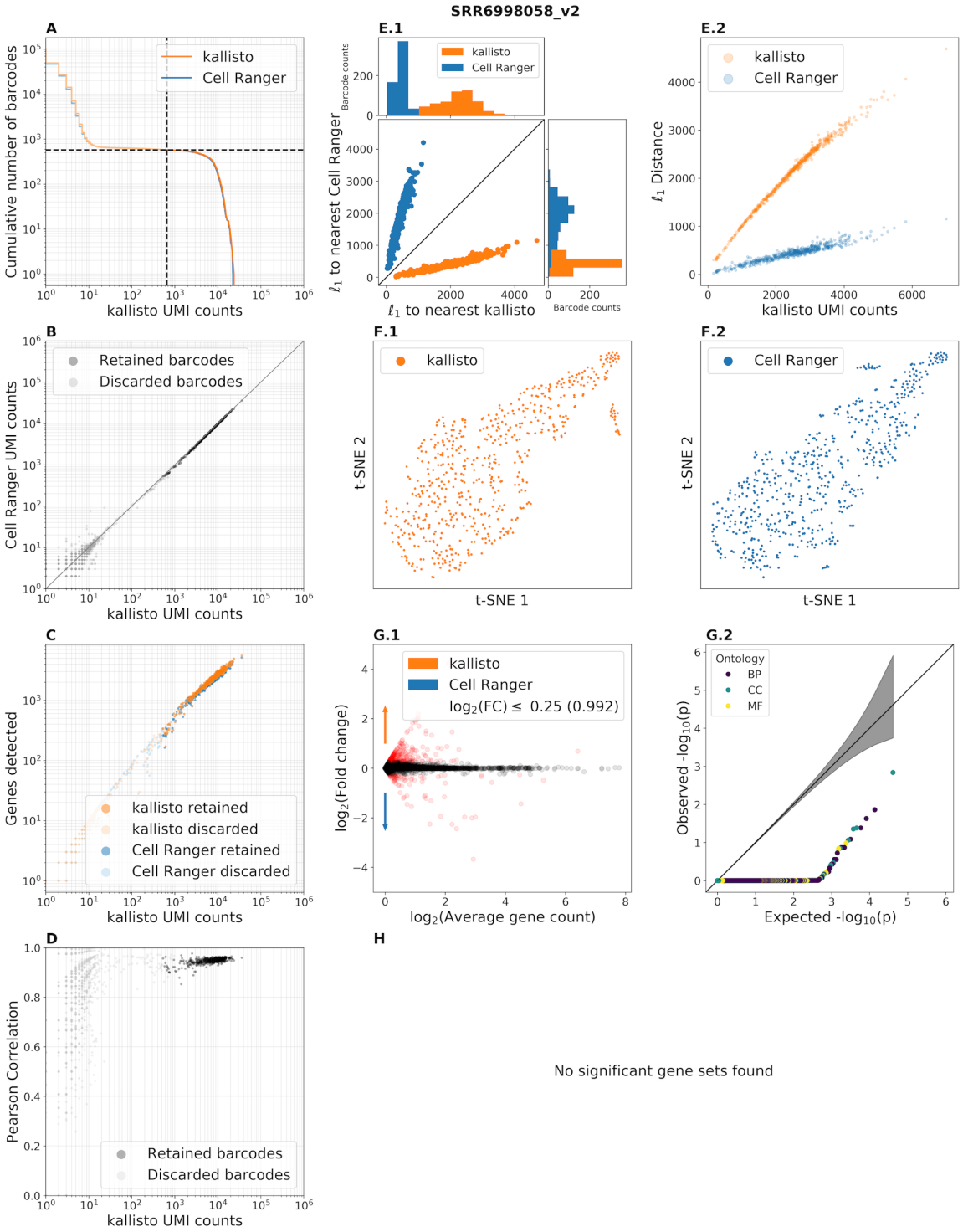
All **Supplementary Figure 3** benchmark panels are configured as follows: (A) "Knee plots" for kallisto and Cell Ranger showing, for a given UMI count (x-axis), the number of cells that contain at least that many UMI counts (y-axis). The dashed lines correspond to the Cell Ranger filtered cells. (B) Correspondence in the number of distinct UMIs per cell between the workflows. (C) Genes detected by kallisto and Cell Ranger as a function of distinct UMI counts per cell. (D) Pearson correlation between gene counts as a function of the distinct UMI counts per cell. (E.1) The l_1 distance between gene abundances for each kallisto cell and its nearest neighbor plotted against each kallisto cell and its corresponding Cell Ranger cell (orange) and the l_1 distance between the gene abundances for each Cell Ranger cell and its nearest neighbor plotted against each Cell Ranger cell and its corresponding kallisto cell (orange). Marginal distributions show that each kallisto cell is closest to its corresponding Cell Ranger cell and that each Cell Ranger cell is closest to its corresponding kallisto cell. (E.2) l_1 distance between kallisto and Cell Ranger cells as a function of UMI counts. (F.1) kallisto t-SNE from the first 10 principal components. (F.2) Cell Ranger t-SNE from the first 10 principal components. (G.1) MA plot for all genes between kallisto and Cell Ranger. Most of the genes have a $log_2(FC) \leq 0.25$ (G.2) QQ plot comparing the distribution of observed distribution of p-values of GSEA, after Bonferroni correction for multiple testing across ontologies and datasets, with the expected distribution of a uniform distribution between 0 and 1. If the observed distribution does not significantly deviate from the expected distribution, then the points should lie close to the diagonal line, $y = x$. The gray ribbon around the line is the 95% confidence interval. Here most GO terms have adjusted $p = 1$, meaning that most GO terms are very depleted of genes "differentially expressed (DE)" between the kallisto and Cell Ranger matrices. GO terms above $y = x$ are labeled. Generally, GO terms significantly enriched among "DE" genes are related to ribosomal proteins and are labeled by numbers corresponding to GO terms in the figure caption. The points are also colored by ontology: biological processes (BP), cellular components (CC), and molecular functions (MF). (H) Significant differential gene sets between Cell Ranger and kallisto.



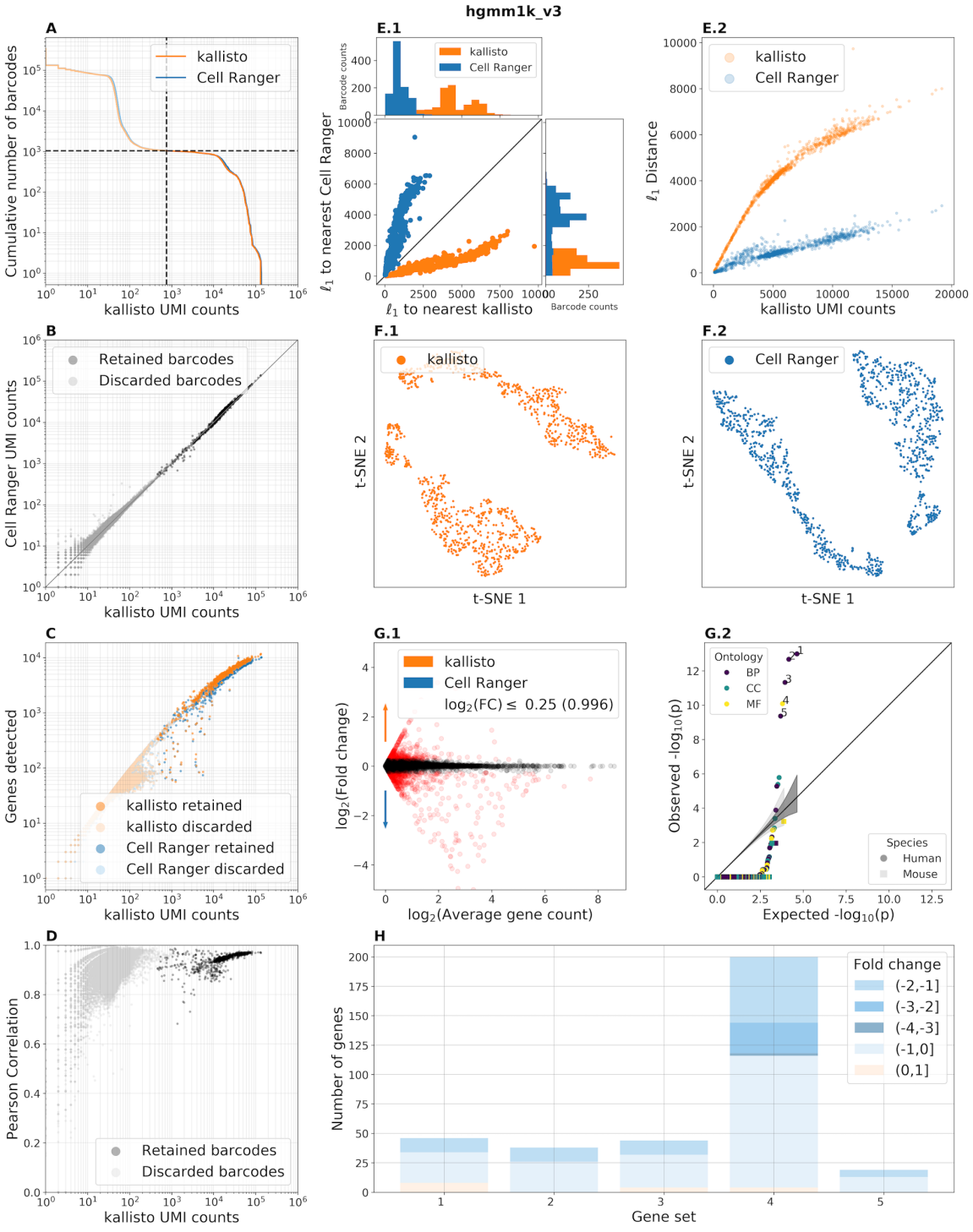
Supplementary Figure 3.1: Benchmark panel of data from O'Koren et al. 2019 (O'Koren et al., 2019) (SRR8599150). Enriched GO terms are 1-structural constituent of ribosome, 2-cytosolic small ribosomal subunit, 3-cytosolic large ribosomal subunit.



Supplementary Figure 3.2: Benchmark panel of data from Packer et al. 2019 (Packer et al., 2019) (SRR8611943).

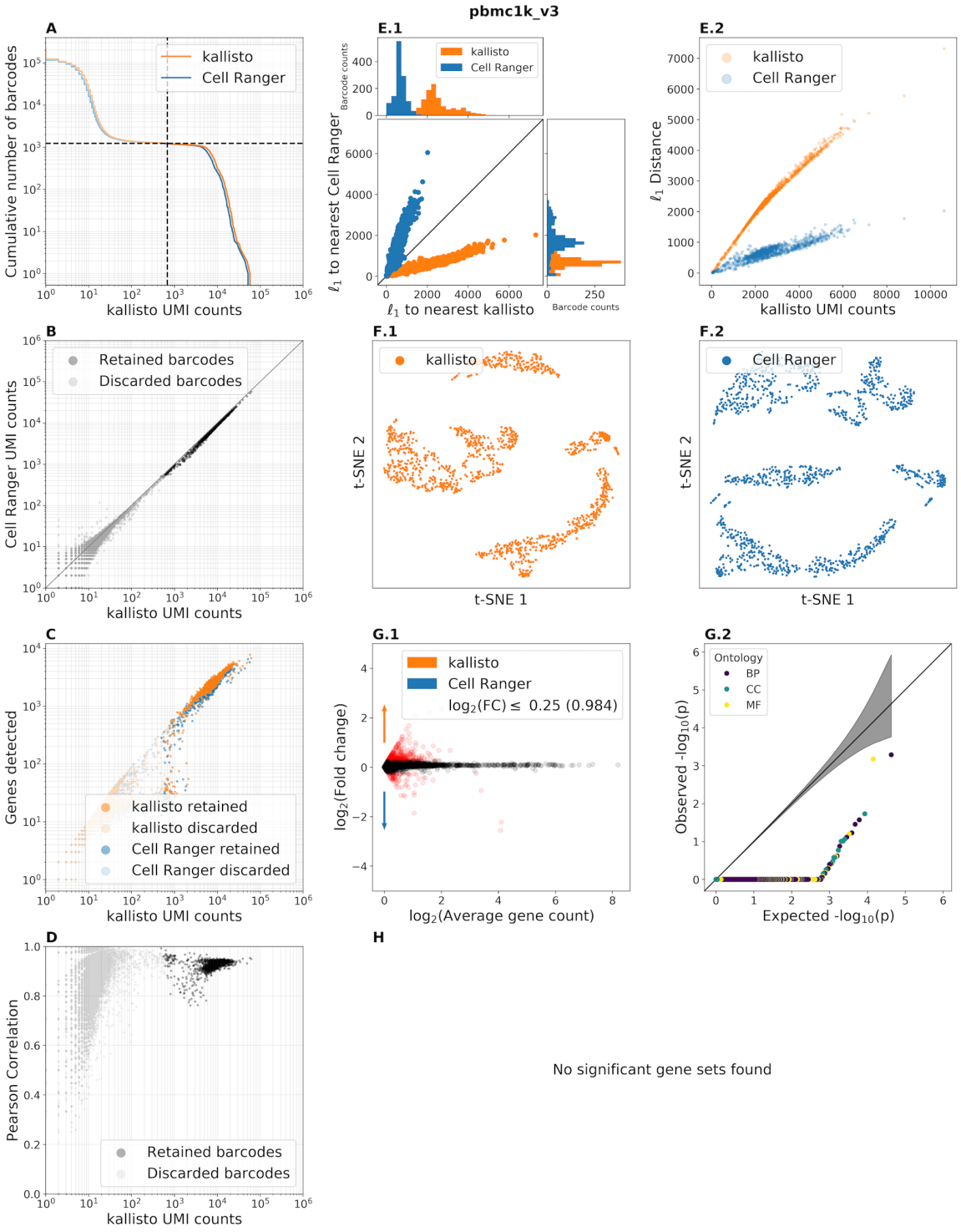


Supplementary Figure 3.3: Benchmark panel of data from Jin et al. 2018 (Jin et al., 2018) (SRR6998058).

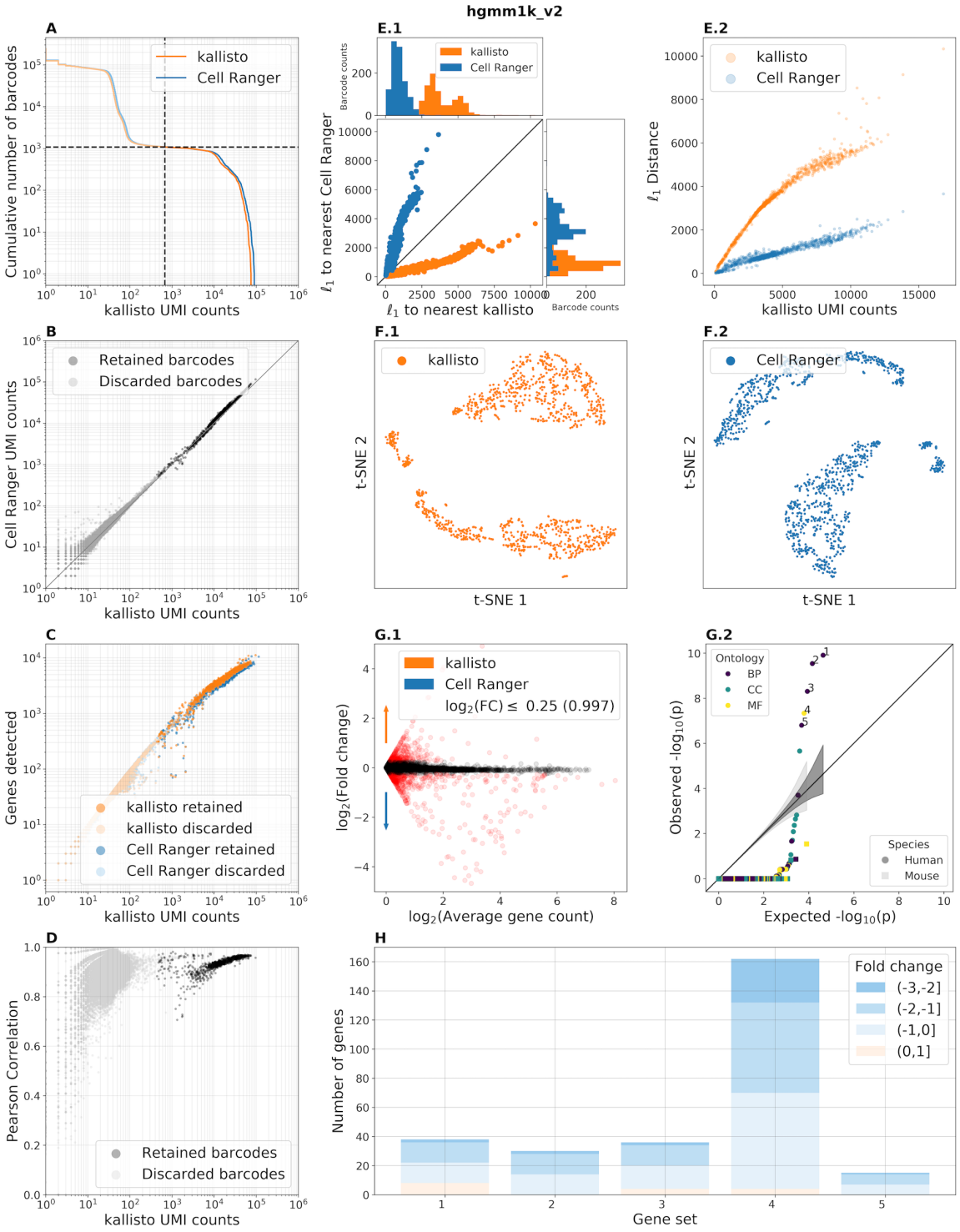


Supplementary Figure 3.4: Benchmark panel of the 10x Genomics hgmm1k_v3 dataset. Enriched GO terms are 1-nuclear-transcribed mRNA catabolic process, nonsense-mediated decay, 2-SRP-dependent cotranslational protein targeting to

membrane, 3-translational initiation, 4-structural constituent of ribosome, 5-viral transcription.

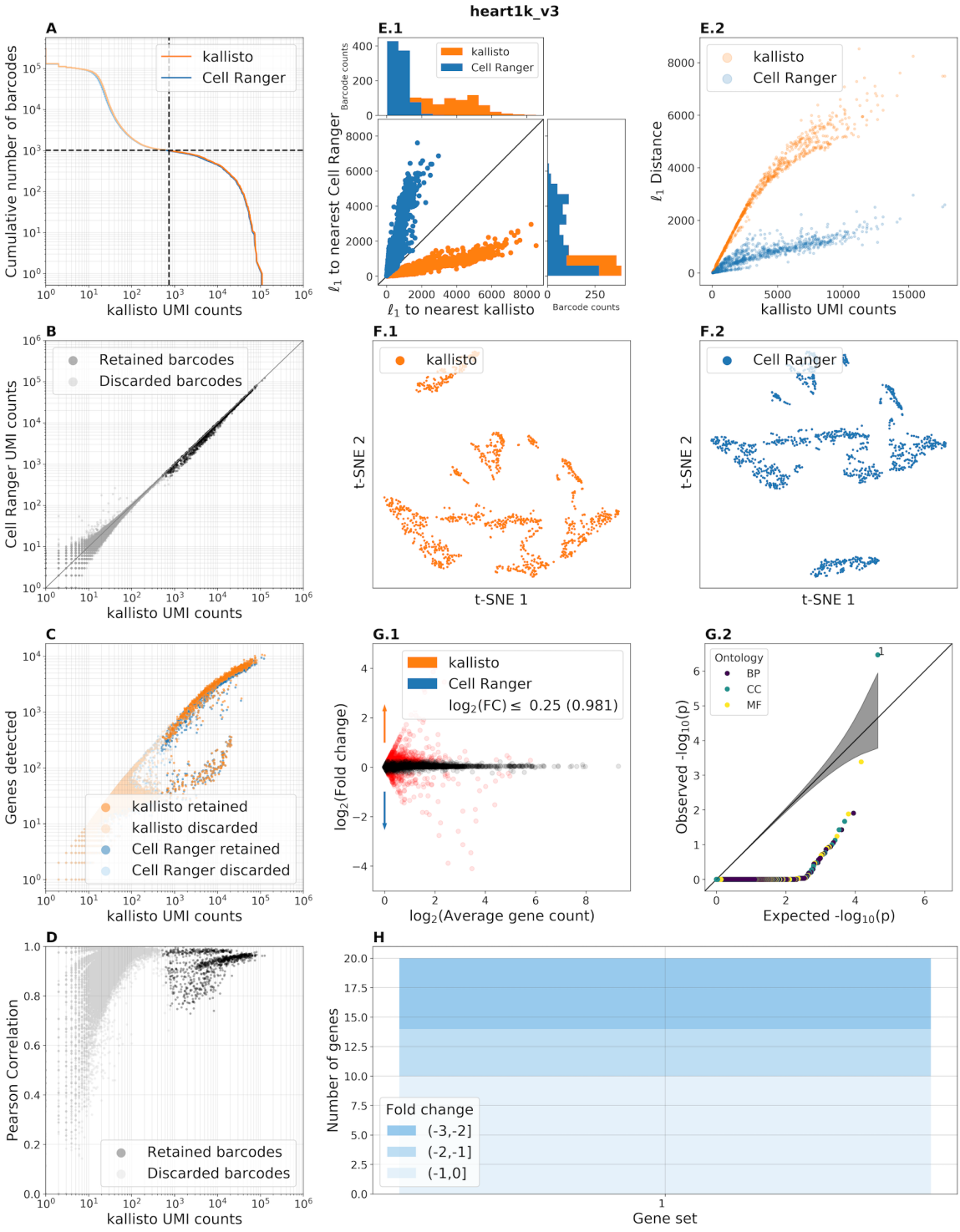


Supplementary Figure 3.5: Benchmark panel of the 10x Genomics pbmc1k_v3 dataset.

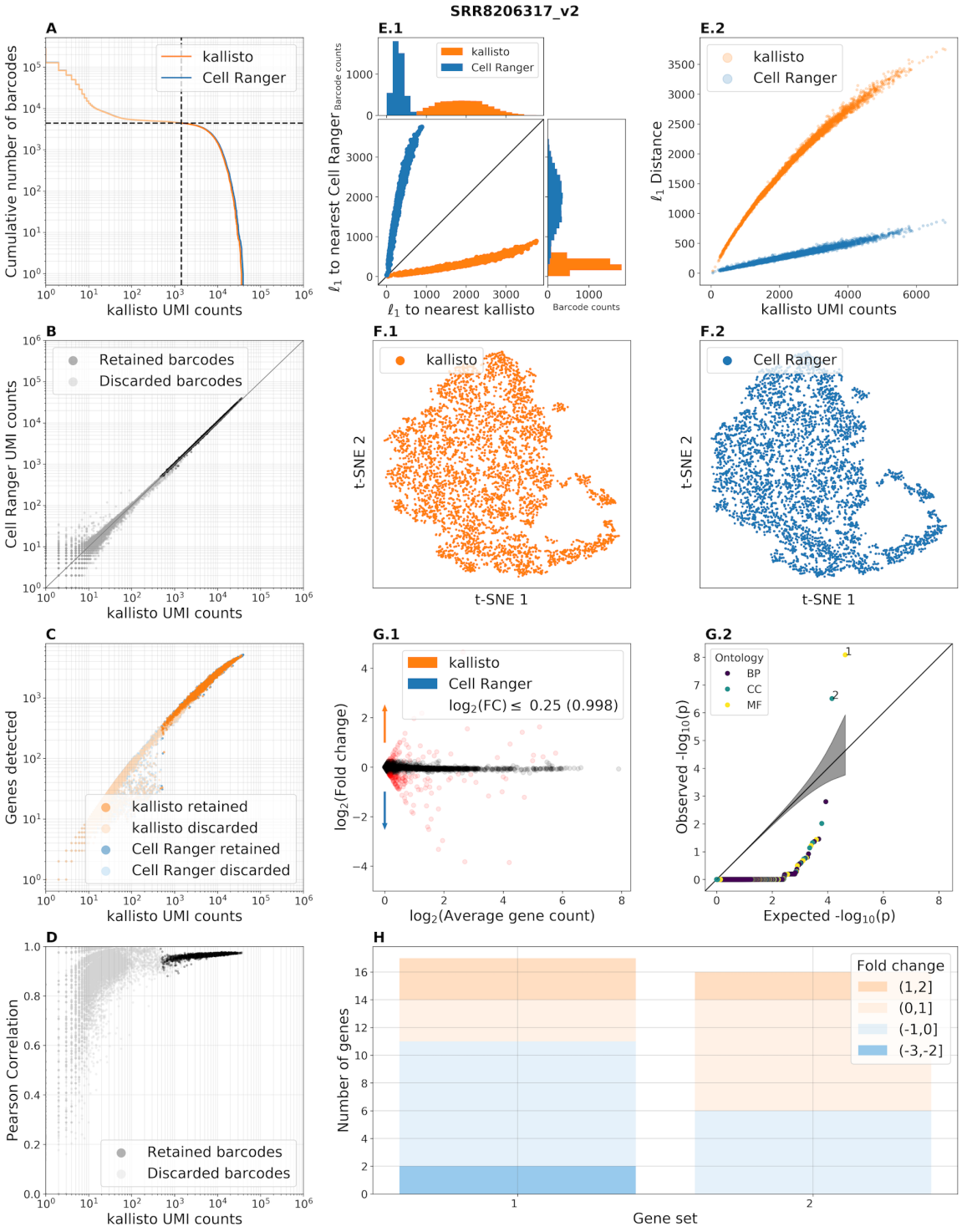


Supplementary Figure 3.6: Benchmark panel of the 10x Genomics hgmm1k_v2 dataset. Enriched GO terms are 1-nuclear-transcribed mRNA catabolic process, nonsense-mediated decay, 2-SRP-dependent cotranslational protein targeting to

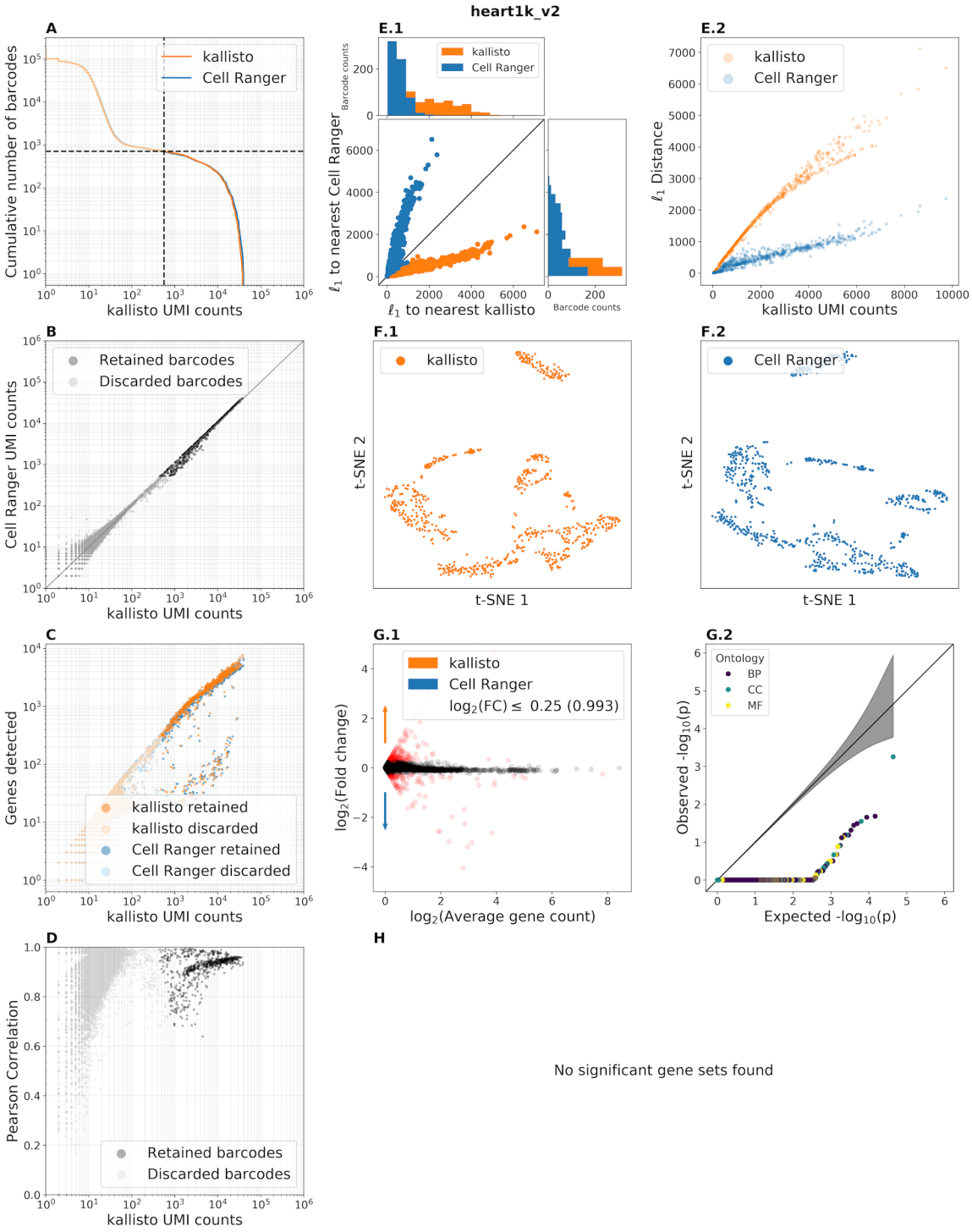
membrane, 3-translational initiation, 4-structural constituent of ribosome, 5-viral transcription.



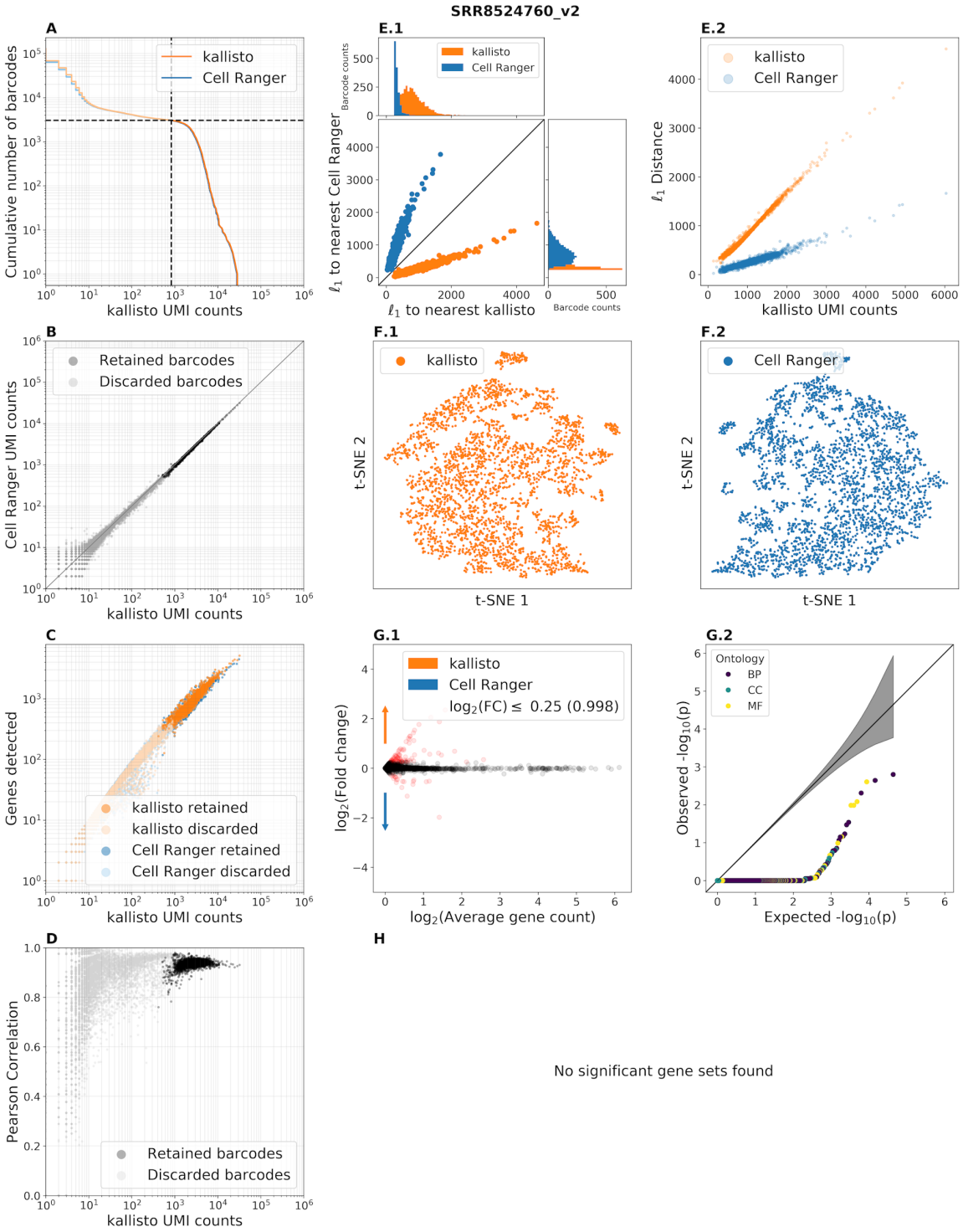
Supplementary Figure 3.7: Benchmark panel of the 10x Genomics heart1k_v3 dataset. Enriched GO terms are 1-cytosolic large ribosomal subunit.



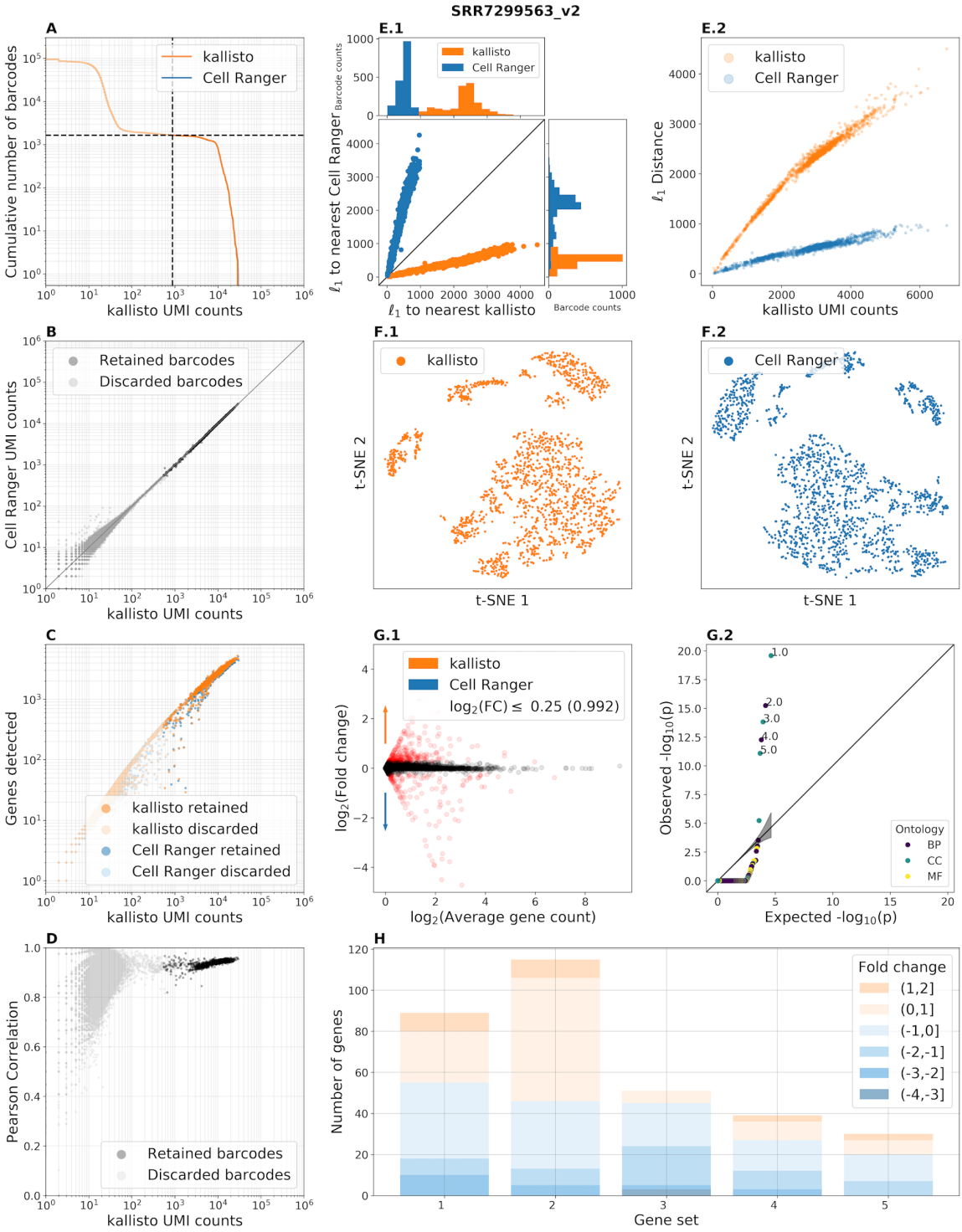
Supplementary Figure 3.8: Benchmark panel of data from Miller et al. 2019 (Miller et al., 2019) (SRR8206317). Enriched GO terms are 1-structural constituent of ribosome, 2-cytosolic large ribosomal subunit.



