Statistical and computational methods for single-cell transcriptome sequencing and metagenomics

by

Fanny Gabrielle Solange Marie Perraudeau

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Biostatistics

and the Designated Emphasis

in

Computational and Genomic Biology

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Sandrine Dudoit, Chair
Professor Elizabeth Purdom
Professor John Taylor

Spring 2018

Statistical and computational methods for single-cell transcriptome sequencing and metagenomics

Abstract

Statistical and computational methods for single-cell transcriptome sequencing and metagenomics

by

Fanny Gabrielle Solange Marie Perraudeau

Doctor of Philosophy in Biostatistics
and the Designated Emphasis in
Computational and Genomic Biology

University of California, Berkeley

Professor Sandrine Dudoit, Chair

I propose statistical methods and software for the analysis of single-cell transcriptome sequencing (scRNA-seq) and metagenomics data. Specifically, I present a general and flexible zero-inflated negative binomial-based wanted variation extraction (ZINB-WaVE) method, which extracts low-dimensional signal from scRNA-seq read counts, accounting for zero inflation (dropouts), over-dispersion, and the discrete nature of the data. Additionally, I introduce an application of the ZINB-WaVE method that identifies excess zero counts and generates gene and cell-specific weights to unlock bulk RNA-seq differential expression pipelines for zero-inflated data, boosting performance for scRNA-seq analysis. Finally, I present a method to estimate bacterial abundances in human metagenomes using full-length 16S sequencing reads.

To my parents and brother,

Contents

List of Figures

List of Tables

Acknowledgments

A tremendous MERCI to my advisor, Sandrine Dudoit. Sandrine is a venerable statistician and a joy to be around. I have learned so much from her and it was an honour to have been her student. Her intellectual rigor and open-mindedness made a lasting impression and is something I will always cherish. The Ph.D. with Sandrine has been a journey and a School of Life.

I want to thank Jim Bullard for giving me such an exciting problem to work on. He taught me critical thinking on real life problems. Even after sleep deprivation and between feeding-bottles and diapers, he has "helped us to graduate". I have thoroughly enjoyed working with Jim and the amazing team at Whole Biome.

Davide Risso and Jean-Philippe Vert provided me with very valuable advice. Their deep understanding of Statistics and Genomics (coupled with numerous espressos and diners) has been great source of inspiration.

I am grateful to Koen Van den Berge, Lieven Clement, and the zingeR team for our enriching collaboration.

Veronique Stoven who inspired me to start the Ph.D. asked me the right questions to find my way after graduate school (and ever after).

I am thankful to John Taylor and Pierre Gladieux for providing me the golden opportunity to come for a research internship at UC Berkeley. It opened up a whole new world to me.

To Kelly Street, the best co-PhD I could have hoped for. To Nelle Varoquaux for the morning/afternoon/evening cafés at BIDS. To the Dudoit/Purdom group members for advice on presentations. To all my family, friends, and roommates. Thank you! My time with you make up most of this experience, and I thank you all for making it special.

I owe my deepest gratitude to my parents and my brother, to whom I have dedicated this thesis. I thank them for their eternal love and support, their grit in life inspired the efforts I put into my own.

And finally, I want to thank Guillaume for his constant love and exceptional patience. This journey is all the better because of him.

Chapter 1

Introduction

This introduction provides an overview of the current state of genomics and gives relevant biological background for readers not familiar with the field. The dissertation is composed of chapters, where, for each chapter, a paper was either already published or submitted for publication. Although each chapter has been written with care and is internally consistent, the notation can be different from one chapter to another. In particular, Chapter 4 addresses a completely distinct question than Chapters 2 and 3.

1.1 The rise of personalized genomics

Genomics is defined as the study of the structure, function, evolution, mapping, and editing of genomes, where a genome is an organism's complete set of DNA, including all of its genes [1]. Genomics aims at the characterization and quantification of genes, which direct the production of proteins with the assistance of enzymes and messenger molecules. In turn, proteins make up body structures such as organs and tissues as well as control chemical reactions and carry signals between cells. Genomics also involves the sequencing and analysis of genomes through uses of high-throughput DNA sequencing and informatics to assemble and analyze the function and structure of entire genomes.

Advances in genomics have given rise to personalized or precision medicine, referring to the fact that patients are separated into different groups, with medical decisions and interventions being tailored to the individual patient based on their predicted response or risk of disease. Skeptics would say that tailoring of treatment to patients is nothing new. This is true. As far back as in ancient Greece, Hippocrates' hypothesis was already that "It is more important to know what sort of person has a disease than to know what sort of disease a person has" [2]. However, the term personalized medicine has risen in usage in recent years driven by the fact that genomics could provide clear evidence on which to stratify (i.e., group) patients according to their genome.

A compelling example showing that personalized medicine could play an effective role in stratifying patient populations is breast cancer. There are actually two categories of women suffering from breast cancer: those with a mutation on a gene coding for a protein called human epidermal growth factor receptor 2 (HER2) - which promotes the growth of cancer cells - and those without this mutation. Testing for the mutation, it is then possible to predict if a patient will response to a particular treatment called Herceptin. Thus, using genomic testing, the right treatment can be applied to the right patient sparing the non-responsive patients the cost and pain of an intense cancer treatment.

1.2 Sequencing technologies: three generations

Over the past quarter-century, advances and discoveries in the field of genomics have been eased by substantial reductions in the cost of genome sequencing. Methods and technologies to sequence genomes have evolved over the years and here is a brief comparison between the three generations of sequencing technologies. See Table 4.1.

In early sequencing studies, the Sanger technology was the most used. This approach, though informative, is time-consuming, expensive, and provides a limited amount of sequencing [3]. Today, Sanger sequencing retains an essential place in genomics for two main purposes [4]. First, it serves as an orthogonal method for confirming results obtained using the second and third-generation sequencing technologies, that is, Sanger approaches provide ground truth against which the other assays can be benchmarked. Second, Sanger sequencing provides a means to patch the coverage of regions that are poorly covered by the second-generation sequencing technology (e.g., regions with high GC-content). One way to restoring coverage of these areas using second-generation technologies would be to increase the quantity of input DNA, but it would be expensive and the quantity available may be limited.

In contrast to Sanger sequencing, second-generation sequencing technologies, e.g., Illumina or 454/Roche, provide much higher sequencing depth and high throughput. This technology has been widely used for two important genomic projects: the 1000 Genomes Project and the Encyclopedia of DNA Elements (ENCODE). The 1000 Genomes Project, that ran between 2008 and 2015, was an international research effort to establish a detailed catalogue of human genetic variation. The final dataset contains data for 2,504 individuals from 26 populations providing an overview of all human genetic variation. The ENCODE is a research project launched by the US National Human Genome Research Institute (NHGRI) in September 2003 aiming at identifying all functional elements in the human genome. These two projects as well as the field of genomics in general benefited from the high throughput of the second-generation sequencing technologies.

However, the higher throughput comes at the expense of read length. By virtue of

Generation	First	Second	Third
Companies	Sanger	Illumina, 454/Roche	Pacific Biosciences, Oxford Nanopore
Read length (bp)	10^3	10^2	10^4
High throughput	no	yes	yes
Sequencing error rate	low	low	high

Table 1.1: *Overview of the three generations of sequencing technologies.* First-generation sequencing technology allows sequencing of long reads. However, low throughput limits the use of the technology. Second-generation sequencing is currently the most used technology, as it has high throughput and low sequencing error rate. However, the short length of the sequencing reads is a drawback for some applications. Third-generation sequencing seems promising as the length of the sequenced reads is higher. However, third-generation technologies have high sequencing error rates.

sequencing single contiguous molecules, third-generation technologies are capable of producing long reads. Thus, the recent development of third-generation sequencing technologies offers a promising approach for applications where read length is important, e.g., when high resolution could be needed to analyze gut microbial communities (see Chapter 4). Unfortunately, with the current technology, gains in length can come at the expense of accuracy. In particular, the sequencing error rate for the current Pacific Biosciences RS sequencer is around 10%, whereas second-generation technologies like Illumina have a sequencing error rate around 0.02%.

1.3 Single-cell genomics

"Imagine being able to shrink down to a small enough size to peer into the human body at the single-cell level. Now take a deep breath and plunge into that cell to see all of the ongoing biological processes, including the full complement of molecules and their locations within the cell. This has long been the realm of science fiction, but not for much longer." [5] These are the first lines of the introduction to the special Issue on single-cell genomics published in Science on October, 7th 2017.

Single-cell genomics refers to the study of the genomes at the cellular level. Before single-cell genomics, genomes were sequenced from mRNA or DNA extracted from a large number of cells (so called "bulk sequencing"), from which it is only possible to get an averaged gene expression profile across a set of cells. With the new single-cell technology, it is now possible to dissect the contributions of individual cells to the biology of ecosystems and organisms

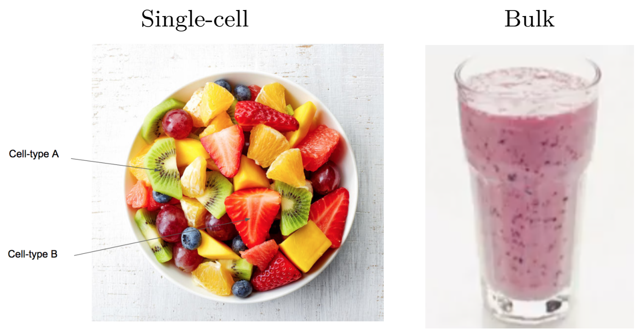


Figure 1.1: *Representation of the difference between bulk and single-cell sequencing.* Using single-cell sequencing it is possible to get the gene expression of each cell in a sample providing a way to identify different cell-types within one sample. On the contrary, using bulk sequencing, one gets a averaged gene expression profile across all the cells is a sample. The idea to illustrate the difference between bulk and single-cell sequencing using fruits is from a talk given by Shalek and Regev in 2016.

opening up myriads of applications and hopes to be able to study the heterogeneity of tissues. For example, it is now conceivable to determine the contributions of each individual cell to cancer development or response to a treatment. See Figure 1.1 a illustration of the difference between bulk and single-cell sequencing.

The Human Cell Atlas (HCA) [6] has been created in 2016 to create comprehensive reference maps of all human cells and intensively takes advantage of recent advances in single-cell genomics. The HCA is a global collaboration to characterize all cells in a healthy human body: cell types, numbers, locations, relationships, and molecular components. Once complete, it will be a fundamental resource for scientists, allowing them to better understand how healthy cells work, and what goes wrong when disease strikes. It will help researchers around the world understand how genetic variants impact disease risk, define drug toxicities, discover better therapies, and advance regenerative medicine [7]. The HCA project is in part supported by the Chan Zuckerberg Initiative, a company founded by Facebook founder Mark Zuckerberg and his wife Priscilla Chan with an investment of up to one billion in Facebook shares in each of the next three years.

1.4 Metagenomics

Single-cell genomics is also a very promising technique in the field of microbiology. The microbiome is defined as the genetic material of all the microbes - bacteria, fungi, protozoa, and viruses - that live on and inside the human body. It is estimated that about 100 trillion microbes live in and on us, that is ten times more microbial cells in our body than human cells, and the majority of these bacteria live in our guts. A recent paper [8] states that actually the number of bacteria in the body might be more of the same order as the number of human cells. But, the conclusion stays the same: we do have a lot of bacteria in and on us. These bacteria are increasingly recognized as having important roles in human health leading both academic labs and the industry to intensively study microbial samples.

Traditionally, microscopy and biochemical testing techniques have been the mainstay for identification of microorganisms. With the advances of DNA sequencing, identification of microbes has now been replaced by sequencing allowing researchers to characterize in a standardized way microbial populations. In particular, bulk genomics is used to produce a profile of diversity in a microbial sample where all the bacteria in a sample are sequenced at once. An ultimate step is then needed to deconvolve the mixture of bacteria in the sample.

All current methods, however, are limited to reconstructing only coarse approximations of the relative abundances of the bacteria in a sample mainly because of biases introduced during the preparation of the sample before sequencing. The main source of bias is the polymerase chain reaction (PCR), a technique used to amplify a single copy of a segment of DNA across several orders of magnitude, generating thousands to millions of copies of a particular DNA sequence. It has been studied for decades that PCR amplification efficiencies vary drastically from one bacterium to another introducing bias to estimate bacteria frequencies in microbial communities. Despite these limitations, the identification of bacteria using bulk genomics is a valuable method and the standard in the field. For example, the Human Microbiome Project (HMP), an initiative launched in 2008 to catalogue healthy human microbiome at multiple body sites, used bulk sequencing to provide a reference for future sequencing and metagenomic efforts.

As explained in the previous section, single-cell genomics has become available in the recent years. This new technology could offer solutions to the major limitations of bulk genomics for microbiology. For example, it could eliminate PCR bias as every single cell can be sequenced separately, suppressing the need to perform the deconvolution. Therefore, single-cell metagenomics could offer a powerful lens for viewing the microbial world. Single-cell sequencing in metagenomics is still immature though and would require more research, especially to isolate single cells from primary samples such as swabs and biopsies.

1.5 Dissertation overview

ZINB-WaVE, a model to analysis scRNA-seq data

In chapter 2, I present a general and flexible zero-inflated negative binomial model (ZINB-WaVE), which leads to low-dimensional representations of the data that account for the count nature of the data, over-dispersion, and zero inflation where zeros are of two types: biological zeros, when a gene is simply not expressed in the cell, and technical zeros (i.e., dropouts), when a gene is expressed in the cell but not detected. I show, with simulated and real data, that the model and its associated estimation procedure are able to give a more stable and accurate low-dimensional representation of the data than principal component analysis (PCA) and zero-inflated factor analysis (ZIFA), without the need for a preliminary normalization step.

Unlock bulk RNA-seq tools for zero inflation and single-cell
In Chapter 3, I introduce a weighting strategy using our ZINB-WaVE model, that identifies excess zero counts and generates gene and cell-specific weights to unlock bulk RNA-seq differential expression (DE) pipelines for zero-inflated data, boosting performance for scRNA-seq analysis. The method is shown to outperform competing methods on simulated and real single-cell RNA-seq datasets.

Estimation of bacterial abundances in microbial samples
In Chapter 4, a method to estimate bacterial abundances in microbial samples using full-length 16S sequencing reads is presented. The method models the sequencing error process using a generalized pair hidden Markov chain model and the Viterbi algorithm. I demonstrate, with simulated and real data, that the model and its associated estimation procedure are able to give accurate estimates at the species (or subspecies) level, and is more flexible than existing methods like SImple Non-Bayesian TAXonomy (SINTAX).

Chapter 2

ZINB-WaVE, a method for the analysis of scRNA-seq data

This chapter has been published in the journal Nature Communications. I specifically focused on simulations and analysis of simulated datasets while Davide Risso conducted the analysis of the real datasets. [1]

2.1 Introduction

Single-cell RNA sequencing (scRNA-seq) is a powerful and relatively young technique enabling the characterization of the molecular states of individual cells through their transcriptional profiles [9]. It represents a major advance with respect to standard "bulk" RNA sequencing, which is only capable of measuring average gene expression levels within a cell population. Such averaged gene expression profiles may be enough to characterize the global state of a tissue, but completely mask signal coming from individual cells, ignoring tissue heterogeneity. Assessing cell-to-cell variability in expression is crucial for disentangling complex heterogeneous tissues [10, 11, 12] and for understanding dynamic biological processes, such as embryo development [13] and cancer [14]. Despite the early successes of scRNA-seq, in order to fully exploit the potential of this new technology, it is essential to develop statistical and computational methods specifically designed for the unique challenges of this type of data [15].

Because of the tiny amount of RNA present in a single cell, the input material needs to go through many rounds of amplification before being sequenced. This results in strong

[1]I would like to thank my co-authors, Davide Risso, Svetlana Gribkova, Sandrine Dudoit, and Jean-Philippe Vert. I also would like to thank Olivier Mirabeau from the Curie Institute for the pre-processing of the Glioblastoma dataset, and Aaron Lun and the Marioni Lab for making the code needed for the simulations available online. Finally, I am grateful to Russell Fletcher for his help with the re-analysis of the OE data.

amplification bias, as well as *dropouts*, i.e., genes that fail to be detected even though they are expressed in the sample [16]. Researchers have recently reduced amplification bias by using random molecular barcodes or unique molecular identifiers (UMIs). The concept of UMIs is that each original DNA fragment, within the same cell, is attached to a unique DNA sequence which is designed as a string of totally random nucleotides. As a result, sequence reads that have different UMIs represent different original DNA fragments, while reads that have the same UMIs are the result of the same amplification from the same original DNA fragment. The inclusion in the library preparation of unique molecular identifiers reduces amplification bias [17], but does not remove dropout events, nor the need for data normalization [18, 19] - a critical step in the analysis pipeline that adjusts for unwanted biological and technical effects that can mask the signal of interest. In addition to the host of unwanted technical effects that affect bulk RNA-seq, scRNA-seq data exhibit much higher variability between technical replicates, even for genes with medium or high levels of expression [20].

The large majority of published scRNA-seq analyses include a dimensionality reduction step. This achieves a two-fold objective: (i) the data become more tractable, both from a statistical (cf. curse of dimensionality) and computational point of view; (ii) noise can be reduced while preserving the often intrinsically low-dimensional signal of interest. Dimensionality reduction is used in the literature as a preliminary step prior to clustering [21, 11, 22], the inference of developmental trajectories [23, 24, 25, 26], spatio-temporal ordering of the cells [13, 27], and, of course, as a visualization tool [28, 29]. Hence, the choice of dimensionality reduction technique is a critical step in the data analysis process.

A natural choice for dimensionality reduction is principal component analysis (PCA), which projects the observations onto the space defined by linear combinations of the original variables with successively maximal variance. However, several authors have reported on shortcomings of PCA for scRNA-seq data. In particular, for real datasets, the first or second principal components often depend more on the proportion of detected genes per cell (i.e., genes with at least one read) than on an actual biological signal [30, 31]. In addition to PCA, dimensionality reduction techniques used in the analysis of scRNA-seq data include independent components analysis (ICA) [23], Laplacian eigenmaps [32, 26], and t-distributed stochastic neighbor embedding (t-SNE) [33, 12, 10]. Note that none of these techniques can account for dropouts, nor for the count nature of the data. Typically, researchers transform the data using the logarithm of the (possibly normalized) read counts, adding an offset to avoid taking the log of zero.

Recently, Pierson and Yau [34] proposed a zero-inflated factor analysis (ZIFA) model to account for the presence of dropouts in the dimensionality reduction step. Although the method accounts for the zero inflation typically observed in scRNA-seq data, the proposed model does not take into account the count nature of the data. In addition, the model makes a strong assumption regarding the dependence of the probability of detection on the mean expression level, modeling it as an exponential decay. The fit on real datasets is not always good and, overall, the model lacks flexibility, with its inability to include covariates and/or normalization factors.

Here, we propose a general and flexible method that uses a zero-inflated negative binomial (ZINB) model to extract low-dimensional signal from the data, accounting for zero inflation (dropouts), over-dispersion, and the count nature of the data. We call this approach Zero-Inflated Negative Binomial-based Wanted Variation Extraction (ZINB-WaVE). The proposed model includes a sample-level intercept, which serves as a global-scaling normalization factor, and gives the user the ability to include both gene-level and sample-level covariates. The inclusion of observed and unobserved sample-level covariates enables normalization for complex, non-linear effects (often referred to as batch effects), while gene-level covariates may be used to adjust for sequence composition effects, such as gene length and GC-content effects. ZINB-WaVE is an extension of the RUV model [35, 36], which accounts for zero inflation and over-dispersion and for which unobserved sample-level covariates may either capture variation of interest or unwanted variation. We demonstrate, with simulated and real data, that the model and its associated estimation procedure are able to give a more stable and accurate low-dimensional representation of the data than PCA and ZIFA, without the need for a preliminary normalization step. The approach is implemented in the open-source R package `zinbwave`, publicly available through the Bioconductor Project (`https://bioconductor.org/packages/zinbwave`).