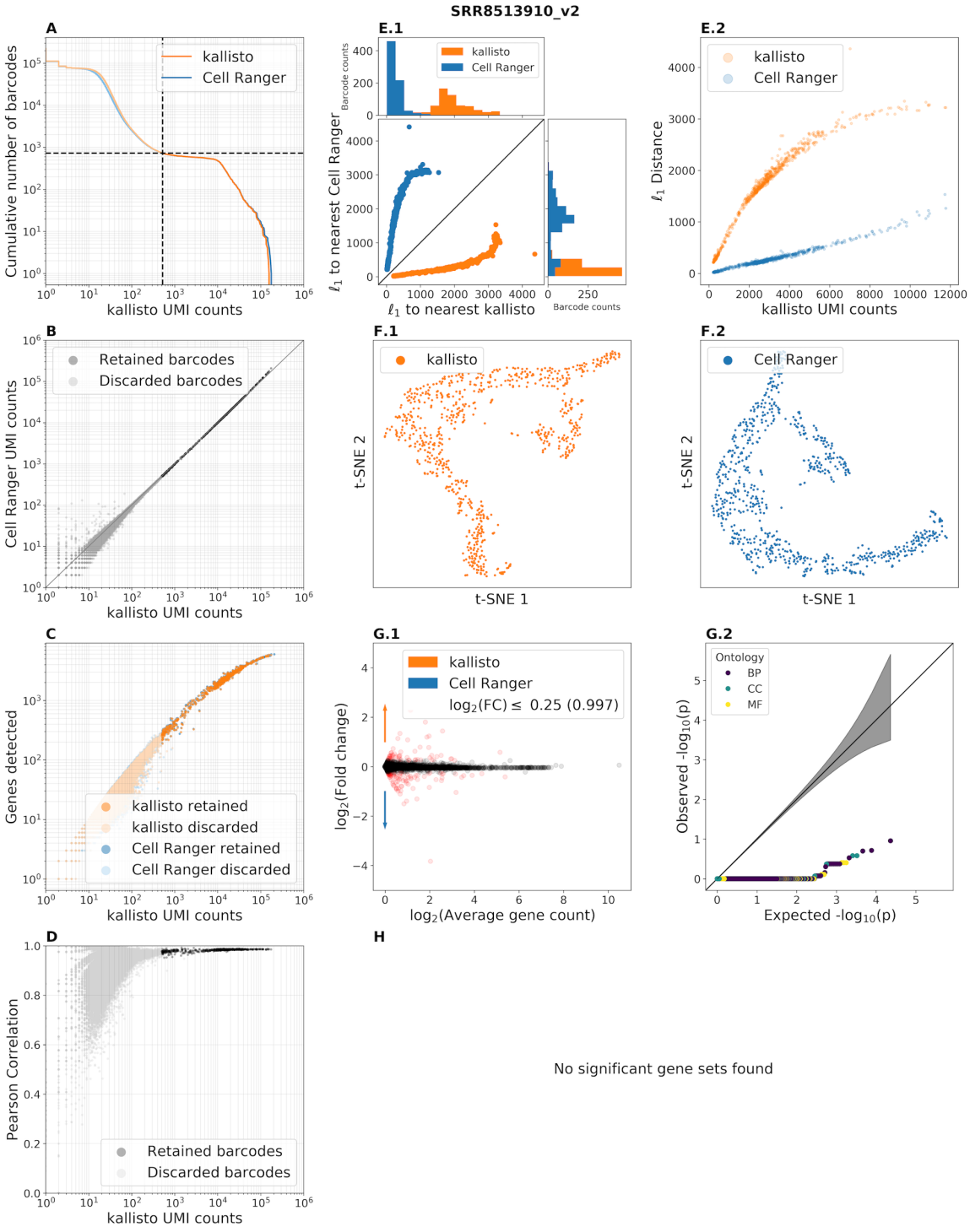
Supplementary Figure 3.9: Benchmark panel of the 10x Genomics heart1k_v2 dataset.



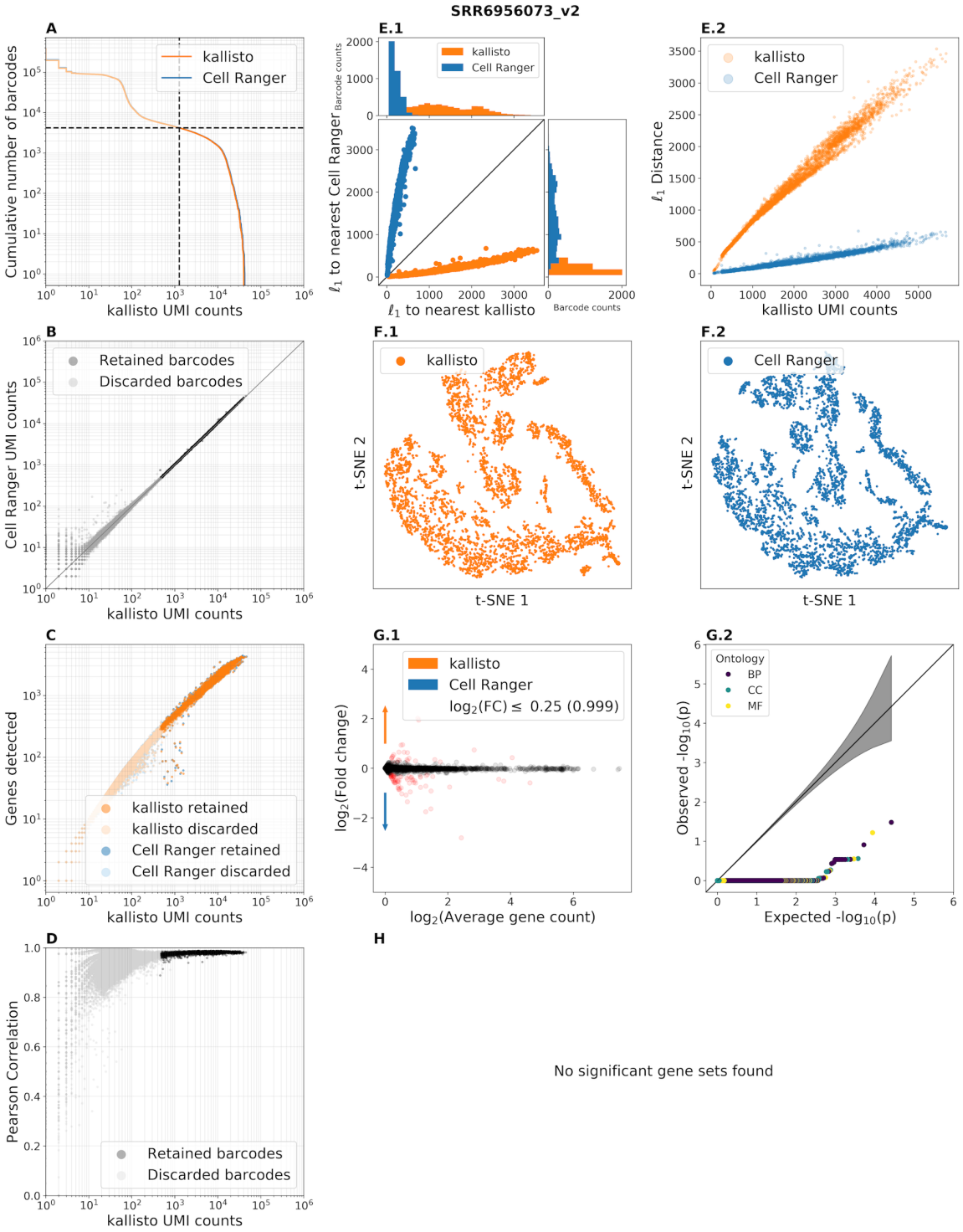
Supplementary Figure 3.10: Benchmark panel of data from Carosso et al. 2019 (Carosso et al., 2018) (SRR8524760).



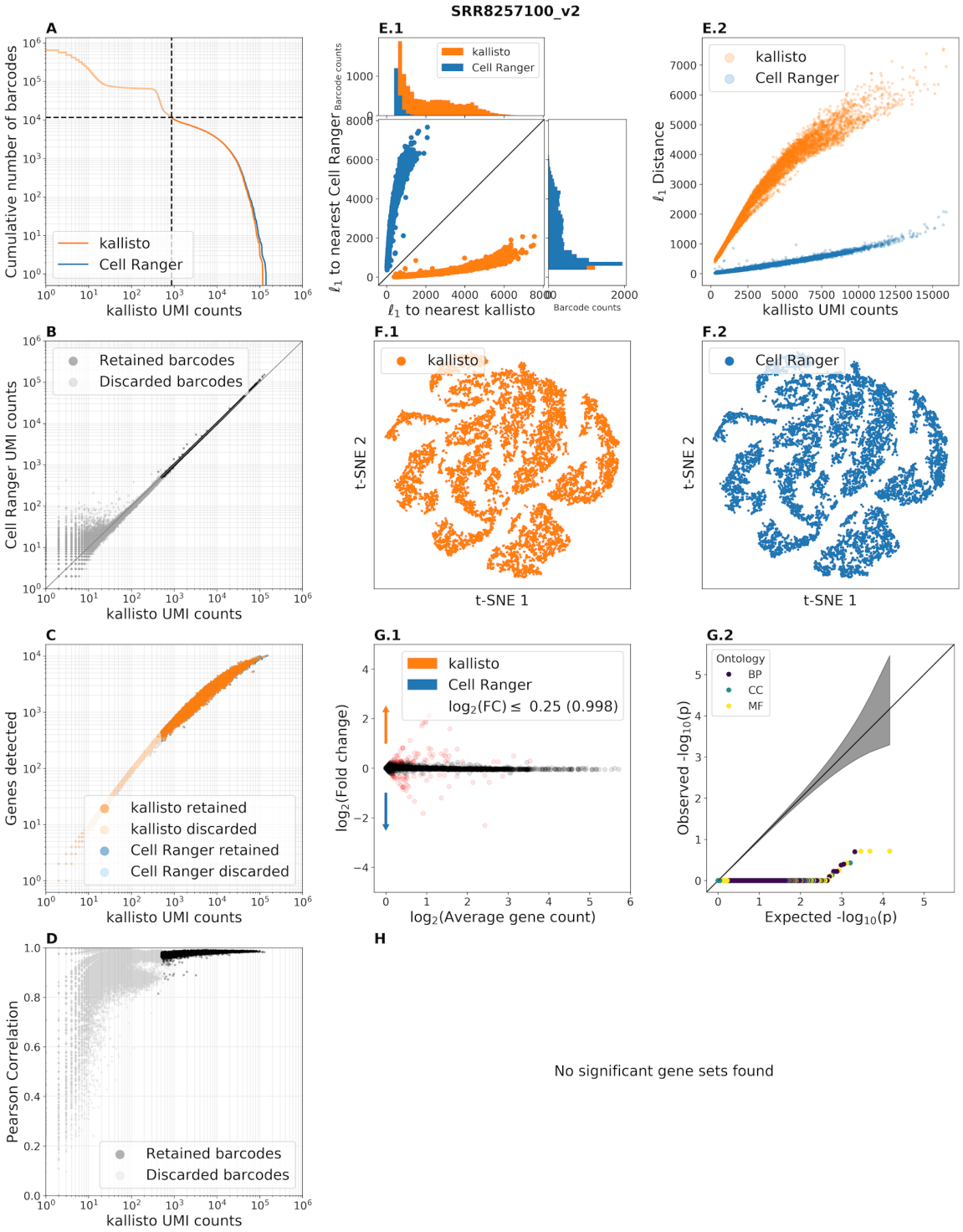
Supplementary Figure 3.11: Benchmark panel of data from Mays et al. 2018 (Mays et al., 2018) (SRR7299563). Enriched GO terms are 1-cytosolic large ribosomal subunit, 2-translation, 3-cytosolic small ribosomal subunit, 4-cytoplasmic translation, 5-polysomal ribosome.



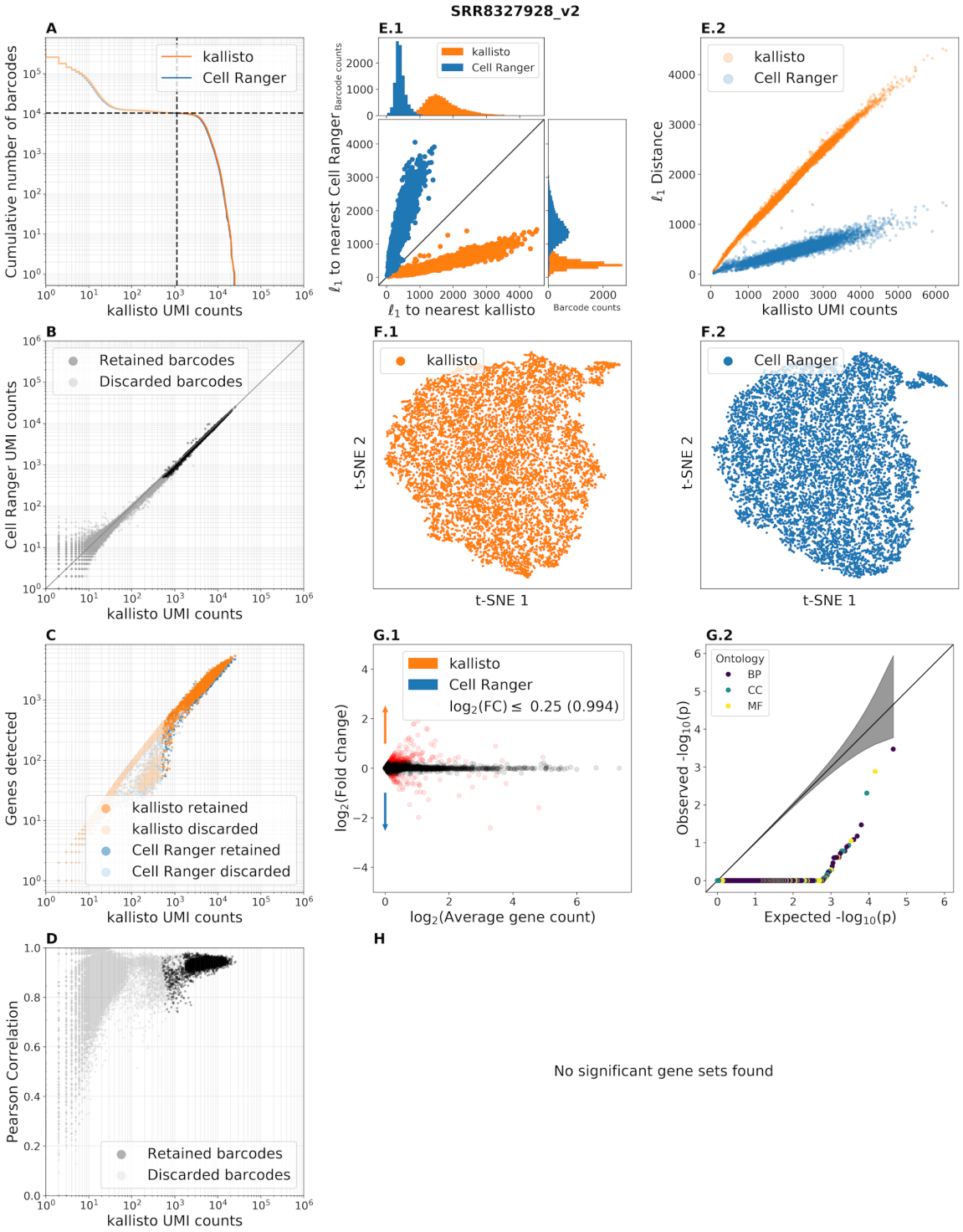
Supplementary Figure 3.12: Benchmark panel of data from the gene expression omnibus(Mahadevaraju et al., 2019) (SRR8513910).



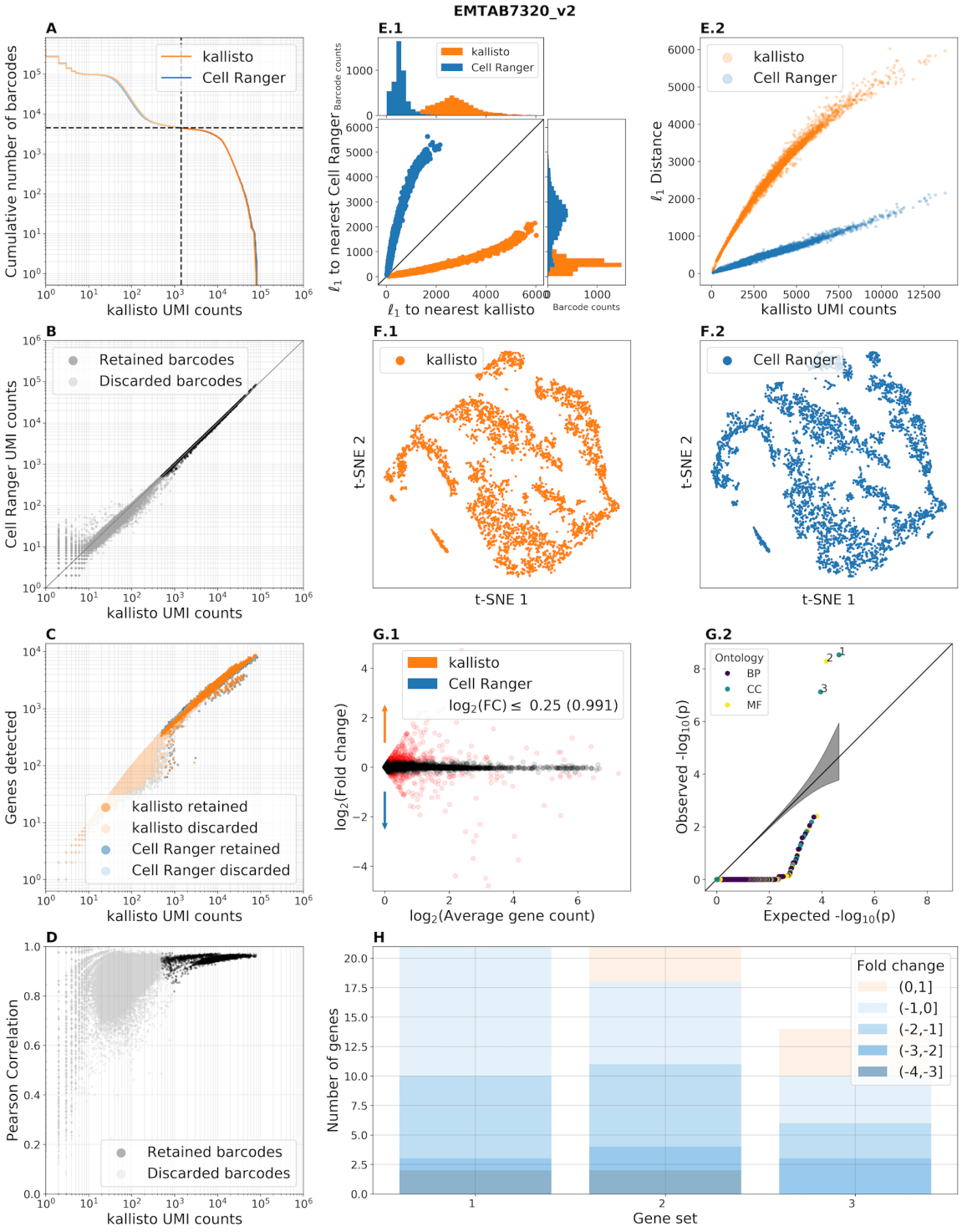
Supplementary Figure 3.13: Benchmark panel of data from Farrell et al. 2018(Farrell et al., 2018) (SRR6956073).



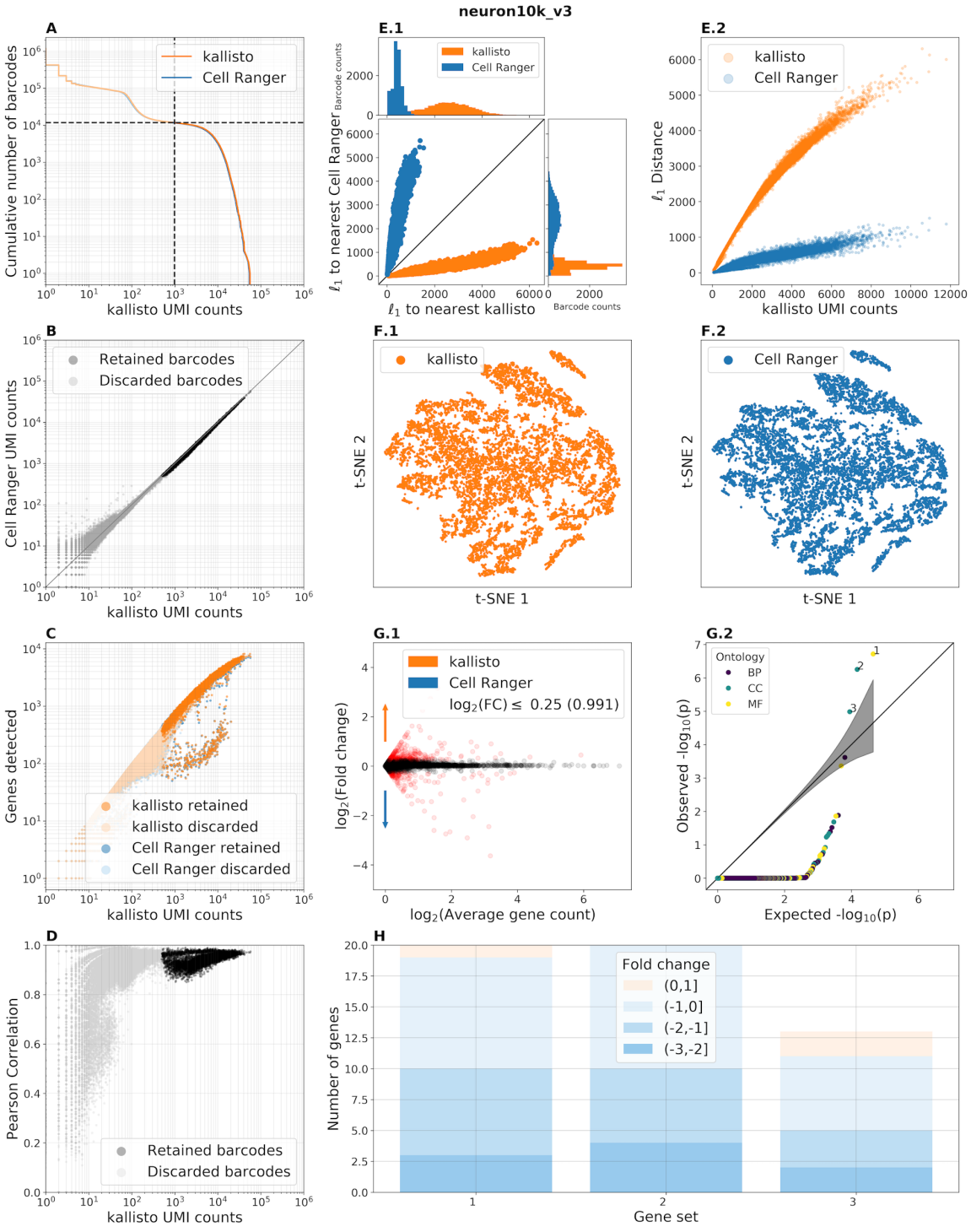
Supplementary Figure 3.14: Benchmark panel of data from Ryu et al. 2019(Ryu et al., 2019) (SRR8257100).



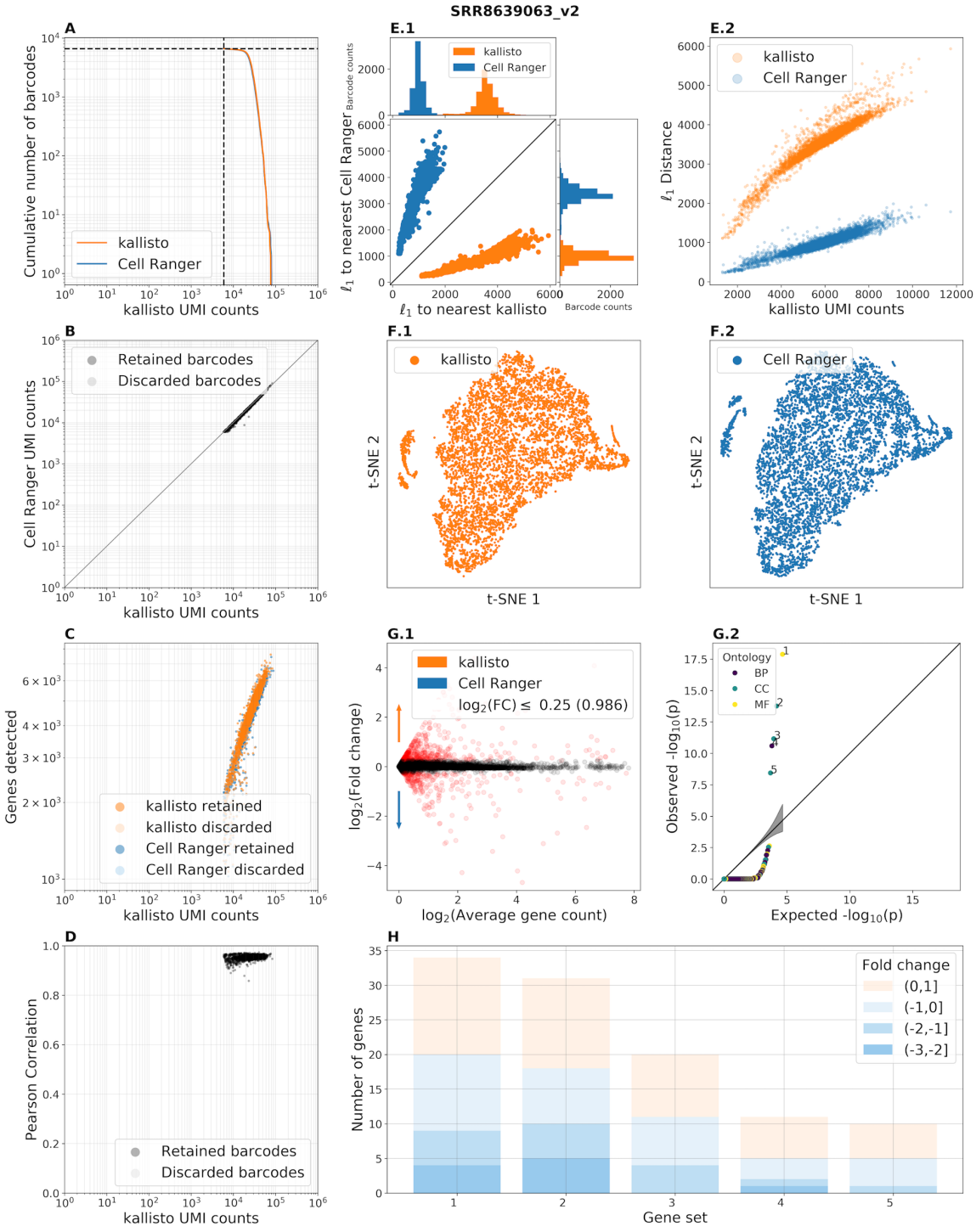
Supplementary Figure 3.15: Benchmark panel of data from Merino et al. 2019(Merino et al., 2019) (SRR8327928).



Supplementary Figure 3.16: Benchmark panel of data from Delile et al. 2019(Delile et al., 2019) (EMTAB7320). Enriched GO terms are 1-cytosolic large ribosomal subunit, 2-structural constituent of ribosome, 3-cytosolic small ribosomal subunit.

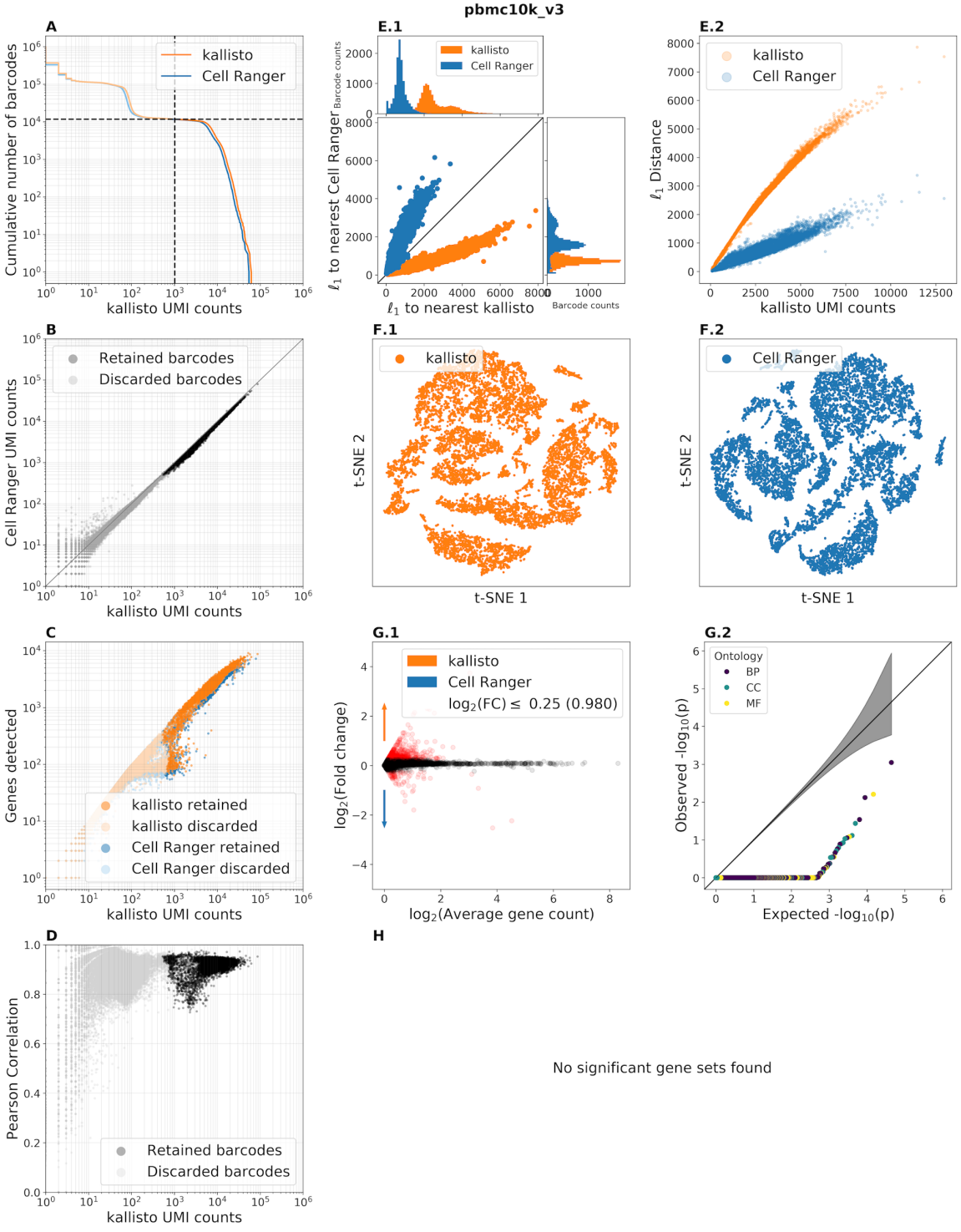


Supplementary Figure 3.17: Benchmark panel of the 10x Genomics neuron10k_v3 dataset. Enriched GO terms are 1-structural constituent of ribosome, 2-cytosolic large ribosomal subunit, 3-cytosolic small ribosomal subunit.

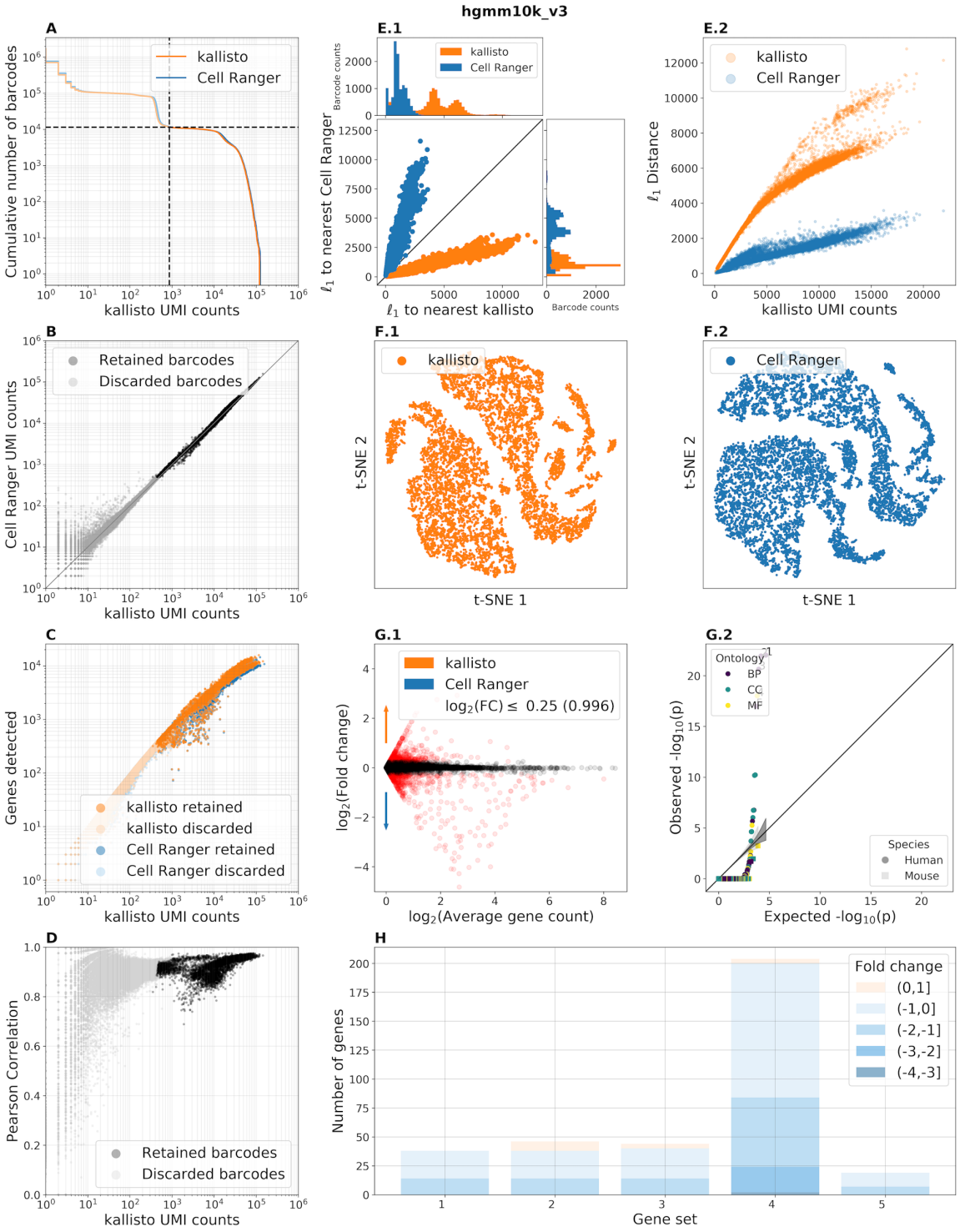


Supplementary Figure 3.18: Benchmark panel of data from Guo et al. 2019 (Guo et al., 2019) (SRR8639063). Note that the FASTQ files distributed with this experiment contained only retained barcodes. Enriched GO terms are 1-structural constituent of

ribosome, 2- cytosolic large ribosomal subunit, 3-cytosolic small ribosomal subunit, 4-cytoplasmic translation, 5-polysomal ribosome.

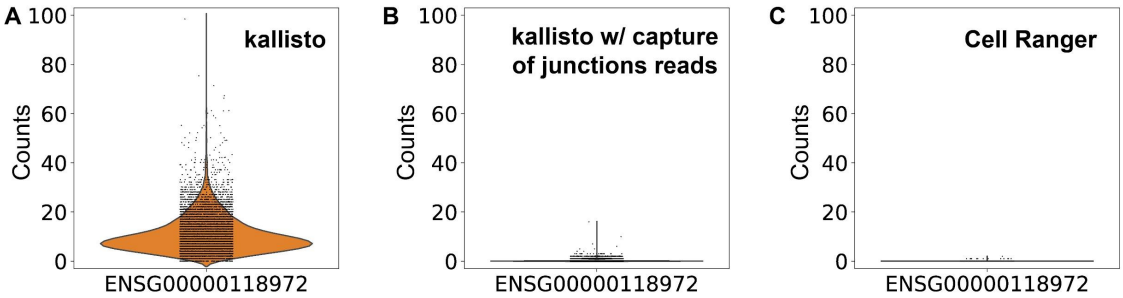


Supplementary Figure 3.19: Benchmark panel of the 10x Genomics pbmc10k_v3 dataset.

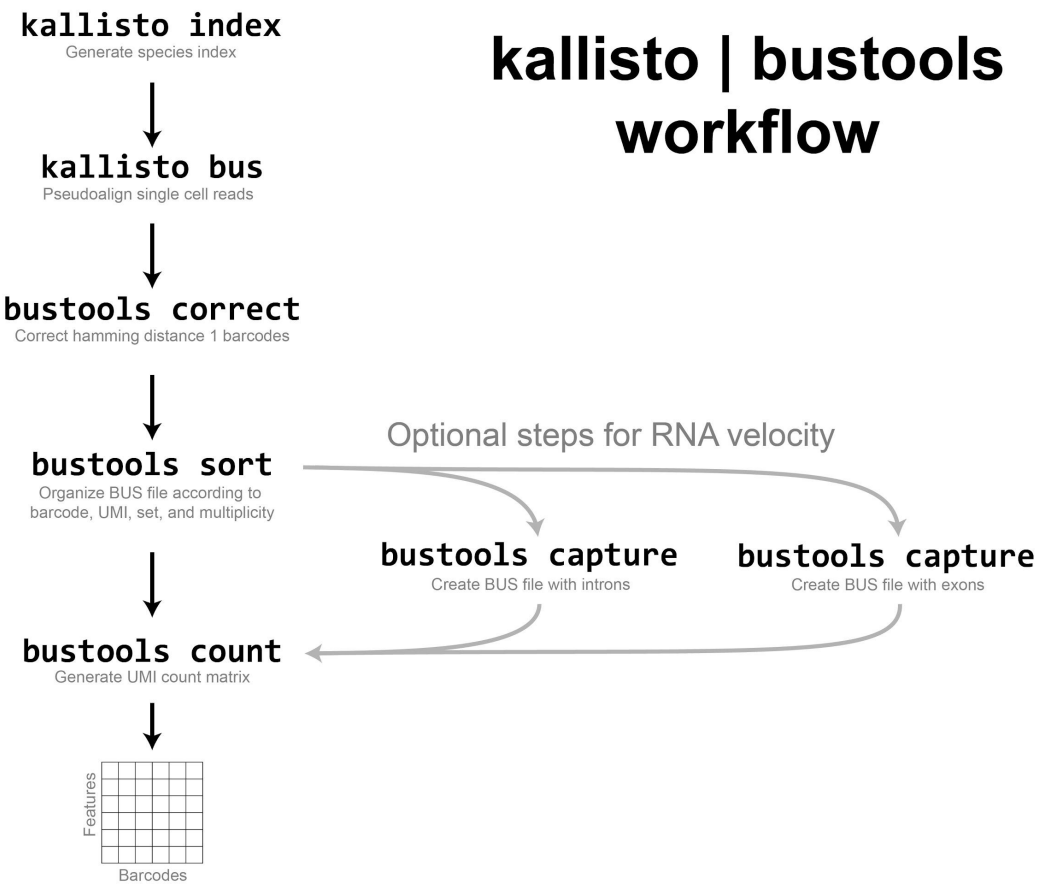


Supplementary Figure 3.20: Benchmark panel of the 10x Genomics hgmm10k_v3 dataset. Enriched GO terms are 1-SRP-dependent cotranslational protein targeting to

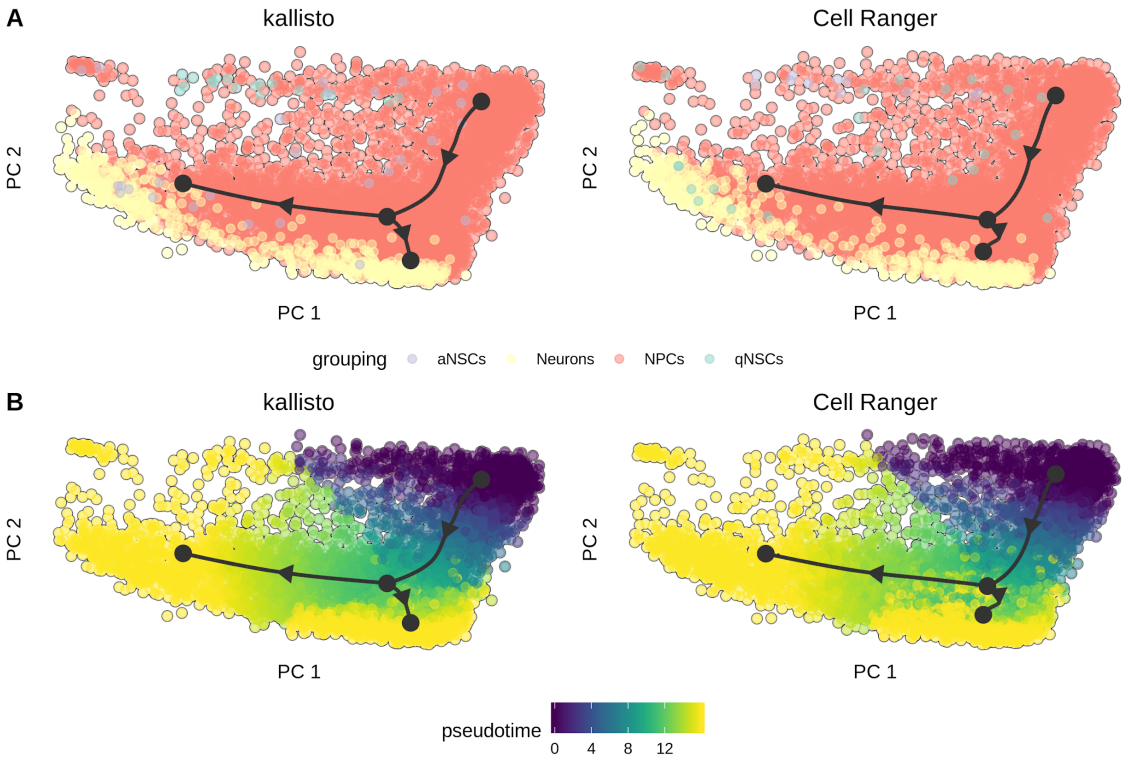
membrane, 2-nuclear-transcribed mRNA catabolic process, nonsense-mediated decay, 3-translational initiation, 4-structural constituent of ribosome, 5-viral transcription.



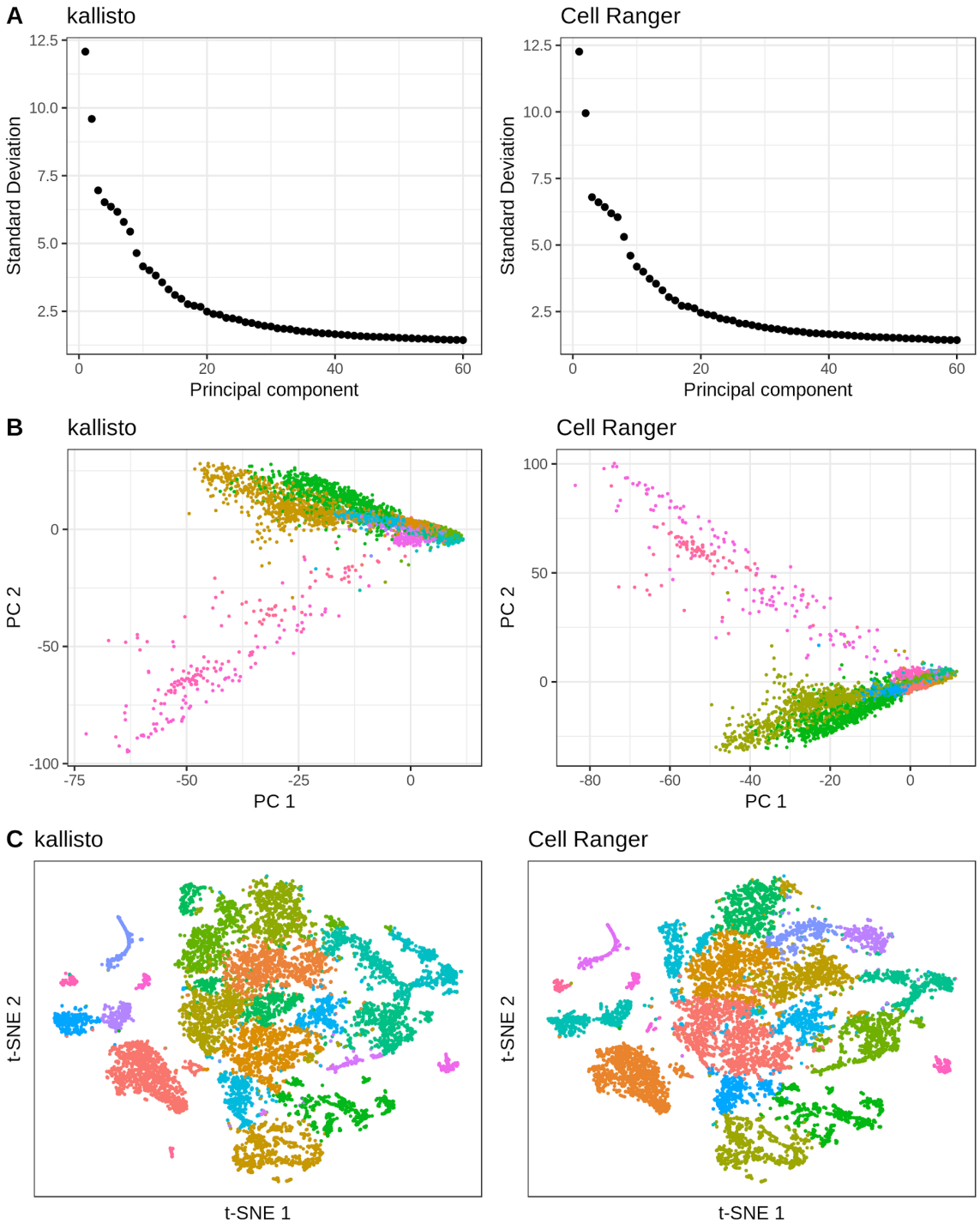
Supplementary Figure 4: Violin plots displaying distribution of counts for the FGF23 (ENSG00000118972) gene in all cells in the pbmc_10k_v3 dataset using different alignment methods. (A) Transcriptome pseudoalignment with kallisto using a standard index constructed from ENSEMBL transcripts. (B) Transcriptome pseudoalignment with kallisto using a modified index that includes, separately, sequences from splice junctions to capture unspliced junction reads. (C) Genome alignment with Cell Ranger. The gene was selected as an example as it was an outlier in discrepancy between kallisto and Cell Ranger when quantification was done with the standard index.



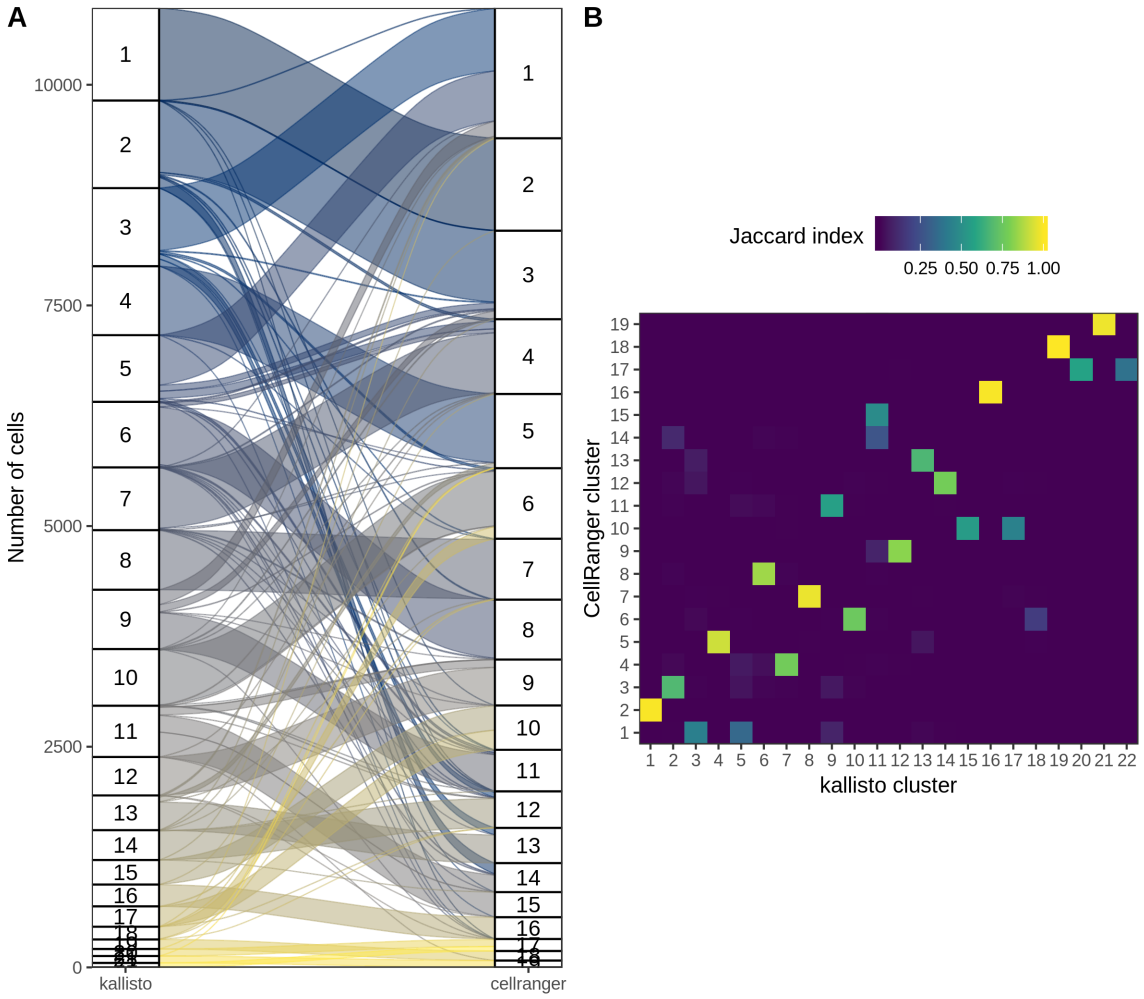
Supplementary Figure 5: Overview of the kallisto | bustools workflow. First an index for kallisto is built from a set of transcript sequences using the **kallisto index** command. Then **kallisto bus** is run on the FASTQ files; this generates a BUS file that contains records corresponding to reads, with data on the cell barcode, UMI, and transcript compatibility of each read. The barcodes are then corrected by processing the BUS file with the **bustools correct** command, after which the BUS file is sorted with **bustools sort**. Here, duplicate reads (those reads sharing an identical cell barcode, UMI, and equivalence class triplet) are collapsed into a single record and their abundance saved as a new metadata column in the BUS file named "multiplicity". Finally, **bustoools count** produces *cells x features* count matrices. If **kallisto bus** is run with an index containing intron sequences, the **bustools capture** command can be used to produce spliced and unspliced matrices for RNA velocity after sorting and before counting.



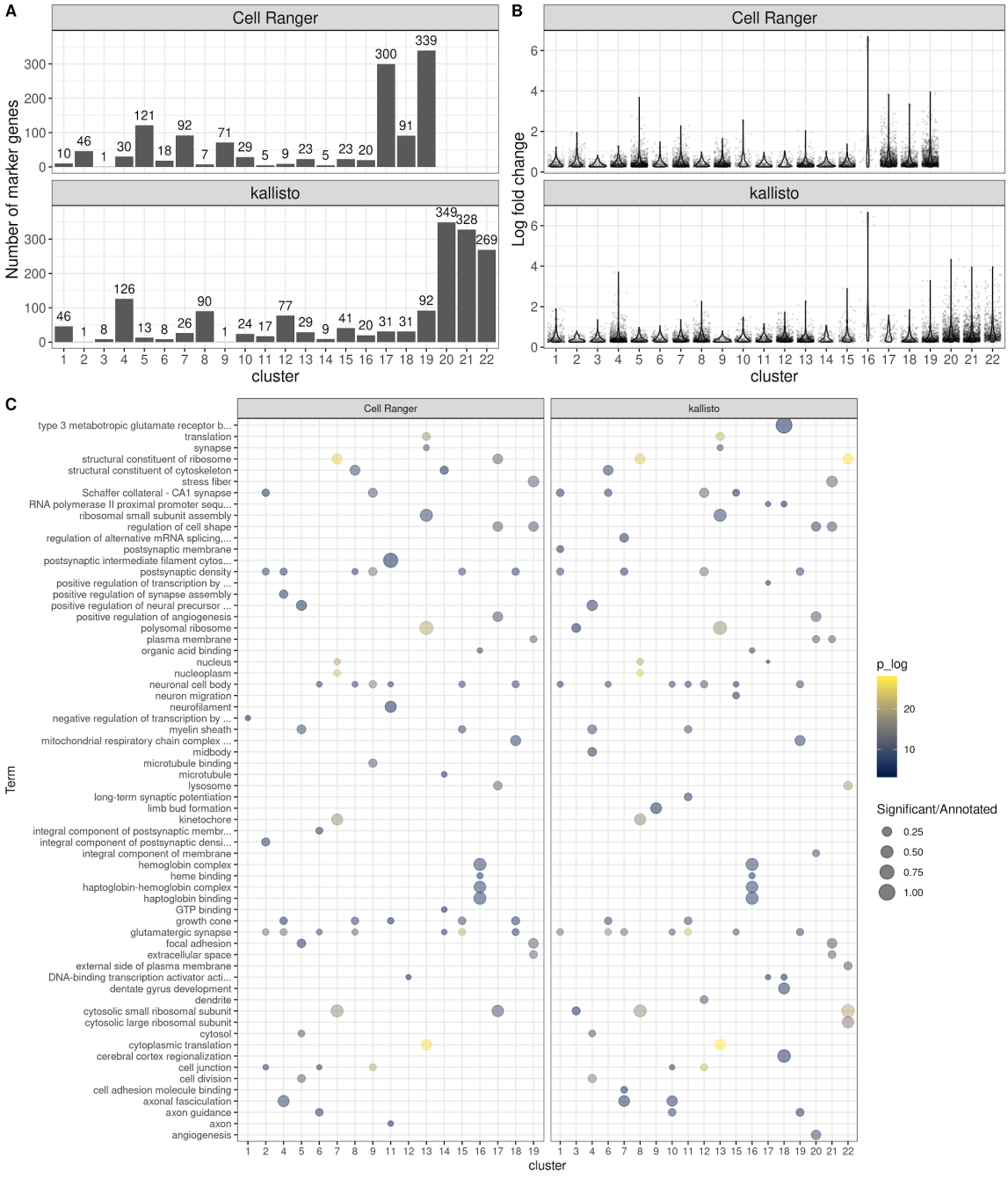
Supplementary Figure 6: Pseudotime trajectories. (A) Lineage inference of neuron10k_v3 dataset with slingshot projected to the first 2 principal components, with cells colored by cell type inferred by SingleR. aNSCs stands for active neuronal stem cells. NPCs stands for neuronal precursor cells. qNSCs stands for quiescent neuronal stem cells. (B) Coloring by pseudotime values from slingshot.



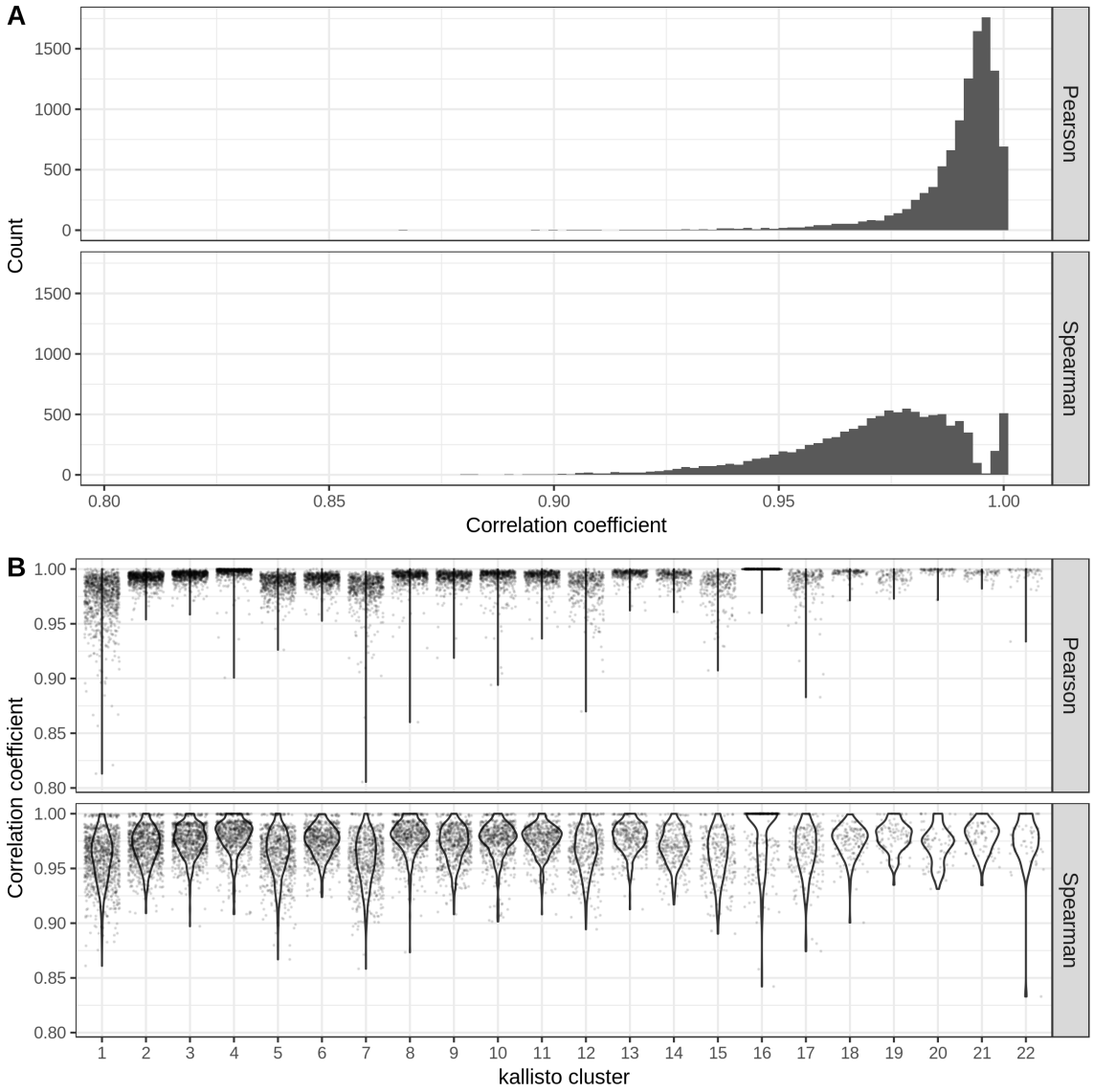
Supplementary Figure 7.1: (A) Elbow plot of standard deviation explained by each principal component of the gene count matrix from kallisto and Cell Ranger. (B) Cell embedding in the first 2 principal components colored by cluster. (C) Cell embedding in tSNE colored by cluster.



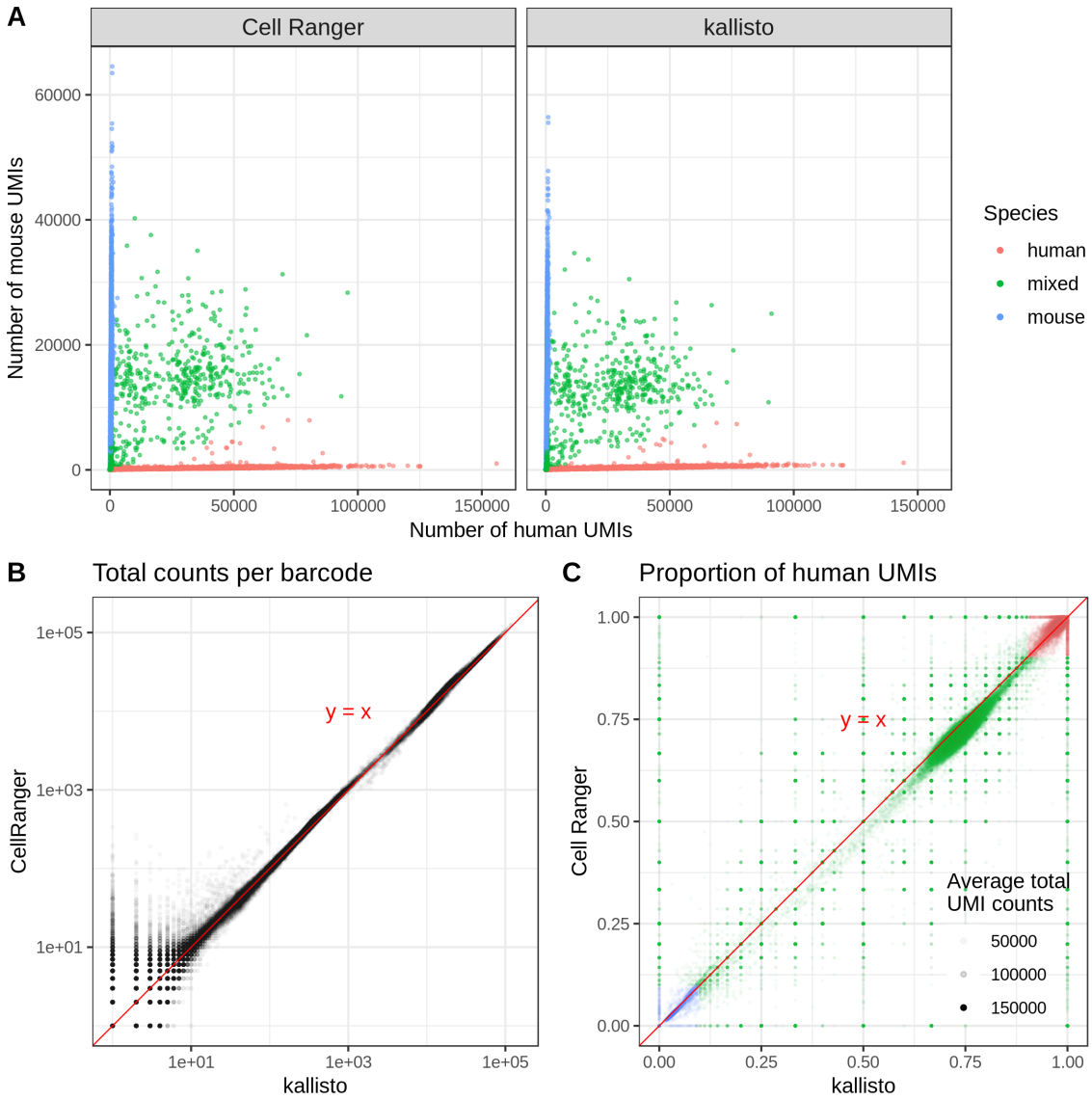
Supplementary Figure 7.2: (A) Number of cells assigned to each cluster by kallisto and Cell Ranger and the correspondence between the clusters. (B) Jaccard indices between each kallisto cluster and each Cell Ranger cluster.



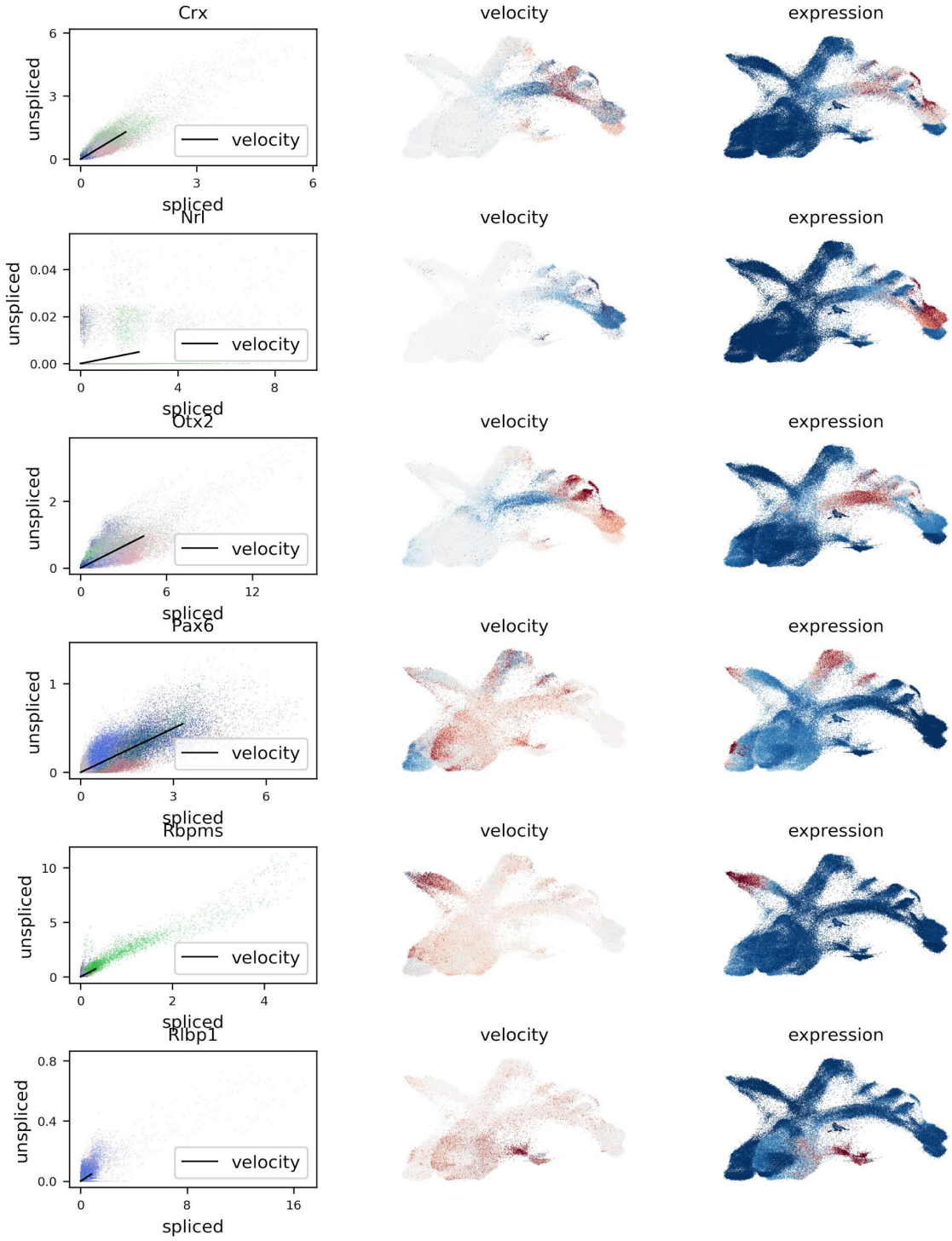
Supplementary Figure 7.3: Number of marker genes with log fold change of at least 0.75 and adjusted $p < 0.05$ in each cluster. (B) Log fold change of marker genes in each cluster. (C) Top 5 enriched GO terms of marker genes (adjusted $p < 0.05$) each cluster.



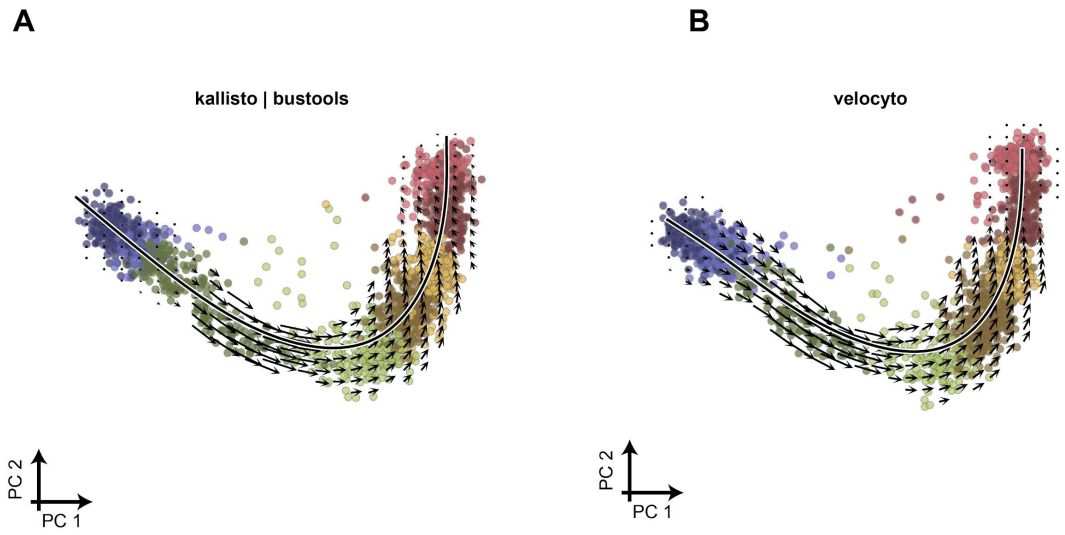
Supplementary Figure 7.4: (A) Histograms of Spearman and Pearson correlation coefficients between barcodes from kallisto and the same barcodes from Cell Ranger for the top 15 marker genes (by log fold change) of each cluster. (B) Spearman and Pearson correlation coefficients, as in (A), for cells in each kallisto cluster. Here cluster 16 corresponds to erythrocytes, while most other cells are neuronal precursor cells.