2.2 Methods

2.2.1 Model

For any $\mu \geq 0$ and $\theta > 0$, let $f_{NB}(\cdot\,; \mu, \theta)$ denote the probability mass function (PMF) of the negative binomial (NB) distribution with mean μ and inverse dispersion parameter θ, namely:

$$f_{NB}(y; \mu, \theta) = \frac{\Gamma(y+\theta)}{\Gamma(y+1)\Gamma(\theta)} \left(\frac{\theta}{\theta+\mu}\right)^{\theta} \left(\frac{\mu}{\mu+\theta}\right)^{y}, \quad \forall y \in \mathbb{N}. \tag{2.1}$$

Note that another parametrization of the NB PMF is in terms of the dispersion parameter $\phi = \theta^{-1}$ (although θ is also sometimes called dispersion parameter in the literature). In both cases, the mean of the NB distribution is μ and its variance is:

$$\sigma^2 = \mu + \frac{\mu^2}{\theta} = \mu + \phi\mu^2\,. \tag{2.2}$$

In particular, the NB distribution boils down to a Poisson distribution when $\phi = 0 \Leftrightarrow \theta = +\infty$.

For any $\pi \in [0, 1]$, let $f_{ZINB}(\cdot\,; \mu, \theta, \pi)$ be the PMF of the zero-inflated negative binomial (ZINB) distribution given by:

$$f_{ZINB}(y; \mu, \theta, \pi) = \pi\delta_0(y) + (1-\pi) f_{NB}(y; \mu, \theta), \quad \forall y \in \mathbb{N}, \tag{2.3}$$

where $\delta_0(\cdot)$ is the Dirac function. Here, π can be interpreted as the probability that a 0 is observed instead of the actual count, resulting in an inflation of zeros compared to the NB distribution, hence the name ZINB.

Given n samples (typically, n single cells) and J features (typically, J genes) that can be counted for each sample, let Y_{ij} denote the count of feature j (for $j = 1, \ldots, J$) for sample i ($i = 1, \ldots, n$). To account for various technical and biological effects frequent, in particular, in single-cell sequencing technologies, we model Y_{ij} as a random variable following a ZINB distribution with parameters μ_{ij}, θ_{ij}, and π_{ij}, and consider the following regression models for the parameters:

$$\ln(\mu_{ij}) = \left(X\beta_\mu + (V\gamma_\mu)^\top + W\alpha_\mu + O_\mu\right)_{ij} , \tag{2.4}$$

$$\text{logit}(\pi_{ij}) = \left(X\beta_\pi + (V\gamma_\pi)^\top + W\alpha_\pi + O_\pi\right)_{ij} , \tag{2.5}$$

$$\ln(\theta_{ij}) = \zeta_j , \tag{2.6}$$

where

$$\text{logit}(\pi) = \ln\left(\frac{\pi}{1-\pi}\right)$$

and elements of the regression models are as follows.

- X is a known $n \times M$ matrix corresponding to M cell-level covariates and $\beta = (\beta_\mu, \beta_\pi)$ its associated $M \times J$ matrices of regression parameters. X can typically include covariates that induce variation of interest, such as cell types, or covariates that induce unwanted variation, such as batch or quality control measures. It can also include a constant column of ones, $\mathbf{1}_n$, to account for gene-specific intercepts.

- V is a known $J \times L$ matrix corresponding to J gene-level covariates, such as gene length or GC-content, and $\gamma = (\gamma_\mu, \gamma_\pi)$ its associated $L \times n$ matrices of regression parameters. V can also include a constant column of ones, $\mathbf{1}_J$, to account for cell-specific intercepts, such as size factors representing differences in library sizes.

- W is an unobserved $n \times K$ matrix corresponding to K unknown cell-level covariates, which could be of "unwanted variation" as in RUV [35, 36] or of interest (such as cell type), and $\alpha = (\alpha_\mu, \alpha_\pi)$ its associated $K \times J$ matrices of regression parameters.

- O_μ and O_π are known $n \times J$ matrices of offsets.

- $\zeta \in \mathbb{R}^J$ is a vector of gene-specific dispersion parameters on the log scale.

This model deserves a few comments.

- By default, X and V contain a constant column of ones, to account, respectively, for gene-specific (e.g., baseline expression level) and cell-specific (e.g., library size) variation. In that case, X and V are of the form $X = [\mathbf{1}_n, X^0]$ and $V = [\mathbf{1}_J, V^0]$ and we can similarly decompose the corresponding parameters as $\beta = [\beta^1, \beta^0]$ and $\gamma = [\gamma^1, \gamma^0]$, where $\beta^1 \in \mathbb{R}^{1\times J}$ is a vector of gene-specific intercepts and $\gamma^1 \in \mathbb{R}^{1\times n}$ a vector of cell-specific intercepts. The representation $\mathbf{1}_n\beta^1 + (\mathbf{1}_J\gamma^1)^\top$ is then not unique, but could be made unique by adding a constant and constraining β^1 and γ^1 to each have elements summing to zero.

- Although W is the same, the matrices X and V could differ in the modeling of μ and π, if we assume that some known factors do not affect both μ and π. To keep notation simple and consistent, we use the same matrices, but will implicitly assume that some parameters may be constrained to be 0 if needed.

- By allowing the models to differ for μ and π, we can model and test for differential expression in terms of either the NB mean or the ZI probability.

- We limit ourselves to a gene-dependent dispersion parameter. More complicated models for θ_{ij} could be investigated, such as a model similar to μ_{ij} or a functional of the form $\theta_{ij} = f(\mu_{ij})$, but we restrict ourselves to a simpler model that has been shown to be largely sufficient in bulk RNA-seq analysis.

2.2.2 Estimation procedure

The input to the model are the matrices X, V, O_μ, and O_π and the integer K; the parameters to be inferred are $\beta = (\beta_\mu, \beta_\pi)$, $\gamma = (\gamma_\mu, \gamma_\pi)$, W, $\alpha = (\alpha_\mu, \alpha_\pi)$, and ζ. Given an $n \times J$ matrix of counts Y, the log-likelihood function is

$$\ell(\beta, \gamma, W, \alpha, \zeta) = \sum_{i=1}^{n}\sum_{j=1}^{J} \ln f_{ZINB}(Y_{ij}; \mu_{ij}, \theta_{ij}, \pi_{ij}) \,, \tag{2.7}$$

where μ_{ij}, θ_{ij}, and π_{ij} depend on $(\beta, \gamma, W, \alpha, \zeta)$ through Equations (2.4)–(2.6).

To infer the parameters, we follow a penalized maximum likelihood approach, by trying to solve

$$\max_{\beta,\gamma,W,\alpha,\zeta} \left\{ \ell(\beta, \gamma, W, \alpha, \zeta) - \mathrm{Pen}(\beta, \gamma, W, \alpha, \zeta) \right\} \,,$$

where $\mathrm{Pen}(\cdot)$ is a regularization term to reduce overfitting and improve the numerical stability of the optimization problem in the setting of many parameters. For nonnegative regularization parameters $(\epsilon_\beta, \epsilon_\gamma, \epsilon_W, \epsilon_\alpha, \epsilon_\zeta)$, we set

$$\mathrm{Pen}(\beta, \gamma, W, \alpha, \zeta) = \frac{\epsilon_\beta}{2}\|\beta^0\|^2 + \frac{\epsilon_\gamma}{2}\|\gamma^0\|^2 + \frac{\epsilon_W}{2}\|W\|^2 + \frac{\epsilon_\alpha}{2}\|\alpha\|^2 + \frac{\epsilon_\zeta}{2}\mathrm{Var}(\zeta) \,,$$

where β^0 and γ^0 denote the matrices β and γ without the rows corresponding to the intercepts if an unpenalized intercept is included in the model, $\| \cdot \|$ is the Frobenius matrix norm ($\|A\| = \sqrt{\mathrm{tr}(A^*A)}$, where A^* denotes the conjugate transpose of A), and $\mathrm{Var}(\zeta) = 1/(J-1)\sum_{i=1}^J \left(\zeta_i - (\sum_{j=1}^J \zeta_j)/J\right)^2$ is the variance of the elements of ζ (using the unbiased sample variance statistic). The penalty tends to shrink the estimated parameters to 0, except for the cell and gene-specific intercepts which are not penalized and the dispersion parameters which are not shrunk towards 0 but instead towards a constant value across genes. Note also that the likelihood only depends on W and α through their product $R = W\alpha$ and that the penalty ensures that at the optimum W and α have the structure described in the following result which generalizes standard results such as Srebro, Rennie, and Jaakkola [37, Lemma1] and Mazumder, Hastie, and Tibshirani [38, Lemma 6].

Lemma 1. *For any matrix R and positive scalars s and t, the following holds:*

$$\min_{S,T\,:\,R=ST} \frac{1}{2}\left(s\|S\|^2 + t\|T\|^2\right) = \sqrt{st}\|R\|_* \,,$$

where $\|A\|_* = tr\left(\sqrt{A^*A}\right)$. If $R = R_L R_\Sigma R_R$ is a singular value decomposition (SVD) of R, then a solution to this optimization problem is:

$$S = \left(\frac{t}{s}\right)^{\frac{1}{4}} R_L R_\Sigma^{\frac{1}{2}}\,, \quad T = \left(\frac{s}{t}\right)^{\frac{1}{4}} R_\Sigma^{\frac{1}{2}} R_R\,.$$

Proof. Let $\tilde{S} = \sqrt{s}S$, $\tilde{T} = \sqrt{t}T$, and $\tilde{R} = \sqrt{st}R$. Then, $\|\tilde{S}\|^2 = s\|S\|^2$, $\|\tilde{T}\|^2 = t\|T\|^2$, and $\tilde{S}\tilde{T} = \sqrt{st}ST$, so that the optimization problem is equivalent to:

$$\min_{\tilde{S},\tilde{T}\,:\,\tilde{S}\tilde{T}=\tilde{R}} \frac{1}{2}\left(\|\tilde{S}\|^2 + \|\tilde{T}\|^2\right)\,,$$

which by Mazumder, Hastie, and Tibshirani [38, Lemma 6] has optimum value $\|\tilde{R}\|_* = \sqrt{st}\|R\|_*$ reached at $\tilde{S} = \tilde{R}_L \tilde{R}_\Sigma^{\frac{1}{2}}$ and $\tilde{T} = \tilde{R}_\Sigma^{\frac{1}{2}} \tilde{R}_R$, where $\tilde{R}_L \tilde{R}_\Sigma \tilde{R}_R$ is a SVD of $\tilde{R}$. Observing that $\tilde{R}_L = R_L$, $\tilde{R}_R = R_R$, and $\tilde{R}_\Sigma = \sqrt{st}R_\Sigma$, gives that a solution of the optimization problem is $S = s^{-1/2}\tilde{S} = s^{-1/2}R_L(st)^{1/4}R_\Sigma^{1/2} = (t/s)^{1/4}R_L R_\Sigma^{1/2}$. A similar argument for T concludes the proof. □

This lemma implies in particular that at any local maximum of the penalized log-likelihood, W and $\alpha^\top$ have orthogonal columns, which is useful for visualization or interpretation of latent factors.

To balance the penalties applied to the different matrices in spite of their different sizes, a natural choice is to fix $\epsilon > 0$ and set

$$\epsilon_\beta = \frac{\epsilon}{J}\,, \quad \epsilon_\gamma = \frac{\epsilon}{n}\,, \quad \epsilon_W = \frac{\epsilon}{n}\,, \quad \epsilon_\alpha = \frac{\epsilon}{J}\,, \quad \epsilon_\zeta = \epsilon\,.$$

In particular, from Lemma 1, we easily deduce the following characterization of the penalty on W and α, which shows that the entries in the matrices W and α have similar standard deviation after optimization:

Corollary 1. *For any $n \times J$ matrix R and positive scalars ϵ, the following holds.*

$$\min_{W,\alpha\,:\,R=W\alpha} \frac{\epsilon}{2}\left(\frac{1}{n}\|W\|^2 + \frac{1}{J}\|\alpha\|^2\right) = \frac{\epsilon}{\sqrt{nJ}}\|R\|_* \,.$$

If $R = R_L R_\Sigma R_R$ is a SVD decomposition of R, then a solution to this optimization problem is:

$$W = \left(\frac{n}{J}\right)^{\frac{1}{4}} R_L R_\Sigma^{\frac{1}{2}}\,, \quad T = \left(\frac{J}{n}\right)^{\frac{1}{4}} R_\Sigma^{\frac{1}{2}} R_R \,.$$

In particular, for any $i = 1, \ldots, \min(n, J)$,

$$\frac{1}{n}\sum_{j=1}^{n} W_{j,i}^2 = \frac{1}{J}\sum_{j=1}^{J} \alpha_{i,j}^2 = \frac{[R_\Sigma]_{i,i}}{\sqrt{nJ}}.$$

The penalized likelihood is however not concave, making its maximization computationally challenging. We instead find a local maximum, starting from a smart initialization and iterating a numerical optimization scheme until local convergence, as described below.

Initialization

To initialize the set of parameters we approximate the count distribution by a log-normal distribution and explicitly separate zero and non-zero values, as follows:

1. Set $\mathcal{P} = \{(i,j) \,:\, Y_{ij} > 0\}$.
2. Set $L_{ij} = \ln(Y_{ij}) - (O_\mu)_{ij}$ for all $(i,j) \in \mathcal{P}$.
3. Set $\hat{Z}_{ij} = 0$ if $(i,j) \in \mathcal{P}$, $\hat{Z}_{ij} = 1$ otherwise.
4. Estimate β_μ and γ_μ by solving the convex ridge regression problem:

$$\min_{\beta_\mu,\gamma_\mu} \sum_{(i,j)\in\mathcal{P}} \left(L_{ij} - (X\beta_\mu)_{ij} - (V\gamma_\mu)_{ji}\right)^2 + \frac{\epsilon_\beta}{2}\|\beta_\mu^0\|^2 + \frac{\epsilon_\gamma}{2}\|\gamma_\mu^0\|^2 \,.$$

This is a standard ridge regression problem, but with a potentially huge design matrix, with up to nJ rows and $MJ + nL$ columns. To solve it efficiently, we alternate the estimation of β_μ and γ_μ. Specifically, we initialize parameter values as:

$$\hat{\beta}_\mu \leftarrow 0\,, \quad \hat{\gamma}_\mu \leftarrow 0$$

and repeat the following two steps a few times (or until convergence):

a) Optimization in γ_μ, which can be performed independently and in parallel for each cell:

$$\hat{\gamma}_\mu \in \arg\min_{\gamma_\mu} \sum_{(i,j)\in\mathcal{P}} \left(L_{ij} - (X\hat{\beta}_\mu)_{ij} - (V\gamma_\mu)_{ji}\right)^2 + \frac{\epsilon_\gamma}{2}\|\gamma_\mu^0\|^2 .$$

b) Optimization in β_μ, which can be performed independently and in parallel for each gene:

$$\hat{\beta}_\mu \in \arg\min_{\beta_\mu} \sum_{(i,j)\in\mathcal{P}} \left(L_{ij} - (V\hat{\gamma}_\mu)_{ji} - (X\beta_\mu)_{ij}\right)^2 + \frac{\epsilon_\beta}{2}\|\beta_\mu^0\|^2 .$$

5. Estimate W and α_μ by solving

$$\left(\hat{W}, \hat{\alpha}_\mu\right) \in \arg\min_{W,\alpha_\mu} \sum_{(i,j)\in\mathcal{P}} \left(L_{ij} - (X\hat{\beta}_\mu)_{ij} - (V\hat{\gamma}_\mu)_{ji} - (W\alpha_\mu)_{ij}\right)^2 + \frac{\epsilon_W}{2}\|W\|^2 + \frac{\epsilon_\alpha}{2}\|\alpha_\mu\|^2 .$$

Denoting by $D = L - X\hat{\beta} - (V\hat{\gamma})^\top$, this problem can be rewritten as:

$$\min_{W,\alpha} \|D - W\alpha\|_{\mathcal{P}}^2 + \frac{1}{2}\left(\epsilon_W\|W\|^2 + \epsilon_\alpha\|\alpha\|^2\right) ,$$

where $\|A\|_{\mathcal{P}}^2 = \sum_{(i,j)\in\mathcal{P}} A_{ij}^2$. By Lemma 1, if K is large enough, one can first solve the convex optimization problem:

$$\hat{R} \in \arg \min_{R\,:\,\mathrm{rank}(R)\le K} \|D - R\|_{\mathcal{P}}^2 + \sqrt{\epsilon_W\epsilon_\alpha}\|R\|_* \qquad (2.8)$$

and set

$$W = \left(\frac{\epsilon_\alpha}{\epsilon_W}\right)^{\frac{1}{4}} R_L R_\Sigma^{\frac{1}{2}} , \quad \alpha = \left(\frac{\epsilon_W}{\epsilon_\alpha}\right)^{\frac{1}{4}} R_\Sigma^{\frac{1}{2}} R_R ,$$

where $\hat{R} = R_L R_\Sigma R_R$ is the SVD of $\hat{R}$. This solution is exact when K is at least equal to the rank of the solution of the unconstrained problem (2.8), which we solve with the `softImpute::softImpute()` function [38]. If K is smaller, then (2.8) becomes a non-convex optimization problem whose global optimum may be challenging to find. In that case we also use the rank-constrained version of `softImpute::softImpute()` to obtain a good local optimum.