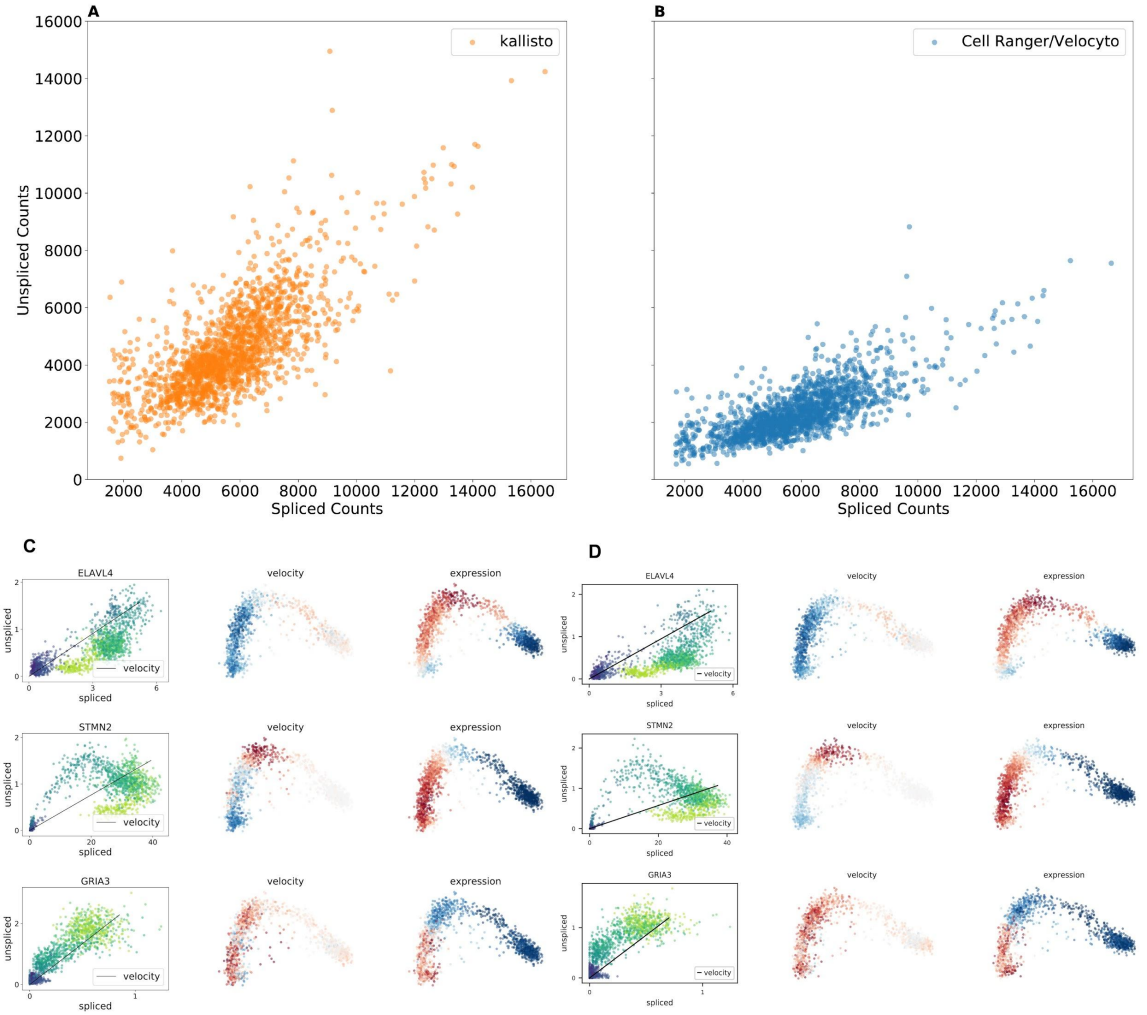
Supplementary Figure 8: Comparison of Cell Ranger to kallisto on the 10x Genomics hgmm10k_v3 species mixing experiment. (A) Barnyard plot with droplets colored according to species of origin: human (red), mouse (blue) and mixed (green). Mixed droplets correspond to cell doublets. (B) The number of total counts per barcode in Cell Ranger and kallisto. (C) The proportion of UMIs in each droplet originating from human. The cluster of droplets in the lower left corner correspond to mouse cells. The cluster of cells in the upper right corner to human cells. The middle band of droplets are doublets. Droplets are shaded according to the number of distinct UMIs they contain.



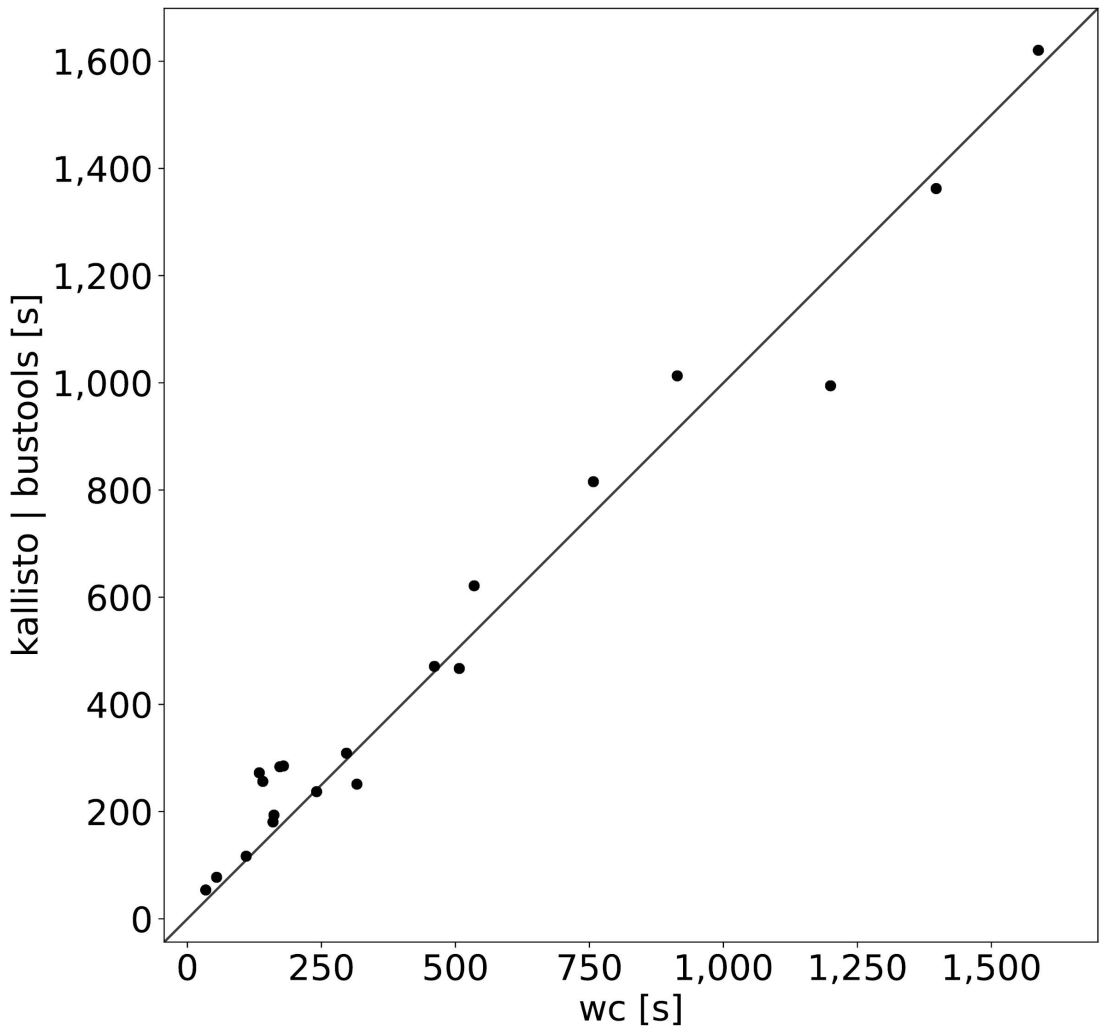
Supplementary Figure 9: Phase diagrams and expression/velocity for six marker genes studied in Clark et al. 2019. The expression results are concordant with pseudotime analysis.



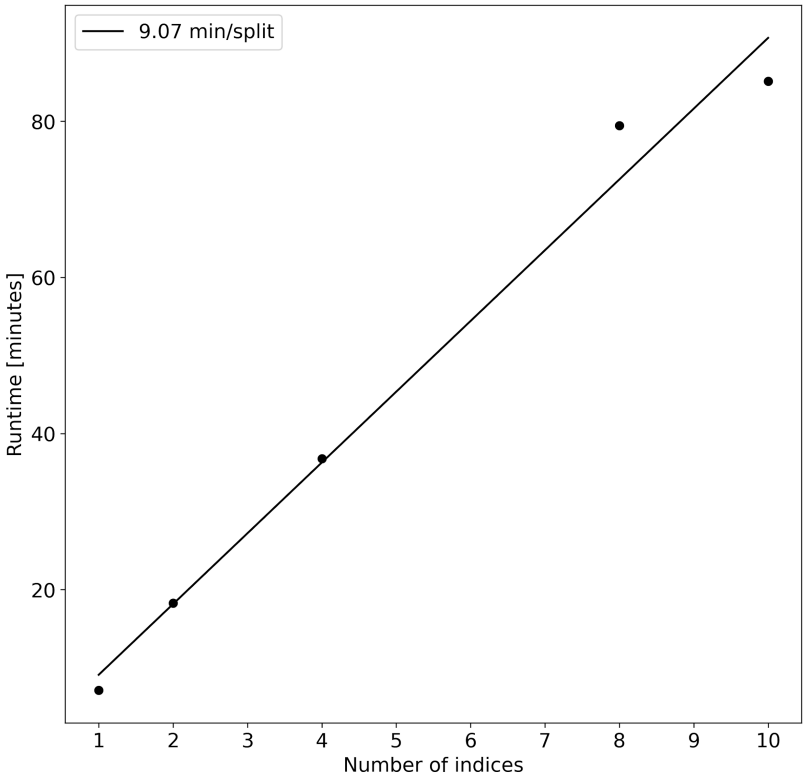
Supplementary Figure 10: (A) RNA velocity based on spliced and unspliced matrices from a dataset of 1,720 human glutamatergic neuron differentiation cells at post-conception week 10. The colors correspond to cell types and intermediate states and a principal "velocity curve" is shown in bold. (A) RNA velocity analysis based on spliced and unspliced matrices computed with kallisto and bustools. B) RNA velocity based on the spliced and unspliced matrices computed with velocyto. Colors correspond to clusters as assigned by the velocyto notebook.



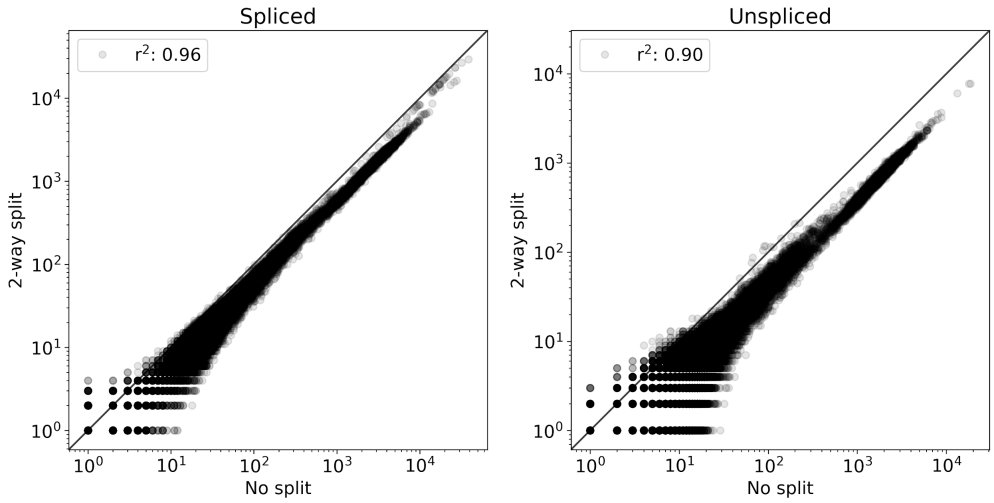
Supplementary Figure 11: Comparison of Cell Ranger and velocyto to kallisto in an RNA velocity analysis of human glutamatergic neuron differentiation cells at post-conception week 10. (A) Number of distinct UMIs from spliced vs. unspliced transcripts from kallisto (orange). (B) Number of distinct UMIs from spliced vs. unspliced transcripts from Cell Ranger (blue). Cell Ranger has similar numbers of spliced counts but fewer unspliced counts. (C) Phase diagrams from the kallisto RNA velocity analysis for 3 genes highlighted in La Manno et al. 2018. (D) Corresponding phase diagrams from the Cell Ranger RNA velocity analysis showing agreement with the kallisto results.



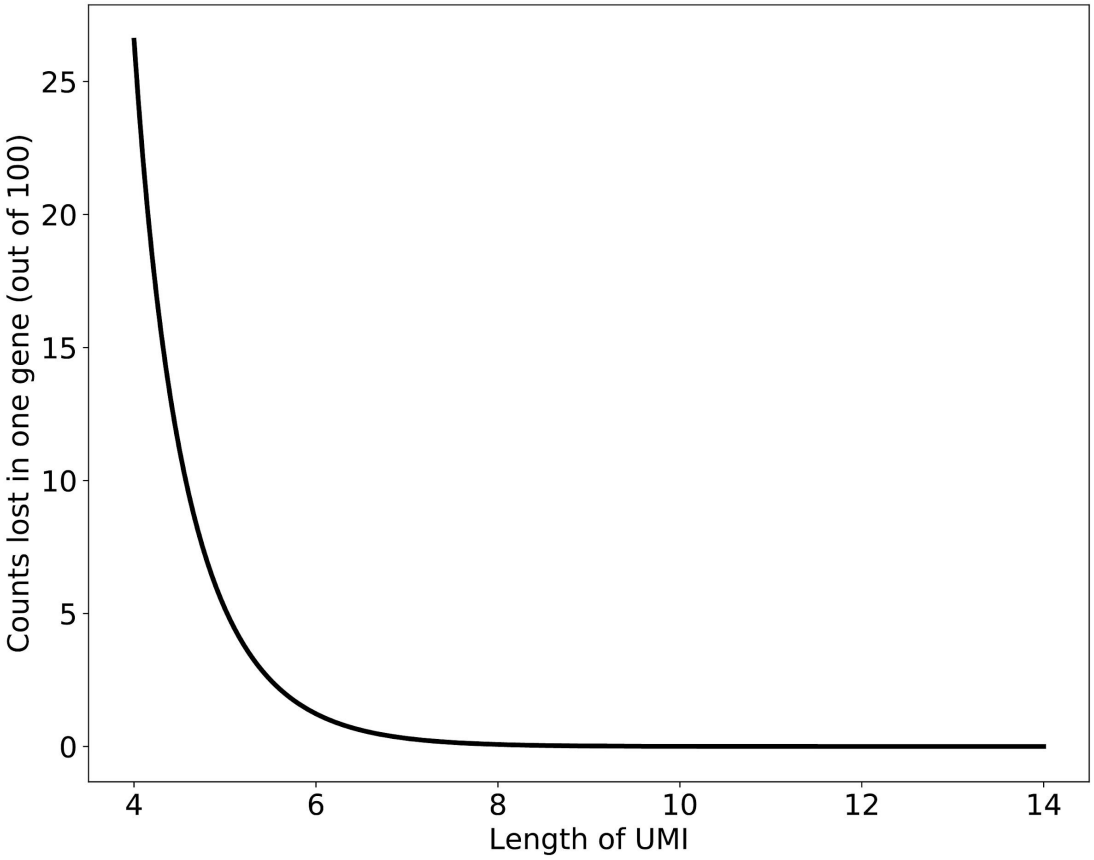
Supplementary Figure 12: Comparison of kallisto runtimes with those of the Unix word count (**wc**) command. Each point corresponds to a different dataset.



Supplementary Figure 13: The runtime to process 50 million reads as a function of the number of indices. The reference transcriptome was split into two, four, eight, and ten parts and the time to align all of the reads to each of the set of indices was recorded.



Supplementary Figure 14: The counts per cell, summed across all genes, when pseudoaligning 50 million single cell reads against the full spliced and unspliced indices and the 2-way split index for spliced and unspliced count matrices. The BUS files generated for the 2-way split index were merged together using bustools mash followed by bustools sort and bustools merge.



Supplementary Figure 15: The number of counts lost due to naïve collapsing of UMIs as a function of the length of the UMIs for a gene with 100 counts. The calculation, based on Supplementary Note equation (11), assumes that the effective number of UMIs is 4^L when UMIs are of length L.

Supplementary Table 1: Runtime, memory, and cost (supplementary_table_S1_runtime_mem_cost.xlsx).

bustools	Description	Enables
capture	Capture records from a BUS file	RNA Velocity
correct	Error correct a BUS file	Barcode Error correction
count	Generate count matrices from a BUS file	Gene count or transcript count matrices
extract	Extract FASTQ reads corresponding to reads in BUS file	FASTQ sampling
inspect	Produce a report summarizing a BUS file	Summary statistics
linker	Remove section of barcodes in BUS files	Excise sections of barcode for custom technologies
mash	Combine BUS records and match EC to the same reference	Combining BUS files from different indices
merge	Merge kmer alignments for a single read	Low memory alignment
project	Project a BUS file to gene sets	Change coordinate system from transcripts to genes
sort	Sort a BUS file by barcodes and UMIs	Constant memory sorting
text	Convert a binary BUS file to a tab-delimited text file	Custom BUS file parsing
whitelist	Generate a whitelist from a BUS file	Technologies without a whitelist

Supplementary Table 2: All of the bustools commands that have been developed and the types of analyses they enable.

Supplementary Table 3: Benchmark panel summary (supplementary_table_S3 _benchmark_panel_summary.xlsx).

Supplementary References

Carosso, G.A., Boukas, L., Augustin, J.J., Nguyen, H.N., Winer, B.L., Cannon, G.H., Robertson, J.D., Zhang, L., Hansen, K.D., Goff, L.A., et al. (2018). Transcriptional suppression from KMT2D loss disrupts cell cycle and hypoxic responses in neurodevelopmental models of Kabuki syndrome: BioRxiv.

Delile, J., Rayon, T., Melchionda, M., Edwards, A., Briscoe, J., and Sagner, A. (2019). Single cell transcriptomics reveals spatial and temporal dynamics of gene expression in the developing mouse spinal cord. Development *146*.

Farrell, J.A., Wang, Y., Riesenfeld, S.J., Shekhar, K., Regev, A., and Schier, A.F. (2018). Single-cell reconstruction of developmental trajectories during zebrafish embryogenesis. Science *360*.

Guo, L., Lin, L., Wang, X., Gao, M., Cao, S., Mai, Y., Wu, F., Kuang, J., Liu, H., Yang, J., et al. (2019). Resolving Cell Fate Decisions during Somatic Cell Reprogramming by Single-Cell RNA-Seq. Mol. Cell *73*, 815-829.e7.

Jin, R.M., Warunek, J., and Wohlfert, E.A. (2018). Chronic infection stunts macrophage heterogeneity and disrupts immune-mediated myogenesis. JCI Insight *3*.

Mahadevaraju, S., Fear, J.M., and Oliver, B. (2019). GEO Accession viewer.

Mays, J.C., Kelly, M.C., Coon, S.L., Holtzclaw, L., Rath, M.F., Kelley, M.W., and Klein, D.C. (2018). Single-cell RNA sequencing of the mammalian pineal gland identifies two pinealocyte subtypes and cell type-specific daily patterns of gene expression. PLoS ONE *13*, e0205883.

Merino, D., Weber, T.S., Serrano, A., Vaillant, F., Liu, K., Pal, B., Di Stefano, L., Schreuder, J., Lin, D., Chen, Y., et al. (2019). Barcoding reveals complex clonal behavior in patient-derived xenografts of metastatic triple negative breast cancer. Nat. Commun. *10*, 766.

Miller, B.C., Sen, D.R., Al Abosy, R., Bi, K., Virkud, Y.V., LaFleur, M.W., Yates, K.B., Lako, A., Felt, K., Naik, G.S., et al. (2019). Subsets of exhausted CD8+ T cells differentially mediate tumor control and respond to checkpoint blockade. Nat. Immunol. *20*, 326–336.

O'Koren, E.G., Yu, C., Klingeborn, M., Wong, A.Y.W., Prigge, C.L., Mathew, R., Kalnitsky, J., Msallam, R.A., Silvin, A., Kay, J.N., et al. (2019). Microglial Function Is Distinct in Different Anatomical Locations during Retinal Homeostasis and Degeneration. Immunity *50*, 723-737.e7.

Packer, J.S., Zhu, Q., Huynh, C., Sivaramakrishnan, P., Preston, E., Dueck, H., Stefanik, D., Tan, K., Trapnell, C., Kim, J., et al. (2019). A lineage-resolved molecular atlas of C. elegans embryogenesis at single cell resolution. BioRxiv.

Ryu, K.H., Huang, L., Kang, H.M., and Schiefelbein, J. (2019). Single-Cell RNA Sequencing Resolves Molecular Relationships Among Individual Plant Cells. Plant Physiol. *179*, 1444–1456.

Supplementary Note

Preliminaries

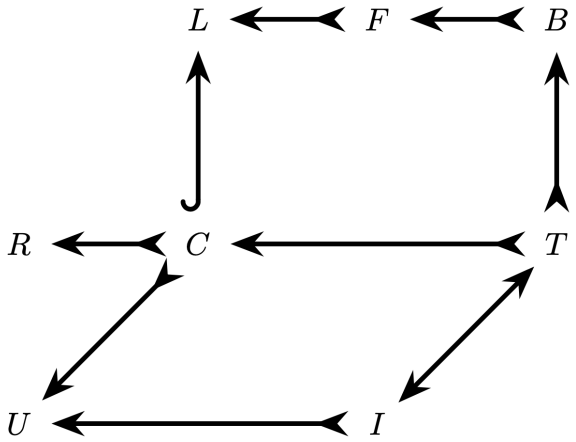


Figure 1: Diagram of sets associated with a cell in a single-cell RNA-seq sequencing experiment.

A single-cell RNA-seq experiment can be described as follows: the goal of the experiment is to identify the ensemble of RNA molecules in multiple cells; in Figure 1 the ensemble of RNA molecules contained within a single cell is denoted by R. To investigate R a library (L) is constructed from the set of molecules captured from R (the set C). Typically, L is the result of various fragmentation and amplification steps performed on C, meaning each element of C may be observed in L with some multiplicity. Thus, there is an inclusion map from C to L, and an injection from C to R. The library is interrogated via sequencing of some of the molecules in L, resulting in a set F of fragments. Subsequently, the set F is aligned or pseudoaligned to create a set B, which in this paper is a BUS file (Melsted, Ntranos, and Pachter 2019). Not every fragment F is represented in B, hence the injection, rather than bijection, from B to F, and similarly from F to L The set T consists of transcripts that correspond to molecules in C that were represented in B. Note that $|R| >= |C| >= |T|$. Separately, the set U consists of the UMIs on the bead that the cell was trapped with, and I is a multiset of UMIs associated with transcripts in C and UMIs in U that are in B (Table 1).

R : multiset of all RNAs in a cell.
L : multiset of all molecules in the library.
F : multiset of reads.
B : multiset of {barcode, UMI, equivalence class} triplets.
C : multiset of captured RNA molecules represented in the library.
T : multiset of transcripts represented in B corresponding to molecules in C.
U : set of UMIs on a bead.
I : multiset of UMIs represented in B corresponding to molecules in U.

Table 1: Notation for description of a single-cell experiment

The data in a single-cell experiment consists of the sets *F* for each cell. In our workflow, a combined BUS file (merge of the sets *B*) is generated using kallisto (Bray et al. 2016). While the multiset *I* is not directly measured, its support *supp(I)* (the set of distinct UMIs) can be extracted from the BUS file. The goal of single-cell RNA-seq pre-processing is to infer the multiset *T*. What we describe in this note is an approach to estimating two different quantities: the effective size $|U|$ of the set of UMIs associated with each bead, and the number of captured molecules represented in the BUS file, i.e. $|I|$ or equivalently $|T|$. Specifically, we are interested in the restriction of the latter to individual genes in cells, for the purpose of estimating the error in the number of counts that can be introduced when naïvely collapsing UMIs by gene. The reason for estimating $|U|$ is that it is necessary to estimate $|I|$.

Modeling an experiment

The number of distinct UMIs on one bead is at most *4^L* where *L* is the number of UMI bases (10xv2 technology uses *L=10* and 10xv3 technology *L=12*). For a bead captured along with a cell in a droplet, we denote the number of UMIs on the bead by $n=|U|$. We model the process by which UMIs are associated with molecules as follows: each UMI is selected by sampling uniformly at random from the set of UMIs *U*. In other words, the molecules are labeled with UMIs by sampling with replacement. This model has been used previously (Grün, Kester, and van Oudenaarden 2014), and is justified by distributions of UMIs seen empirically (Figure 2). If $k=|I|$ is the number of UMIs represented in *B* corresponding to molecules in *U* derived from a single droplet then the assumption of uniform random sampling of UMIs from the bead implies that the probability that a specific UMI is observed zero times is *(1-1/n)^k*. Therefore the expected number of UMIs observed at least once, i.e. the expected number of distinct UMIs in a cell, is

$$n\left(1-\left(1-\frac{1}{n}\right)^{k}\right). \qquad (1)$$

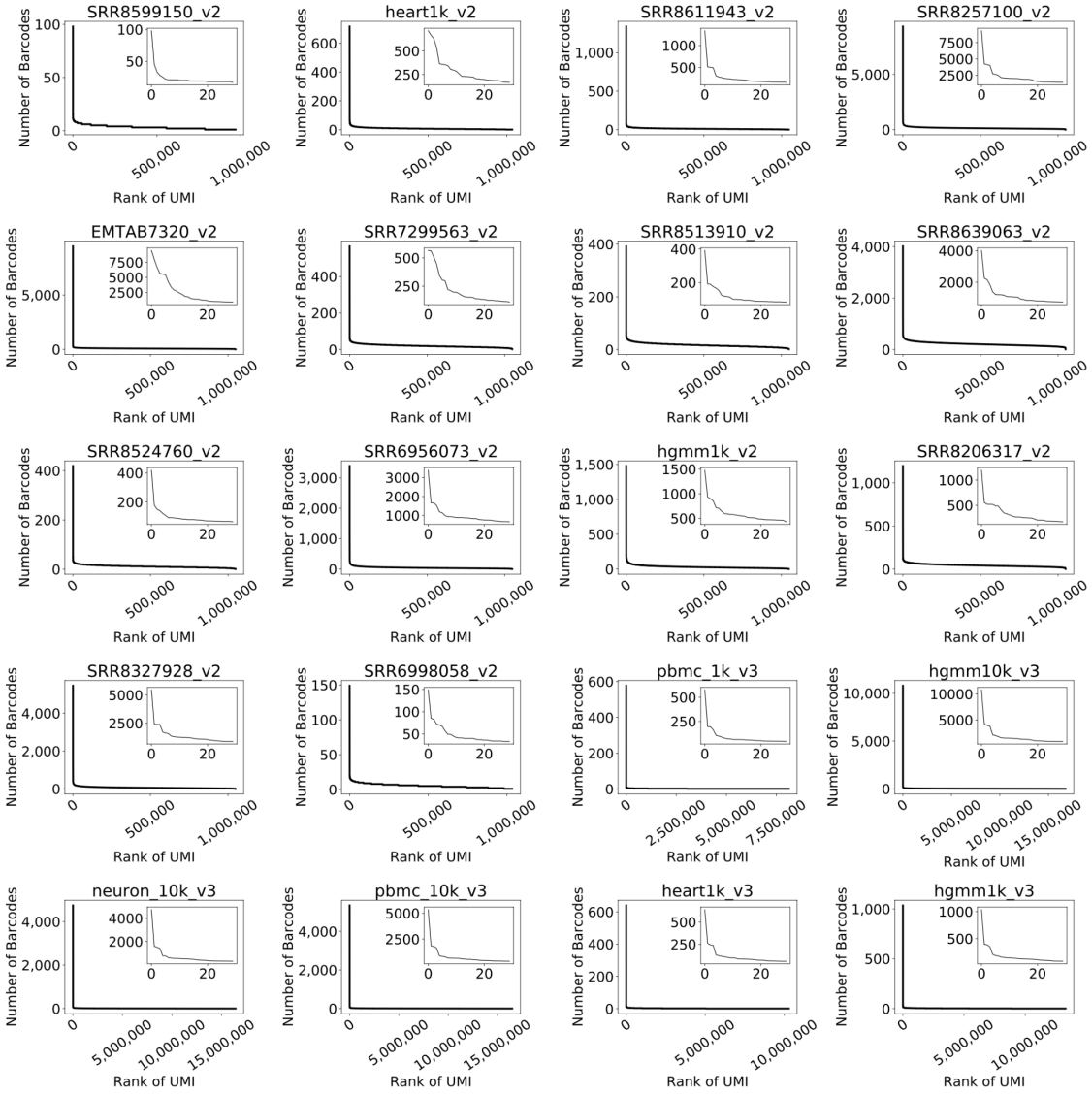


Figure 2: Distribution of UMIs across cells. With the exception of a handful of artifacts, UMIs are uniformly distributed across cells.

Estimating the effective number of UMIs

To estimate $n\ (=|U|)$ we utilize two observations:

1. Reads that originated from different genes correspond to distinct molecules, so if they share the same UMI then the UMI was sampled more than once (i.e. the UMI is not duplicated due to PCR).
2. While the number of sampled UMIs k is unknown, the number of distinct UMIs can be measured directly.

We say that a UMI that has been sampled more than once has a *collision* (Figure 3), and we denote the number of UMIs that appear in more than one gene by random variable

(r.v.) *X* (see Figure 3). We denote the number of distinct UMIs sequenced, i.e. *|supp(I)|*, by a r.v. *D*, and the number of distinct UMIs observed to be from a gene *g* by r.v. *D_g*. We denote the number of sampled molecules originating from a gene *g* by *k_g*. Note that $\sum_g E[d_g] >= E[d]$.

We obtain method-of-moment estimates for the parameters *k,k_g* and *n* (*k_hat,k_g_hat* and *n_hat*) by relating them to realizations of the r.v. *D_g, D,* and *X*. First, from equation (1) (see also (Grün, Kester, and van Oudenaarden 2014)), we have that

$$\mathbb{E}[D] = n\left(1 - \left(1 - \frac{1}{n}\right)^{k}\right), \tag{2}$$

at the gene level,

$$\mathbb{E}[D_g] = n\left(1 - \left(1 - \frac{1}{n}\right)^{k_g}\right). \tag{3}$$

The number of UMIs that occur in more than one gene can be found by knowing the number of UMIs that are seen zero times in all genes, and that the number of UMIs that are seen in only one gene is given by the number of unique UMIs in gene *g* and only gene *g*, summed across all genes. This gives an estimate for the number of UMIs that collide between genes:

$$\mathbb{E}[X] = n\left(1 - \left(\left(1 - \frac{1}{n}\right)^{k} + \sum_{g}\left(1 - \left(1 - \frac{1}{n}\right)^{k_g}\right) \cdot \left(1 - \frac{1}{n}\right)^{k - k_g}\right)\right). \tag{4}$$

From equations (2) and (3) and using the realizations of the r.v. *D* and *D_g*, i.e. *d* and *d_g*, we have that

$$\left(1 - \frac{1}{\hat{n}}\right)^{\hat{k}} = \frac{\hat{n} - d}{\hat{n}} \tag{5}$$

and at the gene level

$$\left(1 - \frac{1}{\hat{n}}\right)^{k_g} = \frac{\hat{n} - d_g}{\hat{n}}. \tag{6}$$

Therefore, substituting equations (5), (6) into equation (4) we obtain

$$x = \hat{n}\left(1 - \left(\frac{\hat{n}-d}{\hat{n}} + \sum_g \left(1 - \frac{\hat{n}-d_g}{\hat{n}}\right) \cdot \left(\frac{\hat{n}-d}{\hat{n}-d_g}\right)\right)\right) \quad (7)$$

$$= \hat{n}\left(\frac{d}{\hat{n}} - \frac{\hat{n}-d}{\hat{n}} \sum_g \left(\frac{d_g}{\hat{n}-d_g}\right)\right) \quad (8)$$

$$= d - (\hat{n}-d) \sum_g \left(\frac{d_g}{\hat{n}-d_g}\right). \quad (9)$$

Since *d*, *d_g* (for all *g*) and *x* are known the number of UMIs, *n_hat*, can be estimated.

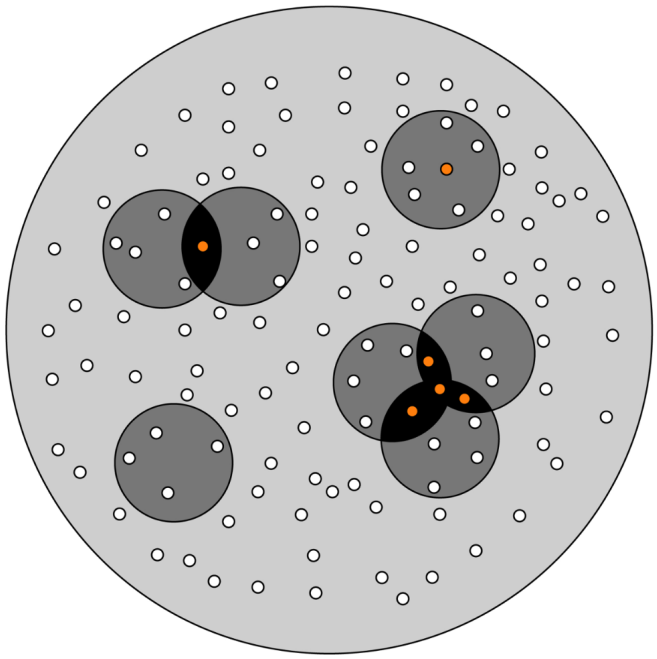


Figure 3: Collisions of UMIs. Each small circle represents a distinct UMI. Each medium sized circle is a gene, and the enclosing circle is the set of all distinct UMIs. UMIs that have collided are shown in orange. Inter-gene collisions consist of UMIs present in two or more genes. An intra-gene collision is also shown.

Estimating counts lost for each gene

Returning to equation (3), we see that

$$\hat{k_g} = \frac{ln\left(1 - \frac{d_g}{\hat{n}}\right)}{ln\left(1 - \frac{1}{\hat{n}}\right)}. \quad (10)$$

With *n_hat*, and measurement of *d_g*, we evaluate the number of molecules captured per gene, *k_g_hat*. The loss of counts due to collapsing of UMIs by gene is

$$\hat{k_g} - d_g = \frac{ln\left(1 - \frac{d_g}{\hat{n}}\right)}{ln\left(1 - \frac{1}{\hat{n}}\right)} - d_g \quad (11)$$

$$\approx \frac{d_g(d_g - 1)}{2\hat{n} + 1}, \quad (12)$$

where 12 is found by Taylor expanding 11.

Constant and low memory processing

The kallisto bustools workflow enables constant and low memory single-cell RNA-seq pre-processing by using small pseudoalignment reference indices, and streaming all processing of BUS records which is possible after a constant memory sort of the initial BUS file produced in an analysis. In order to pseudoalign reads, kallisto first loads up a small index file constructed from a reference transcriptome. The size of this index is not dependent on the number of reads that will be processed. Reads are pseudoaligned by streaming through FASTQ files, and BUS records are incrementally added to as reads are processed. The bustools commands operate on BUS files and are used to perform many required operations on BUS files in order to generate count matrices. These operations include sorting the BUS file, correcting barcodes, and counting UMIs among many others; all operations are performed in constant memory in the number of reads being processed. The first step in working with BUS files is sorting. Sorting the BUS file allows all other bustools to operate on the BUS file in a stream-wise fashion thus keeping memory constant and low. The `bustools sort` command operates in constant memory by utilizing disk when necessary.

While pseudoalignment of reads and processing of BUS files to perform RNA-velocity has only constant memory requirements (in terms of the number of reads) with the kallisto bustools workflow, the indices involved can be large due to the intronic sequences that must be indexed. The modularity of bustools makes possible, in principle, a reduction in absolute memory requirements by virtue of splitting the target sequences prior to indexing, pseudoalignment to the separate indices, and finally merging of the resultant BUS files. We implemented this strategy, which required modifying the kallisto bustools workflow to first align reads to a transcriptome that has been split into an arbitrary number (*n*) of parts and then merging the alignments by interval set intersection. Splitting the transcriptome into *n* parts yields smaller indices to be loaded into memory and requires *n* alignments of the reads which comes with a run-time trade-off (Supplementary Figure 13). For each read that aligns, we record the interval of kmer start positions from the read such that the kmers contained within this interval align to an associated equivalence class. A single read of length L, with a kmer size of *k* can have at

most *L-k+1* possible kmer alignments where each possible kmer in that read aligns to a different equivalence class. We then merge these intervals appropriately in order to assign an equivalence class to the read.

By way of example, suppose that we split an index into three parts and perform pseudoalignment three times. Additionally, suppose that a single read has only two kmers that align, *k_1* and *k_2*. *k_1* aligns to an equivalence class which contains transcripts one and two (*EC_1={T_1, T_2}*) in index one and *k_1* also aligns to *EC_2={T_7, T_9}*. The second kmer *k_2* aligns to *EC_4={T_5, T_6, T_7}* of index two and *EC_5={T_10, T_11)* of index three.

In the case of a full transcriptome, *EC_1* and *EC_2* would have been indexed together since they share the same kmer, *EC_1,2 = {T_1, T_2, T_7, T_9}* and *k_1* would have aligned to this equivalence class. Similarly, in the case of a full transcriptome, *EC_4* and *EC_5* would have have been indexed together since they share the same kmer, *EC_4,5 = {T_5, T_6, T_7,T_10*, *T_11}* and *k_2* would have aligned to this equivalence class. The read would then have been assigned the equivalence class *EC= EC_1,2 ∩ EC_7,9 = T_7*.

In the case of the split indices, in order to accurately assign the read to this equivalence class we must:

1. determine all of the kmer alignments for a single read from each index,
2. appropriately merge overlapping kmer alignments and equivalence classes,
3. determine the set of elementary intervals (note: Given a list of intervals where for any interval the left endpoint is smaller than the right endpoint, an elementary interval is defined as any interval from the set of intervals constructed by taking every adjacent pair of points from a sorted list of unique endpoints. E.g. *I={[3, 5), [0, 4), [4, 9)}* and *E={[0, 3), [3,4), [4, 5), [5, 9)}})* and the set of equivalence classes contained within those intervals,
4. and intersect all of the elementary intervals.

A single read can have multiple kmers align to multiple equivalence classes in each of the *n* split indices. To keep track of these alignments we store a 0-indexed interval with endpoints corresponding to kmer start positions on the read and the equivalence class corresponding to that interval. Note that the interval is closed on the left and open on the right.

After *n* separate alignments, we combine all of the *n* BUS files into one BUS file by simply remapping the equivalence class so that the set of transcripts defined by an

equivalence is based on the combined transcripts from all *n* splits instead of just the transcripts from each separate split.

For all of the BUS records corresponding to a single read, we find the set of elementary intervals and the set of transcripts corresponding to each interval, and then intersect the intervals to ultimately assign an equivalence class to the read.

The example above for the split indices would then result in the following alignment (superscript corresponds to the index number that the equivalence class is from):

1. Find split alignments. *k_1: EC_1^1={T_1, T_2}, EC_2^2={T_7, T_9}* and *k_2: EC_4^2={T_5, T_6, T_7}, EC_5^3={T_{10}, T_{11})*
2. Merge. *k_1: EC_1^1 ∪ EC_2^2 = EC_1,2 = {T_1, T_2, T_7, T_9}* and *k_2: EC_4^2* ∪ EC_5^3 = EC_4,5 = {T_5, T_6, T_7,T_10, T_11}
3. Intersect. *EC = EC_1,2 ∩ EC_7,9 = {T_7}*

To validate this approach, we split the human polyadenylated transcriptome, as well as an intronic sequences used for RNA velocity into two parts respectively and indexed each part. We then aligned 50 million reads to the four indices and merged the resultant BUS files as described above. Additionally we aligned the reads to the full indices. We then computed the cell-count correlation between the quantification generated with the full indices and the quantification generated with the separate indices and found the results to be highly concordant (*r^2=0.97* for counts from the polyadenylated transcriptome and *r^2=0.90* for counts from the intronic sequences, Supplementary Figure 14).

The results obtained from merging BUS records that were pseudoaligned to split indices will not necessarily exactly recapitulate the results obtained from pseudoaligning to the full index. This is due to the ambiguity introduced when kmers from a single read map to multiple transcripts. When reducing the number of transcripts in the index in each split index, there are fewer sets of shared k-mers between transcripts. Given a k-mer alignment to an equivalence class in a read, the strategy in kallisto is to skip ahead in the index graph and check if the final k-mer in the read maps to the same equivalence class, a different equivalence class, or none at all. This skip ahead strategy, while appropriate for the full index, can skip intermediate k-mer alignments that only result from an equivalence class in the full transcriptome, thereby resulting in fewer alignments and a slight loss in pseudoalignments when splitting indices and subsequently merging results (Supplementary Figure 14).

Supplementary Note References

Bray, Nicolas L., Harold Pimentel, Páll Melsted, and Lior Pachter. 2016. “Near-Optimal Probabilistic RNA-Seq Quantification.” *Nature Biotechnology* 34 (5): 525–27.

Grün, Dominic, Lennart Kester, and Alexander van Oudenaarden. 2014. “Validation of Noise Models for Single-Cell Transcriptomics.” *Nature Methods* 11 (6): 637–40.

Melsted, Páll, Vasilis Ntranos, and Lior Pachter. 2019. “The Barcode, UMI, Set Format and BUStools.” *Bioinformatics* 35 (21): 4472–73.

Section 2: count data normalization with log(x+1)* and log(1+x)*

* These formulae contributed equally to the title

Preamble

Methods to perform single-cell RNA-sequencing analysis often apply variance stabilizing transformations in order to reduce unwanted technical variation. We show, for the first time, that one such technique, log1p, can disproportionately weight genes with low counts and produce misleading results in analysis of SARS-CoV-2 scRNAseq data.

Summary

Single-cell RNA-seq technologies have been successfully employed over the past decade to generate many high resolution cell atlases. These have proved invaluable in recent efforts aimed at understanding the cell type specificity of host genes involved in SARS-CoV-2 infections. While single-cell atlases are based on well-sampled highly-expressed genes, many of the genes of interest for understanding SARS-CoV-2 can be expressed at very low levels. Common assumptions underlying standard single-cell analyses don't hold when examining low-expressed genes, with the result that standard workflows can produce misleading results.

Introduction

The *ACE2* receptor, which facilitates entry of SARS-Cov-2 into cells (Zhang *et al.*, 2020), has become one of the most studied genes in the history of genomics over the past two months. There are already hundreds of preprints about the gene (Google Scholar), and it is currently the default gene displayed on the UCSC genome browser (Lee *et al.*, 2020). Several studies have reported on the expression of *ACE2* at single-cell resolution, and papers have been rife with speculation about implications of differential *ACE2* mRNA abundance for severity of disease. As is common in single-cell RNA-seq, the expression estimates of *ACE2* are derived from counts that are filtered and normalized.

Results

Figure 1a shows an analysis of *ACE2* mRNA in mice lungs (data from (Angelidis *et al.*, 2019)). The expression is computed from cells containing at least one copy of the gene. To understand the implication of this restriction, suppose the counts are modeled by a Poisson random variable X with parameter λ. Application of the filter amounts to computing

$$E[X|X > 0] = \frac{\lambda}{1 - e^{-\lambda}}.$$

While this is approximately λ when λ is large, it is close to 1 when λ is small (de L'Hospital, 1768). Figure 1b shows the fraction of cells containing at least one copy of *ACE2* (Booeshaghi and Pachter, 2020). Evidently, Figure 1a creates a misleading impression. In fact, *ACE2* has significantly lower mRNA expression in the lungs of aged mice than young mice.

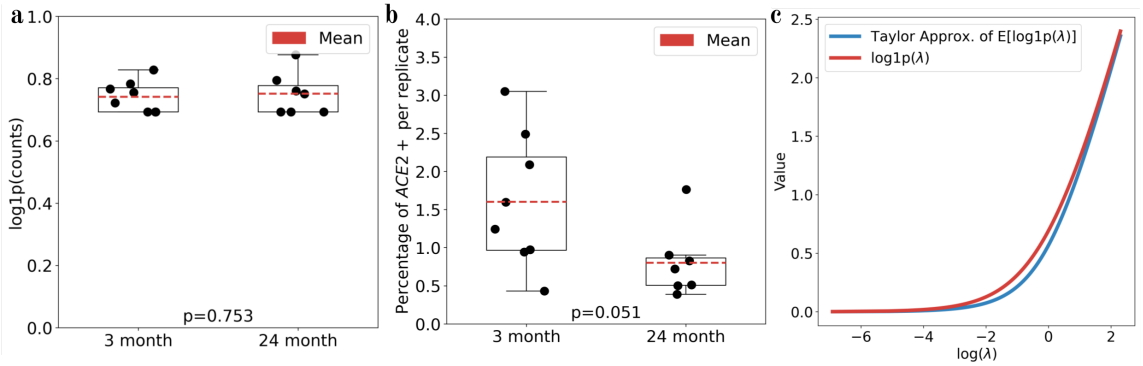


Figure 1: a) Changes in *ACE2* expression in the lungs of eight 3-month old mice and seven 24-month old mice after log1p transformation of the raw counts on the cells with non-zero *ACE2* expression. The *p*-value was computed using a *t*-test. b) Changes in *ACE2* expression as determined by the fraction of *ACE2* positive cells. The *p*-value was computed using a *t*-test. c) A comparison of the naïve estimate of the expectation of log1p (red) to the Taylor approximation of the expectation of log1p (blue). The code to produce the panels in the figure is available here.

The fraction f of cells with nonzero expression of a gene has a useful statistical interpretation. We leave it as an exercise for the reader to show that the the following estimator for the Poisson rate is consistent:

$$\hat{\lambda} = -\log(1 - f).$$

Since f is approximately equal to this expression when is f small, this provides an interpretation of the fraction of cells with at least one copy of a low-abundance gene as an estimate of the rate parameter λ in a Poisson distribution.

Another mistake that we've found to be common in reporting *ACE2* expression has to do with the log transformation, frequently used as part of a normalization of counts. Counts are log transformed for two reasons: the first is to stabilize the variance, as the log transform has the property that it stabilizes the variance for random variables whose variance is quadratic in the mean (Bartlett, 1947). The rationale of this step for single-cell RNA-seq is manifold: first when performing PCA on the gene expression matrix to find a reduced-dimensional representation that captures the variance, it is desirable that all genes contribute equally. The second rationale for the log transform is that it converts multiplicative relative changes to additive differences. In the context of PCA, this allows for interpreting the projection axes in terms of relative, rather than absolute, abundances of genes.

A seemingly minor technical issue in log transforming counts is that zero counts cannot be "logged", as log(0) is undefined. To circumvent this problem, it is customary to add a "pseudocount", e.g. +1, to each gene count prior to log transforming the data. We denote this by log1p (see units of Figure 1a), in accordance with nomenclature standard in scientific computing (Liu, 1988). For a gene with an average of λ counts where λ is large, it is intuitive that the average of the log1p transformed counts is approximately $\log(\lambda)$. However, this is not true for

small λ. An understanding of the result of applying the log1p transform begins with the observation that for a random variable X, $E[f(X)]$ is not, in general, equal to $f(E[X])$. For example, if X is a Poisson random variable with parameter λ, it is not true that $E[log(1 + X)] = log(\lambda + 1)$. By Taylor approximation,

$$E[\log(X+1)] \approx \log(E[X+1]) - \frac{E[X]}{2(E[X]+1)^2},$$
$$= \log(\lambda+1) - \frac{\lambda}{2(\lambda+1)^2}.$$

This shows that for low-expressed genes, the average log1p expression differs considerably from $\log(\lambda)$ (see Figure 1c). Thus, while a 2-fold change for large λ translates to a log(2) difference after log1p, that is not the case for small λ.

Discussion

In summary, while single-cell RNA-seq atlases offer detailed information about the transcriptomic profiles of distinct cell types, their use to examine specific genes, as has been done recently in the study of SARS-CoV-2 infection related genes, requires care. Methods should not be used unless their limitations are understood. For example, while it doesn't matter whether one uses $\log(x+1)$ or $\log(1+x)$, the filtering and normalization applied to counts can affect comparative estimates in non-intuitive ways. Moreover, there are subtle problems that arise when working with small counts that transcend the elementary issues we have raised (Warton, 2018; Lun, 2018). These matters are not theoretical; we leave the identification of published preprints and papers that have ignored the issues we've raised, and hence reported misleading results, as another exercise for the reader.

Methods

Code and data to reproduce all analysis in this paper can be found on the GitHub repository: https://github.com/pachterlab/BP_2021_2.

Acknowledgments

We thank Charles Herring, Michael Hoffman, Harold Pimentel, Jeffrey Spence, and Valentine Svensson for helpful comments.

References

Angelidis, I. *et al.* (2019) An atlas of the aging lung mapped by single cell transcriptomics and deep tissue proteomics. *Nat. Commun.*, **10**, 963.

Bartlett, M.S. (1947) The Use of Transformations. *Biometrics*, **3**, 39–52.

Booeshaghi, A.S. and Pachter, L. (2020) Decrease in ACE2 mRNA expression in aged mouse lung. *bioRxiv*, 2.

Lee, C.M. *et al.* (2020) UCSC Genome Browser enters 20th year. *Nucleic Acids Res.*, **48**, D756–D761.

de L'Hospital, G.F.A. (1768) Analyse des infiniment petits Moutard.

Liu, Z.A. (1988) Berkeley Elementary Functions Test Suite.
Lun, A. (2018) Overcoming systematic errors caused by log-transformation of normalized single-cell RNA sequencing data. *bioRxiv*, 404962.
Warton, D.I. (2018) Why you cannot transform your way out of trouble for small counts. *Biometrics*, **74**, 362–368.
Zhang, H. *et al.* (2020) Angiotensin-converting enzyme 2 (ACE2) as a SARS-CoV-2 receptor: molecular mechanisms and potential therapeutic target. *Intensive Care Med.*, **46**, 586–590.

Chapter 4

Analysis of the mouse primary motor cortex

Preamble

Current single-cell RNA-sequencing processing produces matrices that aggregate counts at the gene level. To date, atlas-scale isoform-level scRNAseq analysis is lacking primarily because tools to generate isoform-level quantifications are lacking. I have extended the *kallisto* | *bustools* workflow to process full-length single-cell RNA-sequencing taking advantage of expectation maximization algorithm to disambiguate ambiguous sequence alignments. I demonstrate a framework for inferring spatially resolved isoform expression using the newly computed *cell* by *isoform* matrices, along with data from two other single-cell RNA-sequencing technologies, and demonstrate the application of this framework to develop the first ever spatially resolved single-cell isoform atlas of the mouse primary motor cortex.

Summary

Full-length SMART-Seq[1] single-cell RNA-seq can be used to measure gene expression at isoform resolution, making possible the identification of isoform markers for cell types and for an isoform atlas. Used in conjunction with spatial RNA capture and gene tagging methods, spatially resolved cell type isoform expression can be inferred. In a comprehensive analysis of 6,160 mouse primary motor cortex cells assayed with SMART-Seq, 280,327 cells assayed with MERFISH[2], and 94,162 cells assayed with 10x Genomics[3], we find examples of isoform specificity in cell types, including isoform shifts between cell types that are masked in gene-level analysis as well as examples of transcriptional regulation. Additionally, we show that isoform specificity helps to refine cell types, and that a multi-platform analysis of single-cell transcriptomic data leveraging multiple measurements provides a comprehensive atlas of transcription in the mouse primary motor cortex that eclipses what is possible with any single technology.

Introduction

Transcriptional and post-transcriptional control of individual isoforms of genes is crucial for neuronal differentiation[4–8], and isoforms of genes have been shown to be specific to cell types in mouse and human brains[9–14]. It is therefore not surprising that dysregulation of splicing has been shown to be associated with several neurodevelopmental and neuropsychiatric diseases[6,15,16]. As such, it is of interest to study gene expression in the brain at single-cell and isoform resolution.

Nevertheless, current single-cell studies aiming to characterize cell types in the brain via single-cell RNA-seq (scRNA-seq) have relied mostly on gene-level analysis. This is, in part, due to the nature of the data produced by the highest throughput single-cell methods. Popular high-throughput assays such as Drop-seq [17], 10x Genomics' Chromium [3], and inDrops [18], produce 3'-end reads which are, in initial pre-processing, collated by gene to produce per-cell gene counts. The SMART-Seq scRNA-seq method introduced in 2012[19] is a full-length scRNA-seq method, allowing for quantification of individual isoforms of genes with the expectation-maximization algorithm[20]. However, such increased resolution comes at the cost of throughput; SMART-Seq requires cells to be deposited in wells, thus limiting the throughput of the assay. In addition, SMART-Seq requires more sequencing per cell [21].

The tradeoffs are evident in analysis of scRNA-seq data from the primary motor cortex (MOp) produced by the BRAIN Initiative Cell Census Network[22]. We examined 6,160 (filtered) SMART-Seq v4 cells and 94,162 (filtered) 10x Genomics Chromium (10xv3) cells (Extended Data Fig. 1 and Figure 2a,b) and found that while 10xv3 and SMART-Seq are equivalent in defining broad classes of cell types, 3'-end technology that can assay more cells can identify some rare cell types that are missed at lower cell coverage (Extended Data Fig. 2a). Overall 56 clusters with gene markers could be identified in the 10xv3 data but not in the SMART-Seq data while only 39 clusters with gene markers could be identified in the SMART-Seq data and not the 10xv3 data, and this differential is consistent with previously reported comparisons of 10x Genomics Chromium and SMART-Seq clusters[23] [21]. However, while SMART-Seq has lower throughput than some other technologies, it has a significant advantage: by virtue of probing transcripts across their full length, SMART-Seq makes possible isoform quantification and the detection of isoform markers for cell types that cannot be detected with 3'-end technologies (Extended Data Figs. 2b,c) . Moreover, the uniformity of read coverage of SMART-Seq data [1] and its quantification with state-of-the-art tools [24] yields higher sensitivity than other methods, which can make possible refined cell type classification.

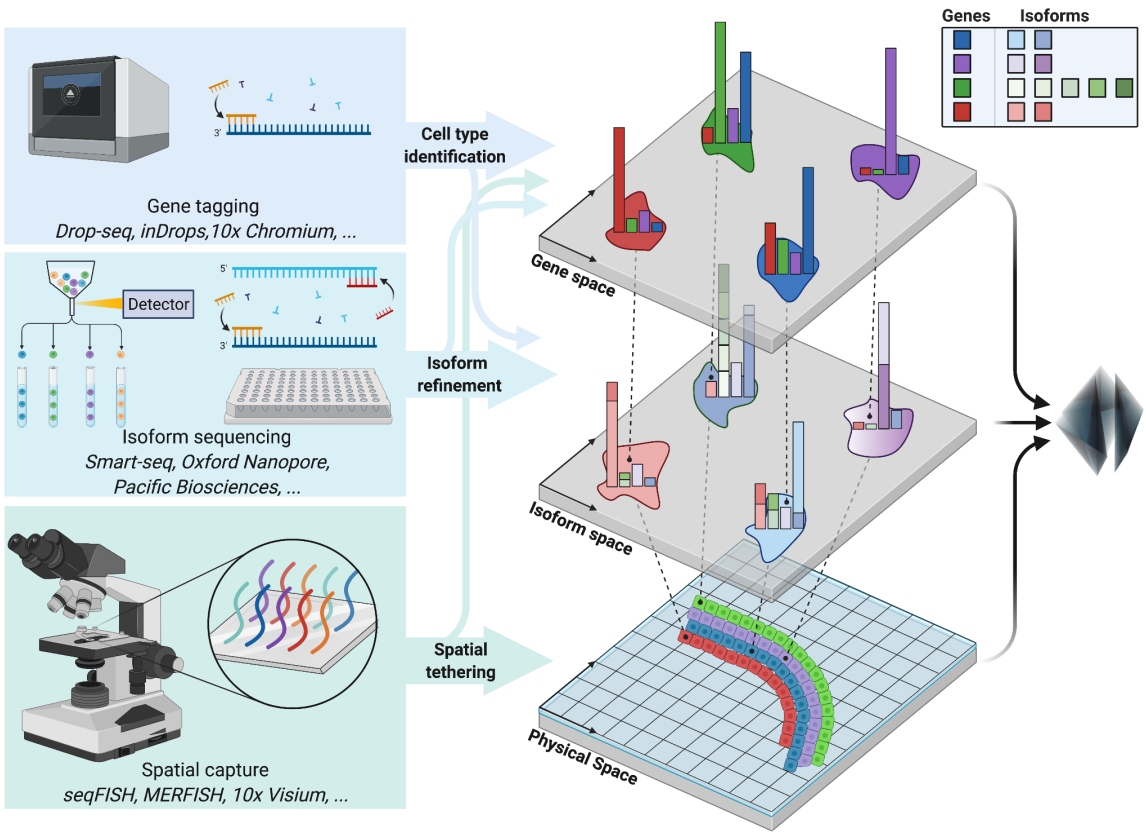


Figure 1: Measuring RNA with multiple platforms. RNA is measured using gene tagging techniques such as the 10x Chromium single-cell RNA-seq protocol, isoform sequencing techniques such as SMART-Seq, and spatial RNA capture techniques such as MERFISH. High cell-throughput gene tagging enables cell type identification with marker genes and deep full-length isoform sequencing enables cell type marker refinement with isoform markers. Spatial RNA capture coupled with gene tagging and isoform sequencing enables spatially resolved cell type specific isoform markers. The multimethod procedure for sampling RNA enables spatially resolved cell type specific isoform inference that no single technology could achieve independently [51].

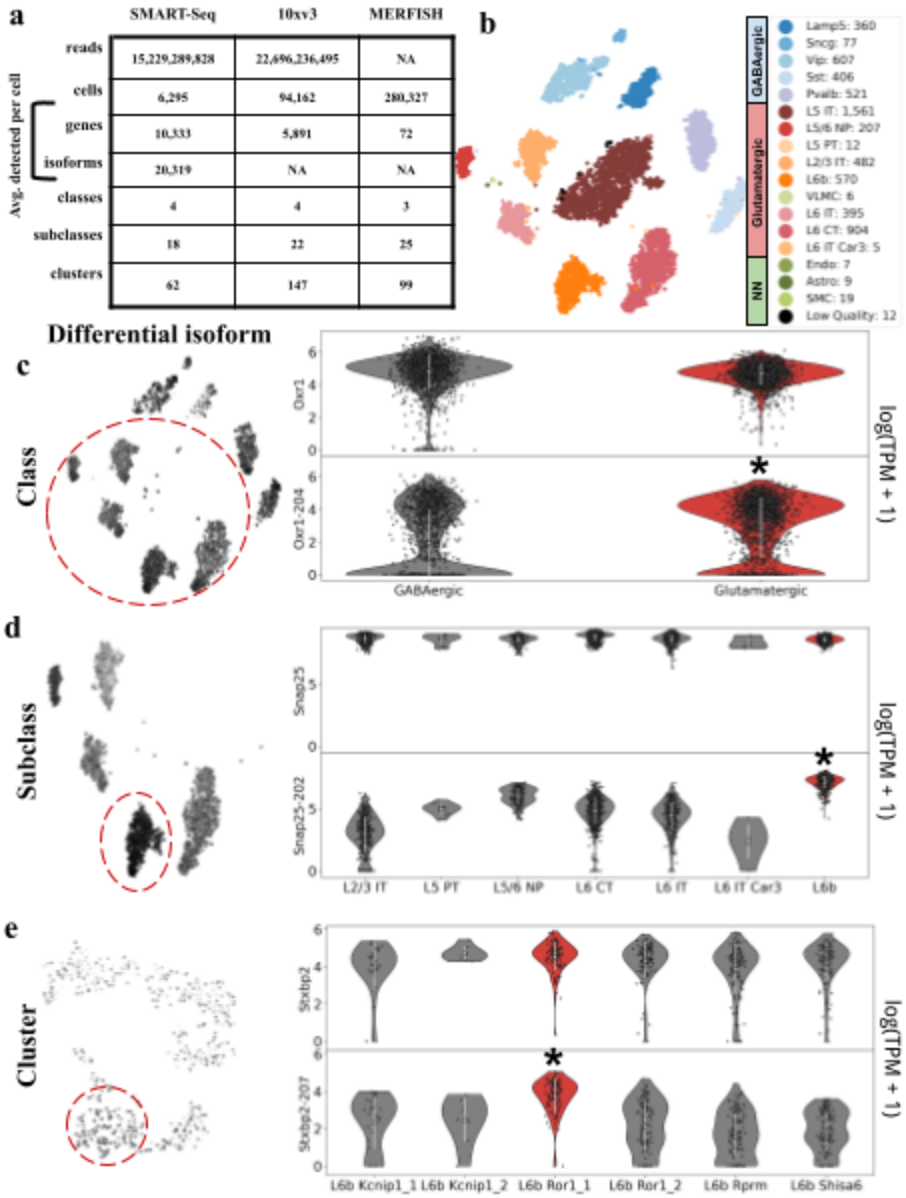


Figure 2: Isoform specificity in the absence of gene specificity. a) Overview of data analyzed. The clustering method used by the BICCN consortium generates three hierarchies of cells: classes, subclasses, and clusters. b) A t-SNE of 10 neighborhood components of 6,160 SMART-Seq cells colored according to subclass. c) Example of a gene with an isoform specific to the Glutamatergic class. The *Oxr1-204* isoform distribution in log1p(Transcripts per million (TPM)) across cells (left) painted on the t-SNE of the cells. The cells belonging to the Glutamatergic class are circled. The violin plots of the gene and isoform distributions show that the gene is not differential but the isoform is (right). d) Example of a gene with an isoform specific to the L6b subclass. The *Snap25-202* isoform distribution across cells (left) painted on the t-SNE of the cells. The

cells belonging to the L6b subclass are circled. The violin plots of the gene and isoform distributions show that the gene is not differential but the isoform is (right). e) Example of a gene with an isoform specific to the L6b Ror1_1 cluster. The *Stxbp2-207* isoform distribution in log1p(TPM) across cells (left) painted on the t-SNE of the cells. The cells belonging to the L6b Ror1_1 cluster are circled. The violin plots of the gene and isoform distributions show that the gene is not differential but the isoform is (right). A * indicates statistically significant difference $p<0.01$ between the group and its complement. The white circles within the violin plots represent the mean and the white bars represent + / - one standard deviation. [Code a, Code b, Code c, Code d, Code e]

To take advantage of the complementary strengths of these different platforms, we introduce an approach to scRNA-seq that links the SMART-seq resolved isoforms to the 10x Chromium defined cell types, and in addition merges this information with spatial transcriptomic measurements obtained by MERFISH [25] (Figure 1). In addition to revealing extensive isoform diversity and cell type specificity in the MOp, we find evidence for previously missed transcriptionally distinct cell subtypes in the MOp. Our results extend the notion of a single-cell database beyond a list of gene markers, and we produce a gene-isoform-space single-cell atlas for the MOp using the 10xv3, SMART-seq and MERFISH data together. Our methods are open-source, reproducible, easy-to-use and constitute an effective workflow for leveraging full-length scRNA-seq data in combination with data from other technologies.

Results

Isoforms markers for cell types

To identify isoform markers of cell types, we first sought to visualize our SMART-Seq data using gene derived cluster labels from the BICCN analysis (see Methods). Rather than layering cluster labels on cells mapped to 2-D with an unsupervised dimensionality reduction technique such as t-SNE[26] or UMAP[27], we utilized a supervised learning approach to project cells so that they are best separated according to BICCN consortium[22] annotations using neighborhood component analysis (NCA). This method produces meaningful representation of the global structure of the data (Figure 2b), without overfitting (Supplementary Fig. 1a). Analysis of the projections revealed batch effect in the 10xv3 data, which we addressed by restricting analysis to a single batch and minimal evidence of batch effect in the MERFISH data (see Methods, Supplementary Fig. 2a,b).

Next, motivated by the discovery of genes exhibiting differential exon usage between glutamatergic and GABAergic neurons in the primary visual cortex[14], we performed a differential analysis between these two classes of neurons. We searched for significant shifts in isoform abundances in genes whose expression was stable across cell types (for

details see Methods). We discovered 398 such isoform markers belonging to 310 genes (Supplementary Table 1, [Code]). Figure 2c shows an example of such an isoform from the Oxidative resistance 1 (*Oxr1*) gene which is known to be essential for protection against oxidative stress-induced neurodegeneration [28,29]. While we see no change in gene expression of *Oxr1* between these two neuron types, we find that among the 16 isoforms of the gene, one of them, *Oxr1-204*, is highly expressed in glutamatergic neurons. The *Oxr1* gene undergoes an isoform shift in GABAergic neurons where the expression of the *Oxr1-204* isoform is significantly lower, suggesting distinct subcellular isoform localization in the two neuron types [30]. A gene-level analysis is blind to this isoform shift (Figure 2c rightmost panel, top).

We hypothesized that there exists genes exhibiting cell type isoform specificity at all levels of the MOp cell ontology. However, detection of such genes and their associated isoforms requires meaningful cell type assignments and accurate isoform quantifications. To assess the reliability of the SMART-Seq clusters produced by the BICCN[31], we examined the correlation in gene expression by cluster with an orthogonal single-cell RNA-seq technology, the 10xv3 3'-end assay. 94,162 10xv3 cells, also derived from the MOp, were clustered using the same method as the SMART-Seq cells (see Methods). The clustering method generates three hierarchies of cells: classes, subclasses, and clusters. The SMART-Seq data has 2 major classes (Glutamatergic, GABAergic), 18 subclasses that subdivide the classes, and 62 clusters that subdivide the subclasses. The 10x data similarly contains three hierarchies of cells: two major classes (Glutamatergic, GABAergic), 21 subclasses, and 85 clusters. We found high correlation of gene expression between the two assays at the subclass and cluster levels (Extended Data Fig. 3).

Next, we assessed the accuracy of the SMART-Seq isoform quantification and its concordance with 10xv3 quantifications of isoforms. Since not all isoforms can be quantified from 10xv3 3'-end data, we examined only isoforms containing some unique 3' UTR sequence. This allowed for a validation of our isoform quantifications with a different technology (see Methods). To extract isoform quantifications from 10xv3 data in cases where there was a unique 3' sequence, we relied on transcript compatibility counts[32] produced by pseudoalignment with kallisto[24]. We were able to validate the SMART-Seq isoform shift predictions at both the subclass and cluster levels (Extended Data Fig. 4). The isoform abundance correlations are slightly lower than those of gene abundance estimates (Extended Data Fig. 3), but sufficiently accurate to identify significant isoform shifts, consistent with benchmarks showing that isoforms can be quantified accurately from full-length bulk RNA-seq[33].

Having validated the cluster assignments and isoform abundance estimates, we tested for isoform switches for 16 cell subclasses excluding Low Quality cells (example in Figure 2d), and then for 48 distinct clusters for subclasses that have more than one cluster (example in Figure 2e) and more than 5 cells per cluster; see Methods. At the higher level of 16 cell subclasses, we found a total of 654 isoforms from 550 genes within the glutamatergic class and 381 isoforms from 332 genes within the GABAergic class exhibiting isoform shifts among the 16 cell subclasses despite constant gene abundance (Supplementary Table 2a,b, [Code a, Code b]). There are several intriguing examples of isoform shifts at this level. For example, we find a shift in the *Snap25-202* isoform, whose expression has been specifically shown to be correlated with age and to differentially regulate synaptic transmission and synaptic plasticity at central synapses [34,35]. This isoform marks the L6b subclass Figure 2d. At the cluster level, we found 923 isoforms from 823 genes exhibiting isoform shifts among the 48 clusters passing filter despite constant gene abundance (Supplementary Table 3, [Code]). One particularly intriguing isoform that marks the L6b Ror1_ 1 cluster, a subset of cells in the L6b subclass, is the *Stxbp2-207* isoform whose gene *Stxbp2* has been previously detected in the subthalamic nucleus and the posterior hypothalamus [36].

Assaying both males and females allowed us to look at sex specific effects in all subclasses except for the L5 IT which was excluded due to batch effect (Supplementary Fig. 4). In total these subclasses exhibited 418 sex specific isoforms averging 40 isoforms per subclass (Supplementary Table 7, [Code]). Unlike a recent study [37] which found a sex-specific cell type in the ventromedial nucleus of the hypothalamus, we do not find any sex-specific subclasses. We did however find a handful of sex-differential autosomal isoforms. Among these, we find that the *Shank1-203* isoform is differential in Vip neurons, a finding that refines previous data showing that *Shank1*, which has been shown to localize in Purkinje cells in the cortex [38], is a sex specific gene who's expression is regulated by sex hormones [39].

We also looked for instances where clusters could be refined according to isoform expression. After re-clustering each 10xv3 derived cluster using SMART-seq isoform quantifications (see Methods), we found 12 clusters can be split by isoforms. Examining the L6 CT Grp_1 cluster we find that the average effect size for differential isoforms that split the cluster into two sub clusters is higher than that for genes (Extended Data Fig. 5). One isoform in particular, that splits the L6 CT Grp_1 cluster, is a protein coding isoform of the Amyloid Precursor Protein *App*. Dysregulation of splicing for isoforms of *App* have been found to be associated with disease pathogenesis in Alzheimer disease models [40]. Our findings show that isoform level expression can help refine cell types in the mouse primary motor cortex beyond what is possible only with gene-level expression estimates.

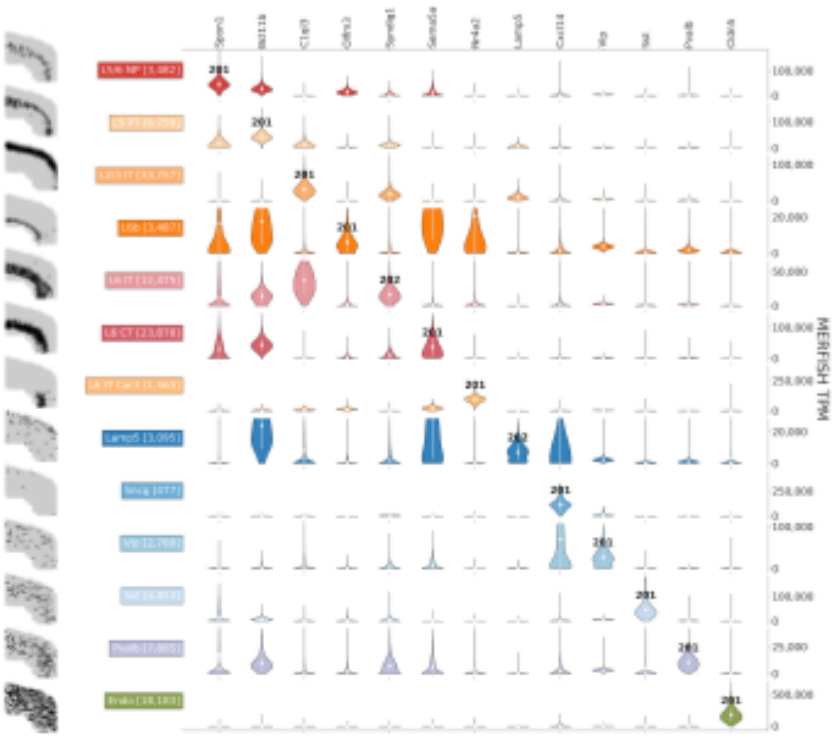


Figure 3: Isoform atlas. Spatial isoform atlas for the MOp. The scatter plots in the leftmost column show the location of the cells (black points) in the subclass (row) within a single representative slice of the mouse primary motor cortex as assayed by MERFISH. To the right of the scatter plot, each column corresponds to a marker gene in the MERFISH dataset for the subclass (row) where one of the marker genes' underlying isoform (labeled on the diagonal) was differential in the SMART-Seq dataset for that subclass. This spatial isoform inference links isoform expression from the SMART-Seq data with physical location of the cells expressing that isoform from the MERFISH data. The normalized gene expression values are plotted for each subclass-gene pair in TPM units. The isoform that is differential for each subclass is displayed in bold on the corresponding violin plot. The white circles within the violin plots represent the mean and the white bars represent + / - one standard deviation. [Code]

Along with isoforms detectable as differential between cell types without change in gene abundance, we identified isoform markers for the classes, subclasses, and clusters in the MOp ontology that are differential regardless of gene expression. We found 5,658 isoforms belonging to 3,132 genes that are specific to the glutamatergic and GABAergic classes (Figure 3, Supplementary Table 4, [Code]), 7,588 isoforms belonging to 4,171

genes within the glutamatergic class and 4,359 isoforms belonging to 2,614 genes within the GABAergic class exhibiting isoform shifts specific to subclasses (Supplementary Table 5a,b, [Code a, Code b]), and for the 48 clusters passing filter 3,171 isoforms belonging to 2,461 genes exhibiting isoform shifts in clusters (Supplementary Table 6, [Code]). Together, these form an isoform catalog for the MOp (Supplementary Fig. 5a,b).

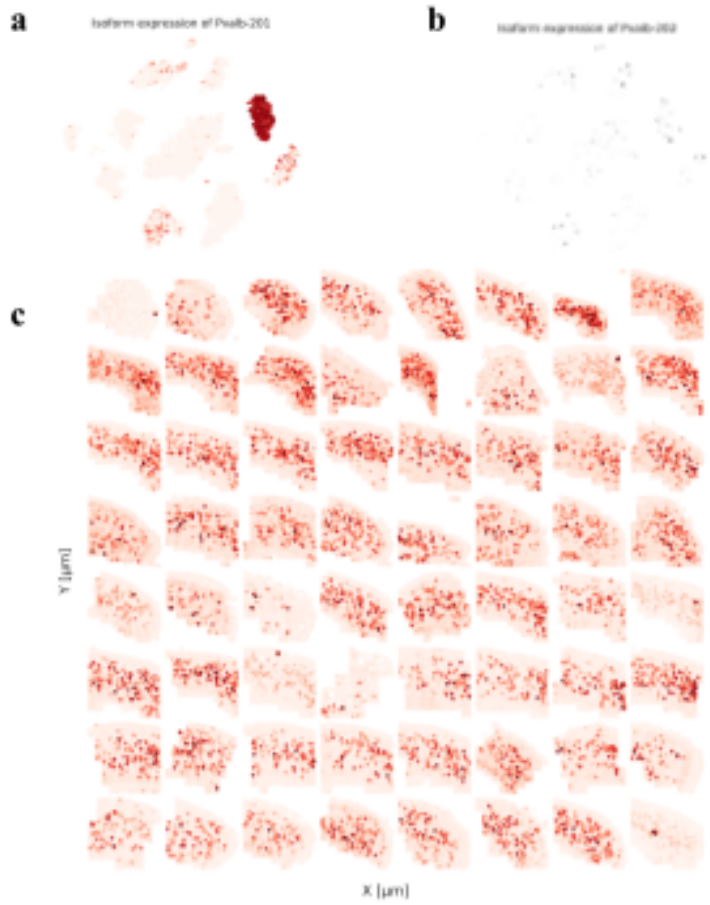


Figure 4: Spatial extrapolation of isoform expression. a) Expression of the *Pvalb-201* isoform and b) expression of the *Pvalb-202* isoform in log1p(TPM) units for each cell painted on the NCA-tSNE, as assayed by SMART-Seq. c) Spatial expression of the *Pvalb-201* isoform across 64 slices from the MOp, as extrapolated from probes for the *Pvalb* gene assayed by MERFISH. Each MERFISH cell is colored by its expression of Pvalb in log1p(TPM) units. [Code a, Code b, Code c]

Spatial isoform specificity

While spatial single-cell RNA-seq methods are not currently well-suited to directly probing isoforms of genes due to the number and lengths of probes required, spatial analysis at the gene-level can be refined to yield isoform-level results by extrapolating SMART-Seq isoform quantifications (Figure 3, Supplementary Fig. 5c).