6. Estimate β_π, γ_π, and α_π by solving the regularized logistic regression problem:

$$\min_{(\beta_\pi,\gamma_\pi,\alpha_\pi)} \sum_{(i,j)} \Big[-\hat{Z}_{ij}(X\beta_\pi + (V\gamma_\pi)^\top + \hat{W}\alpha_\pi)_{ij} + \ln\left(1 + e^{(X\beta_\pi + (V\gamma_\pi)^\top + \hat{W}\alpha_\pi)_{ij}}\right)\Big] + \frac{\epsilon_\beta}{2}\|\beta_\pi\|^2 + \frac{\epsilon_\gamma}{2}\|\gamma_\pi\|^2 + \frac{\epsilon_\alpha}{2}\|\alpha_\pi\|^2 . \qquad (2.9)$$

This is a standard ridge logistic regression problem, but with a potentially huge design matrix, with up to nJ rows and $MJ + nL$ columns. To solve it efficiently, we alternate the estimation of β_π, γ_π, and α_π. Specifically, we initialize parameter values as:

$$\hat{\beta}_\pi \leftarrow 0\,, \quad \hat{\gamma}_\pi \leftarrow 0\,, \quad \hat{\alpha}_\pi \leftarrow 0$$

and repeat the following two steps a few times (or until convergence):

a) Optimization in γ_π:

$$\hat{\gamma}_\pi \in \arg\min_{\gamma_\pi} \sum_{(i,j)} \Big[-\hat{Z}_{ij}(X\hat{\beta}_\pi + (V\gamma_\pi)^\top + \hat{W}\hat{\alpha}_\pi)_{ij} + \ln\left(1 + e^{(X\hat{\beta}_\pi + (V\gamma_\pi)^\top + \hat{W}\hat{\alpha}_\pi)_{ij}}\right) \Big] + \frac{\epsilon_\gamma}{2}\|\gamma_\pi\|^2\,. \tag{2.10}$$

Note that this problem can be solved for each cell (i) independently and in parallel. When there is no gene covariate besides the constant intercept, the problem is easily solved by setting $(\hat{\gamma}_\pi)_i$ to the logit of the proportion of zeros in each cell.

b) Optimization in β_π and α_π:

$$\left(\hat{\beta}_\pi, \hat{\alpha}_\pi\right) \in \arg\min_{(\beta_\pi,\alpha_\pi)} \sum_{(i,j)} \Big[-\hat{Z}_{ij}(X\beta_\pi + (V\hat{\gamma}_\pi)^\top + \hat{W}\alpha_\pi)_{ij} + \ln\left(1 + e^{(X\beta_\pi + (V\hat{\gamma}_\pi)^\top + \hat{W}\alpha_\pi)_{ij}}\right) \Big] + \frac{\epsilon_\beta}{2}\|\beta_\pi\|^2 + \frac{\epsilon_\alpha}{2}\|\alpha_\pi\|^2\,. \tag{2.11}$$

7. Initialize $\hat{\zeta} = 0$.

Optimization

After initialization, we maximize locally the penalized log-likelihood by alternating optimization over the dispersion parameters and left and right-factors, iterating the following steps until convergence:

1. Dispersion optimization:

$$\hat{\zeta} \leftarrow \arg\max_{\zeta} \left\{ \ell(\hat{\beta}, \hat{\gamma}, \hat{W}, \hat{\alpha}, \zeta) - \frac{\epsilon_\zeta}{2}\mathrm{Var}(\zeta) \right\}\,.$$

To solve this problem, we start by estimating a common dispersion parameter for all the genes, by maximizing the objective function under the constraint that $\mathrm{Var}(\zeta) = 0$; in practice, we use a derivative-free one-dimensional optimization vector over the range $[-50, 50]$. We then optimize the objective function by a quasi-Newton optimization scheme starting from the constant solution found by the first step. To derive the gradient of the objective function used by the optimization procedure, note that the derivative of the NB log-density is:

$$\frac{\partial}{\partial\theta}\ln f_{NB}(y;\mu,\theta) = \Psi(y+\theta) - \Psi(\theta) + \ln\theta + 1 - \ln(\mu+\theta) - \frac{y+\theta}{\mu+\theta},$$

where $\Psi(z) = \Gamma'(z)/\Gamma(z)$ is the digamma function. We therefore get the derivative of the ZINB density as follows, for any $\pi \in [0,1]$:

- If $y > 0$, $f_{ZINB}(y;\mu,\theta,\pi) = (1-\pi)f_{NB}(y;\mu,\theta)$ therefore

$$\frac{\partial}{\partial\theta}\ln f_{ZINB}(y;\mu,\theta) = \Psi(y+\theta) - \Psi(\theta) + \ln\theta + 1 - \ln(\mu+\theta) - \frac{y+\theta}{\mu+\theta}.$$

- For $y = 0$, $\frac{\partial}{\partial\theta}\ln f_{NB}(0;\mu,\theta) = \ln\theta + 1 - \ln(\mu+\theta) - \frac{\theta}{\mu+\theta}$, therefore

$$\frac{\partial}{\partial\theta}\ln f_{ZINB}(y;\mu,\theta) = \frac{\ln\theta + 1 - \ln(\mu+\theta) - \frac{\theta}{\mu+\theta}}{1 + \frac{\pi(\mu+\theta)^\theta}{(1-\pi)\theta^\theta}}.$$

The derivative of the objective function w.r.t. ζ_j, for $j = 1,\dots,J$, is then easily obtained by

$$\sum_{i=1}^{n}\theta_j\frac{\partial}{\partial\theta}\ln f_{ZINB}(y_{ij};\mu_{ij},\theta_j) - \frac{\epsilon_\zeta}{J-1}\left(\zeta_j - \frac{1}{J}\sum_{k=1}^{J}\zeta_k\right).$$

(Note that the $J-1$ term in the denominator comes from the use of the unbiased sample variance statistic in the penalty for ζ.)

2. Left-factor (cell-specific) optimization:

$$\left(\hat{\gamma},\hat{W}\right) \leftarrow \arg\max_{(\gamma,W)}\left\{\ell(\hat{\beta},\gamma,W,\hat{\alpha},\hat{\zeta}) - \frac{\epsilon_\gamma}{2}\|\gamma^0\|^2 - \frac{\epsilon_W}{2}\|W\|^2\right\}. \tag{2.12}$$

Note that this optimization can be performed independently and in parallel for each cell $i = 1,\dots,n$. For this purpose, we consider a subroutine `solveZinbRegression` to find a set of vectors (a_μ, a_π, b) that locally maximize the log-likelihood of a ZINB model for a vector of counts y parametrized as follows:

$$\begin{aligned}\ln(\mu) &= A_\mu a_\mu + B_\mu b + C_\mu\,,\\ \text{logit}(\pi) &= A_\pi a_\pi + B_\pi b + C_\pi\,,\\ \ln(\theta) &= C_\theta\,.\end{aligned}$$

We give more details on how to solve `solveZinbRegression` in the next section. To solve (2.12) for cell i we call `solveZinbRegression` with the following parameters:

$$\begin{cases} a_\mu & = \gamma_\mu[.,i] \\ a_\pi & = \gamma_\pi[.,i] \\ b & = W[i,.]^\top \\ y & = Y[i,]^\top \\ A_\mu & = V_\mu \\ B_\mu & = \alpha_\mu^\top \\ C_\mu & = (X_\mu[i,.]\beta_\mu + O_\mu[i,.])^\top \\ A_\pi & = V_\pi \\ B_\pi & = \alpha_\pi^\top \\ C_\pi & = (X_\pi[i,.]\beta_\pi + O_\pi[i,.])^\top \\ C_\theta & = \zeta \end{cases} .$$

3. Right-factor (gene-specific) optimization:

$$\left(\hat{\beta}, \hat{\alpha}\right) \leftarrow \arg\max_{(\beta,\alpha)} \left\{ \ell(\beta, \hat{\gamma}, \hat{W}, \alpha, \hat{\zeta}) - \frac{\epsilon_\beta}{2}\|\beta^0\|^2 - \frac{\epsilon_\alpha}{2}\|\alpha\|^2 \right\} .$$

Note that this optimization can be performed independently and in parallel for each gene $j = 1, \ldots, J$, by calling `solveZinbRegression` with the following parameters:

$$\begin{cases} a_\mu & = (\beta_\mu[.,j]; \alpha_\mu[.,j]) \\ a_\pi & = (\beta_\pi[.,j]; \alpha_\pi[.,j]) \\ b & = \emptyset \\ y & = Y[.,j] \\ A_\mu & = [X_\mu, W] \\ B_\mu & = \emptyset \\ C_\mu & = (V_\mu[j,.]\gamma_\mu)^\top + O_\mu[.,j] \\ A_\pi & = [X_\pi, W] \\ B_\pi & = \emptyset \\ C_\pi & = (V_\pi[j,.]\gamma_\pi)^\top + O_\pi[.,j] \\ C_\theta & = \zeta_j \mathbf{1}_n \end{cases} .$$

4. Orthogonalization:

$$\left(\hat{W}, \hat{\alpha}\right) \leftarrow \arg \min_{(W,\alpha)\,:\, W\alpha = \hat{W}\hat{\alpha}} \frac{1}{2}\left(\epsilon_W \|W\|^2 + \epsilon_\alpha \|\alpha\|^2\right) .$$

This is obtained by applying Lemma 1, starting from an SVD decomposition of the current $\hat{W}\hat{\alpha}$. Note that this step not only allows to maximize locally the penalized log-likelihood, but also ensures that the columns of W stay orthogonal to each other during optimization.

Solving `solveZinbRegression`

Given a N-dimensional matrix of counts $y \in \mathbb{N}^N$, and matrix $A_\mu \in \mathbb{R}^{N\times p}$, $B_\mu \in \mathbb{R}^{N\times r}$, $C_\mu \in \mathbb{R}^N$, $A_\pi \in \mathbb{R}^{N\times q}$, $B_\pi \in \mathbb{R}^{N\times r}$, $C_\pi \in \mathbb{R}^N$ and $C_\theta \in \mathbb{R}^N$, for some integers p, q, r, the function `solveZinbRegression`$(y, A_\mu, B_\mu, C_\mu, A_\pi, B_\pi, C_\pi, C_\theta)$ attempts to find parameters $(a_\mu, a_\pi, b) \in \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^r$ that maximize the ZINB log-likelihood of y with parameters:

$$\begin{aligned}
\ln(\mu) &= A_\mu a_\mu + B_\mu b + C_\mu \,, \\
\text{logit}(\pi) &= A_\pi a_\pi + B_\pi b + C_\pi \,, \\
\ln(\theta) &= C_\theta \,.
\end{aligned}$$

Starting from an initial guess (as explained in the different steps above), we perform a local minimization of this function $F(a_\mu, a_\pi, b)$ using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton method. Let us now give more details on how the gradient of F is computed.

Given a single count y (i.e., $N = 1$), we first explicit the derivatives of the log-likelihood of y with respect to the (μ, π) parameters of the ZINB distribution. We first observe that

$$\frac{\partial}{\partial \mu} \ln f_{NB}(y; \mu, \theta) = \frac{y}{\mu} - \frac{y+\theta}{\mu+\theta} \,,$$

and that by definition of the ZINB distribution the following holds:

$$\begin{aligned}
\frac{\partial}{\partial \mu} \ln f_{ZINB}(y; \mu, \theta, \pi) &= \frac{(1-\pi) f_{NB}(y; \mu, \theta) \frac{\partial}{\partial \mu} \ln f_{NB}(y; \mu, \theta)}{f_{ZINB}(y; \mu, \theta, \pi)} \,, \\
\frac{\partial}{\partial \pi} \ln f_{ZINB}(y; \mu, \theta, \pi) &= \frac{\delta_0(y) - f_{NB}(y; \mu, \theta)}{f_{ZINB}(y; \mu, \theta, \pi)} \,.
\end{aligned}$$

Let us explicit these expressions, depending on whether or not y is null:

- If $y > 0$, then $\delta_0(y) = 0$ and $f_{ZINB}(y; \mu, \theta, \pi) = (1-\pi) f_{NB}(y; \mu, \theta)$, so we obtain:

$$\begin{aligned}
\frac{\partial}{\partial \mu} \ln f_{ZINB}(y; \mu, \theta, \pi) &= \frac{y}{\mu} - \frac{y+\theta}{\mu+\theta} \,, \\
\frac{\partial}{\partial \pi} \ln f_{ZINB}(y; \mu, \theta, \pi) &= \frac{-1}{1-\pi} \,.
\end{aligned}$$

- If $y = 0$, then $\delta_0(y) = 0$, and we get

$$\begin{aligned}
\frac{\partial}{\partial \mu} \ln f_{ZINB}(0; \mu, \theta, \pi) &= \frac{-(1-\pi)\left(\frac{\theta}{\mu+\theta}\right)^{\theta+1}}{\pi + (1-\pi)\left(\frac{\theta}{\mu+\theta}\right)^{\theta}}, \\
\frac{\partial}{\partial \pi} \ln f_{ZINB}(0; \mu, \theta, \pi) &= \frac{1-\left(\frac{\theta}{\mu+\theta}\right)^{\theta}}{\pi + (1-\pi)\left(\frac{\theta}{\mu+\theta}\right)^{\theta}}.
\end{aligned}$$

When $N \geq 1$, using standard calculus for the differentiation of compositions and the facts that:

$$\begin{aligned}
\left(\ln^{-1}\right)'(\ln \mu) &= \mu, \\
\left(\text{logit}^{-1}\right)'(\text{logit}\pi) &= \pi(1-\pi),
\end{aligned}$$

we finally get that

$$\begin{aligned}
\nabla_{a_\mu} F &= A_\mu^\top G, \\
\nabla_{a_\pi} F &= A_\pi^\top H, \\
\nabla_b F &= B_\mu^\top G + B_\pi^\top H.
\end{aligned}$$

where G and H are the N-dimensional vectors given by

$$\begin{aligned}
\forall i \in [1, N], \quad G_i &= \mu_i \frac{\partial}{\partial \mu} \ln f_{ZINB}(y_i; \mu_i, \theta_i, \pi_i), \\
H_i &= \pi_i(1-\pi_i) \frac{\partial}{\partial \pi} \ln f_{ZINB}(y_i; \mu_i, \theta_i, \pi_i).
\end{aligned}$$

2.2.3 Software implementation

The ZINB-WaVE method is implemented in the open-source R package zinbwave, available as part of the Bioconductor Project (`https://bioconductor.org/packages/zinbwave`). See the package vignette for a detailed example of a typical use. The code to reproduce all the analyses and figures of this article is available at `https://github.com/drisso/zinb_analysis`.

2.2.4 Real datasets

V1 dataset. Tasic et al. [11] characterized more than 1,600 cells from the primary visual cortex (V1) in adult male mice, using a set of established Cre lines. Single cells were isolated by FACS into 96-well plates and RNA was reverse transcribed and amplified using the SMARTer kit. Sequencing was performed using the Illumina HiSeq platform, yielding 100 bp-long reads. We selected a subset of three Cre lines, Ntsr1-Cre, Rbp4-Cre, and Scnn1a-Tg3-Cre, that label Layer 4, Layer 5, and Layer 6 excitatory neurons, respectively. This subset consists of 379 cells, grouped by the authors into 17 clusters; we excluded the cells that did not pass the authors' quality control filters and that were classified by the authors as "intermediate" cells between two clusters, retaining a total of 285 cells. Gene expression was quantified by gene-level read counts. Raw gene-level read counts and QC metrics (see below) are available as part of the `scRNAseq` Bioconductor R package (`https://bioconductor.org/packages/scRNAseq`). We applied the dimensionality reduction methods to the 1,000 most variable genes.

S1/CA1 dataset. Zeisel et al. [12] characterized 3,005 cells from the primary somatosensory cortex (S1) and the hippocampal CA1 region, using the Fluidigm C1 microfluidics cell capture platform followed by Illumina sequencing. Gene expression was quantified by unique molecular identifier (UMI) counts. In addition to gene expression measures, we have access to metadata that can be used to assess the methods: batch, sex, number of mRNA molecules. Raw UMI counts and metadata were downloaded from `http://linnarssonlab.org/cortex/`.

mESC dataset. Kolodziejczyk et al. [39] sequenced the transcriptome of 704 mouse embryonic stem cells (mESCs), across three culture conditions (serum, 2i, and a2i), using the Fluidigm C1 microfluidics cell capture platform followed by Illumina sequencing. We selected only the cells from the second and third batch, after excluding the samples that did not pass the authors' QC filtering. This allowed us to have cells from each culture condition in each batch and resulted in a total of 169 serum cells, 141 2i cells, and 159 a2i cells. In addition to gene expression measures, we have access to batch and plate information that can be included as covariates in our model. Raw gene-level read counts were downloaded from `http://www.ebi.ac.uk/teichmann-srv/espresso/`. Batch and plate information was extracted from the sample names, as done in Lun and Marioni [40]. We applied the dimensionality reduction methods to the 1,000 most variable genes.

Glioblastoma dataset. Patel et al. [14] collected 672 cells from five dissociated human glioblastomas. Transcriptional profiles were generated using the SMART-Seq protocol. We analyzed only the cells that passed the authors' QC filtering. The raw data were downloaded from the NCBI GEO database (accession GSE57872). Reads were aligned using TopHat with the following parameters: –rg-library Illumina –rg-platform Illumina –keep-fasta-order -G -N 3 –read-edit-dist 3 –no-coverage-search -x 1 -M -p 12. Counts were obtained using `htseq-count` with the following parameters (`http://www-huber.embl.de/HTSeq/doc/count.html`): -a 10 -q -s no -m union. We applied the dimensionality reduction methods to the 1,000 most variable genes.

OE dataset. Fletcher et al. [41] characterized 849 FACS-purified cells from the mouse olfactory epithelium (OE), using the Fluidigm C1 microfluidics cell capture platform followed by Illumina sequencing. Gene-level read counts were downloaded from GEO (GSE95601; file `GSE95601_oeHBCdiff_Cufflinks_eSet_counts_table.txt.gz`). As done in Perraudeau et al. [42], we filtered the cells that exhibited poor sample quality using SCONE [43] (v. 1.1.2). A total of 747 cells passed this filtering procedure. To compare with the original results, we also re-analyze the final repertoire of 13 stable clusters found in Fletcher et al. [41], consisting of 616 cells, downloaded from `https://github.com/rufletch/p63-HBC-diff`. See Fletcher et al. [41] for details on the original analysis and Perraudeau et al. [42] for details on the ZINB-WaVE based workflow.

10x Genomics 68k PBMCs dataset. Zheng et al. [44] characterized 68,579 peripheral blood mononuclear cells (PBMCs) from a healthy donor. The gene-level UMI counts were downloaded from `https://www.10xgenomics.com/single-cell/` using the cellrangerRkit R package (Version 1.1.0). We applied the dimensionality reduction methods to the 1,000 most variable genes.

2.2.5 Simulated datasets

Simulating from the ZINB-WaVE model

Setting the simulation parameters. In order to simulate realistic data, we fitted our ZINB-WaVE model to two real datasets (V1 and S1/CA1) and used the resulting parameter estimates as the truth to be estimated in the simulation. Genes that did not have at least 5 reads in at least 5 cells were filtered out and $J = 1,000$ genes were then sampled at random for each dataset. The ZINB-WaVE model was fit to the count matrix Y with the number of unknown cell-level covariates set to $K = 2$, genewise dispersion ($\zeta = \ln\theta = -\ln\phi$), X_μ, X_π, V_μ, and V_π as columns of ones (i.e., intercept only), and no offset matrices, to get estimates for W, α_μ, α_π, β_μ, β_π, γ_μ, γ_π, and ζ. The parameters which were varied in the simulations are as follows.

- The number of cells: $n = 100, 1,000, 10,000$.
- The proportion of zero counts, $zfrac = \sum_{i,j} \mathbf{1}(Y_{ij} = 0)/nJ$, via the parameter γ_π: $zfrac \approx 0.25, 0.50, 0.75$. Since $\text{logit}(\pi) = X\beta_\pi + (V\gamma_\pi)^T + W\alpha_\pi$, the value of γ_π is directly linked to the dropout probability π, thus to the zero fraction. Note that by changing only γ_π but not γ_μ, we change the dropout rate but not the underlying, unobserved mean expression, i.e., this varies the number of *technical* zeros but not *biological* zeros.
- The ratio of within to between-cluster sums of squares for W. Specifically, let C denote the number of clusters and n_c the number of cells in cluster c. For a given column of W (out of K columns), let W_{ic} denote the value for cell i in cluster c, $\bar{W}$ the overall

average across all n cells, $\bar{W}_c$ the average for cells in cluster c, and TSS the total sum of squares. Then,

$$\begin{aligned} TSS &= \sum_{c=1}^{C}\sum_{i=1}^{n_c}(W_{ic}-\bar{W})^2 \\ &= \sum_{c=1}^{C}\sum_{i=1}^{n_c}(W_{ic}-\bar{W}_c)^2 + \sum_{c=1}^{C} n_c(\bar{W}_c-\bar{W})^2 \\ &= WSS + BSS, \end{aligned}$$

with $WSS = \sum_{c=1}^{C}\sum_{i=1}^{n_c}(W_{ic}-\bar{W}_c)^2$ and $BSS = \sum_{c=1}^{C} n_c(\bar{W}_c-\bar{W})^2$ the within and between-cluster sums of squares, respectively. The level of difficulty of the clustering problem can be controlled by the ratio of within to between-cluster sums of squares. However, we want to keep the overall mean $\bar{W}$ and overall variance (i.e., TSS) constant, so that the simulated values of W stay in the same range as the estimated W from the real dataset; this prevents us from simulating an unrealistic count matrix Y.

Let us scale the between-cluster sum of squares by a^2 and the within-cluster sum of squares by a^2b^2, i.e., replace $(\bar{W}_c-\bar{W})$ by $a(\bar{W}_c-\bar{W})$ and $(W_{ic}-\bar{W}_c)$ by $ab(W_{ic}-\bar{W}_c)$, with $a \geq 0$ and $b \geq 0$ such that TSS and $\bar{W}$ are constant. The total sum of squares TSS remains constant, i.e.,

$$TSS = a^2b^2WSS + a^2BSS,$$

provided

$$a^2 = \frac{TSS}{b^2WSS + BSS}.$$

Requiring the overall mean $\bar{W}$ to remain constant implies that

$$\bar{W}_c^* = (1-a)\bar{W} + a\bar{W}_c,$$

where the $*$ superscript refers to the transformed W. Thus,

$$\begin{aligned} W_{ic}^* &= \bar{W}_c^* + ab(W_{ic}-\bar{W}_c) \\ &= (1-a)\bar{W} + a\bar{W}_c + ab(W_{ic}-\bar{W}_c) \\ &= (1-a)\bar{W} + a(1-b)\bar{W}_c + abW_{ic}. \end{aligned} \tag{2.13}$$

The above transformation results in a scaling of the ratio of within to between-cluster sums of squares by b^2, while keeping the overall mean and variance constant.

In our simulations, we fixed $C = 3$ clusters and considered three values for b^2, where the same value of b^2 is applied to each of the K columns of W: $b^2 = 1$, corresponding to the case where the within and between-cluster sums of squares are the same as the ones of the fitted W from the real dataset; $b^2 = 5$, corresponding to a larger ratio of within to between-cluster sums of squares and hence a harder clustering problem; $b^2 = 10$, corresponding to a very large ratio of within to between-cluster sums of squares and hence almost no clustering.

Overall, 2 (real datasets) $\times$ 3 (n) $\times$ 3 ($zfrac$) $\times$ 3 (clustering) = 54 scenarios were considered in the simulation.

Simulating the datasets. For each of the 54 scenarios, we simulated $B = 10$ datasets, resulting in a total of $54 \times 10 = 540$ datasets. Using the fitted W, α_μ, α_π, β_μ, β_π, γ_μ, γ_π, and ζ from one of the two real datasets, the datasets were simulated according to the following steps.

1. Simulate W with desired clustering strength. First fit a K-variate Gaussian mixture distribution to W inferred from one of the real datasets using the R function `Mclust` from the mclust package and specifying the number of clusters C. Then, for each of $B = 10$ datasets, simulate W cluster by cluster from K-variate Gaussian distributions using the `mvrnorm` function from the MASS package, with the cluster means, covariance matrices, and frequencies output by `Mclust`. Transform W as in Equation (2.13) to get the desired ratio of within to between-cluster sums of squares.
2. Simulate γ_μ and γ_π to get the desired zero fraction. We only considered cell-level intercept n-vectors γ_μ and γ_π, i.e., $L = 1$ and a matrix V of gene-level covariates consisting of a single column of ones. As the fitted γ_μ and γ_π from the original datasets are correlated n-vectors, fit a bivariate Gaussian distribution to γ_μ and γ_π using the function `Mclust` from the mclust package with $C = 1$ cluster. Then, for each of $B = 10$ datasets, simulate γ_μ and γ_π from a bivariate Gaussian distribution using the `mvrnorm` function from the MASS package, with the mean and covariance matrix output by `Mclust`. To increase/decrease the zero fraction, increase/decrease each element of the mean vector for γ_π inferred from `Mclust` (shifts of $\{0, 2, 5\}$ for V1 dataset and $\{-1.5, 0.5, 2\}$ for S1/CA1 dataset).
3. Create the ZINB-WaVE model using the function `zinbModel` from the package zinbwave.
4. Simulate counts using the function `zinbSim` from the package zinbwave.

Simulating from the Lun and Marioni [40] model

To simulate datasets from a different model than our ZINB-WaVE model, we simulated counts using the procedure described in Lun and Marioni [40] (details in Supplementary Materials of original publication and code available from the Github repository `https://github.com/MarioniLab/PlateEffects2016`). Although the Lun and Marioni [40] model is also based on a zero-inflated negative binomial distribution, the distribution is parameterized differently and also fit differently, gene by gene. In particular, the negative binomial mean is parameterized as a product of the expression level of interest and two nuisance technical effects, a gene-level effect assumed to have a log-normal distribution and a cell-level effect (cf. library size) whose distribution is empirically derived. The zero inflation probability is assumed to be constant across cells for each gene and is estimated independently of the negative binomial mean. The ZINB distribution is fit gene by gene using the `zeroinfl` function from the R package pscl. We used the raw gene-level read counts for the mESC dataset as input to the script `reference/submitter.sh`, to create a simulation function constructed by fitting a ZINB distribution to these counts. The script `simulations/submitter.sh` was then run to simulate counts based on the estimated parameters (negative binomial mean, zero inflation probability, and genewise dispersion parameter). We simulated $C = 3$ clusters with equal number of cells per cluster. The parameters which were varied in the simulations are as follows.

- The number of cells: $n = 100, 1, 000, 10, 000$.
- The proportion of zero counts, $zfrac = \sum_{i,j} \mathbf{1}(Y_{ij} = 0)/nJ$, via the zero inflation probability: $zfrac \approx 0.4, 0.6, 0.8$. For $zfrac = 0.4$, we did not modify the code in Lun and Marioni [40]. However, to simulate datasets with greater zero fractions, namely $zfrac = 0.6$ and $zfrac = 0.8$, we added respectively 0.3 and 0.6 to the zero inflation probability (p'_i, in their notation).

For each of the 3 (n) $\times$ 3 ($zfrac$) = 9 scenarios, we simulated B ($B = 10$) datasets, resulting in $9 \times 10 = 90$ simulated datasets in total.

2.2.6 Dimensionality reduction methods

Three different dimensionality reduction methods were applied to the real and simulated datasets: ZINB-WaVE, zero-inflated factor analysis, and principal component analysis. For all the methods, we selected $K = 2$ dimensions, unless specified otherwise. A notable exception is the S1/CA1 dataset, for which, given the large number of cells and the complexity of the signal, we specified $K = 3$ dimensions.

ZINB-WaVE

We applied the ZINB-WaVE procedure using the function `zinbFit` from our R package zinbwave, with the following parameter choices.

- Number of unknown cell-level covariates K: $K = 1, 2, 3, 4$.
- Gene-level covariate matrix V: not included or set to a column of ones $\mathbf{1}_J$.
- Cell-level covariate matrix X: set to a column of ones $\mathbf{1}_n$. For the mESC dataset, we also considered including batch covariates in X.

- Dispersion parameter ζ: the same for all genes (common dispersion) or specific to each gene (genewise dispersion).

Zero-inflated factor analysis

We used the zero-inflated factor analysis (ZIFA) method [34], as implemented in the `ZIFA` python package (Version 0.1) available at `https://github.com/epierson9/ZIFA`, with the block algorithm (function `block_ZIFA.fitModel`, with default parameters). The output of ZIFA is an $n \times K$ matrix corresponding to a projection of the counts onto a latent low-dimensional space of dimension K.

Principal component analysis

We used the function `prcomp` from the R package stats for the simulation study and, for computational efficiency, the function `jsvds` from the R package rARPACK for the real datasets.

2.2.7 Normalization methods

As normalization is essential, especially for zero-inflated distributions, PCA and ZIFA were applied to both raw and normalized counts. The following normalization methods were used.

- Total-count normalization (TC). Counts are divided by the total number of reads sequenced for each sample and multiplied by the mean total count across all the samples. This method is related to the popular transcripts per million (TPM) [45] and fragments per kilobase million (FPKM) [46] methods.

- Full-quantile normalization (FQ) [47]. The quantiles of the distributions of the gene-level read counts are matched across samples. We used the function `betweenLaneNormalization` from the Bioconductor R package EDASeq.

- Trimmed mean of M values (TMM) [48]. The TMM global-scaling factor is computed as the weighted mean of log-ratios between each sample and a reference sample. If the majority of the genes are not differentially expressed (DE), TMM should be close to 1, otherwise, it provides an estimate of the correction factor that must be applied in order to fulfill this hypothesis. We used the function `calcNormFactors` from the Bioconductor R package edgeR to compute these scaling factors.

2.2.8 Clustering methods

Clustering of the OE dataset. We used the resampling-based sequential ensemble clustering (RSEC) framework implemented in the `RSEC` function from the Bioconductor R package clusterExperiment [49]. Briefly, RSEC implements a consensus clustering algorithm which generates and aggregates a collection of clusterings, based on resampling cells and using a sequential tight clustering algorithm [50]. See Fletcher et al. [41] for details on the parameters used for RSEC in the original analysis and Perraudeau et al. [42] for details on the parameters used for RSEC in the ZINB-WaVE workflow.

Clustering of the 10x Genomics 68k PBMCs dataset. We used two different clustering methods to assess the ability of ZINB-WaVE to extract biologically meaningful signal from the data. First, we used a a clustering procedure similar to the one implemented in the R package Seurat [51] (Version 2.0.1). In particular, we used ZINB-WaVE ($K = 10$) instead of PCA for dimensionality reduction. The clustering is based on shared nearest neighbor modularity [52]. We used the following parameters: `k.param = 10, k.scale=10, resolution = 0.6`. All the other parameters were left at their default values. We also clustered the data using a sequential k-means clustering approach [50], implemented in the `clusterSingle` function of the clusterExperiment package [49]. We used the following parameters: `sequential = TRUE, subsample = FALSE, k0 = 15, beta = 0.95, clusterFunction = "kmeans"`.

2.2.9 Models for count data

We compared the goodness-of-fit of our ZINB-WaVE model to that of two other models for count data: a standard negative binomial model, that does not account for zero inflation, and the MAST hurdle model, that is specifically designed for scRNA-seq data [31].

Goodness-of-fit was assessed on the V1 dataset using mean-difference plots (MD-plots) of estimated vs. observed mean count and zero probability, as well as plots of the estimated dispersion parameter against the observed zero frequency.

ZINB-WaVE model

For our ZINB-WaVE model, the overall mean and zero probability are

$$\begin{aligned} E[Y_{ij}] &= (1-\pi_{ij})\mu_{ij}, \\ P(Y_{ij}=0) &= \pi_{ij} + (1-\pi_{ij})(1+\phi_j\mu_{ij})^{\frac{1}{\phi_j}}. \end{aligned}$$

ZINB-WaVE was fit to the V1 dataset using the function `zinbFit` from our R package zinbwave, with the following parameter choices: $K = 0$ unknown cell-level covariates, gene-level intercept ($X = \mathbf{1}_n$), cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion.

Negative binomial model

The negative binomial (NB) distribution is a special case of the ZINB distribution that does not account for zero inflation, i.e., for which $\pi = 0$. Thus,

$$\begin{aligned} E[Y_{ij}] &= \mu_{ij}, \\ P(Y_{ij} = 0) &= (1 + \phi_j \mu_{ij})^{\frac{1}{\phi_j}}. \end{aligned}$$

A NB distribution was fit gene by gene to the V1 dataset, after full-quantile normalization, using the Bioconductor R package edgeR [53] (Version 3.16.5) with only an intercept (i.e., default value for the `design` argument as a single column of ones) and genewise dispersion.

Model-based analysis of single-cell transcriptomics

The model-based analysis of single-cell transcriptomics (MAST) hurdle model proposed by Finak et al. [31] is defined as follows:

$$\begin{aligned} \text{logit}(P(Z_{ij} = 1)) &= X_i \beta_j^D, \\ Y_{ij} | Z_{ij} = 1 &\sim \mathcal{N}(X_i \beta_j^C, \sigma_j^2), \end{aligned}$$

where Y_{ij} is log2(TPM +1) for cell i and gene j, $X_i \in \mathbb{R}^k$ is a known covariate vector, and Z_{ij} indicates whether gene j is truly expressed in cell i. For each gene j, the parameters of the MAST model are: the regression coefficients $\beta_j^D \in \mathbb{R}^k$ for the discrete part and the regression coefficients $\beta_j^C \in \mathbb{R}^k$ and variance $\sigma_j^2 \in \mathbb{R}$ for the Gaussian continuous part. Note that we follow the notation in Finak et al. [31], which is different from that in the ZINB-WaVE model of Equations (2.4)–(2.6).

For the MAST model, the overall mean and zero probability are

$$\begin{aligned} E[Y_{ij}] &= \text{logit}^{-1}(X_i \beta_j^D) X_i \beta_j^C, \\ P(Y_{ij} = 0) &= 1 - \text{logit}^{-1}(X_i \beta_j^D). \end{aligned}$$

MAST is implemented in the Bioconductor R package MAST [54] (which allows different covariate vectors X_i for the continuous and discrete components). We use the function `zlm` to fit MAST with an intercept and a covariate for the cellular detection rate (as recommended in the MAST vignette for the MAIT data analysis) for both the discrete and continuous parts.

Note that in contrast to the NB and ZINB models, the MAST hurdle model is for log2(TPM+1) instead of counts and has no dispersion parameter. To be able to compare the fits of the three models, the MAST goodness-of-fit plots display estimated vs. observed mean log2(TPM+1) and estimated variance σ_j^2 vs. observed zero frequency (Figure 2.28). Although not a direct comparison, we think this allows a fair assessment of the goodness-of-fit of the different models.

2.2.10 Evaluation criteria

Clustering: Silhouette width

Given a set of labels for the cells (e.g., biological condition, batch), silhouette widths provide a measure of goodness of the clustering of the cells with respect to these labels. Silhouette widths may be averaged within clusters or across all observations to assess clustering strength at the level of clusters or overall, respectively. The silhouette width s_i of a sample i is defined as follows:

$$s_i = \frac{b_i - a_i}{\max\{a_i, b_i\}},$$

where $a_i = d(i, \mathcal{C}_{cl(i)})$, $b_i = \min_{l \neq cl(i)} d(i, \mathcal{C}_l)$, $\mathcal{C}_{cl(i)}$ is the cluster to which i belongs, and $d(i, \mathcal{C}_l)$ is the average distance between sample i and the samples in cluster $\mathcal{C}_l$.

Average silhouette widths were used to compare ZINB-WaVE, PCA, and ZIFA on both real and simulated datasets. For simulated data, the cluster labels correspond to the true simulated W and, for each scenario, silhouette widths were computed and averaged over B datasets. For real data, the authors' cluster labels or known cell types were used.

Clustering: Precision and recall

Two different clusterings (i.e., partitions) may be compared quantitatively using the precision and recall coefficients. These measures involve assessing whether pairs of cells cluster together in each of the two clusterings. The four different possibilities are enumerated in the following table.

		Clustering 2	
		Pairs in same cluster	
		No	Yes
Clustering 1	No	NN	NY
Pairs in same cluster	Yes	YN	YY

Then, using Clustering 1 as a reference, the precision coefficient is defined as the proportion of cells clustered together in Clustering 2 which are also (correctly) clustered together in Clustering 1

$$Precision = \frac{YY}{YY + NY}.$$

Similarly, the recall coefficient is defined as the proportion of cells (correctly) clustered together in Clustering 1 which are also clustered together in Clustering 2

$$Recall = \frac{YY}{YY + YN}.$$

Precision and recall were used to compare ZINB-WaVE, PCA, and ZIFA on simulated datasets, where the reference Clustering 1 corresponds to the true simulated labels and Clustering 2 was performed using the R function `kmeans` in the reduced-dimensional space: the first two columns of W for ZINB-WaVE, the first two principal components for PCA, and the first two latent variables for ZIFA. We simulated 3 clusters and set the number of clusters equal to 3 in `kmeans` (argument `centers`). Precision and recall were computed using the `extCriteria` function from the R package clusterCrit [55].

Dimensionality reduction: Correlation with QC measures

To evaluate the dependence of the inferred low-dimensional signal on unwanted variation, we computed the absolute correlation between each dimension (e.g., principal component) and a set of quality control (QC) measures. For the V1 dataset, FastQC and Picard tools were used to compute a set of 16 QC measures. These measures are available as part of the Bioconductor R package scRNAseq. For the Glioblastoma and mESC datasets, we used the scater Bioconductor R package [56] (Version 1.2.0) to compute a set of 7 QC measures. For the S1/CA1 dataset, we used a set of 6 QC measures provided by the authors.

Dimensionality reduction: Correlation of pairwise distances between observations

For simulated data, we assessed different dimensionality reduction methods (ZINB-WaVE, PCA, and ZIFA) in terms of the correlation between pairwise distances between observations in the true and in the estimated reduced-dimensional space. In particular, we monitored the influence of the number of unknown cell-level covariates K when the other parameters are set correctly (V is a column of ones $\mathbf{1}_J$ and genewise dispersion). For each simulation scenario, the correlation between true and estimated pairwise distances was computed and averaged over B datasets.

Bias and MSE of the ZINB-WaVE estimators

For data simulated from the ZINB-WaVE model, let θ and $\hat{\theta}_b$, $b = 1, \ldots, B$, respectively denote the true parameter and an estimator of this parameter for the b^{th} simulated dataset. Then, for each scenario, the performance of the estimator $\hat{\theta}$ can be assessed in terms of bias and mean squared error (MSE) as follows:

$$\begin{aligned}
\text{Bias} &= \frac{1}{B}\sum_{b=1}^{B}(\hat{\theta}_b - \theta), \\
\text{MSE} &= \frac{1}{B}\sum_{b=1}^{B}(\hat{\theta}_b - \theta)^2.
\end{aligned}$$

Bias and MSE were computed for β_μ, β_π, γ_μ, γ_π, ζ, $W\alpha_\mu$, $W\alpha_\pi$, $\ln(\mu)$, and $\text{logit}(\pi)$. When the parameter to be estimated was an $n \times J$ matrix, the matrix was converted to a $1 \times nJ$ row vector and bias and MSE were averaged over the elements of the vector.

Model selection for ZINB-WaVE

The Akaike information criterion (AIC) and the Bayesian information criterion (BIC) are widely used for model selection and are defined as follows:

$$\begin{aligned} AIC &= 2N - 2\ell(\hat\beta, \hat\gamma, \hat W, \hat\alpha, \hat\zeta), \\ BIC &= \ln(n)N - 2\ell(\hat\beta, \hat\gamma, \hat W, \hat\alpha, \hat\zeta), \end{aligned}$$

where $\ell(\hat\beta, \hat\gamma, \hat W, \hat\alpha, \hat\zeta)$ is the log-likelihood function evaluated at the MLE (Equation (4.5)), n is the sample size, and N is the total number of estimated parameters, i.e., $N = J(M_\mu + M_\pi) + n(L_\mu + L_\pi) + 2KJ + nK + J$ when the model is fit with genewise dispersion and $N = J(M_\mu + M_\pi) + n(L_\mu + L_\pi) + 2KJ + nK + 1$ when the model is fit with common dispersion.

We use AIC and BIC in the simulation to select the number of unknown cell-level covariates (K). Note that because of the complex non-convex likelihood function for the ZINB-WaVE model, there is no closed-form expression for the MLE and our numerical optimization procedure only provides an approximation of the AIC and BIC. In practice, however, we found that our results closely approximated the true MLE (see Results and Figure 2.29).

2.3 Results

2.3.1 A general and flexible model for scRNA-seq

ZINB-WaVE is a general and flexible model for the analysis of high-dimensional zero-inflated count data, such as those recorded in single-cell RNA-seq assays. Given n samples (typically, n single cells) and J features (typically, J genes) that can be counted for each sample, we denote with Y_{ij} the count of feature j ($j = 1, \ldots, J$) for sample i ($i = 1, \ldots, n$). To account for various technical and biological effects, typical of single-cell sequencing technologies, we model Y_{ij} as a random variable following a zero-inflated negative binomial distribution (see Methods for details).