Figure 4a,b shows an example of a gene, *Pvalb*, where the SMART-Seq quantifications reveal that of the two isoforms of the gene, only one, *Pvalb-201*, is expressed. Moreover it can be seen to be specific to the Pvalb cell subclass (Figure 3). In an examination of MERFISH spatial single-cell RNA-seq, derived from 64 slices from the MOp region (Extended Data Fig. 6a), the Pvalb subclass, of which the *Pvalb* gene is a marker, can be seen to be dispersed throughout the motor cortex spanning all layers (Extended Data Fig. 6b). While the MERFISH probes only measure abundance of *Pvalb* at the gene level (Figure 4c), extrapolation from the SMART-Seq quantifications can be used to refine the MERFISH result to reveal the spatial expression pattern of the *Pvalb-201* isoform.

This extrapolation can be done systematically. To build a spatial isoform atlas of the MOp, we identified differentially expressed genes from the MERFISH data (Supplementary Table 8a,b, [Code a, Code b]) and for each of them checked whether there were SMART-seq isoform markers (from Supplementary Table 5a,b, [Code a, Code b]). An example of the result is shown in Figure 3, which displays one gene for each cluster, together with the isoform label specific to that cluster and the spatial location of the specific cluster within a slice of the mouse primary motor cortex.

We hypothesized that the mouse primary motor cortex may exhibit changes in isoform expression associated with physical location of cells [41,42]. To determine whether there are isoforms that increase or decrease in expression along the depth of the motor cortex we first estimated the position of the various layers in the Glutamatergic subclasses (Methods, Extended Data Fig. 7a) and then performed weighted least squares regression on the centroids of the subclasses and inferred isoform expression from the SMART-Seq data. While we find many isoforms that exhibit a significant change in expression across the depth (Extended Data Fig. 7b,c, Supplementary Table 8c, [Code]), none of them that pass our filter exhibit a monotonic change with respect to the mean. This suggests that non-linear models may be better suited to study isoform variability across the depth of the mouse primary motor cortex.

While direct measurement of isoform abundance may be possible with spatial RNA-seq technologies such as SEQFISH[37] or MERFISH[2], such resolution would require dozens of probes to be assayed per gene (Supplementary Fig. 6), each of which is typically tens of base-pairs in length. Thus, while isoforms can in theory be detected in cases where they

contain large stretches of unique sequence, the technology is currently prohibitive for assaying most isoforms, making the extrapolation procedure described here of practical relevance (Supplementary Table 9, [Source]).

Splicing markers

Isoform quantification of RNA-seq can be used to distinguish shifts in expression between transcripts that share transcriptional start sites, and shifts due to the use of distinct transcription start sites. Investigating such differences can, in principle, shed light on transcriptional versus post-transcriptional regulation of detected isoform shifts[43,44]. Extended Data Fig. 8 shows an example of a gene, *Ptk2b* (Extended Data Fig. 8a), in the Glutamatergic class that exhibits differential expression of transcripts between start sites (Extended Data Fig. 8b). This gene is known to be associated with Alzheimer's disease and its transcript usage is mediated by genetic variation [45]. Interestingly, we find that isoforms sharing the preferential start site exhibit no discernible difference in expression (Extended Data Fig. 8c), suggesting the observed differences result from cell-type specific transcriptional, rather than post-transcriptional regulation. We identified 1,971 isoforms from 128 groups of transcription start sites where the transcription start sites are preferentially expressed in either GABAergic, glutamatergic and Non-Neuronal classes, even when the expression of isoforms contained within the transcription start site is constant (Supplementary Table 10a,c,d, [Code a, Code c, Code d]). Such cases are likely instances where the transcription start site shifts between cell types are a result of differential splicing, i.e. the result of a post-transcriptional program.

We also looked at post-transcriptional programs (Supplementary Table 10b, [Code], instances where transcription start sites are not differential between classes but where there are isoform shifts within transcription start sites between classes. We find 31 isoforms from 28 transcription start site groups that are differential between classes when the transcription start site group is not. One such example is expression of isoforms *Rtn1-201* and *Rtn1-203* which share the same transcription start site in the *Rtn1* gene. The glutamatergic class exhibits preferential expression of *Rtn1-201*, previously shown to be expressed in grey matter[46], whereas the GABAergic class does not (Extended Data Fig. 9). These cases are likely instances where isoform shifts between cell types are a result of differential splicing, i.e. the result of a post-transcriptional program.

Discussion

Our mouse primary motor cortex spatially resolved isoform atlas expands on previously identified gene markers in the MOp, greatly expanding the catalog to isoform markers for cell types characterized by the BICCN [22]. Our approach leverages distinct strengths of different technologies, utilizing the isoform resolution of SMART-seq in conjunction with

the complementary cell depth obtainable with 10X Genomics' technology and the spatial resolution produced with MERFISH to spatially place cell-type isoform markers. This validated approach, in which we leverage technologies that are broadly consistent (Extended Data Fig. 10) yet complementary in their strengths, is important because with additional morphological data, isoform specificity could help to explain morphological differences. For example *Pvalb* cells observed in hippocampal *Pvalb* interneurons cannot be distinguished morphologically based only on gene level analysis [9,47,48].

Spatial isoform markers also enable more targeted assays for "automatic expression histology", for example using new genetic tools such as recently developed paired guide RNAs for alternative exon removal (pgFARM) [49]. Finally, the next step after assembling a single-cell isoform atlas is to probe the functional significance of cell type isoform specificity. Recently developed experimental methods for this purpose, e.g. isoforms screens[49], are a promising direction and will be key to understanding the significance of the vast isoform diversity in the brain[50].

Methods

All of the results and figures in the paper are reproducible starting with the raw reads using scripts and code downloadable from https://github.com/pachterlab/BYVSTZP_2020. The repository makes the method choices completely transparent, including all parameters and thresholds used. All p-values were corrected using Bonferroni correction and all error bars denote +/- one standard deviation from the mean.

Tissue collection and isolation of cells

Mouse breeding and husbandry: All procedures were carried out in accordance with Institutional Animal Care and Use Committee protocols at the Allen Institute for Brain Science. Mice were provided food and water ad libitum and were maintained on a regular 12-h day/night cycle at no more than five adult animals per cage. For this study, we enriched for neurons by using Snap25-IRES2-Cre mice[52] (MGI:J:220523) crossed to Ai14[53] (MGI: J:220523), which were maintained on the C57BL/6J background (RRID:IMSR_JAX:000664). Animals were euthanized at 53−59 days of postnatal age. Tissue was collected from both males and females (scRNA SMART, scRNA 10x v3).

Single-cell isolation: We isolated single cells by adapting previously described procedures[14,31]. The brain was dissected, submerged in ACSF[31], embedded in 2% agarose, and sliced into 250-µm (SMART-Seq) or 350-µm (10x Genomics) coronal sections on a

compresstome (Precisionary Instruments). The Allen Mouse Brain Common Coordinate Framework version 3 (CCFv3, RRID:SCR_002978)[54] ontology was used to define MOp for dissections.

For SMART-Seq, MOp was microdissected from the slices and dissociated into single cells with 1 mg/ml pronase (Sigma P6911-1G) and processed as previously described [31]. For 10x Genomics, tissue pieces were digested with 30 U/ml papain (Worthington PAP2) in ACSF for 30 mins at 30 °C. Enzymatic digestion was quenched by exchanging the papain solution three times with quenching buffer (ACSF with 1% FBS and 0.2% BSA). The tissue pieces in the quenching buffer were triturated through a fire-polished pipette with 600-µm diameter opening approximately 20 times. The solution was allowed to settle and supernatant containing single cells was transferred to a new tube. Fresh quenching buffer was added to the settled tissue pieces, and trituration and supernatant transfer were repeated using 300-µm and 150-µm fire polished pipettes. The single cell suspension was passed through a 70-µm filter into a 15-ml conical tube with 500 ul of high BSA buffer (ACSF with 1% FBS and 1% BSA) at the bottom to help cushion the cells during centrifugation at 100xg in a swinging bucket centrifuge for 10 minutes. The supernatant was discarded, and the cell pellet was resuspended in quenching buffer.

All cells were collected by fluorescence-activated cell sorting (FACS, BD Aria II, RRID: SCR_018091) using a 130-µm nozzle. Cells were prepared for sorting by passing the suspension through a 70-µm filter and adding DAPI (to the final concentration of 2 ng/ml). Sorting strategy was as previously described[31], with most cells collected using the tdTomato-positive label. For SMART-Seq, single cells were sorted into individual wells of 8-well PCR strips containing lysis buffer from the SMART-Seq v4 Ultra Low Input RNA Kit for Sequencing (Takara 634894) with RNase inhibitor (0.17 U/µl), immediately frozen on dry ice, and stored at −80 °C. For 10x Genomics, 30,000 cells were sorted within 10 minutes into a tube containing 500 µl of quenching buffer. Each aliquot of 30,000 sorted cells was gently layered on top of 200 µl of high BSA buffer and immediately centrifuged at 230xg for 10 minutes in a swinging bucket centrifuge. Supernatant was removed and 35 µl of buffer was left behind, in which the cell pellet was resuspended. The cell concentration was quantified, and immediately loaded onto the 10x Genomics Chromium controller.

Genomic library preparation and sequencing

For SMART-Seq library preparation, we performed the procedures with positive and negative controls as previously described[31]. The SMART-Seq v4 (SSv4) Ultra Low Input

RNA Kit for Sequencing (Takara Cat# 634894) was used to reverse transcribe poly(A) RNA and amplify full-length cDNA. Samples were amplified for 18 cycles in 8-well strips, in sets of 12–24 strips at a time. All samples proceeded through Nextera XT DNA Library Preparation (Illumina Cat# FC-131-1096) using Nextera XT Index Kit V2 (Illumina Cat# FC-131-2001) and a custom index set (Integrated DNA Technologies). Nextera XT DNA Library prep was performed according to manufacturer's instructions, with a modification to reduce the volumes of all reagents and cDNA input to 0.4x or 0.5x of the original protocol.

For 10x v3 library preparation, we used the Chromium Single Cell 3' Reagent Kit v3 (10x Genomics Cat# 1000075). We followed the manufacturer's instructions for cell capture, barcoding, reverse transcription, cDNA amplification, and library construction. We targeted sequencing depth of 120,000 reads per cell.

Sequencing of SMART-Seq v4 libraries was performed as described previously[31]. Briefly, libraries were sequenced on an Illumina HiSeq2500 platform (paired-end with read lengths of 50 bp). 10x v3 libraries were sequenced on Illumina NovaSeq 6000 (RRID:SCR_016387).

Pre-processing single-cell RNA-seq data

The 6,295 SMART-Seq cells were processed using kallisto with the `kallisto pseudo` command[24]. The 94,162 10x Genomics v3 cells were pre-processed with kallisto and bustools[55]. Gene count matrices were made by using the --genecounts flag and TCC matrices were made by omitting it. The mouse transcriptome reference used was GRCm38.p3 (mm10) RefSeq annotation gff file retrieved from NCBI on 18 January 2016 (https://www.ncbi.nlm.nih.gov/genome/annotation_euk/all/), for consistency with the reference used by the BICCN consortium[22].

The GTF and the GRCm38 genome fasta [Source to both], provided by the consortium, were used to create a transcriptome fasta, transcripts to genes map [Source], and kallisto index using kb ref -i index.idx, -g t2g.txt -f1 transcriptome.fa genome.fa genes.gtf. To validate the SMART-Seq isoform quantifications we first examined the robustness of the quantifications to gene annotation, and found an average correlation at the isoform level of 0.965 between the BICCN derived quantifications we used in our analysis [22] and the mouse GENCODE M25 derived quantifications (Supplementary Fig. 3). The GENCODE M25 mouse transcriptome reference [Source] and the kallisto index was built using kallisto index -i index.idx gencode.vM25.transcripts.fa.gz .

Isoform and gene count matrices were generated for the Smart-seq2 data using the kallisto pseudo command. Cluster assignments were associated with cells using cluster labels generated by the BICCN consortium[22]. The labels are organized in a hierarchy of three levels: classes, subclasses and clusters. The cluster labels for the cells can be downloaded from https://github.com/pachterlab/BYVSTZP_2020.

Clustering and cell type assignment

Our analyses utilized SMART-seq cell labels produced by the BICCN [22]. Briefly, the assignment of cell types to the SMARTSeq cells was based on an extension of the cluster merging algorithm in the scrattch.hicat package [31]. The clustering method generates three hierarchies of cells: classes, subsets of cells within classes called subclasses, and subsets of cells within subclasses called clusters.
To build a common adjacency matrix incorporating samples from all the datasets, first a subset of datasets was chosen ("reference datasets"). The 10x v2 single cell dataset from Allen (scRNA 10x v2 A) and 10x v3 single nucleus dataset from Broad (snRNA 10x v3 B) were used as references.

The key steps of the pipeline are as follows: 1. Perform single-dataset clustering, 2. select anchor cells for each reference dataset, 3. select highly variable genes (HVG), 4. compute K nearest neighbors (KNN), 5. compute the Jaccard similarity, 6. perform louvain clustering, 7. merge clusters, 8. cluster iteratively, 9. compile and merge clusters.
For further details please see [22].

Normalization and filtering of SMART-Seq data

Isoform counts were first divided by the length of transcript to obtain abundance estimates proportional to molecule copy numbers. Since reads can come from anywhere in the transcriptome, it is likely that longer isoforms are enriched. Therefore normalizing isoform abundances by length is crucial to accurately estimating mRNA copy number. This has been shown in numerous studies on the accurate estimate of isoform abundance [56,57].

After normalizing by length, we then removed isoforms that had fewer than one count and that were in fewer than one cell. We also removed genes and their corresponding isoforms that had a dispersion of less than 0.001.

To generate the cell by gene matrix we summed the isoforms that correspond to the same gene. Cells with less than 250 gene counts and with greater than 10% mitochondrial

content were removed. Cells were normalized to transcripts per million (TPM) by dividing the counts in each cell by the sum of the counts for that cell, then multiplying by 1,000,000. The count matrices were then transformed with log1p and the columns scaled to unit variance and zero mean. The resulting gene and isoform matrix contained 6,160 cells and 19,190 genes corresponding to 69,172 isoforms.

Highly variable isoforms and genes were identified by first computing the dispersion for each feature, and then binning all of the features into 20 bins. The dispersion for each feature was normalized by subtracting the mean dispersion and dividing by the variance of the dispersions within each bin. Then the top 5000 features were retained based on the normalized dispersion. This was computed by using the scanpy.pp.highly_variable_genes with n_top_genes = 5000, flavor=seurat, and n_bins=20[58].

Normalization and filtering of 10xv3 data

To generate the cell by gene matrix we used ‘bustools count --genecounts’. The cell by isoform matrix was generated using ‘bustools count’ and restricting to the equivalence classes that contained only one isoform thus generating a cell by isoform matrix. Both matrices were loaded into python using kb python. Cells with less than 250 gene counts and with greater than 21.5% mitochondrial content were removed. Cells were normalized to counts per million (CPM) by dividing the counts in each cell by the sum of the counts for that cell, then multiplying by 1,000,000. The count matrices were then transformed with log1p and the columns scaled to unit variance and zero mean. The resulting gene matrix contained 94,162 cells and 24,575 genes. We removed the cells that were identified as Low Quality by the BICCN consortium. We identified batch effect among cells assayed on different dates so we restricted our analysis to only the cells assayed on the same date and selected the date with the most number of cells (Supplementary Fig. 2). Additionally we performed pairwise comparison of gene counts for each of the 4 10xv3 batches and found the Pearson correlation to be very high for all pairs, with a mean of 0.9979 indicating limited batch effect between batches assayed on the same date.

Highly variable isoforms and genes were identified by first computing the dispersion for each feature, and then binning all of the features into 20 bins. The dispersion for each feature was normalized by subtracting the mean dispersion and dividing by the variance of the dispersions within each bin. Then the top 5000 features were retained based on the normalized dispersion. This was computed by using the scanpy.pp.highly_variable_genes with n_top_genes = 5000, flavor=seurat, and n_bins=20[58].

Dimensionality reduction and visualization

In order to visualize the SMART-Seq data with predefined cluster labels produced via a joint analysis with many other data types we performed neighborhood component analysis[59] (NCA) on the full scaled log(TPM +1) matrix using the subcluster labels, to ten components. t-distributed stochastic neighbor embedding (t-SNE)[26] was then performed on the 10 NCA components. NCA takes as input not just a collection of cells with their associated abundances, but also cluster labels for those cells, and seeks to find a projection that minimizes leave-one-out k-nearest neighbor error[59]. Moreover, t-SNE applied to PCA (Supplementary Fig. 1b) scrambles the proximity of glutamatergic and GABAergic cell types, while t-SNE of NCA appears to respect global structure of the cells. While UMAP applied to PCA of the data (Supplementary Fig. 1c) appears to be better than t-SNE in terms of preserving global structure, it still does not separate out the cell types as well as NCA (Supplementary Fig. 1d). t-SNE was computed using sklearn.manifold. t-SNE was generated with default parameters and random state 42. Similarly uniform manifold approximation was performed on the 10 NCA components and the 50 truncated SVD components. Uniform Manifold Approximation and Projection (UMAP)[27] was computed with the umap package with default parameters.

To ensure that NCA was not overfitting cells to their corresponding subclasses, we randomly permuted all of the subclasses labels and reran the NCA to t-SNE dimensionality reduction method. We observed uniform mixing of the permuted subclass labels, indicating that NCA was not overfitting the cells to their corresponding subclasses.

Sample Size

No explicit calculations were performed to determine sample size. We analyzed 6,160 mouse primary motor cortex cells assayed with SMART-Seq, 280,327 cells assayed with MERFISH, and 94,162 cells assayed with 10x Genomics Chromium v3. We analyzed both male and female mice to understand differences in gene and isoform expression. The sample size for differential expression was set to be such that 90% of cells in a cluster have a non-zero expression of the tested gene. The smallest cluster size contained 7 cells with all cells having non-zero expression of the tested genes. We computed error bars for all tests to ensure that sample sizes were sufficient.

Batch effects

After finding a meaningful projection that appears to respect global structure of the cells we searched for possible sources of batch effect within the datasets. We found evidence

of batch effect in the 10xv3 data by assay date (Supplementary Fig. 2a). To ensure that our findings were not confounded by this batch effect we selected the set of cells from only one assay date and picked the set with the largest number of cells and the one with cells present in all clusters. We then looked at the MERFISH data and found minimal evidence of batch effect across samples based on the distribution of batch labels across clusters where the observed fraction of cells per batch in each cluster was almost exactly the expected fraction of cells per batch assuming uniform mixing (Supplementary Fig. 2b).

In further examining the single 10xv3 batch we settled on, we noted a low correlation in one case, the L5 IT subclass. The low correlation was also observed in a comparison between SMART-Seq and MERFISH gene expression data (Extended Data Fig. 10a), and 10xv3 and MERFISH data (Extended Data Fig. 10b). We hypothesized that this low correlation stems from a subclass specific sex effect within the L5 IT where those cells differ drastically in their overall expression compared to other subclasses. The L5 IT subclass contains seven clusters in the SMART-Seq data, four clusters in the 10xv3 data, and four clusters in the MERFISH data.

To determine the source of the low correlation within the L5 IT between SMART-Seq, 10x, and MERFISH we looked at differential genes between male and female cells within each subclass. We found that cells within the L5 IT of the 10x and SMART-Seq data exhibited sex specfic segregation (Supplementary Fig. 4a,b). After performing differential expression between male and female cells within all subclasses we found that the L5 IT had the highest amount of uniquely differential genes (Supplementary Fig. 4c) and that the SMART-Seq and 10x data had 37 common genes that were differentially expressed (Supplementary Fig. 4d). The other subclasses, however, did not exhibit sex-based segregation. Without being able to rule out that the low correlation for L5 IT cells across the technologies was due to confounding between batch and sex in the dataset, we decided to excluded the subclass from our analyses.

Measuring number of isoforms per gene

We parsed the transcripts to genes map, grouping together transcripts that had the same end site that were in the same gene. We then counted the number of these end site sets within a gene and plotted them against the number of isoforms within that gene.

Cross-technology cluster correlation

The correlation between 10xv3-Smart-seq, 10xv3-MERFISH, and SMART-Seq-MERFISH, was performed at the gene level and between cells grouped by

subclasses for all three pairs of technologies, and at the isoform level and between cells grouped by cluster for only the 10xv3 and SMART-Seq. For each pair we started with two raw matrices and restricted to the set of genes/isoforms common to the two. Then we normalized the counts for each matrix per cell to one million, log1p transformed the entire matrix, and scaled the features to zero mean and unit variance. Within each cluster we restricted the features to those present in at least 50% of the cells. We then found the mean cell within the respective clusters in the two matrices, and computed the Pearson correlation between them. These methods were implemented for Extended Data Figs. 3, 4, and 10. In terms of accuracy of different technologies, we found good agreement between quantifications from SMART-Seq, 10xv3 and MERFISH (Extended Data Fig. 10).

Comparisons of different scRNA-seq technologies have tended to focus on throughput, cost, and gene-level accuracy[60] in a winner-takes-all competition. Our results shed some light on the matter: it has been previously shown that quantification of isoform abundance is necessary for accurate gene-level estimates[61], and we found that it matters in practice (Supplementary Fig. 7, Supplementary Table 11a,b, [Code a, Code b] and Supplementary Table 12a,b, [Code a, Code b]). This highlights the importance of proper isoform quantification of SMART-Seq data, even for gene-centric analysis [56,60–63], when used in conjunction with 10x Genomics and MERFISH data.

Isoform atlas

For each level of clustering: class, subclass, cluster, we performed a t-test for each gene/isoform between the cluster and its complement, on the log1p counts. To identify isoform enrichment that is masked at a gene level analysis, we looked for isoforms that were upregulated by checking that the gene containing that isoform was not significantly expressed in that cluster, relative to the complement of that cluster. Isoforms that were expressed in less than 90% of the cells in that cluster were ignored. All t-tests used a significance level of 0.01 and all p-values were corrected for multiple testing using Bonferroni correction.

MERFISH isoform extrapolation

First we identified the genes that mark the specific subclass within the MERFISH data. The Pvalb gene is a marker for the Pvalb subclass. Then we performed differential analysis on the SMART-Seq data at the isoform level on the subclasses to identify the isoforms that mark each of the SMART-Seq subclasses. Only one of the two isoforms for

Pvalb marked the Pvalb cluster. This allowed us to extrapolate the fact that the specific Pvalb isoform is being detected in the MERFISH data.

Additionally, we identified all of the genes that mark the specific subclasses in the MERFISH data through differential analysis and checked if their underlying isoforms were also differentially expressed. We then noted which isoforms were differentially expressed for the spatial isoform atlas.

Weighted least squares regression

First we selected a representative slice of the MOp. Then we found the outer hull of the MOp by using scipy.spatial.ConvexHull. We selected the points that defined the upper boundary of the MOp then performed linear regression to fit a line to those points using sklearn.linear_model.LinearRegression(). For each subclass in the Glutamatergic class of cells we identified the centroid of the subclass and determined the perpendicular distance of the centroid to the MOp boundary line. We normalized the set of distances by dividing by the centroid with the largest distance to the boundary.

We look at the isoforms for which all of the subclasses had non-zero expression in at least 90% of cells.For each isoform we performed weighted least squares for all of the subclasses with the weights equal to the variance of isoform expression for each subclass. We used the statsmodel.api.sm.WLS function. All WLS tests used a significance level of 0.01 and all F-score p-values were corrected for multiple testing using Bonferroni correction. Monotonicity was checked for isoforms with an absolute value slope greater than 1.5.

Grouping transcripts by start site

Using the transcripts to genes map and the filtered isoform matrix generated before, we grouped isoforms by their transcription start site into TSS classes and summed the raw counts for the isoforms within each TSS class to create a *cell x TSS* matrix. Differential analysis was then performed in exactly the same way as above. For each cluster and each TSS/isoform, a t-test was performed between the cells in that cluster and the cells in the complement of that cluster. All statistical tests used a significance level of 0.01 and all p-values were corrected for multiple testing using Bonferroni correction.

Comparison of naïve and EM quantification

Naïve gene count matrices were constructed from the SMART-Seq data by summing the counts corresponding to a single gene. Gene count matrices quantified by the EM algorithm and normalized appropriately were made with SMART-Seq by first dividing isoform abundances by the length of their transcripts, and then summing the abundances of isoforms by gene. Differential analysis was performed independently on these two gene count matrices and the resultant differential genes were compared. Differential expression was then performed on all of the genes for both the EM and naïve gene quantifications. All statistical tests used a significance level of 0.01 and all p-values were corrected for multiple testing using Bonferroni correction.

Software versions

Anndata 0.7.1
bustools 0.39.4
awk (GNU awk) 4.1.4
grep (GNU grep) 3.1
kallisto 0.46.1
kb_python 0.24.4
Matplotlib 3.0.3
Numpy 1.18.1
Pandas 0.25.3
Scanpy 1.4.5.post3
Scipy 1.4.1
sed (GNU sed) 4.4
sklearn 0.22.1
statsmodels 0.12.1
tar (GNU tar) 1.29
umap 0.3.10

Methods References

Data Availability

The single-cell RNA-seq data used in this study was generated as part of the BICCN consortium[22]. The 10xv3 and SMART-Seq data can be downloaded from http://data.nemoarchive.org/biccn/lab/zeng/transcriptome/scell/. The MERFISH data is available at https://caltech.box.com/shared/static/dzqt6ryytmjbgyai356s1z0phtnsbaol.gz.

All cell annotations and cluster labels are available at https://github.com/pachterlab/BYVSTZP_2020/tree/master/reference.

Software Availability

The software used to generate the results and figures of the paper is available at https://github.com/pachterlab/BYVSTZP_2020.

Acknowledgments

We thank members of the BICCN consortium, especially the "Mini-MOp" analysis group, for helpful conversations related to transcriptome analysis of the primary motor cortex. Thanks to Nadezda Volovich, Vasilis Ntranos, and Páll Melsted, for help with a preliminary quantification of the SMART-Seq data. Figure 1 was created from scratch using the tools available on Biorender.com. Extended Data Figure 6a was obtained from http://atlas.brain-map.org/atlas. This work was funded by the NIH Brain Initiative via grant U19MH114930 to HZ and LP).

References

1. Picelli, S. *et al.* Full-length RNA-seq from single cells using Smart-seq2. *Nat. Protoc.* **9**, 171–181 (2014).
2. Chen, K. H., Boettiger, A. N., Moffitt, J. R., Wang, S. & Zhuang, X. RNA imaging. Spatially resolved, highly multiplexed RNA profiling in single cells. *Science* **348**, aaa6090 (2015).
3. Zheng, G. X. Y. *et al.* Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.* **8**, 14049 (2017).
4. Weyn-Vanhentenryck, S. M. *et al.* Precise temporal regulation of alternative splicing during neural development. *Nat. Commun.* **9**, 2189 (2018).
5. Walker, R. L. *et al.* Genetic control of gene expression and splicing in the developing human brain. *BioRxiv* (2018) doi:10.1101/471193.
6. Porter, R. S., Jaamour, F. & Iwase, S. Neuron-specific alternative splicing of transcriptional machineries: Implications for neurodevelopmental disorders. *Mol. Cell. Neurosci.* **87**, 35–45 (2018).
7. Lukacsovich, D. *et al.* Single-Cell RNA-Seq Reveals Developmental Origins and Ontogenetic Stability of Neurexin Alternative Splicing Profiles. *Cell Rep.* **27**, 3752-3759.e4 (2019).
8. Song, Y. *et al.* Single-Cell Alternative Splicing Analysis with Expedition Reveals Splicing Dynamics during Neuron Differentiation. *Mol. Cell* **67**, 148-161.e5 (2017).
9. Que, L., Winterer, J. & Földy, C. Deep Survey of GABAergic Interneurons:

Emerging Insights From Gene-Isoform Transcriptomics. *Front. Mol. Neurosci.* **12**, 115 (2019).

10. Huang, C.-C., Lin, Y.-S., Lee, C.-C. & Hsu, K.-S. Cell type-specific expression of Eps8 in the mouse hippocampus. *BMC Neurosci.* **15**, 26 (2014).
11. Zhang, Y. *et al.* An RNA-sequencing transcriptome and splicing database of glia, neurons, and vascular cells of the cerebral cortex. *J. Neurosci.* **34**, 11929–11947 (2014).
12. Sugino, K. *et al.* Mapping the transcriptional diversity of genetically and anatomically defined cell populations in the mouse brain. *elife* **8**, (2019).
13. Bittar, P. G., Charnay, Y., Pellerin, L., Bouras, C. & Magistretti, P. J. Selective distribution of lactate dehydrogenase isoenzymes in neurons and astrocytes of human brain. *J. Cereb. Blood Flow Metab.* **16**, 1079–1089 (1996).
14. Tasic, B. *et al.* Adult mouse cortical cell taxonomy revealed by single cell transcriptomics. *Nat. Neurosci.* **19**, 335–346 (2016).
15. Gandal, M. J. *et al.* Transcriptome-wide isoform-level dysregulation in ASD, schizophrenia, and bipolar disorder. *Science* **362**, (2018).
16. Petri, S. *et al.* The mRNA expression of AMPA type glutamate receptors in the primary motor cortex of patients with amyotrophic lateral sclerosis: an in situ hybridization study. *Neurosci. Lett.* **360**, 170–174 (2004).
17. Macosko, E. Z. *et al.* Highly Parallel Genome-wide Expression Profiling of Individual Cells Using Nanoliter Droplets. *Cell* **161**, 1202–1214 (2015).
18. Klein, A. M. *et al.* Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell* **161**, 1187–1201 (2015).
19. Ramsköld, D. *et al.* Full-length mRNA-Seq from single-cell levels of RNA and individual circulating tumor cells. *Nat. Biotechnol.* **30**, 777–782 (2012).
20. Arzalluz-Luque, Á. & Conesa, A. Single-cell RNAseq for the study of isoforms-how is that possible? *Genome Biol.* **19**, 110 (2018).
21. Seirup, M. *et al.* Reproducibility across single-cell RNA-seq protocols for spatial ordering analysis. *PLoS ONE* **15**, e0239711 (2020).
22. Yao, Z. *et al.* An integrated transcriptomic and epigenomic atlas of mouse primary motor cortex cell types. *bioRxiv* (2020).
23. Wang, X., He, Y., Zhang, Q., Ren, X. & Zhang, Z. Direct Comparative Analysis of 10X Genomics Chromium and Smart-seq2. *BioRxiv* 615013 (2019) doi:10.1101/615013.
24. Bray, N. L., Pimentel, H., Melsted, P. & Pachter, L. Near-optimal probabilistic RNA-seq quantification. *Nat. Biotechnol.* **34**, 525–527 (2016).
25. Zhang, M. *et al.* Molecular, spatial and projection diversity of neurons in primary motor cortex revealed by in situ single-cell transcriptomics. *BioRxiv* (2020) doi:10.1101/2020.06.04.105700.

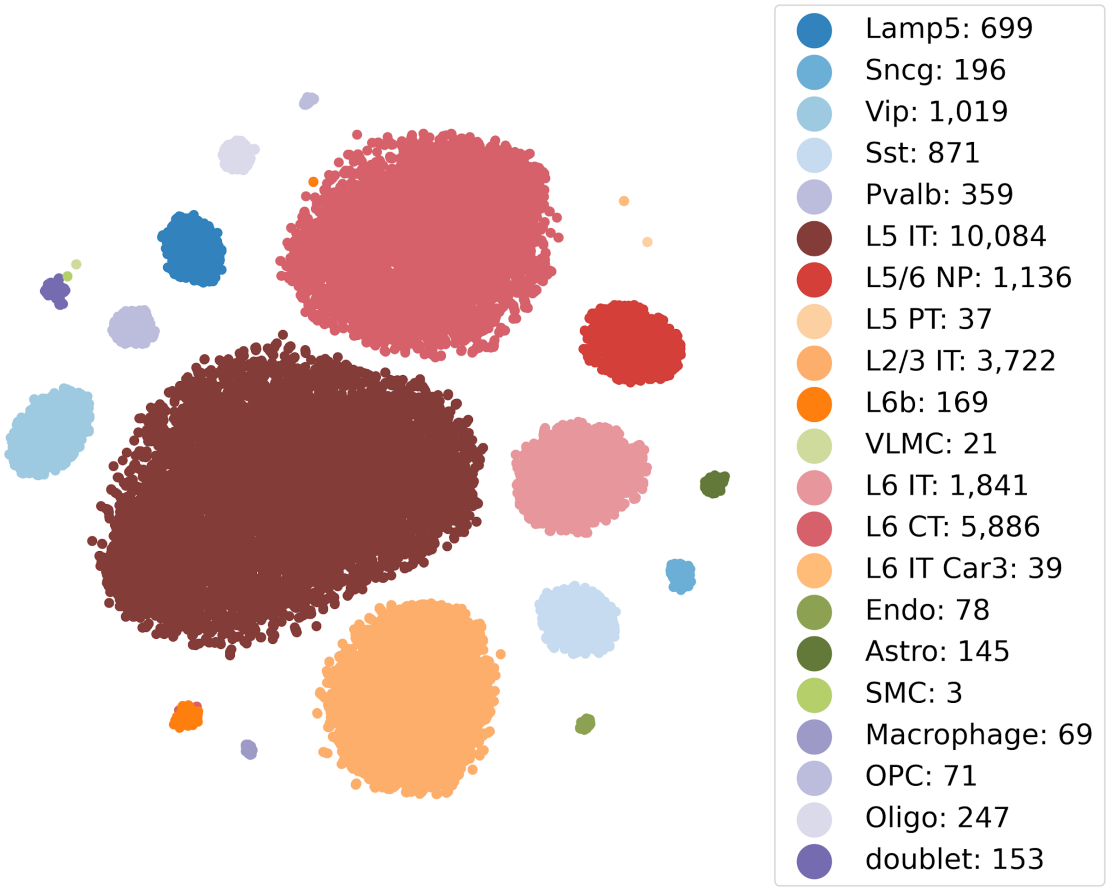
26. Maaten, L. van der & Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research* (2008).
27. Becht, E. *et al.* Dimensionality reduction for visualizing single-cell data using UMAP. *Nat. Biotechnol.* **37**, 38–44 (2018).
28. Oliver, P. L. *et al.* Oxr1 is essential for protection against oxidative stress-induced neurodegeneration. *PLoS Genet.* **7**, e1002338 (2011).
29. Volkert, M. R. Preventing Neurodegeneration by Controlling Oxidative Stress: The Role of OXR1. *Front Neurosci* (2020).
30. Wu, Y., Davies, K. E. & Oliver, P. L. The antioxidant protein Oxr1 influences aspects of mitochondrial morphology. *Free Radic. Biol. Med.* **95**, 255–267 (2016).
31. Tasic, B. *et al.* Shared and distinct transcriptomic cell types across neocortical areas. *Nature* **563**, 72–78 (2018).
32. Ntranos, V., Yi, L., Melsted, P. & Pachter, L. A discriminative learning approach to differential expression analysis for single-cell RNA-seq. *Nat. Methods* **16**, 163–166 (2019).
33. Unable to find information for 7206968.
34. Bark, I. C., Hahn, K. M., Ryabinin, A. E. & Wilson, M. C. Differential expression of SNAP-25 protein isoforms during divergent vesicle fusion events of neural development. *Proc Natl Acad Sci USA* **92**, 1510–1514 (1995).
35. Irfan, M. *et al.* SNAP-25 isoforms differentially regulate synaptic transmission and long-term synaptic plasticity at central synapses. *Sci. Rep.* **9**, 6403 (2019).
36. Wallén-Mackenzie, Å. *et al.* Spatio-molecular domains identified in the mouse subthalamic nucleus and neighboring glutamatergic and GABAergic brain structures. *Commun. Biol.* **3**, 338 (2020).
37. Kim, D.-W. *et al.* Multimodal analysis of cell types in a hypothalamic node controlling social behavior. *Cell* **179**, 713-728.e17 (2019).
38. Böckers, T. M. *et al.* Differential expression and dendritic transcript localization of Shank family members: identification of a dendritic targeting element in the 3' untranslated region of Shank1 mRNA. *Mol. Cell. Neurosci.* **26**, 182–190 (2004).
39. Berkel, S. *et al.* Sex hormones regulate SHANK expression. *Front. Mol. Neurosci.* **11**, 337 (2018).
40. Zhang, Y., Thompson, R., Zhang, H. & Xu, H. APP processing in Alzheimer's disease. *Mol. Brain* **4**, 3 (2011).
41. Rash, B. G. & Grove, E. A. Area and layer patterning in the developing cerebral cortex. *Curr. Opin. Neurobiol.* **16**, 25–34 (2006).
42. Sansom, S. N. & Livesey, F. J. Gradients in the brain: the control of the development of form and function in the cerebral cortex. *Cold Spring Harb. Perspect. Biol.* **1**, a002519 (2009).
43. Trapnell, C. *et al.* Transcript assembly and quantification by RNA-Seq reveals

unannotated transcripts and isoform switching during cell differentiation. *Nat. Biotechnol.* **28**, 511–515 (2010).
44. Unable to find information for 5087166.
45. Raj, T. *et al.* Integrative transcriptome analyses of the aging brain implicate altered splicing in Alzheimer's disease susceptibility. *Nat. Genet.* **50**, 1584–1592 (2018).
46. Mills, J. D. *et al.* Unique transcriptome patterns of the white and grey matter corroborate structural and functional heterogeneity in the human frontal lobe. *PLoS ONE* **8**, e78480 (2013).
47. Klausberger, T. & Somogyi, P. Neuronal diversity and temporal dynamics: the unity of hippocampal circuit operations. *Science* **321**, 53–57 (2008).
48. Harris, K. D. *et al.* Classes and continua of hippocampal CA1 inhibitory neurons revealed by single-cell transcriptomics. *PLoS Biol.* **16**, e2006387 (2018).
49. Thomas, J. D. *et al.* RNA isoform screens uncover the essentiality and tumor-suppressor activity of ultraconserved poison exons. *Nat. Genet.* **52**, 84–94 (2020).
50. Karlsson, K. & Linnarsson, S. Single-cell mRNA isoform diversity in the mouse brain. *BMC Genomics* **18**, 126 (2017).

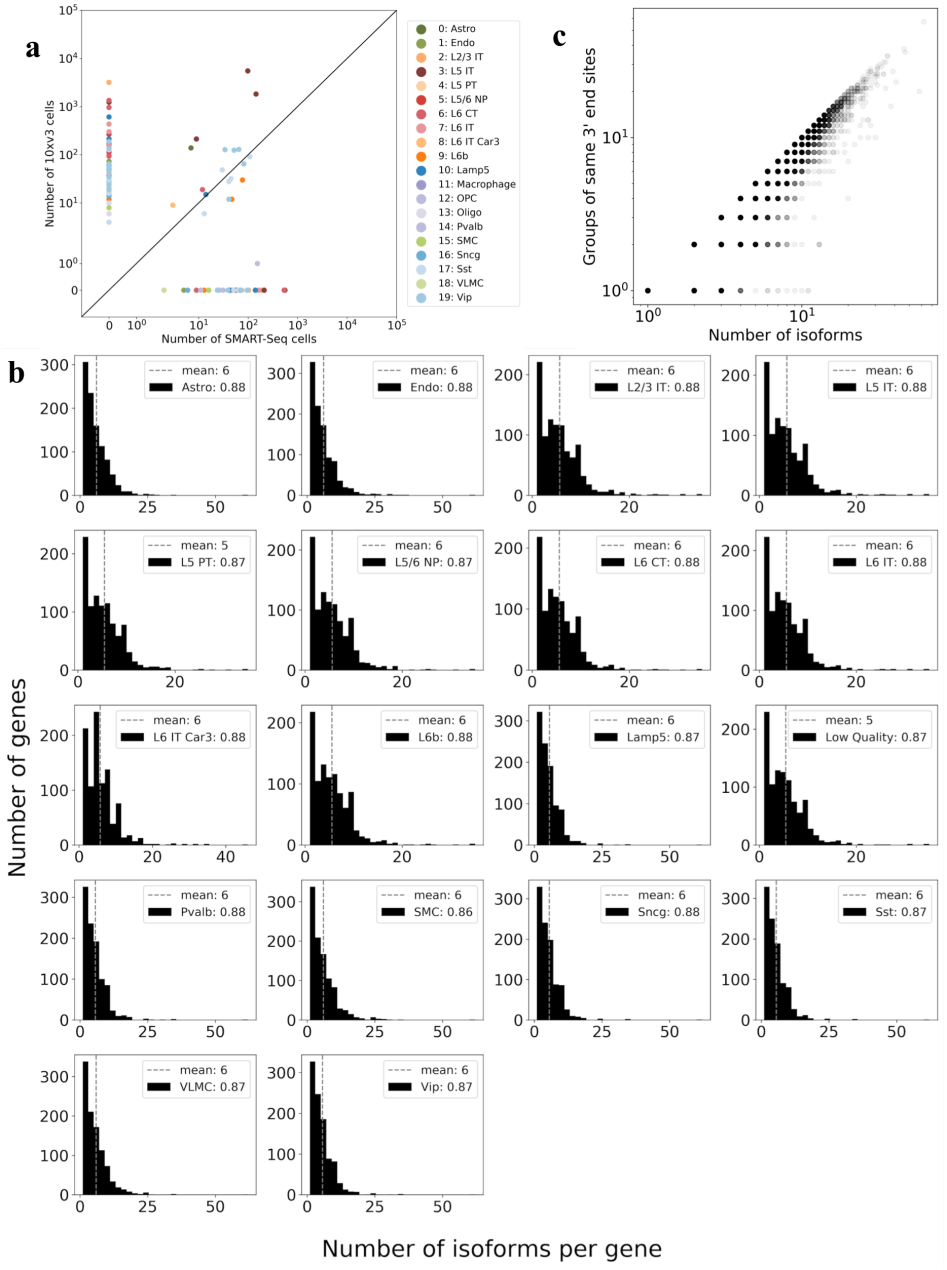
Methods References

51. Beerenwinkel, N., Pachter, L. & Sturmfels, B. Epistasis and shapes of fitness landscapes. *Statistica Sinica* 1317–1342 (2007).
52. Harris, J. A. *et al.* Anatomical characterization of Cre driver mice for neural circuit mapping and manipulation. *Front. Neural Circuits* **8**, 76 (2014).
53. Madisen, L. *et al.* A robust and high-throughput Cre reporting and characterization system for the whole mouse brain. *Nat. Neurosci.* **13**, 133–140 (2010).
54. Lein, E. S. *et al.* Genome-wide atlas of gene expression in the adult mouse brain. *Nature* **445**, 168–176 (2007).
55. Melsted, P. *et al.* Modular and efficient pre-processing of single-cell RNA-seq. *BioRxiv* (2019) doi:10.1101/673285.
56. Jiang, H. & Wong, W. H. Statistical inferences for isoform expression in RNA-Seq. *Bioinformatics* **25**, 1026–1032 (2009).
57. Trapnell, C. Defining cell types and states with single-cell genomics. *Genome Res.* **25**, 1491–1498 (2015).
58. Unable to find information for 4822624.
59. Salakhutdinov, R. & Hinton, G. Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure. (2007).
60. Ziegenhain, C. *et al.* Comparative Analysis of Single-Cell RNA Sequencing Methods. *Mol. Cell* **65**, 631-643.e4 (2017).
61. Trapnell, C. *et al.* Differential analysis of gene regulation at transcript resolution with RNA-seq. *Nat. Biotechnol.* **31**, 46–53 (2013).
62. Westoby, J., Herrera, M. S., Ferguson-Smith, A. C. & Hemberg, M. Simulation-based benchmarking of isoform quantification in single-cell RNA-seq. *Genome Biol.* **19**, 191 (2018).
63. Zhang, C., Zhang, B., Lin, L.-L. & Zhao, S. Evaluation and comparison of computational tools for RNA-seq isoform quantification. *BMC Genomics* **18**, 583 (2017).

Supplementary Material

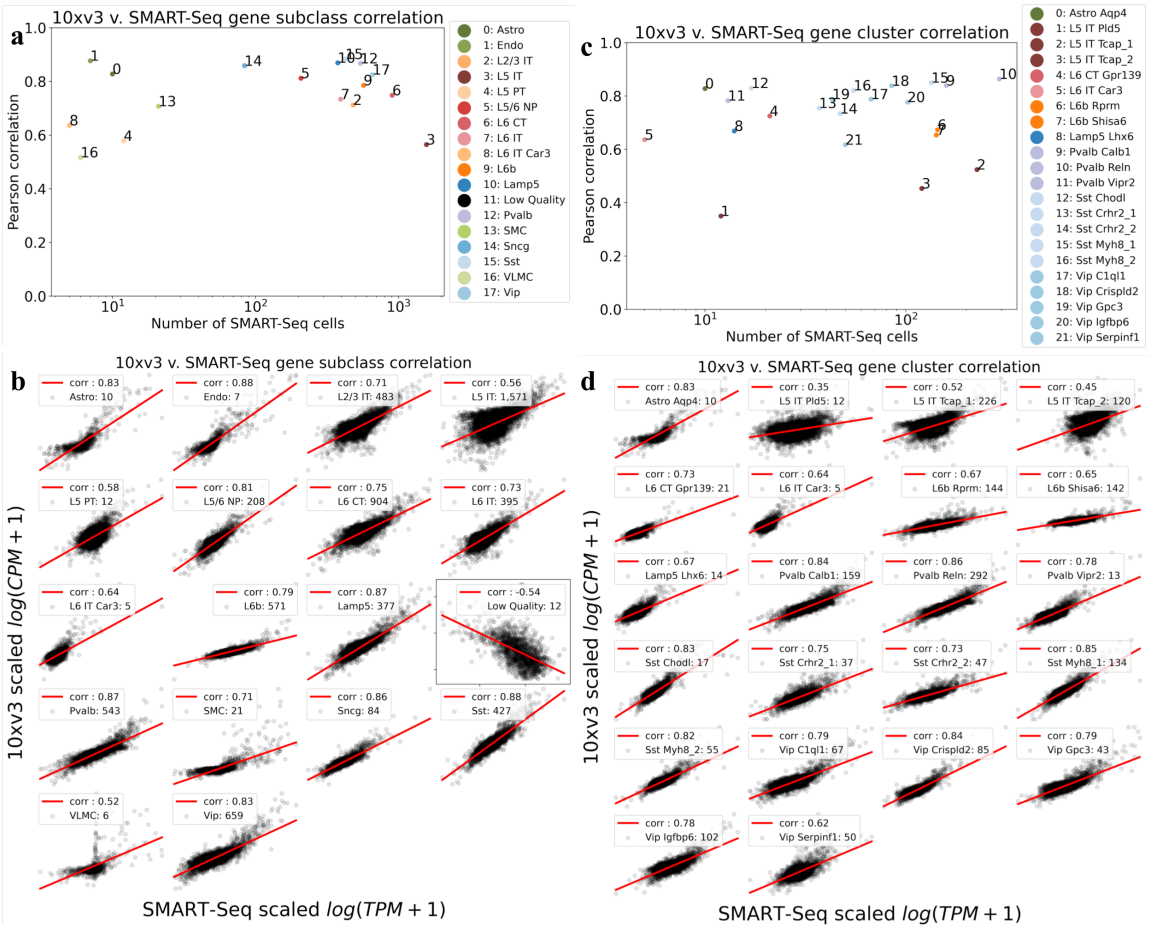


Extended Data Figure 1: 10xv3 neighborhood component analysis. Neighborhood component analysis (NCA) to 10 dimensions followed by t-distributed stochastic neighbor embedding (t-SNE) of 26,845 10xv3 cells from the mouse primary motor cortex annotated with cell subclass assignments. The number of cells in each subclass is displayed next to the subclass label. [Code]

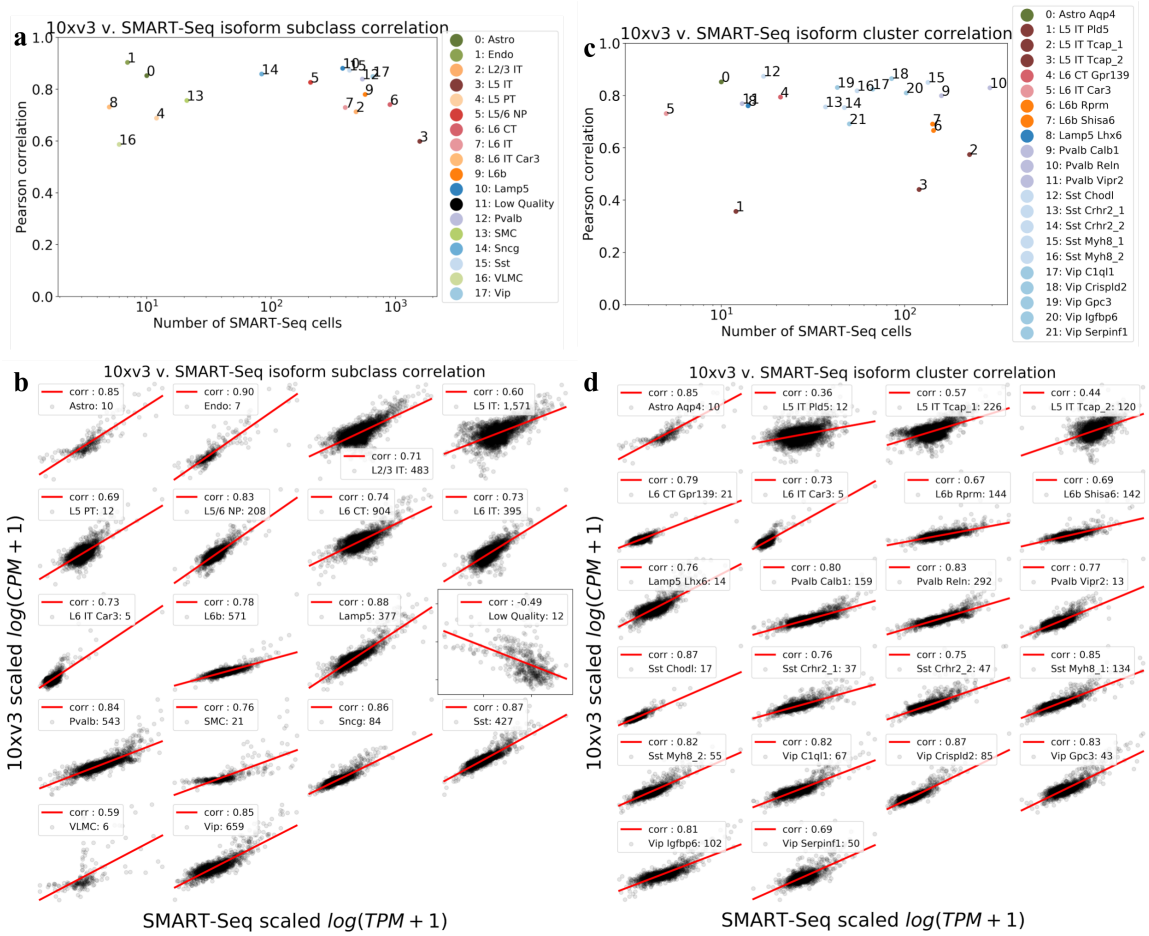


Extended Data Figure 2: Cell type identification and isoform summary statistics. a) Comparison of subclass identification for 10xv3 and SMART-Seq. Each technology identified subclasses separately, but with the same method. 56 clusters with gene markers were identified in the 10xv3 data but not in the SMART-Seq data while 39 clusters with gene markers were identified in the SMART-Seq data and not the Chromium data. b) The distribution of the number of isoforms per gene within each of 18 subclasses, computed from the top 998 most highly expressed genes in the SMART-Seq dataset. The number associated with each class indicates the fraction of genes for which there are more than

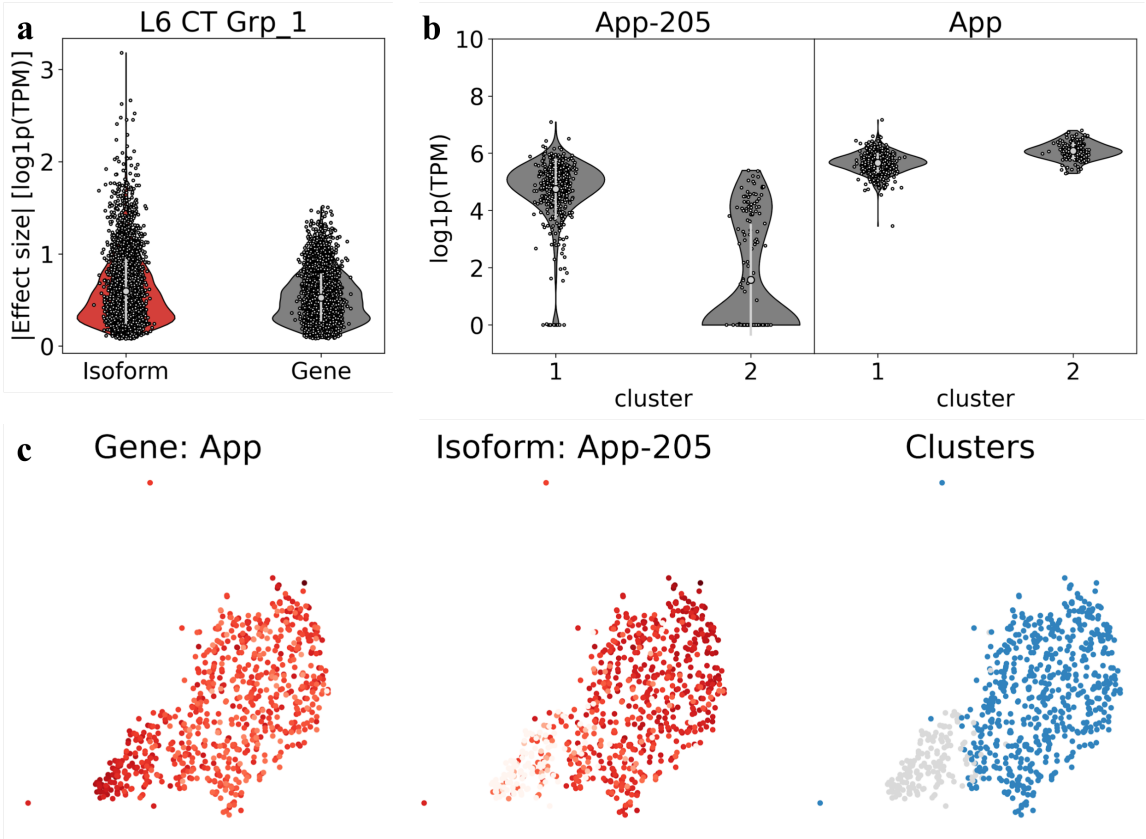
one isoform. c) Extent of isoform diversity in groups of transcripts sharing a 3’ end. Each point displays the density of the number of groups (y-axis) containing a given number of isoforms (x-axis) for a single gene. Points along the line $y = x$ correspond to genes where all transcripts contained within the gene have unique 3’ ends. [Code a, Code b, Code c]



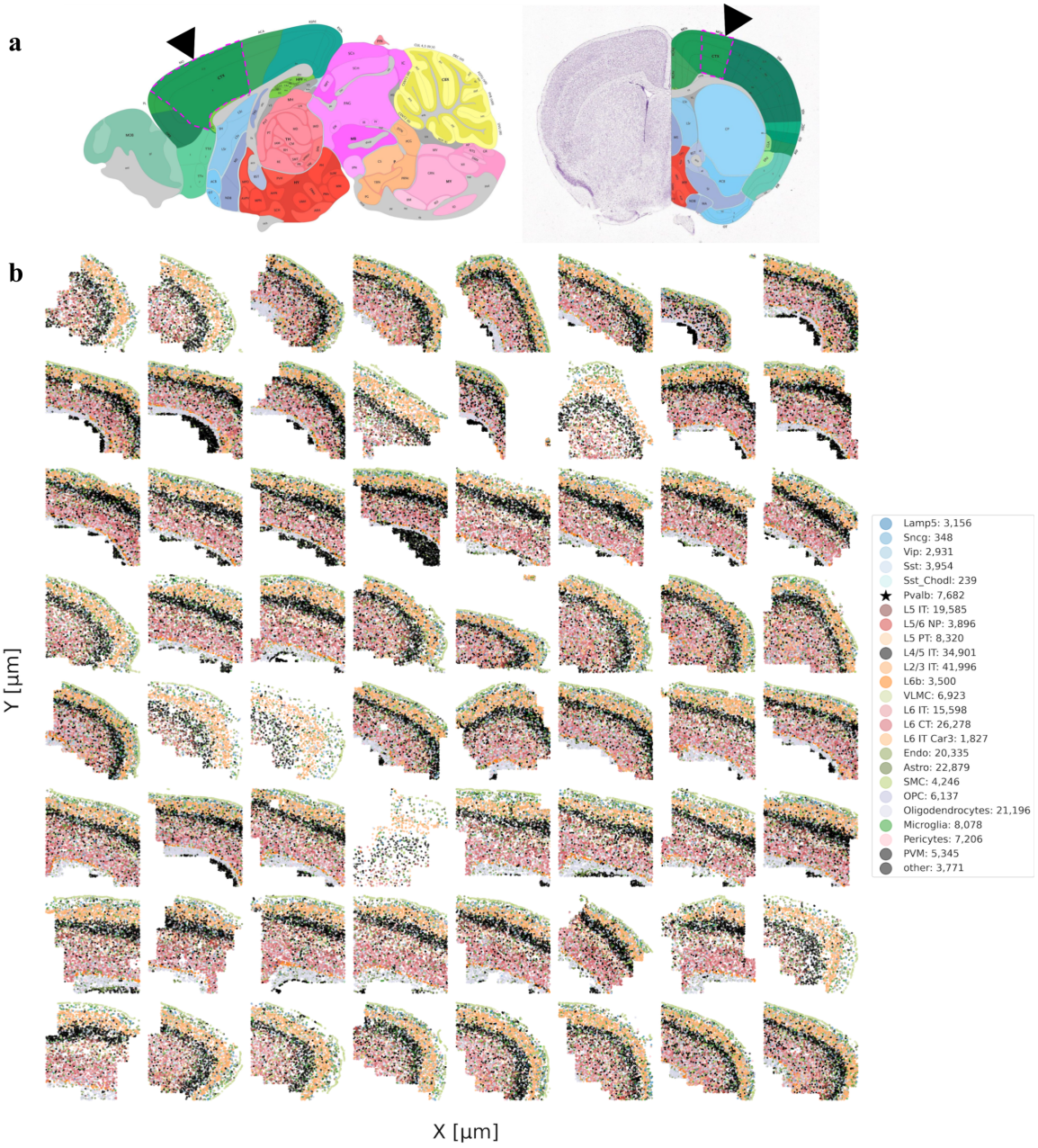
Extended Data Figure 3: Gene level subclass validation with 10xv3 and SMART-Seq. a) Pearson correlation by subclass of the mean gene expression in 10xv3 and the mean gene expression in SMART-Seq, against the size of the subclass, for genes that are expressed in at least 50% of cells in that subclass. b) Scatter plot by subclass of the mean gene expression in 10xv3 vs the mean gene expression in SMART-Seq for genes that are expressed in at least 50% of cells in that subclass. The subclass sizes and Pearson correlation values are also reported. c) Pearson correlation for common clusters in the 10xv3 and SMART-seq datasets, computed for each cluster with respect to genes expressed in at least 50% of cells of the cluster. d) Scatter plot by cluster of the mean gene expression in 10xv3 vs the mean gene expression in SMART-Seq, for genes that are expressed in at least 50% of cells. The cluster sizes and Pearson correlation values are also reported. [Code a, Code b, Code c, Code d]



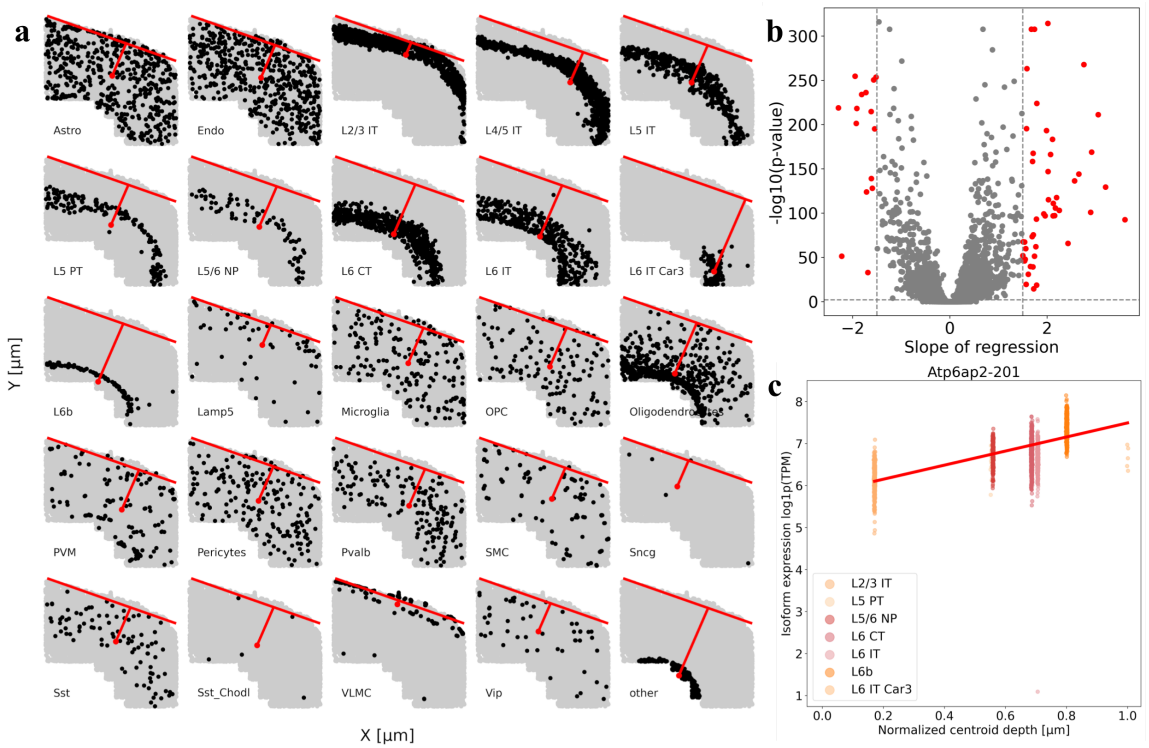
Extended Data Figure 4: Isoform level subclass validation with 10xv3 and SMART-Seq. a) Pearson correlation by subclass of the mean isoform expression in 10xv3 and the mean isoform expression in SMART-Seq, against the size of the subclass, for isoforms that are expressed in at least 50% of cells in that subclass. b) Scatter plot by subclass of the mean isoform expression in 10xv3 vs the mean isoform expression in SMART-Seq, for isoforms that are expressed in at least 50% of cells in that subclass. The subclass sizes and Pearson correlation values are also reported. c) Pearson correlation by cluster of the mean isoform expression in 10xv3 and the mean isoform expression in SMART-Seq, against the size of the cluster, for isoforms that are expressed in at least 50% of cells in that cluster for clusters that are common to both the SMART-Seq and 10x datasets. d) Scatter plot by cluster of the mean isoform expression in 10xv3 vs the mean isoform expression in SMART-Seq for isoforms that are expressed in at least 50% of cells in that cluster. The cluster sizes and Pearson correlation values are also reported. [Code a, Code b, Code c, Code d]



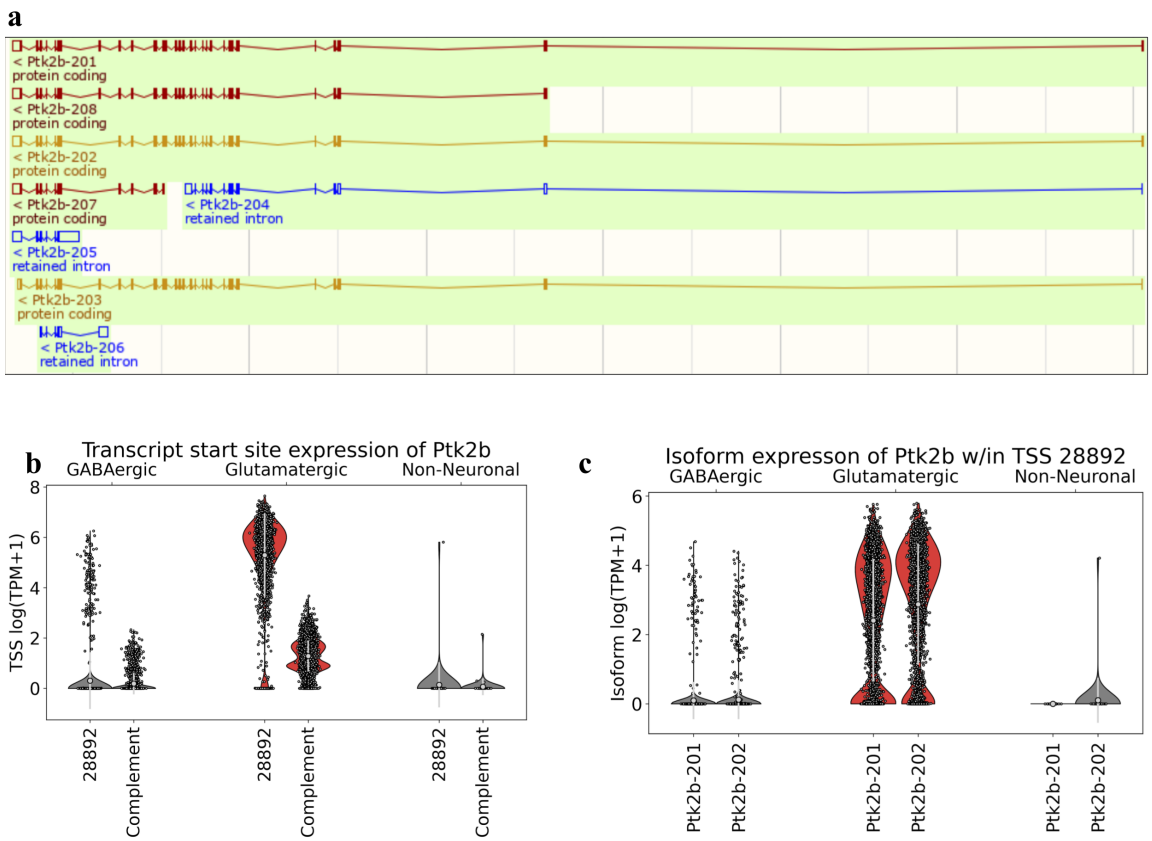
Extended Data Figure 5: Splitting clusters with k-means clustering on isoforms. a) Effect sizes for isoforms that split the L6 CT Grp_1 cluster into two parts is higher on average than that for genes. b) The *App-205* isoform splits the L6 CT Grp_1 cluster into two parts where one part has a higher expression of the isoform whereas both halves have similar *App* gene expression. The white circles within the violin plots represent the mean and the white bars represent + / - one standard deviation. c) Each point is a cell and is painted by the log1p(TPM) expression of the App gene (left) and App-205 isoform (middle). K-means clustering splits the L6 CT Grp_1 cluster into two distinct halves marked by expression of *App-205*. The white circles within the violin plots represent the mean and the white bars represent + / - one standard deviation. [Code a, Code b, Code c]



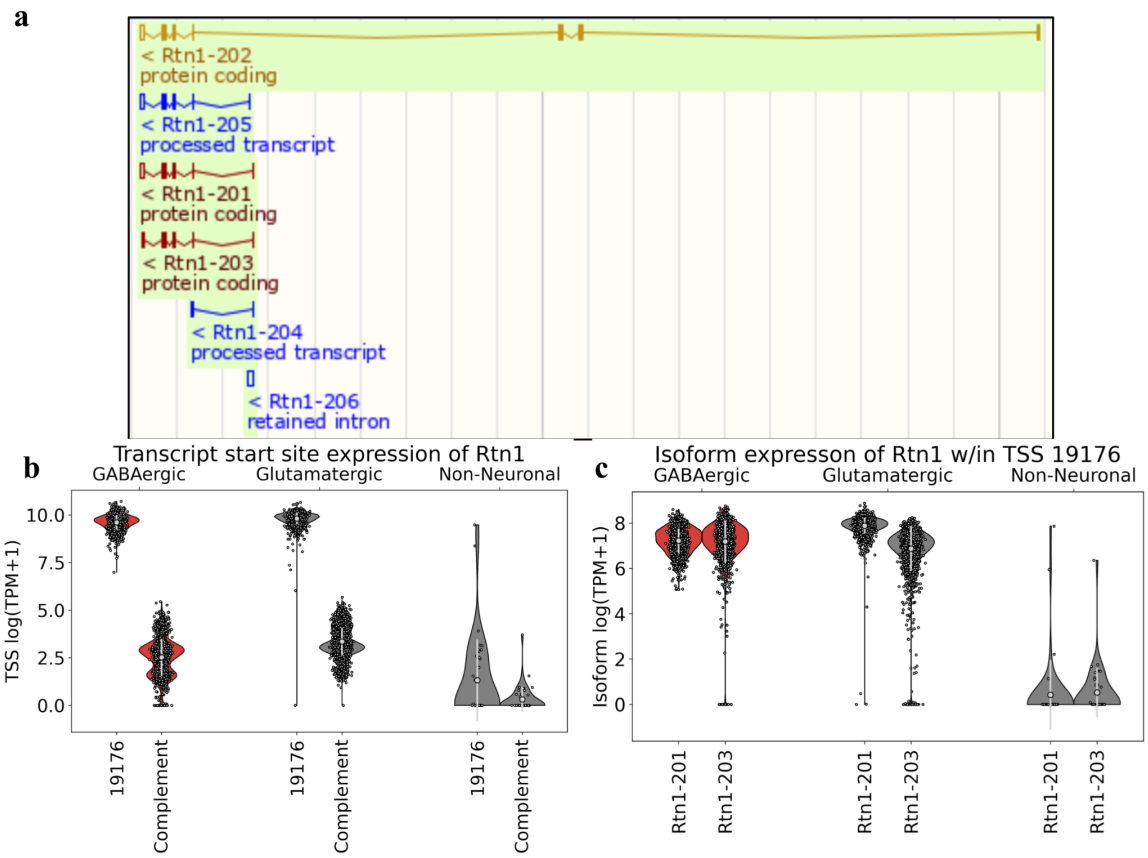
Extended Data Figure 6: Spatial localization of cell types in the mouse primary motor cortex. a) The location of the mouse primary motor cortex, outlined in pink and pointed to by a black arrow. The sagittal view (left) and coronal view (right) are shown. Image credit: Allen Institute. b) Spatial location of cells in all subclasses across 64 slices from the MOp assayed with MERFISH, the Pvalb cells are represented by a black star. [Source a, Code b]