Both the mean expression level (μ) and the probability of dropouts (π) are modeled in terms of *observed* sample-level and gene-level covariates (X and V, respectively, Figure 2.1). In addition, we include a set of *unobserved* sample-level covariates (W) that need to be inferred from the data. The matrix X can include covariates that induce variation of interest, such as cell types, or covariates that induce unwanted variation, such as batch or quality control (QC) measures. It can also include a constant column of ones for an intercept that

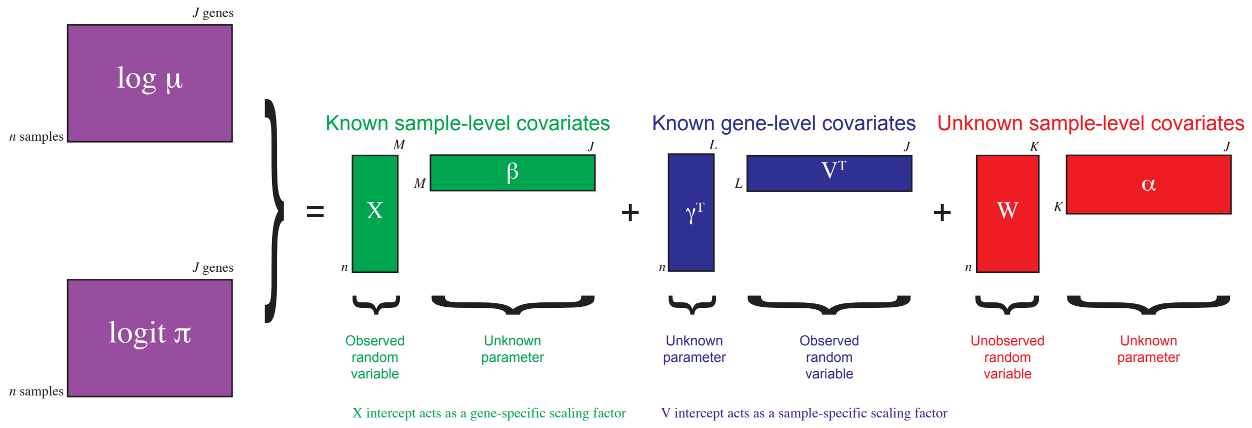

J genes
log μ
n samples
J genes
logit π
n samples
=
Known sample-level covariates
M
J
X
M
β
n
Observed random variable
Unknown parameter
X intercept acts as a gene-specific scaling factor
+
Known gene-level covariates
L
J
γ^T
L
V^T
n
Unknown parameter
Observed random variable
V intercept acts as a sample-specific scaling factor
+
Unknown sample-level covariates
K
J
W
K
α
n
Unobserved random variable
Unknown parameter

Figure 2.1: *Schematic view of the ZINB-WaVE model.* Given n cells and J genes, let Y_{ij} denote the count of gene j ($j = 1, \ldots, J$) for cell i ($i = 1, \ldots, n$) and Z_{ij} an unobserved indicator variable, equal to one if gene j is a dropout in cell i and zero otherwise. Then, $\mu_{ij} = E[Y_{ij}|Z_{ij} = 0, X, V, W]$ and $\pi_{ij} = Pr(Z_{ij} = 1|X, V, W)$. We model $\ln(\mu)$ and $\text{logit}(\pi)$ with the regression specified in the figure. Note that the model allows for different covariates to be specified in the two regressions; we have omitted the μ and π indices for clarity (see Methods for details).

accounts for gene-specific differences in mean expression level or dropout rate (cf. scaling in PCA). The matrix V can also accommodate an intercept to account for cell-specific global effects, such as size factors representing differences in library sizes (i.e., total number of reads per sample). In addition, V can include gene-level covariates, such as gene length or GC-content.

The unobserved matrix W contains unknown sample-level covariates, which could correspond to unwanted variation as in RUV [35, 36] or could be of interest as in cluster analysis (e.g., cell type). The model extends the RUV framework to the ZINB distribution (thus far, RUV had only been implemented for linear [35] and log-linear regression [36]). It differs in interpretation from RUV in the $W\alpha$ term, which is not necessarily considered unwanted; this term generally refers to unknown low-dimensional variation, that could be due to unwanted technical effects (as in RUV), such as batch effects, or to biological effects of interest, such as cell cycle or cell differentiation.

It is important to note that although W is the same, the matrices X and V could differ in the modeling of μ and π, if we assume that some known factors do not affect both. When $X = \mathbf{1}_n$ and $V = \mathbf{1}_J$, the model is a factor model akin to principal component analysis, where W is a factor matrix and α_μ and α_π are loading matrices. However, the model is more general, allowing the inclusion of additional sample and gene-level covariates that might help to infer the unknown factors.

2.3.2 Biologically meaningful clusters

We applied the ZINB-WaVE procedure to several publicly-available real datasets, from microfluidics, plate-based, and droplet-based platforms (see Methods).

As previously shown Hicks, Teng, and Irizarry [30] and Finak et al. [31], the first few principal components of scRNA-seq data, even after normalization, can be influenced by technical rather than biological features, such as the proportion of genes with at least one read (*detection rate*) or the total number of reads (*sequencing depth*) per sample. Figure 2.2 illustrates this using the V1 dataset: although the first two principal components somewhat segregated the data by layer of origin (Figure 2.2a), the clustering was far from perfect. This is at least partly due to unwanted technical effects, such as sequencing depth and amount of starting material. To quantify such technical effects, we computed a set of QC measures, such as detection rate and total number of reads (see Methods). The first two principal components are especially correlated with detection rate (Figure 2.2b).

ZIFA suffered from the same problem: the clustering of the samples in two dimensions was not qualitatively different from PCA (Figure 2.2c) and the second component was highly correlated with detection rates (Figure 2.2d).

Conversely, ZINB-WaVE led to tighter clusters, grouping the cells by layer of origin (Figure 2.2e). Furthermore, the two components inferred by ZINB-WaVE showed lower correlation with the QC features (Figure 2.2f), highlighting that the clusters shown in Figure 2.2e are not driven by technical effects.

We repeated these analyses on the S1/CA1 dataset (Figure 2.8), mESC dataset (Figure 2.9), and Glioblastoma dataset (Figure 2.10). For all datasets, ZINB-WaVE led to tighter clusters in two dimensions. However, it did not always lead to a decrease in correlation with the QC measures. See also Hicks, Teng, and Irizarry [30] for additional datasets in which principal components are strongly correlated with detection rate.

As a measure of the goodness of the clustering results, we used the average silhouette width (see Methods), computed using the labels available from the original study: these were either known *a priori* (e.g., the patient ID in the Glioblastoma dataset) or inferred from the data (and validated) by the authors of the original publication (e.g., the cell types in the S1/CA1 dataset). For all four datasets, ZINB-WaVE led to generally tighter clusters, as shown by an increased per-cluster average silhouette width in the majority of the groups (Figure 2.12).

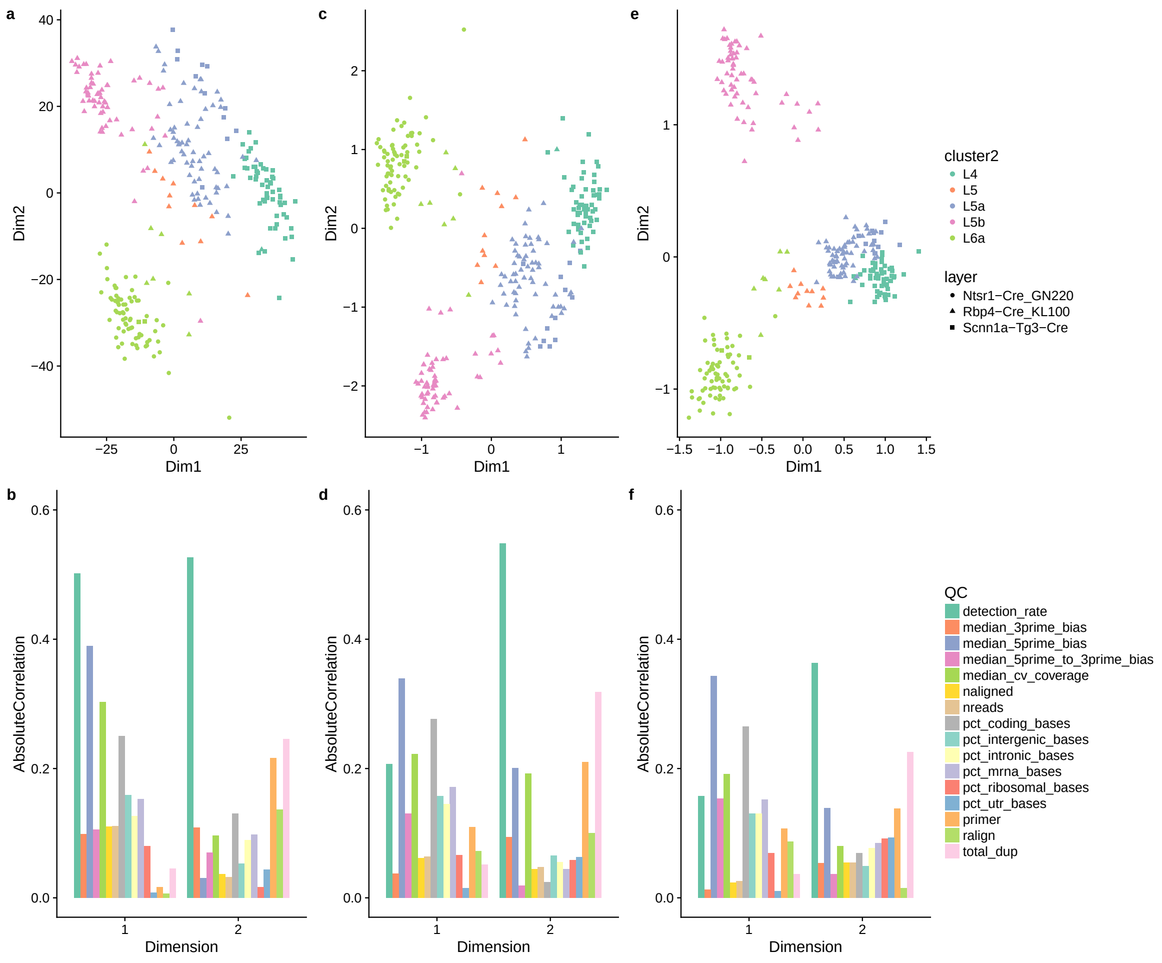


Figure 2.2: *Low-dimensional representation of the V1 dataset.* Upper panels provide two-dimensional representations of the data, after selecting the 1,000 most variable genes. Lower panels provide barplots of the absolute correlation between the first two components and a set of QC measures (see Methods). **(a, b)** PCA (on TC-normalized counts); **(c, d)** ZIFA (on TC-normalized counts); **(e, f)** ZINB-WaVE (no normalization needed). ZINB-WaVE leads to a low-dimensional representation that is less influenced by technical variation and to tighter, biologically meaningful clusters. The ZINB-WaVE projection was robust to the number of genes selected (Figure 2.11).

2.3.3 Novel biological insights

To demonstrate the ability of ZINB-WaVE to lead to novel biological insights, we focused on two inferential questions typical of scRNA-seq studies: (i) the identification of developmental lineages and (ii) the characterization of rare cell types.

First, we reanalyzed a set of cells from the mouse olfactory epithelium (OE) that were collected to identify the developmental trajectories that generate olfactory neurons (mOSN), sustentacular cells (mSUS), and microvillous cells (MV) [41]. In the original publication, the data were normalized by full-quantile normalization, followed by a regression-based adjustment for sample quality. Clustering on the first 50 principal components identified cellular states that were then ordered into developmental lineages by Slingshot [57] using the first 5 PCs. In the original analysis, Slingshot was able to correctly infer the lineages in its supervised mode, by manually specifying lineage endpoints (i.e., clusters corresponding to the mature cell types). Figure 2.3a shows the minimum spanning tree (MST) obtained with the described supervised analysis. When running Slingshot in unsupervised mode, however, the inferred MST only correctly identified the neuronal (mOSN) and microvillous (MV) lineages, while it was unable to identify sustentacular (mSUS) cells as a mature cell type (Figure 2.3b). By repeating the clustering and lineage reconstruction with Slingshot on the first 50 factors of ZINB-WaVE, we were able to infer the correct lineages even in unsupervised mode (Figure 2.3c). This suggests that the low-dimensional signal revealed by ZINB-WaVE more closely matches the true developmental process. See Perraudeau et al. [42] for a complete workflow that involves ZINB-WaVE for dimensionality reduction and Slingshot for lineage reconstruction.

We next focused on a set of 68,579 peripheral blood mononuclear cells (PBMCs) assayed with the 10x Genomics Chromium system [44]. This example allowed us to demonstrate how ZINB-WaVE can be applied to a state-of-the-art dataset that comprises tens of thousands of cells, while also illustrating that existing clustering algorithms can be used on the low-rank matrix inferred by ZINB-WaVE. In particular, we applied a popular clustering method, based on the identification of shared nearest neighbors [52], similar to that of Macosko et al. [10] and implemented in the R package Seurat [51]. We modified the clustering procedure to use ZINB-WaVE (with $K = 10$) instead of PCA as the dimensionality reduction step. To visualize the clustering results, we applied t-SNE to the inferred W matrix (Figure 2.3d).

Our approach recapitulates the major cell populations found in Zheng et al. [44] (Figure 2.3d, e). In particular, 80% of the cells are T-cells (Clusters 0 – 1 CD4+ T-cells; Clusters 2 – 6 CD8+ T-cells). As in Zheng et al. [44], we are able to identify populations of activated cytotoxic T-cells (Cluster 5; 9%), natural killer cells (Cluster 7; 6%), and B-cells (Cluster 8; 5%) (Figure 2.3d). In addition, we are able to identify sub-populations of myeloid cells that were not completely characterized in the original publication. In particular, we identified clusters corresponding to: CD14+ monocytes (Cluster 9; 3%), characterized by

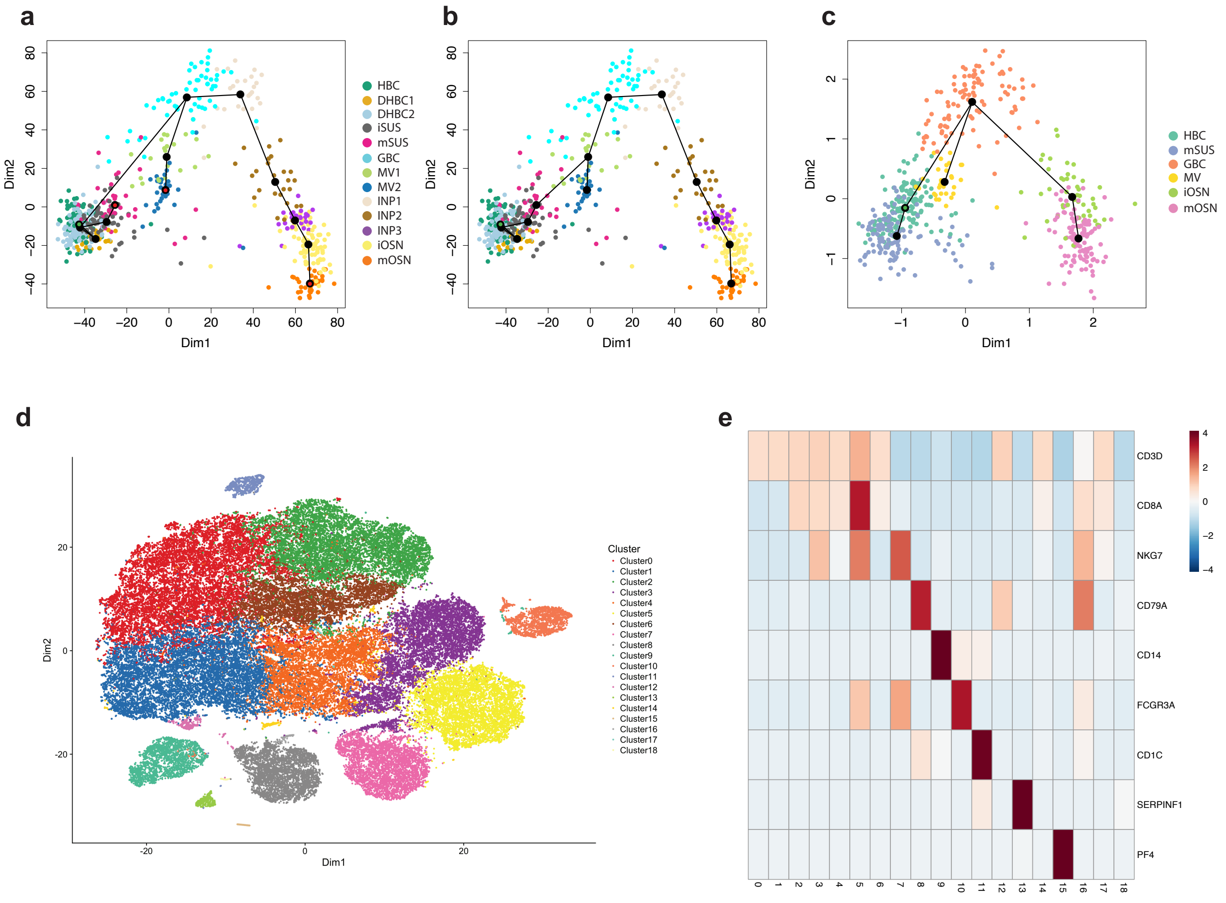


Figure 2.3: *Using ZINB-WaVE to gain novel biological insights: Lineage inference for OE dataset and discovery of rare cell types for 10x Genomics 68k PBMCs dataset.* **(a–c)** Lineage inference on the OE dataset. Slingshot minimum spanning tree on cell clusters: **(a)** PCA with endpoint supervision (marked as red nodes in the tree); **(b)** PCA with no endpoint supervision; **(c)** ZINB-WaVE with no endpoint supervision. **(a–b)** RSEC clustering (see Methods) on the first 50 PCs of the normalized counts led to 13 clusters; **(c)** the same procedure on 50 components of W from ZINB-WaVE led to 6 clusters: horizontal basal cells (HBC); transitional HBC (DHBC1-2); immature sustentacular cells (iSUS); mature sustentacular cells (mSUS); globose basal cells (GBC); microvillous cells (MV); immediate neuronal precursors (INP1-3); immature olfactory neurons (iOSN); mature olfactory neurons (mOSN). mOSN, MV, and mSUS are the mature cell types and should be identified by Slingshot as the three lineage endpoints. **(d–e)** Discovery of novel cell types for the 10x Genomics 68k PBMCs dataset. **(d)** Scatterplot of first two t-SNE dimensions obtained from 10 components of W from ZINB-WaVE; cells are color-coded by cluster. Clustering was performed on the 10 components of W (see Methods). **(e)** Heatmap of expression measures for marker genes for the 18 clusters found by our procedure: columns correspond to clusters and rows to genes; the value in each cell is the average log expression measure per cluster, centered and scaled so that each row has mean zero and standard deviation one.

the expression of *CD14*, *S100A9*, *LYZ* [58]; CD16+ monocytes (Cluster 10; 2.5%), which express *FCGR3A/CD16*, *AIF1*, *FTL*, *LST1* [58]; CD1C+ dendritic cells (DC; Cluster 11; 1%), which express *CD1C*, *LYZ*, *HLA-DR* [58]; plasmacytoid dentritic cells (pDC; Cluster 13; 0.5%), expressing *GZMB*, *SERPINF1*, *ITM2C* [58]; megakaryocyte (Cluster 15; 0.2%), expressing *PF4* [44] (Figure 2.3e). We also identified several small clusters (collectively comprising about 1% of the cells) that were either enriched for mitochondrial genes (Cluster 14) or ribosomal genes (Clusters 17 and 18), or for which we could not find any markers (Clusters 12 and 16). We hypothesize that these clusters represent either doublets or lower quality libraries.

The above analysis differs from that of the original publication not only in terms of dimensionality reduction, but also in terms of clustering (Zheng et al. [44] used k-means on the first 50 PCs). We hence repeated the clustering using a sequential procedure based on k-means (see Methods for details). This led to similar (albeit noisier) results (Figure 2.13), highlighting that ZINB-WaVE is able to extract meaningful biological signal from the data, as input to a variety clustering procedures. In fact, extracting a two-dimensional signal from ZINB-WaVE already allowed us to identify most major cell types (Figure 2.14).

2.3.4 Impact of normalization

As for any high-throughput genomic technology, an important aspect of scRNA-seq data analysis is normalization. Indeed, there are a variety of steps in scRNA-seq experiments that can introduce bias in the data and whose effects need to be corrected for [19]. Some of these effects, e.g., sequencing depth, can be captured by a global-scaling factor (usually referred to as *size factor*). Other more complex, non-linear effects, such as those collectively known as *batch effects*, require more sophisticated normalization [36]. Accordingly, a typical scRNA-seq pipeline will include a normalization step, i.e., a linear or non-linear transformation of read counts, to make the distributions of expression measures comparable between samples. The normalization step is usually carried out prior to any inferential procedure (e.g., clustering, differential expression analysis, or pseudotime ordering). In this work, we considered three popular between-sample normalization methods: total-count (TC), trimmed mean of M values (TMM), and full-quantile (FQ) normalization (see Methods); we also compared the results to unnormalized data (RAW). TC, along with the related methods transcripts per million (TPM) and fragments per kilobase million (FPKM), is by far the most widely used normalization in the scRNA-seq literature; hence, TC-normalized data were used for the results shown in Figures 2.2, 2.8, 2.9, and 2.10.

Normalization highly affected the projection of the data in lower dimensions (Figures 2.17 – 2.22) and, consequently, the clustering results varied greatly between normalization methods in all four datasets (Figure 2.4). Strikingly, the choice of normalization method was more critical than the choice between PCA and ZIFA. For instance, for the mESC dataset

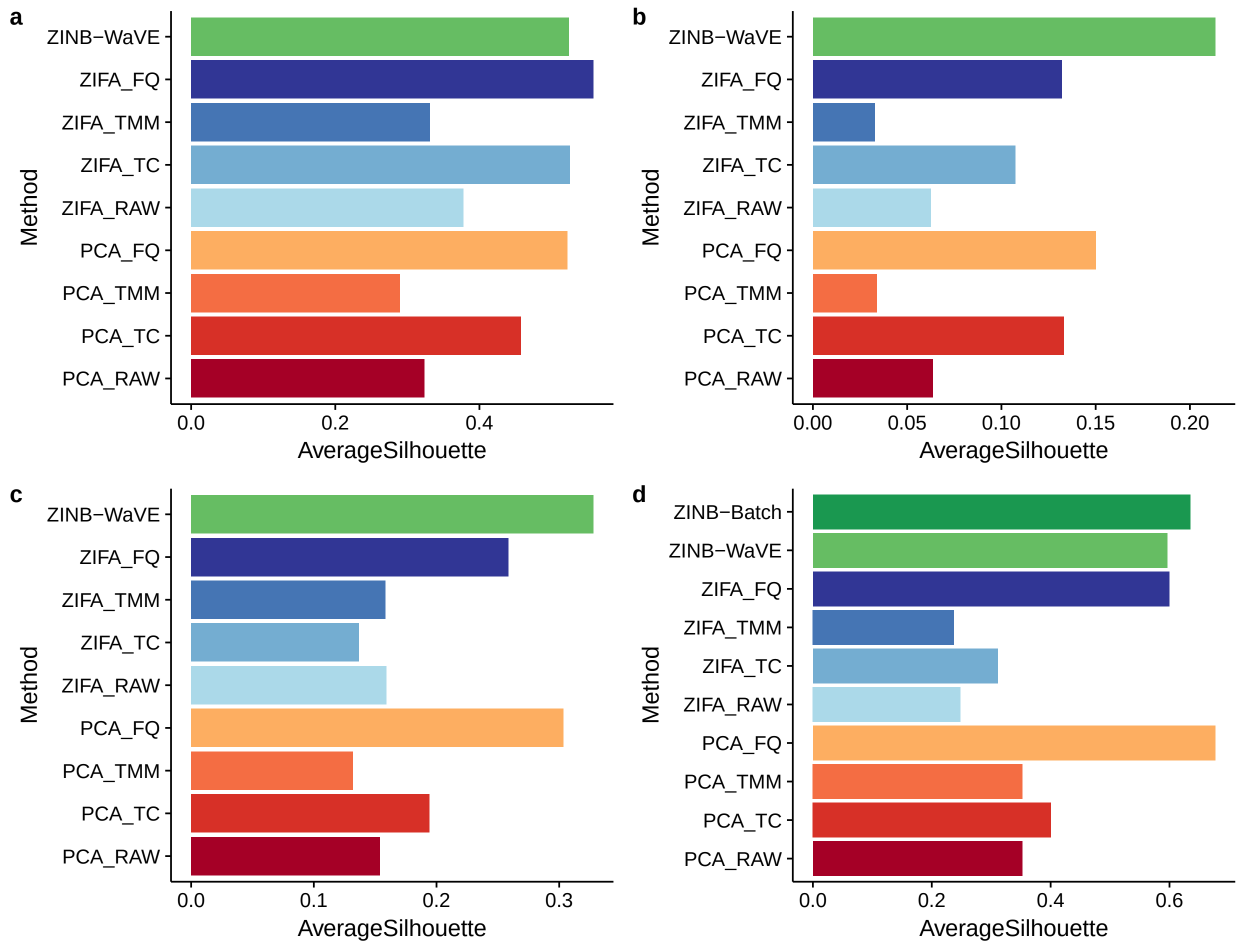


Figure 2.4: *Average silhouette width: Real datasets.* **(a)** V1 dataset; **(b)** S1/CA1 dataset; **(c)** Glioblastoma dataset; **(d)** mESC dataset. Silhouette widths were computed in the low-dimensional space, using the groupings provided by the authors of the original publications: unsupervised clustering procedure **(a, b)**, observed characteristics of the samples, such as patient **(c)** and culture condition **(d)**. PCA and ZIFA were applied after normalization: unnormalized (RAW), total-count (TC), trimmed mean of M values (TMM), and full-quantile (FQ). For the mESC dataset **(d)**, we fitted the ZINB-WaVE model with batch as a sample-level covariate (ZINB-Batch) in addition to the default model with only a sample-level intercept (see Figure 2.5).

(Figure 2.4d), FQ normalization followed by either PCA or ZIFA led to very high average silhouette width. Critically, the ranking of normalization methods was not consistent across datasets (Figure 2.4), highlighting that the identification of the best normalization method for a given dataset is a difficult problem for scRNA-seq [19, 43].

One important feature of the ZINB-WaVE model is that the sample-level intercept γ (corresponding to a column of ones in the gene-level covariate matrix V, see Methods) acts as a global-scaling normalization factor, making a preliminary global-scaling step unnecessary. As a consequence, ZINB-WaVE can be directly applied to unnormalized read counts, hence preserving the distributional properties of count data. ZINB-WaVE applied to unnormalized counts led to results that were comparable, in terms of average silhouette width, to PCA and ZIFA applied using the best performing normalization (Figure 2.4). In particular, ZINB-WaVE outperformed PCA and ZIFA on the S1/CA1 (Figure 2.4b) and Glioblastoma (Figure 2.4c) datasets, while it was slightly worse than PCA (after FQ normalization) on the mESC dataset (Figure 2.4d) and than ZIFA (after FQ normalization) on the V1 dataset (Figure 2.4a). Interestingly, the overall average silhouette width was lower for the S1/CA1 and Glioblastoma datasets than it was for the V1 and mESC datasets, suggesting that ZINB-WaVE leads to better clustering in more complex situations (e.g., lower sequencing depth).

Although the overall average silhouette width is a useful metric to rank the different methods in terms of how well they represent known global biological structure, looking only at the average across many different cell types may be misleading. For the V1 dataset, for instance, ZINB-WaVE led to a slightly lower overall average silhouette width than ZIFA (Figure 2.4a). However, ZINB-WaVE yielded higher cluster-level average silhouette widths for the L6a, L5b, and L5 samples, while ZIFA produced higher average silhouette widths for the L4 and L5a samples (Figure 2.12a). In fact, certain cell types may be easier to cluster than others, leading to silhouette widths that differ greatly between different clusters, as is the case for the S1/CA1 dataset: oligodendrocytes are much easier to cluster than the other cell types and all methods were able to identify them (Figure 2.12b); on the other hand, only ZINB-WaVE was able to achieve a positive silhouette width for the pyramidal SS and CA1 neurons and it performed much better than PCA and ZIFA in the interneuron cluster; finally, certain groups, such as the endothelial-mural and astrocytes, were simply too hard to distinguish in three dimensions (Figure 2.12b).

2.3.5 Accounting for batch effects

Although the sample-level intercept γ (corresponding to a column of ones in the gene-level covariate matrix V) can act as a global-scaling normalization factor, this may not be sufficient to accurately account for complex, non-linear technical effects that may influence the data (e.g., batch effects). Hence, we explored the possibility of including additional sample-level covariates in X to account for such effects (see Methods).

We illustrate this using the mESC dataset, which is a subset of the data described in Kolodziejczyk et al. [39] and which comprises two batches of mESCs grown in three different media (see Methods). ZINB-WaVE applied with no additional covariates led to three clusters, corresponding to the three media (Figures 2.5a and 2.9). However, the clusters corresponding to media 2i and a2i are further segregated by batch (Figure 2.5a). Including indicator variables for batch in ZINB-WaVE (as columns of X) removed such batch effects, consolidating the clustering by medium (Figure 2.5b). This led to slightly larger average silhouette widths, both overall (Figure 2.4d) and at the cluster level (Figure 2.5c). Conversely, the average silhouette width for batch, a measure of how well the cells cluster by batch, was much larger for the model that did not include the batch covariate (Figure 2.5d), indicating that explicitly accounting for batch in the ZINB-WaVE model effectively removed the dependence of the inferred low-dimensional space on batch effects. Similar results were obtained by normalizing the data with the ComBat batch correction method [59], implemented in the Bioconductor R package sva [60] (Figure 2.23). The advantage of ZINB-WaVE is the ability of including batch effects in the same model used for dimensionality reduction, without the need for prior data normalization.

The mESC dataset is an example of good experimental design, where each batch includes cells from each biological condition (this is known as a factorial design). Hence, it is relatively easy to correct for batch effects and, unsurprisingly, both ComBat and ZINB-WaVE successfully do so. The Glioblastoma dataset is an example of a more complex situation, in which there is confounding between batch and biology, each patient being processed separately [30]. Luckily, the Glioblastoma design is not completely confounded, as patient *MGH26* was processed in two batches (Figure 2.5e). We used this feature of the data to test whether ZINB-WaVE was able to adjust for batch effects even in the presence of confounding. ComBat was not able to correctly account for batch, removing the patient effects along with the batch effects (Figure 2.24a). Including the batch variable as a covariate in the ZINB-WaVE model led to similar, unsatisfactory results (Figure 2.24b). One key observation, recently made by Townes et al. [61], is that the detection rate is markedly different between the two batches of *MGH26* (Figure 2.24c): Including the detection rate as a covariate in the ZINB-WaVE model led to the removal of the batch effect, while preserving the biological differences between patients (Figure 2.5f). Note that this is analogous of the inclusion of the "cellular detection rate" in the MAST model [31]. Although this helped adjusting for batch effects, the confounding between patient and batch is still present in the data because of the poor experimental design and no normalization will be able to completely account for such confounding.

2.3.6 Goodness-of-fit of ZINB-WaVE model

We compared the goodness-of-fit of our ZINB-WaVE model to that of a negative binomial (NB) model (as implemented in edgeR [53]) and a hurdle model (as implemented in MAST [31]).

As previously noted in the literature, NB models, which are quite successful for bulk RNA-seq, are not appropriate for single-cell RNA-seq, as they cannot accommodate zero inflation. In particular, NB models poorly fit the data in terms of the overall mean count and zero probability (Figures 2.25 and 2.26) and appears to handle the excess of zeros by

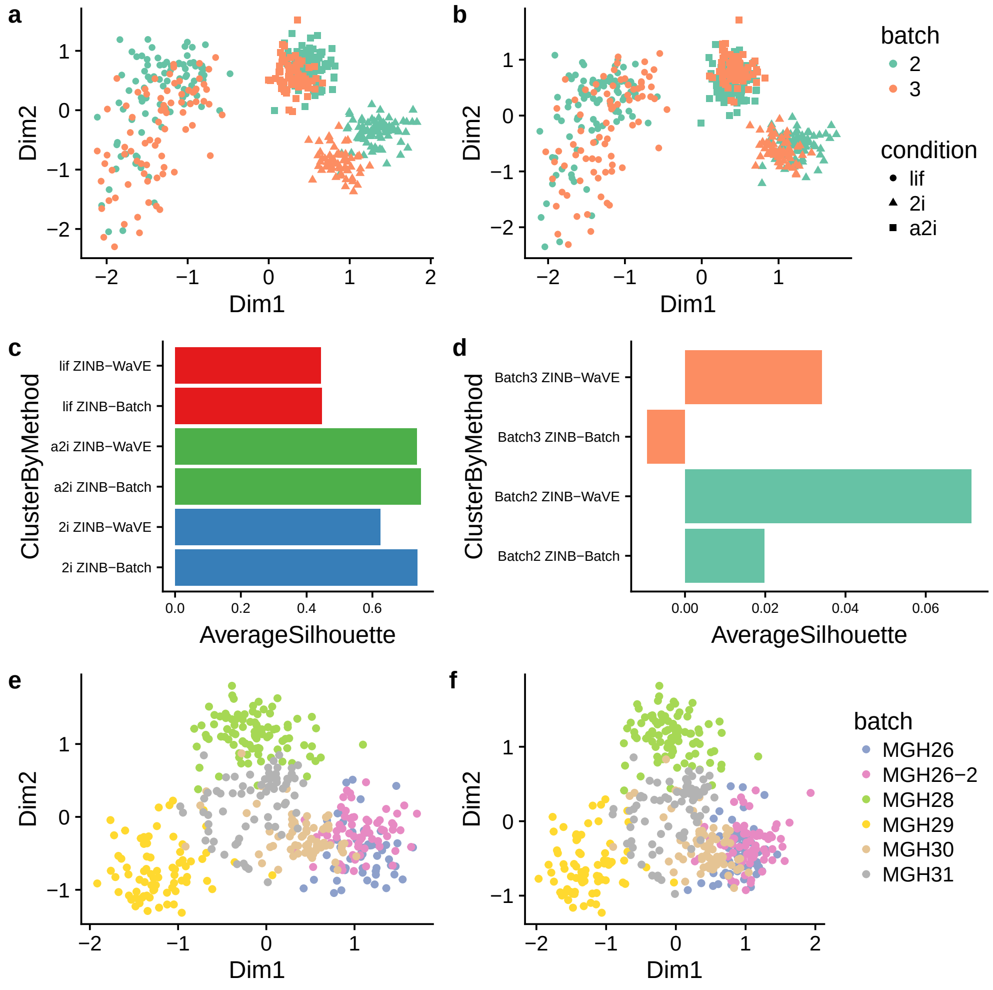


Figure 2.5: *Accounting for batch effects in ZINB-WaVE: mESC and Glioblastoma datasets.* Upper panels provide two-dimensional representations of the mESC data, with cells color-coded by batch and shape reflecting culture conditions: **(a)** Default ZINB-WaVE with only sample-level intercept; **(b)** ZINB-WaVE with batch as known sample-level covariate. **(c)** Average silhouette widths by biological condition for ZINB-WaVE with and without batch covariate; **(d)** Average silhouette widths by batch for ZINB-WaVE with and without batch covariate. Although the cells cluster primarily based on culture condition, batch effects are evident in **(a)**. Accounting for batch effects in the ZINB-WaVE model **(b)** leads to slightly better clustering by biological condition **(c)** and removes the clustering by batch **(d)**. Note the difference in scale between the barplots of **(c)** and **(d)**. Lower panels provide two-dimensional representations of the Glioblastoma data, with cells color-coded by batch: **(e)** Default ZINB-WaVE with only sample-level intercept; **(f)** ZINB-WaVE with total number of expressed genes as sample-level covariate. Despite the confounding between patient and batch, the addition of a covariate that captures the batch difference allows ZINB-WaVE to remove the batch effect without removing the patient effect.

over-estimating the dispersion parameter (Figure 2.27).

We also examined the goodness-of-fit of the MAST hurdle model, which is specifically designed for scRNA-seq. However, a direct comparison of MAST with NB and ZINB is cumbersome, due to differences in parameterization. In particular, MAST models log2(TPM+1) rather than counts and does not have a dispersion parameter but only a variance parameter for the Gaussian component (see Methods for details). We found that MAST underestimated the overall mean log2(TPM+1) and over-estimated the zero probability, but had stable variance estimates over the observed proportion of zero counts (Figure 2.28).

By contrast, our ZINB model lead to better fit in terms of both the overall mean count and zero probability, as well as more stable estimators of the dispersion parameter (Figures 2.25 – 2.27).

2.3.7 Estimators are asymptotically unbiased and robust

We next turned to simulations to explore in greater detail the performance of ZINB-WaVE. First, we evaluated the ZINB-WaVE estimation procedure on simulated data from a zero-inflated negative binomial distribution, to assess both accuracy under a correctly specified model and robustness to model misspecification. The approach involves computing maximum likelihood estimators (MLE) for the parameters of the model of Figure 2.1, namely, α, β, γ, and W, and hence μ and π. MLE are asymptotically unbiased and efficient estimators. However, because the likelihood function of our ZINB-WaVE model is not convex, our numerical optimization procedure may converge to a local maximum, rather than to the true MLE (see Methods). We observed that our estimators are asymptotically unbiased and have decreasing variance as the number of cells n increases (Figure 2.29). This suggests that our estimates are not far from the true MLE.

To assess the robustness of the ZINB-WaVE procedure, we examined bias and mean squared error (MSE) for estimators of μ and π, as well as the log-likelihood function, the Akaike information criterion (AIC), and the Bayesian information criterion (BIC), for models with misspecified number of unobserved covariates K (i.e., number of columns of W), gene-level covariate matrix V, and dispersion parameter ϕ (Figures 2.6 and 2.30 – 2.32). In Figure 2.6, the data were simulated with $K = 2$ unknown factors, $X = \mathbf{1}_n$, $V = \mathbf{1}_J$, and genewise dispersion. The model parameters were then estimated with $K = 1, 2, 3, 4$, both for a model that included a cell-level intercept ($V = \mathbf{1}_J$) and one that did not ($V = \mathbf{0}_J$). When the intercept was correctly included in the model, misspecification of K (with $K > 2$) resulted in no or very small bias (Figure 2.6a, b), small MSE (Figure 2.6c, d), and a greater impact on AIC and especially BIC (Figure 2.30). When no intercept was included in the fitted model, the sensitivity to K became more important. However, although the data were simulated with $K = 2$, specifying $K \geq 3$ led to small bias and low MSE (Figure 2.6). This is likely because one column of W acted as a *de facto* intercept, overcoming the absence of

V and explaining why AIC and BIC are minimized at $K = 4$ (Figure 2.30). In addition, we observed robustness of the results to the choice of dispersion parameter (genewise or common), even though the data were simulated with genewise dispersion (Figure 2.6). The estimators for $\ln(\mu)$ and π were unbiased over the whole range of mean expression and zero inflation probability (Figure 2.33).