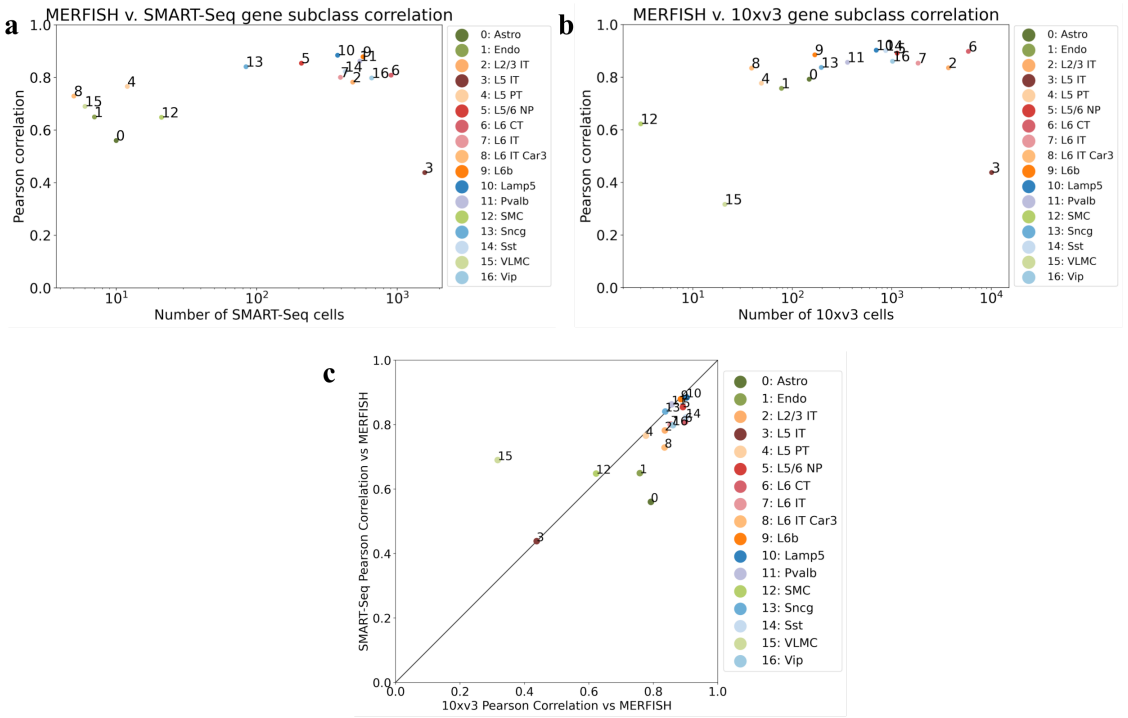
Extended Data Figure 7: Analysis of isoform expression gradients. a) A representative slice of the mouse primary motor cortex, as assayed by MERFISH, where each dot indicates the position of a cell from the corresponding subclass (black). The red point indicates the position of the centroid of those cells within the colored subclass with a line connecting the centroid to the boundary of the brain slice; the distance from the centroid to the slice boundary is indicated by the red line. b) Volcano plot of the set of isoforms found to be differentially expressed across the depth of the mouse primary motor cortex found using weighted least squares regression. The isoforms with a bonferroni corrected p-value less than 0.01 and regression slope greater than 1.5 are colored red. c) An example of an isoform that is colored red in plot (b). The expression of *Tubb2a-201* appears to increase across the depth of the motor cortex on average. [Code a, Code b, Code c]



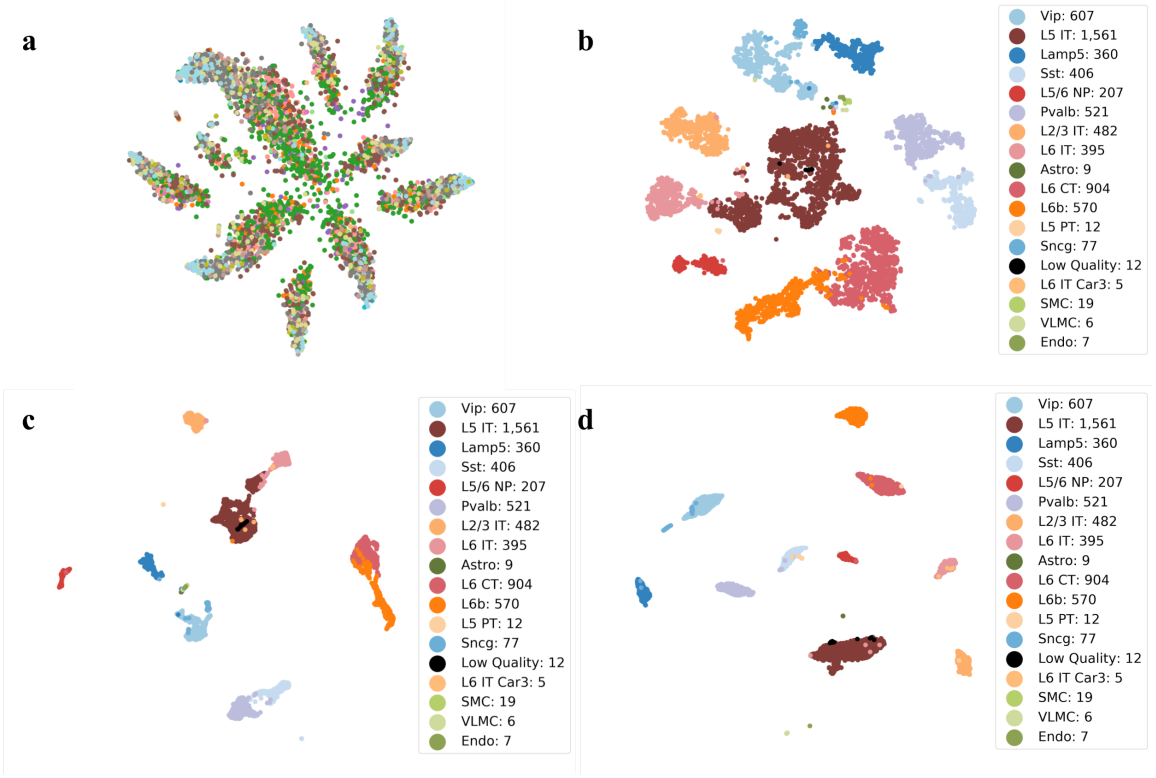
Extended Data Figure 8: Isoform shifts reflecting transcriptional changes. a) The eight isoforms of the *Ptk2b* gene. The 1st and 3rd isoforms from the top have the same transcription start site at the 5' end of the transcript. b) Expression patterns of groups of transcript sharing the same transcript start site (TSS) from the protein tyrosine kinase 2 (*Ptk2b*) gene. c) Expression patterns of isoforms within TSS groups from the *Ptk2b* gene. The white circles within the violin plots represent the mean and the white bars represent + / - one standard deviation. [Source Pkt2b, Code b, Code c]



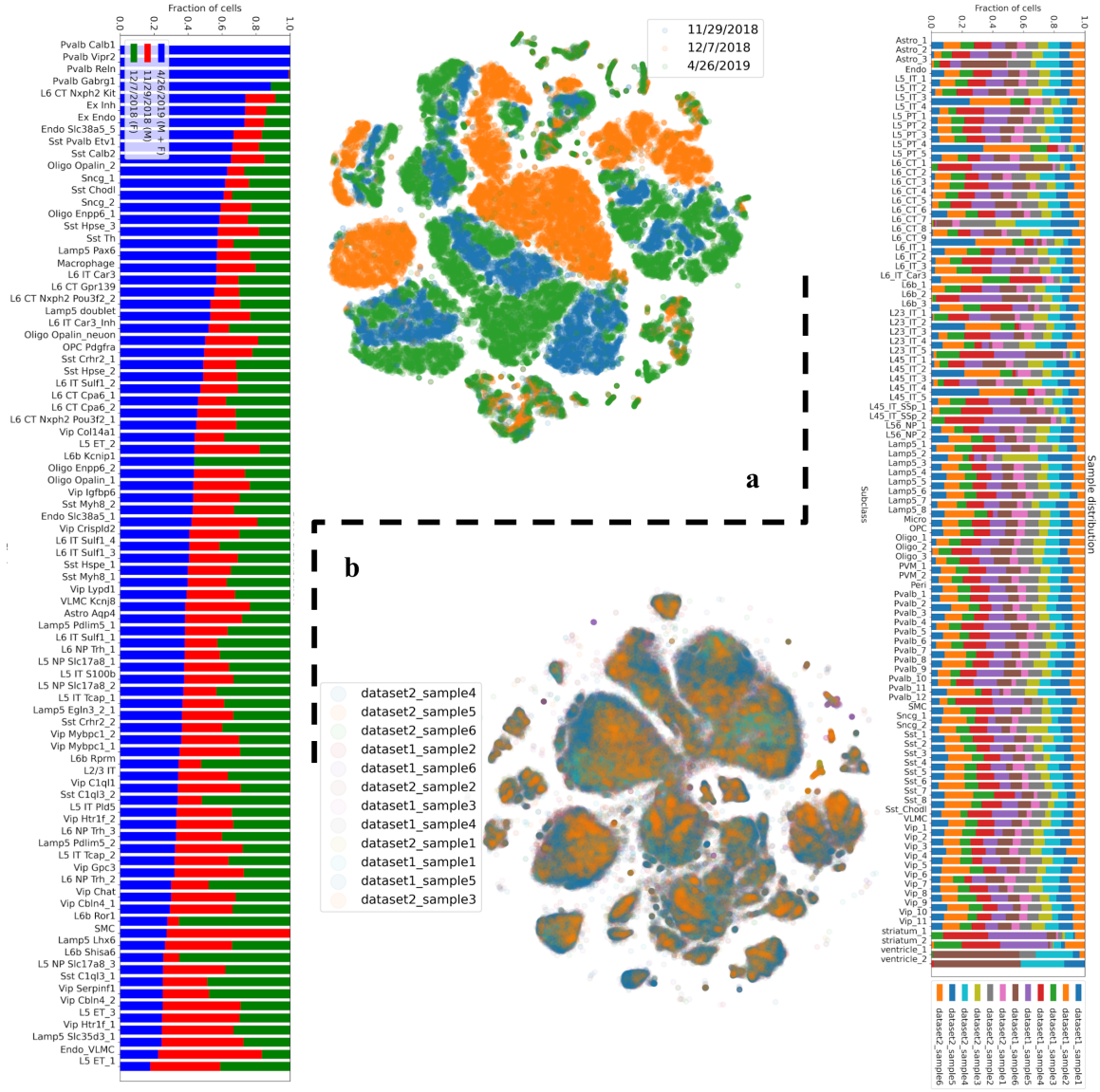
Extended Data Figure 9: Isoform shifts reflecting post-transcriptional changes. a) The six isoforms of the *Rtn1* gene. The 3rd and 4th isoforms from the top have the same transcription start site at the 5' end of the transcript. b) Expression patterns of groups of transcript sharing the same TSS from the reticulon 1 (*Rtn1*) gene. c) Expression patterns of isoforms within TSS groups from the *Rtn1* gene. The white circles within the violin plots represent the mean and the white bars represent + / - one standard deviation. [Source a, Code b, Code c]



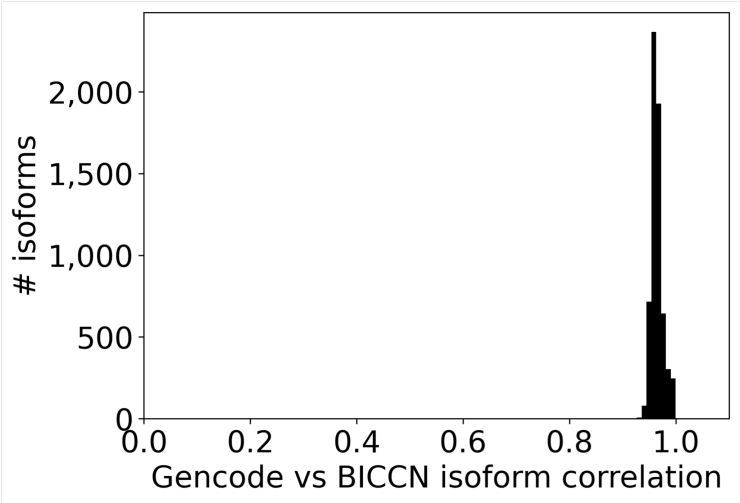
Extended Data Figure 10: Gene level subclass validation with SMART-Seq, 10xv3, and MERFISH. a) Pearson correlation by subclass of the mean gene expression in MERFISH and the mean gene expression in SMART-Seq, against the size of the subclass, for all 254 genes in the MERFISH data . b) Pearson correlation by subclass of the mean gene expression in MERFISH and the mean gene expression in 10xv3, against the size of the subclass, for all 254 genes in the MERFISH data. c) Comparison of gene correlations by cell type between 10xv3 and MERFISH, and SMART-Seq and MERFISH computed using the 254 genes assayed in the MERFISH dataset. [Code a, Code b, Code c]



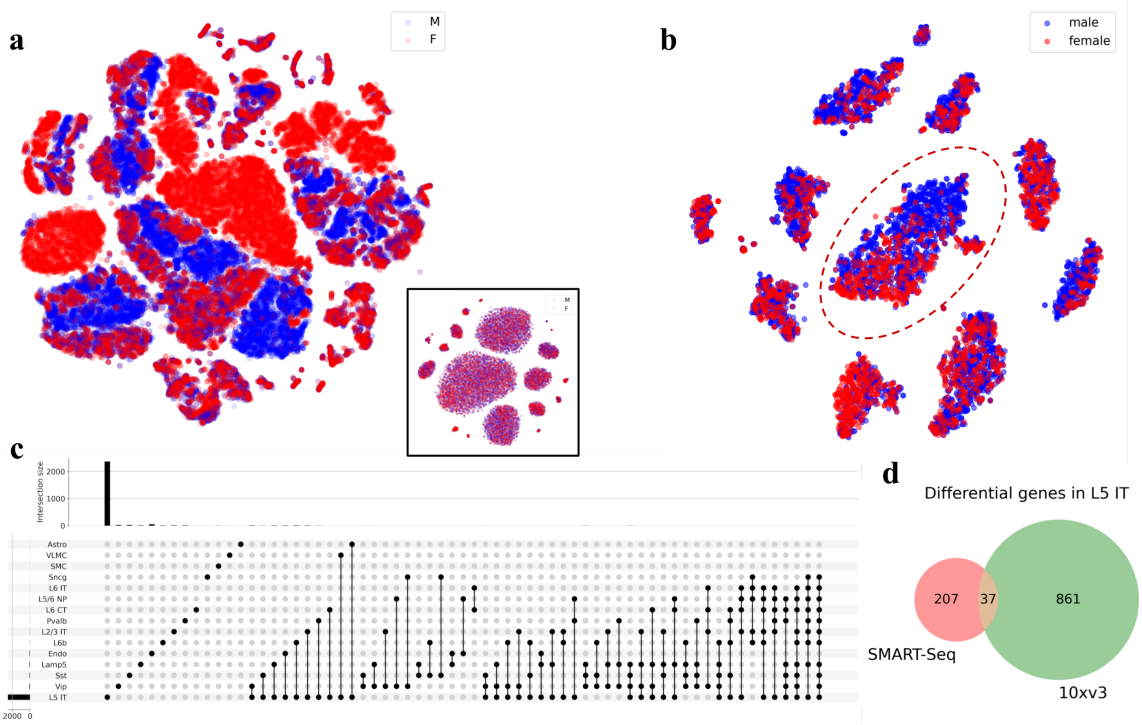
Supplementary Figure 1: a) NCA on the scaled log1p normalized SMART-Seq gene matrix with the subclass labels permuted randomly to ten components, followed by t-SNE to dimension two. The lack of separation of cells by label demonstrates that the NCA procedure is not overfitting the data. b) Truncated singular value decomposition (TSVD) on the scaled log1p normalized SMART-Seq gene matrix to 50 components followed by t-SNE to two dimensions. The numbers next to each subclass label indicate the number of cells in the subclass. c) TSVD on the scaled log1p normalized SMART-Seq gene matrix to 50 components followed by uniform manifold approximation and projection (UMAP) to two dimensions. The numbers next to each subclass label indicate the number of cells in the subclass. d) Ten components of the NCA on the scaled log1p normalized SMART-Seq gene matrix followed by UMAP to two dimensions. The numbers next to each subclass label indicate the number of cells in the subclass. [Code a, Code b, Code c, Code d]



Supplementary Figure 2: a) Fraction of cells per cluster in the 10xv3 dataset originating from a specific processing date. 24,348 male cells were assayed on 11/29/2018, 32,145 female cells were assayed on 12/7/2018, and 18,140 male cells and 15,398 female cells were assayed on 4/26/2019. The accompanying TSVD to 50 components followed by t-SNE to two dimensions of the 10xv3 data shows each cell painted by the date it was assayed. The separation by date indicates a strong batch effect by date. b) t-SNE to two components of the MERFISH dataset where each cell is painted by the sample it originated from, alongside the fraction of cells per cluster in the MERFISH dataset originating from a specific sample. [Code a, Code b]

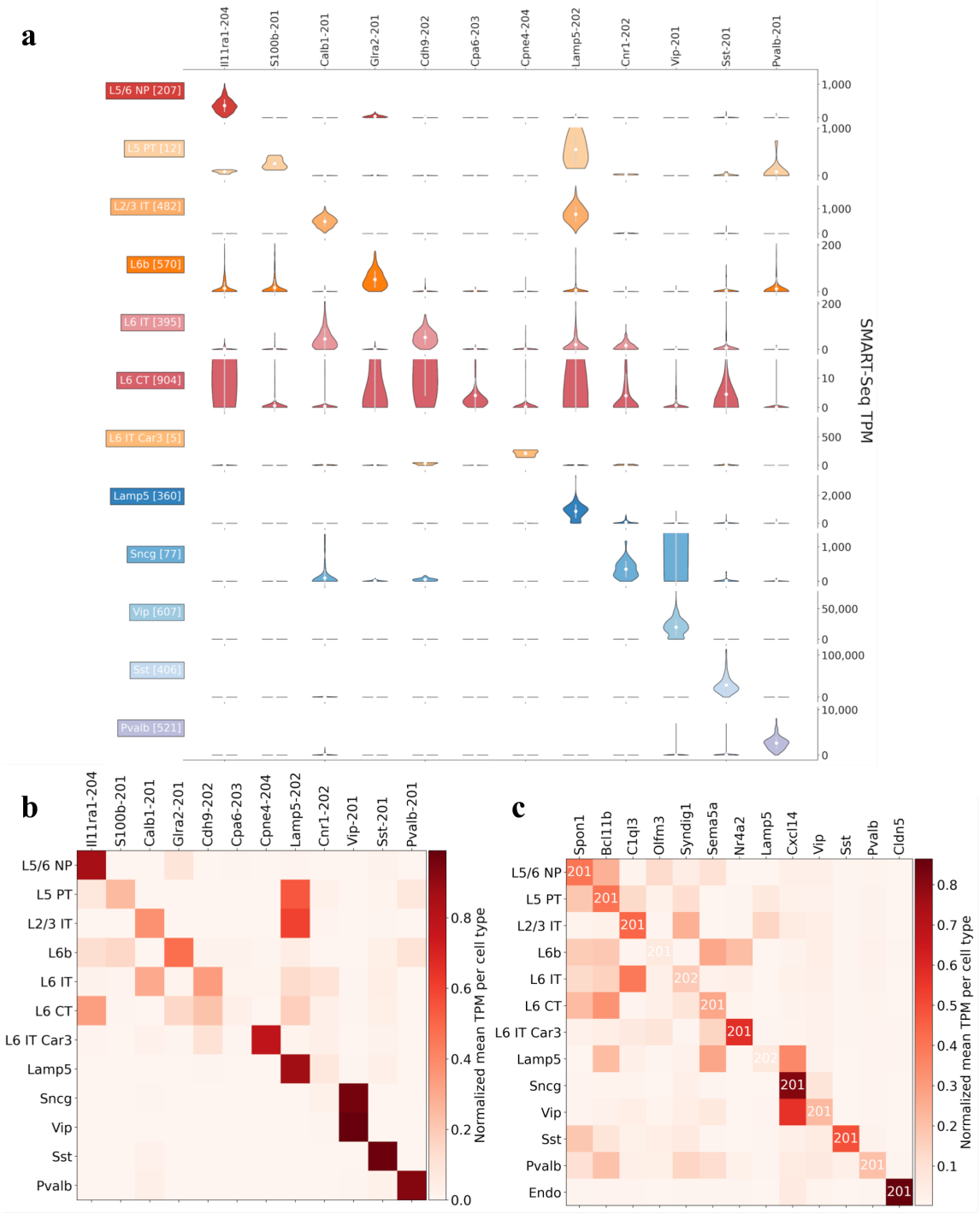


Supplementary Figure 3: Comparison of isoform quantifications obtained with respect to the Gencode M25 and BICCN annotations. The average Pearson correlation across 107,639 isoforms was 0.965. Code



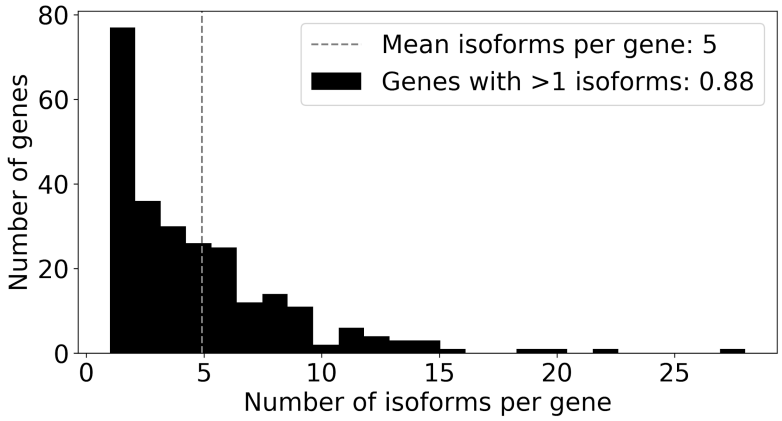
Supplementary Figure 4: a) TSVD followed by t-SNE of all of the 10xv3 data. The clusters segregate by sex. The inset graph shows NCA followed by t-SNE on a subset of the 10xv3 data. The subset graph is showing only the cells assayed on 4/26/2019. The clusters do not appear to segregate by sex. b) NCA followed by t-SNE of the SMART-Seq dataset. Each cell is painted by the sex of the animal it originated from. The

L5 IT subclass is enclosed with a dotted red ellipse. c) Upset plot showing the number of isoforms that are unique and shared between subclasses after differential expression based on sex within each subclass for the SMART-Seq dataset. d) Venn diagram of the number of differential genes shared between the SMART-Seq and 10xv3 quantifications for the L5 IT subclass. [Code a (subset), Code b, Code c, Code d]

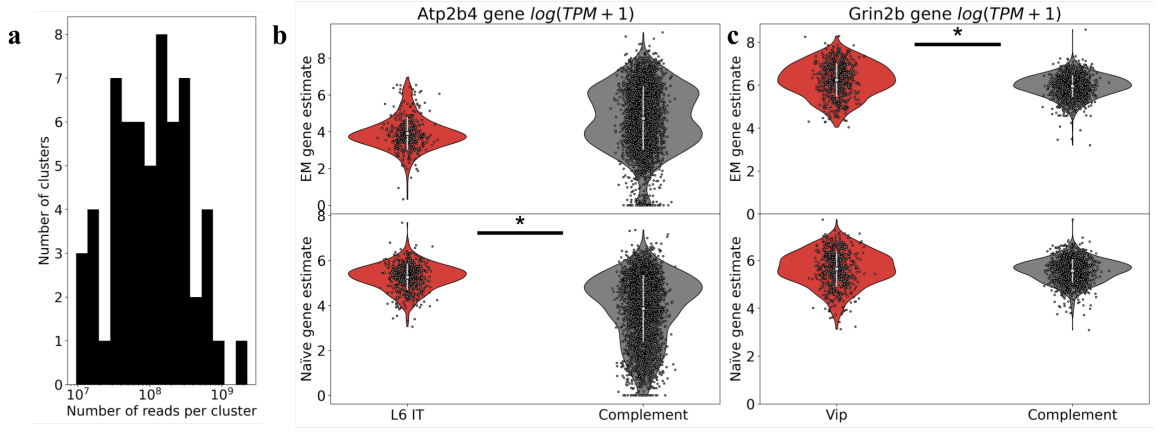


Supplementary Figure 5: a) A sample from an isoform atlas displaying isoform markers differential with respect to subclasses. Each row corresponds to one subclass, and each column corresponds to one isoform. SMART-Seq isoform abundance estimates are in TPM units, and each column is scaled so that the maximum TPM is 4 times the mean of the isoform specific for that row's cluster. The white circles within the violin plots

represent the mean and the white bars represent + / - one standard deviation. b) A sample from an isoform atlas displaying isoform markers differential with respect to subclasses. Each row corresponds to one subclass, and each column corresponds to one isoform. SMART-Seq isoform abundance estimates are in TPM units, and each row is scaled by the sum of the mean expressions of each isoform for all cells in that subclass. c) A sample from a spatial-isoform atlas inferring spatial isoform markers for cell types with known spatial locations and known gene markers. Each row corresponds to one subclass, and each column corresponds to one gene with the inferred differential isoform from that gene, determined by the SMART-Seq data, along the diagonal. SMART-Seq isoform abundance estimates are in TPM units, and each row is scaled by the sum of the mean expressions of each gene for all cells in that subclass. [Code a, Code b, Code c]



Supplementary Figure 6: The distribution of the number of isoforms per gene for all of the genes in the MERFISH dataset. 88% of genes in the MERFISH dataset have more than one isoform. Code



Supplementary Figure 7: a) Distribution of read depth per SMART-seq cell showing sufficient depth for accurate quantification with the expectation-maximization (EM)

algorithm. b) Comparison of two quantifications approaches: EM gene quantification where isoform abundances are estimated, and then added up to obtain a gene abundance estimate, versus naive quantification in which read counts are collated by gene locus (* indicates statistically significant difference $p<0.01$). In this case naïve quantification introduces a possibly incorrect gene marker for the Pvalb subclass. c) In this case naïve quantification does not identify a gene marker for the L6 CT subclass. The white circles within the violin plots represent the mean and the white bars represent + / - one standard deviation. [Code a, Code b,c]

Chapter 5

Conclusion & Outlook

Conclusion

In this thesis I've demonstrated how mechanical engineering and bioinformatics solutions can work hand-in-hand to address challenges that must be met in order to realize the goal of bringing single-cell RNA sequencing to the clinic (Owen et al. 2021). I've demonstrated both how to open-source and reduce cost of key devices, and how to facilitate sequence processing and analysis using novel ideas in scientific computation.

The *poseidon* and *colosseum* devices, presented in Chapter 2, are instruments that enable massive scale single-cell isolation and collection. They each have novel design elements that reduce cost and enable modularity, at a similar accuracy to expensive commercial alternatives. They are open-source tools that are readily modifiable and extensible as proven by their rapid adoption by multiple groups around the world that have used, studied, benchmarked, and modified these devices (Akkoyun and Özçelik 2020; Shin and Choi 2021; Prof and Smit 2021; Carbonell Rubio, Weber, and Klotzsch 2022; Bharadwaj and Verma 2021; Iannone et al. 2022).

The *kallisto* | *bustools* command-line tools, presented in Chapter 3, make scalable scRNAseq analysis fast and efficient. They implement novel algorithms for sequence read-alignment, barcode error correction, and molecular counting that helps resolve ambiguities in sequence mapping. These algorithms offer tradeoffs between speed and accuracy that I have benchmarked and documented (Melsted et al. 2021). For example, alignment using exact subsequence matching against a de-Bruijn graph representation of the reference makes read alignment fast but returns possibly ambiguous read alignments. The *bustools* barcode error correction and molecular counting use the uniqueness of molecular and cellular barcodes to help resolve these sequence alignment ambiguities. Stream-wise processing of the sequencing reads make the *kallisto* | *bustools* workflow memory-efficient, a crucial property for low-cost computing, and, crucially, facilitating adoption of single-cell RNA-seq in the clinic. These novel algorithmic tools and software engineering choices have enabled novel data analysis. Using the *kallisto* | *bustools* workflow, I produced the largest RNA velocity analysis at the time of publication. I was able to infer transcriptional rates for each of 27,740 genes in 113,917 cells undergoing neurogenesis in the mouse retina. Computing transcriptional rates for genes is crucial for

the development of biomolecular feedback circuits where speed and robust responses to input signals are desired (Pandey and Murray 2022).

Refining gene expression data to the isoform level is crucial for understanding transcriptional regulation and the effects of alternative splicing in biological processes. Towards that end, I have extended the *kallisto* | *bustools* workflow to process full-length single-cell RNA-sequencing taking advantage of expectation maximization algorithm to disambiguate ambiguous sequence alignments. Using these tools I assembled the first ever spatially-resolved single-cell isoform atlas, and in particular one of great interest in the neuroscience community (the mouse primary motor cortex) with data generated with three RNA-sequencing assays, as described in Chapter 4. Additionally, I proposed a novel framework for using data generated from multiple modalities of RNA sampling in order to infer spatial isoform expression of cell types (Booeshaghi et al. 2021).

Taken together, these tools make technical advancement in instrumentation and algorithmic methods for large-scale data analysis, as well as biological discovery. Engineering hardware and software tools under speed and cost constraints addresses fundamental issues that represent roadblocks in critical technologies for engineering our biophysical environment. Furthermore, they offer a first-step towards enabling translational research to better understand health and human disease.

Outlook

There are multiple fundamental engineering challenges that remain to be addressed in order to fully translate single-cell RNA-sequencing from a research tool to a routine biomedical readout in the clinic. First, scRNAseq requires cell isolation, and this process is limited by the rate of cell encapsulation. Microfluidics and microwells are two such ways to put one cell and one bead into a single isolated chamber but they have inherent limits. Technological limitations of Poisson loading mean there are many isolation chambers without a cell or without a bead or both(Collins et al. 2015). Coupled with a throughput of 500 cells per minute, encapsulating millions or billions of cells is currently impractical. These throughput limits are imposed by the stochastic nature of particle-in-chamber loading and necessitate further research into nanoscale device design to overcome these limits. Techniques towards this direction take advantage of the elastic nature of the hydrogel beads used in scRNAseq in order to gain sub-poisson loading of droplets (Abate et al. 2009), other technologies deform PDMS microwells to trap individual cells (Bose et al. 2015), and others use acoustics (Link et al. 2021). This increased scale of experimentation is necessary if we are to transcriptomically profile all 37.2 trillion cells in a human's body, routinely profile hundreds of thousands of blood cells from patients undergoing blood tests, or assay the tumor cells of cancer patients as

they undergo treatment. The *poseidon* syringe pump system and *colosseum* fraction collector are first attempts at tackling the cost of current microfluidic-based cell-encapsulation technologies in order to understand and overcome the inherent technological limitations that prevent massive scale experiments.

Second, scRNAseq data analysis requires constitutive models that govern RNA production and degradation. With such models, scRNAseq data can be used to infer maximum likelihood estimates on transcription rate parameters that offer an interpretable view of the physical processes within a cell. Parameter inference, on the increasing amounts of scRNAseq data, will require further development of scalable and efficient algorithms where the speed-accuracy tradeoffs are understood and documented (Gorin and Pachter 2020; Vastola et al. 2021) Towards this goal I've started collaborating with colleagues to understand how this inference procedure can be performed and how the estimated rate parameters can be analyzed.

Third, an interdisciplinary approach must be taken to close the device-to-analysis feedback loop whereby data generated informs analysis methods and methods informs the design, improvement, and modification of data generating devices.

Lastly, I'm excited about monitoring the impact of anthropogenic climate change on species health, using the individual species as an "analog sensor" and scRNAseq as a data acquisition sampling procedure on that sensor. Species' health is impacted by environmental changes. For example, increasing temperatures are associated with smaller body sizes in North American migratory birds (Weeks et al. 2020) and reduced fitness of beetles to freezing soils, due to shallower snow depths (Harris, Rodenhouse, and Holmes 2019). However, tools to model the diverse set of ways in which perturbed habitats impact species molecular health are currently lacking. Additionally, exponentially increasing amounts of sensor and genomic data require correspondingly scalable methods. These methods will help us better understand how cells engineer the physical environment that makes up the bodies of those organisms and how these internal environments interact with the external one.

The challenges of single-cell RNA sequencing transcend the boundaries of traditional academic disciplines, and the field of mechanical engineering that aims to address roadblocks in critical technologies towards engineering our environment, is central to this endeavor. My hope is that the work in this thesis is valuable not only for the technical addressing key challenges to pressing problems in single-cell genomics, but also that the coupling between hardware and software for a key biomedical application demonstrates the centrality of mechanical engineering for emerging technologies in biology.

References

Abate, Adam R., Chia-Hung Chen, Jeremy J. Agresti, and David A. Weitz. 2009. "Beating Poisson Encapsulation Statistics Using Close-Packed Ordering." *Lab on a Chip* 9 (18): 2628–31.

Akkoyun, Fatih, and Adem Özçelik. 2020. "A Simple Approach for Controlling an Open-Source Syringe Pump." *European Mechanical Science* 4 (4): 166–70.

Bharadwaj, Tanmay, and Devendra Verma. 2021. "Open Source Bioprinters: Revolutionizing the Accessibility of Biofabrication." *Bioprinting* 23 (August): e00155.

Booeshaghi, A. Sina, Zizhen Yao, Cindy van Velthoven, Kimberly Smith, Bosiljka Tasic, Hongkui Zeng, and Lior Pachter. 2021. "Isoform Cell-Type Specificity in the Mouse Primary Motor Cortex." *Nature* 598 (7879): 195–99.

Bose, Sayantan, Zhenmao Wan, Ambrose Carr, Abbas H. Rizvi, Gregory Vieira, Dana Pe'er, and Peter A. Sims. 2015. "Scalable Microfluidics for Single-Cell RNA Printing and Sequencing." *Genome Biology* 16 (June): 120.

Carbonell Rubio, Dani, Willi Weber, and Enrico Klotzsch. 2022. "Maasi: A 3D Printed Spin Coater with Touchscreen." *HardwareX* 11 (April): e00316.

Collins, David J., Adrian Neild, Andrew deMello, Ai-Qun Liu, and Ye Ai. 2015. "The Poisson Distribution and beyond: Methods for Microfluidic Droplet Production and Single Cell Encapsulation." *Lab on a Chip* 15 (17): 3439–59.

Gorin, Gennady, and Lior Pachter. 2020. "Special Function Methods for Bursty Models of Transcription." *Physical Review. E* 102 (2-1): 022409.

Harris, Jennifer E., Nicholas L. Rodenhouse, and Richard T. Holmes. 2019. "Decline in Beetle Abundance and Diversity in an Intact Temperate Forest Linked to Climate Warming." *Biological Conservation* 240 (December): 108219.

Iannone, Marco, Diego Caccavo, Anna Angela Barba, and Gaetano Lamberti. 2022. "A Low-Cost Push-Pull Syringe Pump for Continuous Flow Applications." *HardwareX* 11 (April): e00295.

Link, Andreas, John S. McGrath, Mustafa Zaimagaoglu, and Thomas Franke. 2021. "Active Single Cell Encapsulation Using SAW Overcoming the Limitations of Poisson Distribution." *Lab on a Chip* 22 (1): 193–200.

Melsted, Páll, A. Sina Booeshaghi, Lauren Liu, Fan Gao, Lambda Lu, Kyung Hoi Joseph Min, Eduardo da Veiga Beltrame, Kristján Eldjárn Hjörleifsson, Jase Gehring, and Lior Pachter. 2021. "Modular, Efficient and Constant-Memory Single-Cell RNA-Seq Preprocessing." *Nature Biotechnology* 39 (7): 813–18.

Owen, Mallory J., Anna-Kaisa Niemi, David P. Dimmock, Mark Speziale, Mark Nespeca, Kevin K. Chau, Luca Van Der Kraan, et al. 2021. "Rapid Sequencing-Based Diagnosis of Thiamine Metabolism Dysfunction Syndrome." *The New England Journal of Medicine* 384 (22): 2159–61.

Pandey, Ayush, and Richard M. Murray. 2022. "Robustness Guarantees for Structured Model Reduction of Dynamical Systems with Applications to Biomolecular Models." *International Journal of Robust and Nonlinear Control*, January. https://doi.org/10.1002/rnc.6013.

Prof, Meijboom R., and E. Smit. 2021. "Design and Optimization of a 3D-Printed Flow

System for Semi-Automated Sample Preparation." https://ujcontent.uj.ac.za/vital/access/manager/Repository/uj:45133?site_name=GlobalView.

Shin, Joong Ho, and Sungyoung Choi. 2021. "Open-Source and Do-It-Yourself Microfluidics." *Sensors and Actuators. B, Chemical* 347 (November): 130624.

Vastola, John J., Gennady Gorin, Lior Pachter, and William R. Holmes. 2021. "Learning the Dynamics of Bursty Transcription and Splicing Using Ultra-Fast Parameter Inference and New Analytical Solutions of the Chemical Master Equation." *Biophysical Journal* 120 (3, Supplement 1): 135a.

Weeks, Brian C., David E. Willard, Marketa Zimova, Aspen A. Ellis, Max L. Witynski, Mary Hennen, and Benjamin M. Winger. 2020. "Shared Morphological Consequences of Global Warming in North American Migratory Birds." *Ecology Letters* 23 (2): 316–25.