2.3.8 Accuracy

We next compared ZINB-WaVE to PCA and ZIFA, in terms of their ability to recover the true underlying low-dimensional signal and clustering structure. For datasets simulated from a ZINB model, our estimation procedure with correctly specified K led to almost perfect correlation between distances in the true and estimated low-dimensional space (Figures 2.7a, b). The correlation remained high even for misspecified K, in most cases higher than for PCA and ZIFA. ZINB-WaVE performed consistently well across a range of simulation scenarios, including different numbers of cells n, different zero fractions, and varying cluster tightness (Figure 2.35). We observed a consistent ranking, although noisier, when the methods were compared in terms of silhouette widths (Figures 2.7c, d).

As with the real datasets, the choice of normalization influenced the simulation results. Critically, there was not an overall best normalization method; rather, the performance of the normalization methods depended on the dimensionality reduction method and on intrinsic characteristics of the data, such as the fraction of zero counts and the number and tightness of the clusters (Figure 2.7 and Figure 2.35). For instance, TC normalization worked best for PCA on data simulated from the V1 dataset (Figure 2.7a, c), while FQ and TMM normalization worked best for PCA and ZIFA, respectively, on data simulated from the S1/CA1 dataset (Figure 2.7b, d).

It is important to note that the previous results were obtained for data simulated from the ZINB-WaVE model underlying our estimation procedure. It is hence not surprising that ZINB-WaVE outperformed its competitors. To provide a fairer comparison, we also assessed the methods on data simulated from the model proposed by Lun and Marioni [40]. Although this model is also based on a zero-inflated negative binomial distribution, the distribution is parameterized differently and, in particular, does not involve regression on gene-level and sample-level covariates (see Methods).

When the data were simulated to have a moderate fraction of zeros (namely, 40%), all methods performed well in terms of average silhouette width (Figure 2.7e–g) and precision and recall (Figure 2.36). However, the performance of PCA decreased dramatically with 60% of zeros, independently of the number of cells n. Although ZIFA worked well with 60% or fewer zeros, its performance too decreased at 80% of zeros, especially when only 100 cells were simulated (Figure 2.7e). Conversely, the performance of ZINB-WaVE remained good

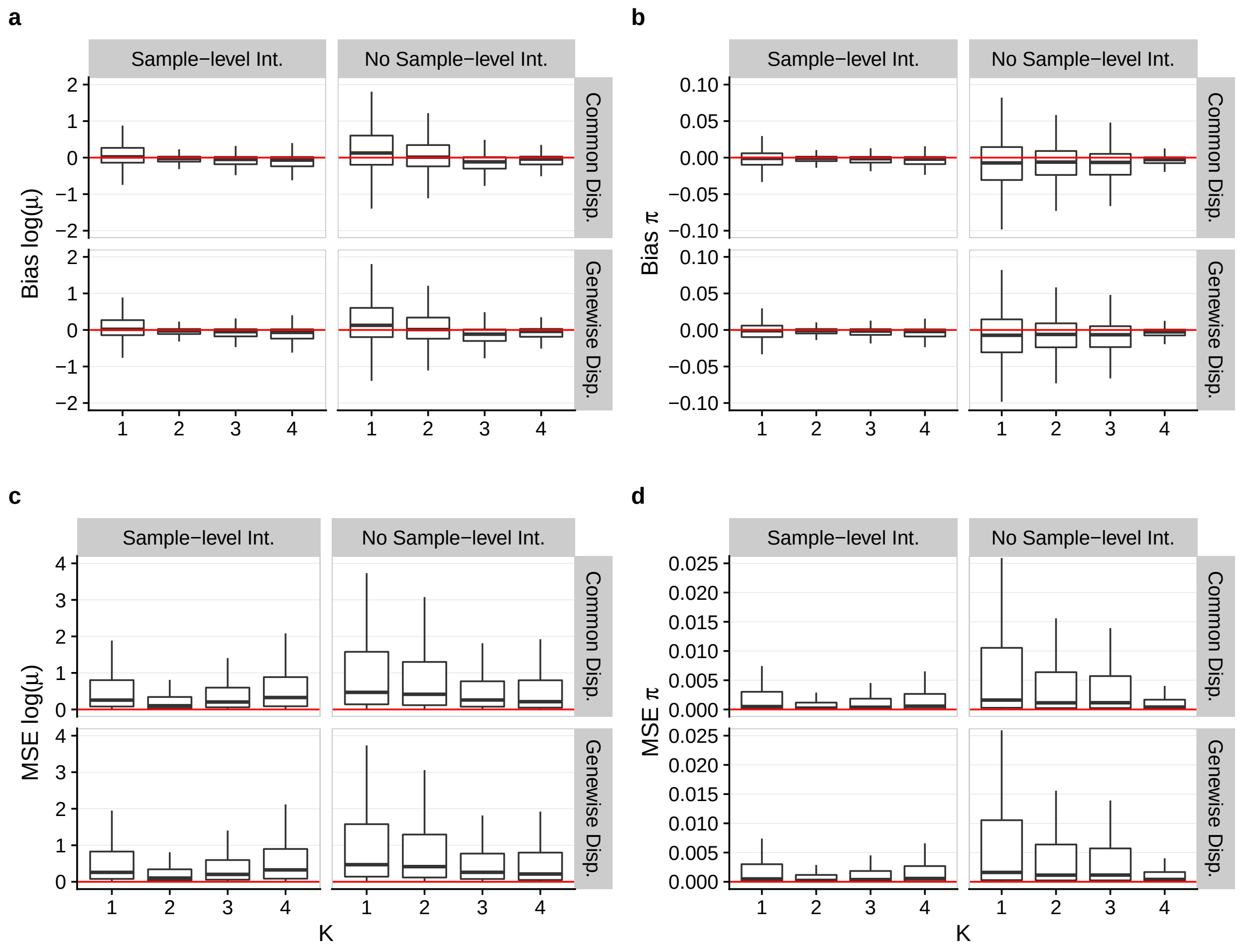


Figure 2.6: *Bias and MSE for ZINB-WaVE estimation procedure: ZINB-WaVE simulation model.* Panels show boxplots of **(a)** bias $\ln(\mu)$, **(b)** bias π, **(c)** MSE $\ln(\mu)$, and **(d)** MSE π for ZINB-WaVE estimation procedure, as a function of the number of unknown covariates K. ZINB-WaVE was fit with $X = \mathbf{1}_n$, common/genewise dispersion, and with/without sample-level intercept (i.e., column of ones in gene-level covariate matrix V). For each gene and cell, bias and MSE were averaged over $B = 10$ datasets simulated from our ZINB-WaVE model, based on the S1/CA1 dataset and with $n = 1,000$ cells, $J = 1,000$ genes, scaling of one for the ratio of within to between-cluster sums of squares ($b^2 = 1$), $K = 2$ unknown factors, zero fraction of about 80%, $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion. The lower and upper hinges of the boxplots correspond to the first and third quartiles (the 25th and 75th percentiles), respectively. The upper/lower whisker extends from the hinge to the largest/smallest value no further than $1.5 \times IQR$ from the hinge (where IQR is the inter-quartile range or difference between the third and first quartiles). Data beyond the whiskers are called outliers and are not plotted here. Figure 2.31 provides the same boxplots with individually plotted outliers.

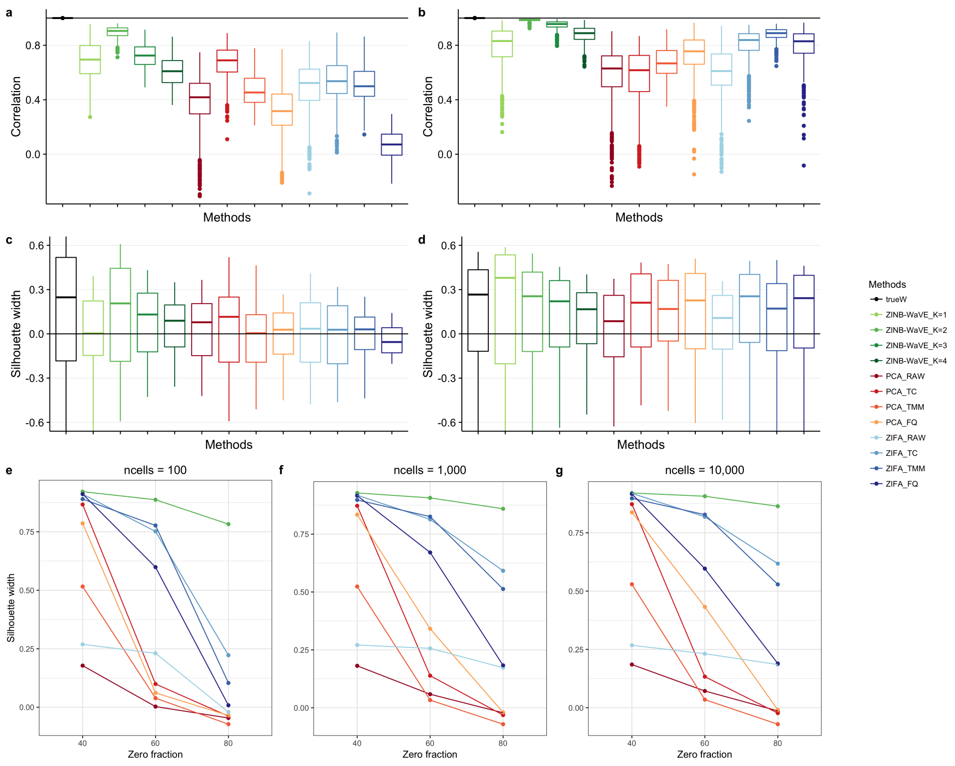


Figure 2.7: *Between-sample distances and silhouette widths for ZINB-WaVE, PCA, and ZIFA: Simulation models.* **(a)** Boxplots of correlations between between-sample distances based on true and estimated low-dimensional representations of the data for simulations based on the V1 dataset. **(b)** Same as **(a)** for simulations based on the S1/CA1 dataset. **(c)** Boxplots of silhouette widths for true clusters for simulations based on the V1 dataset. **(d)** Same as **(c)** for simulations based on the S1/CA1 dataset. For **(a–d)**, all datasets were simulated from our ZINB-WaVE model with $n = 1,000$ cells, $J = 1,000$ genes, "harder" clustering ($b^2 = 5$), $K = 2$ unknown factors, zero fraction of about 80%, $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion. Each boxplot is based on n values corresponding to each of the n samples and defined as averages of correlations **(a, b)** or silhouette widths **(c, d)** over $B = 10$ simulations. See Figure 2.34 for the same scenario but with $n = 10,000$ cells and Figure 2.35 for additional scenarios. **(e–g)** Average silhouette widths (over n samples and $B = 10$ simulations) for true clusters vs. zero fraction, for $n \in \{100, 1,000, 10,000\}$ cells, for datasets simulated from the Lun and Marioni [40] model, with $C = 3$ clusters and equal number of cells per cluster. While ZINB-WaVE was relatively robust to the sample size n and zero fraction, the performance of PCA and ZIFA decreased with larger zero fraction. Between-sample distance matrices and silhouette widths were based on W for ZINB-WaVE, the first two principal components for PCA, and the first two latent variables for ZIFA. ZINB-WaVE was applied with $X = \mathbf{1}_n$, $V = \mathbf{1}_J$, genewise dispersion, and $K \in \{1, 2, 3, 4\}$ (only $K = 2$ is shown in **(e–g)**). For PCA and ZIFA, different normalization methods were used. Colors correspond to the different methods.

even when simulating data with 80% of zeros, independently of the sample size.

2.4 Discussion

Recent advances in single-cell technologies, such as droplet-based methods [10], make it easy and inexpensive to collect hundreds of thousands of scRNA-seq profiles, allowing researchers to study very complex biological systems at high resolution. The resulting libraries are often sequenced at extremely low depth (tens of thousands of reads per cell, only), making the corresponding read count data truly sparse. Hence, there is a growing need for developing reliable statistical methods that are scalable and that can account for zero inflation.

ZINB-WaVE is a general and flexible approach to extract low-dimensional signal from noisy, zero-inflated data, such as those from scRNA-seq experiments. We have shown with simulated and real data analyses that ZINB-WaVE leads to robust and unbiased estimators of the underlying biological signals. The better performance of ZINB-WaVE with respect to PCA comes at a computational cost, since we need to numerically optimize a non-convex likelihood function. However, we empirically found that the computing time was approximately linear in both the number of cells and the number of genes, and approximately quadratic in the number of latent factors (Figure 2.37). The algorithm benefits from parallelization on multicore machines and takes a few minutes on a modern laptop to converge for thousands of cells.

One major difference between ZINB-WaVE and previously proposed factor analysis models (such as PCA and ZIFA) is the ability to include sample-level and gene-level covariates. In particular, by including a column of ones in the gene-level covariate matrix, the corresponding cell-level intercept acts as a global-scaling normalization factor, allowing the modeling of raw count data, with no need for prior normalization.

However, there is no guarantee that the low-dimensional signal extracted by ZINB-WaVE is biologically relevant: If unwanted technical variation affects the data and is not accounted for in the model (or in prior normalization), the low-rank matrix W inferred by ZINB-WaVE will capture such confounding effects. It is therefore important to explore the correlation between the latent factors estimated by our procedure and known measures of quality control that can be computed for scRNA-seq libraries, using, for instance, the Bioconductor R package `scater` [62] (see Figure 2.2f). If one observes high correlation between one or more latent factors and some QC measures, it may be beneficial to include these QC measures as covariates in the model.

Several authors have recognized that high-dimensional genomic data are affected by a variety of unwanted technical effects (e.g., batch effects) that can be confounded with the biological signal of interest, and have proposed methods to account for such effects in either a supervised [59] or unsupervised way [35, 63]. Recently, Lin et al. [64] proposed a model that can extend PCA to adjust for confounding factors. This model, however, does not seem to be ideal for zero-inflated count data. In the scRNA-seq literature, MAST [31] uses the inferred cellular detection rate to adjust for the main source of confounding, in a differential expression setting, but is not designed to infer low-dimensional signal.

The removal of batch effects is an important example of how including additional covariates in the ZINB-WaVE model may lead to better low-dimensional representations of the data. However, ZINB-WaVE is not limited to including batch effects, as other sample-level (e.g., QC metrics) and/or gene-level (e.g., GC-content) covariates may be included in the model. Although we did not find any compelling examples in which adding a gene-level covariate leads to improve signal extraction, it is interesting to note the relationship between GC-content and batch effects [65]. With large collaborative efforts, such as the Human Cell Atlas [66], on the horizon, we anticipate that the ability of our model to include gene-level covariates that can potentially help accounting for differences in protocols will prove important.

Although the low-dimensional signal inferred by ZINB-WaVE can be used to visually inspect hidden structure in the data, visualization is not the main point of our proposed method. The low-dimensional factors are intended to be the closest possible approximation to the true signal, which is assumed to be intrinsically low-dimensional. Such a low-dimensional representation can be used in downstream analyses, such as clustering or pseudotime ordering of the cells [23].

Visualization of high-dimensional datasets is an equally important area of research and many algorithms are available, among which t-SNE [33] has become the most popular for scRNA-seq data. Recently, Wang et al. [67] have proposed a novel visualization algorithm that can account for zero inflation and showed improvement over t-SNE. As t-SNE takes as input a matrix of cell pairwise distances, which may be noisy in high dimensions, a typical pipeline involves computing such distances in PCA space, selecting, for example, the first 50 PCs. An alternative approach is to derive such distances from the low-dimensional space defined by the factors inferred by ZINB-WaVE. This strategy was used effectively in Figure 2.3c to visualize the PBMC dataset.

In this article, we have focused on an unsupervised setting, where the goal is to extract a low-dimensional signal from noisy zero-inflated data. However, our proposed ZINB model is more general and can be used, in principle, for supervised differential expression analysis, where the parameters of interest are regression coefficients β corresponding to known sample-level covariates in the matrix X (e.g., cell type, treatment/control status). Differentially expressed genes may be identified via likelihood ratio tests or Wald tests, with standard errors of estimators of β obtained from the Hessian matrix of the likelihood function. Additionally, posterior dropout probabilities can be readily derived from the model and used as weights to unlock standard bulk RNA-seq methods [68], such as edgeR [53]. We envision a future version of the zinbwave package with this added capability.

2.5 Supplement

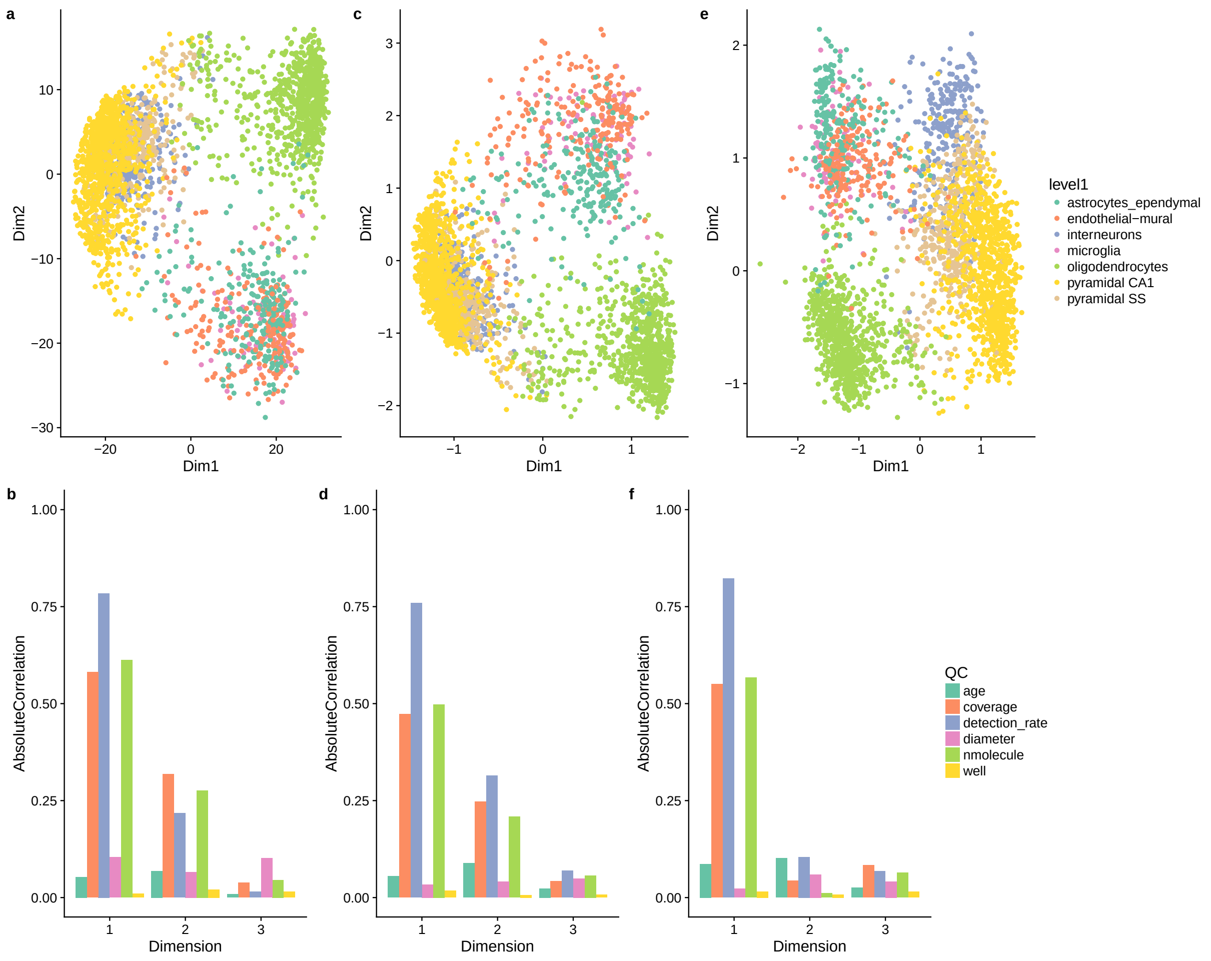


Figure 2.8: *Low-dimensional representation of the S1/CA1 dataset.* Upper panels provide two-dimensional representations of the data. Lower panels provide barplots of the absolute correlation between the first three components and a set of QC measures (see Methods). **(a, b)** PCA (on TC-normalized counts); **(c, d)** ZIFA (on TC-normalized counts); **(e, f)** ZINB-WaVE (no normalization needed). ZINB-WaVE leads to a low-dimensional representation that is less influenced by technical variation and to tighter, biologically meaningful clusters.

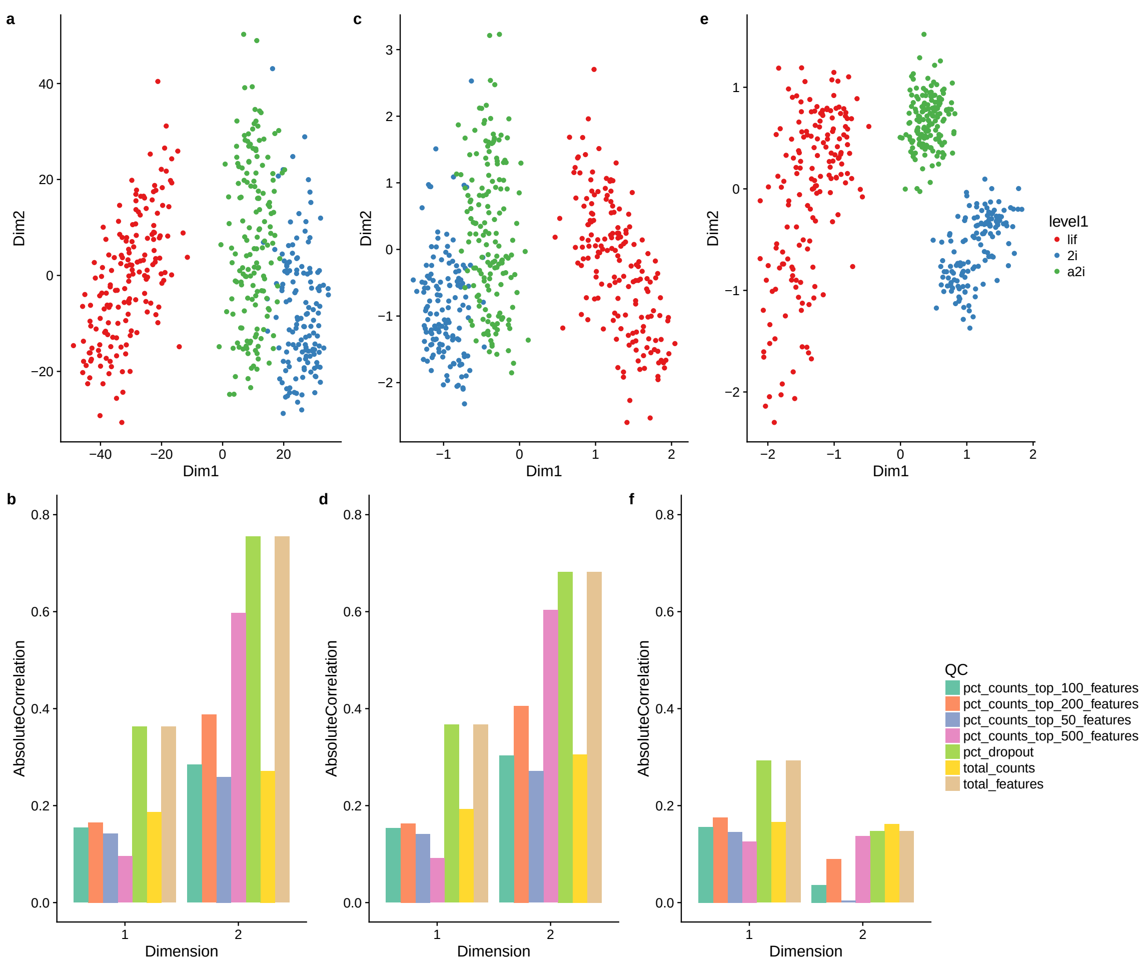


Figure 2.9: *Low-dimensional representation of the mESC dataset.* Upper panels provide two-dimensional representations of the data, after selecting the 1,000 most variable genes. Lower panels provide barplots of the absolute correlation between the first two components and a set of QC measures (see Methods). **(a, b)** PCA (on TC-normalized counts); **(c, d)** ZIFA (on TC-normalized counts); **(e, f)** ZINB-WaVE (no normalization needed). ZINB-WaVE leads to a low-dimensional representation that is less influenced by technical variation and to tighter, biologically meaningful clusters.

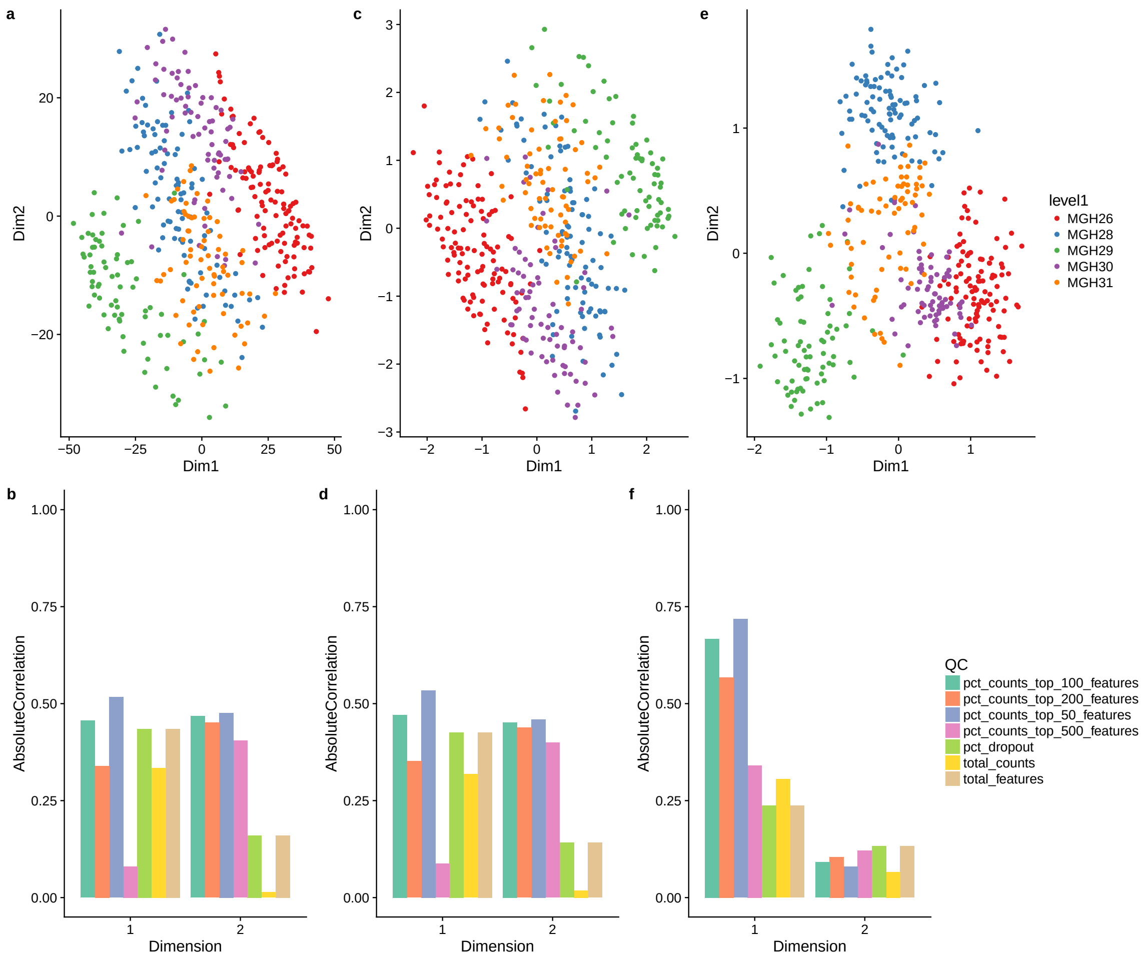


Figure 2.10: *Low-dimensional representation of the Glioblastoma dataset.* Upper panels provide two-dimensional representations of the data, after selecting the 1,000 most variable genes. Lower panels provide barplots of the absolute correlation between the first two components and a set of QC measures (see Methods). **(a, b)** PCA (on TC-normalized counts); **(c, d)** ZIFA (on TC-normalized counts); **(e, f)** ZINB-WaVE (no normalization needed). ZINB-WaVE leads to a low-dimensional representation that is less influenced by technical variation and to tighter, biologically meaningful clusters.

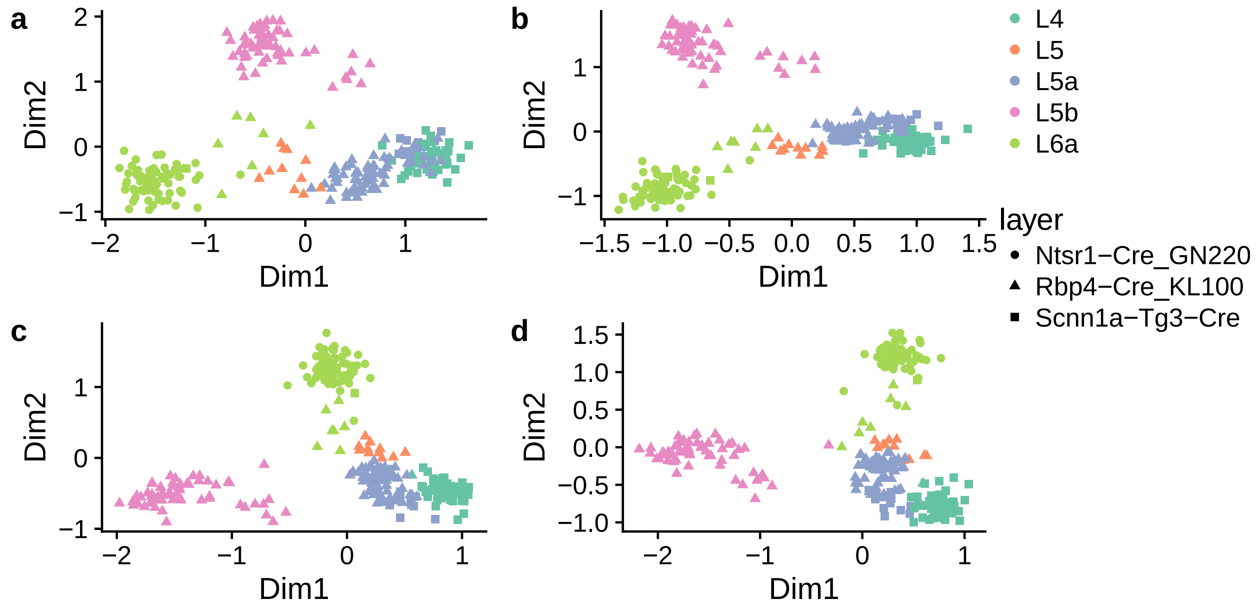


Figure 2.11: *Low-dimensional representation of the V1 dataset.* ZINB-WaVE two-dimensional representation of the data, after selecting the **(a)** 500, **(b)** 2,000, **(c)** 5,000, **(d)** 10,000 most variable genes. ZINB-WaVE leads to a stable low-dimensional representation, robust to the number of highly variable genes selected.

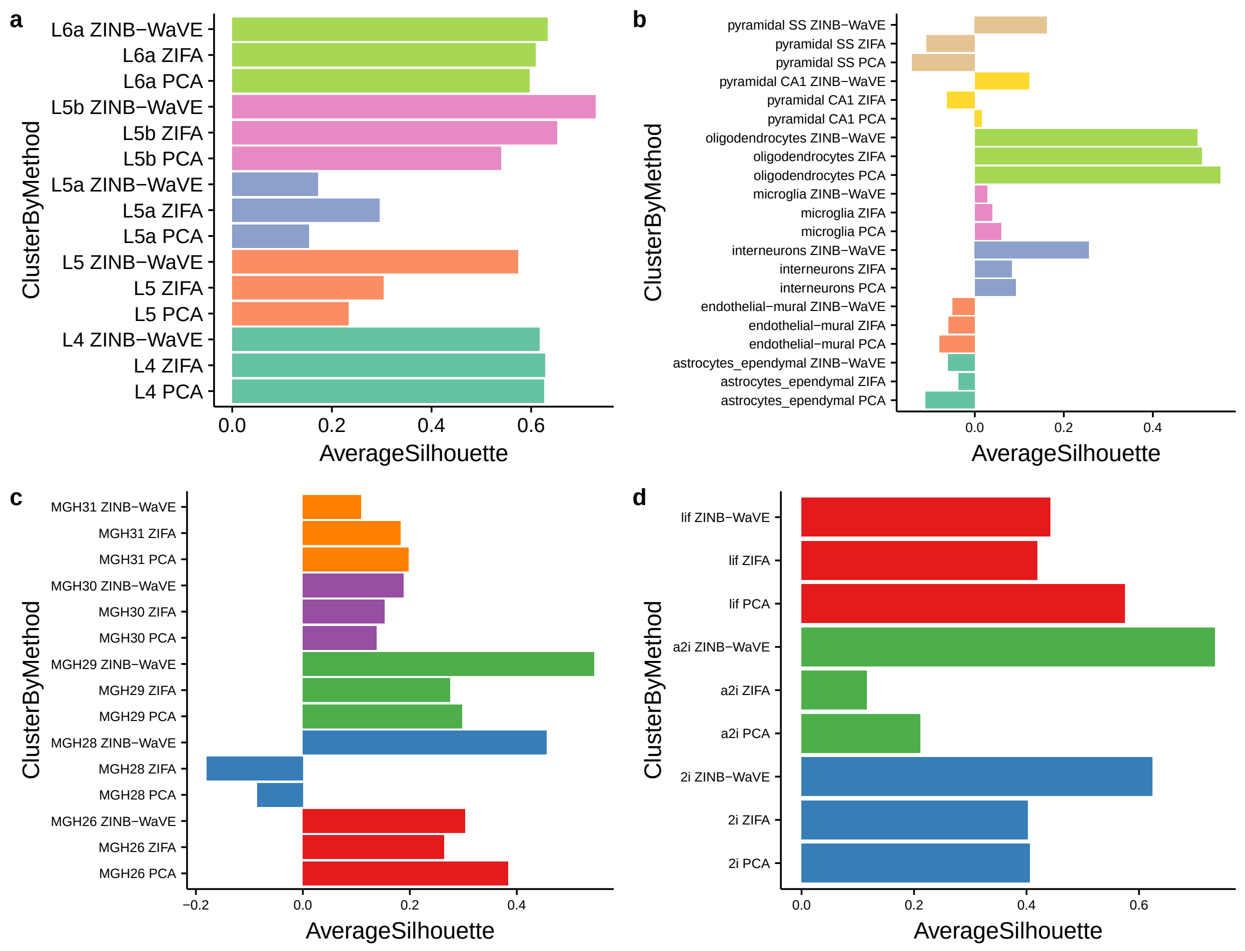


Figure 2.12: *Per-cluster average silhouette widths: Real datasets.* **(a)** V1 dataset; **(b)** S1/CA1 dataset; **(c)** Glioblastoma dataset; **(d)** mESC dataset. For each of the four scRNA-seq datasets of Figures 2.2 and 2.8 – 2.10, barplots of the per-cluster average silhouette widths for ZINB-WaVE, ZIFA, and PCA (the best normalization method was used for ZIFA and PCA). Silhouette widths were computed in the low-dimensional space, using the groupings provided by the authors of the original publications: unsupervised clustering procedure **(a–b)**, observed characteristics of the samples, such as patient **(c)** and culture condition **(d)**
.

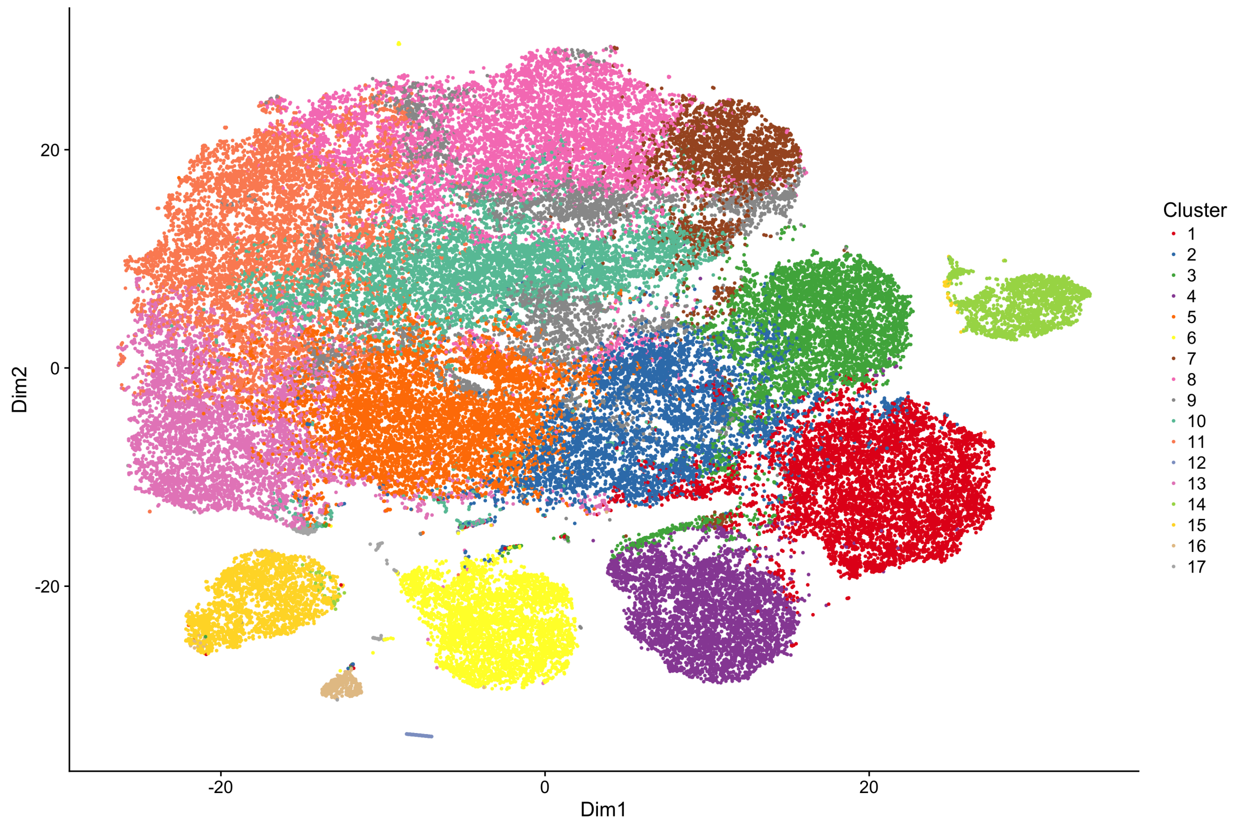


Figure 2.13: *Analysis of the 10x Genomics 68k PBMCs dataset.* Two-dimensional t-SNE representation of W ($K = 10$) color-coded by sequential k-means clustering (see Methods for details on the clustering procedure).

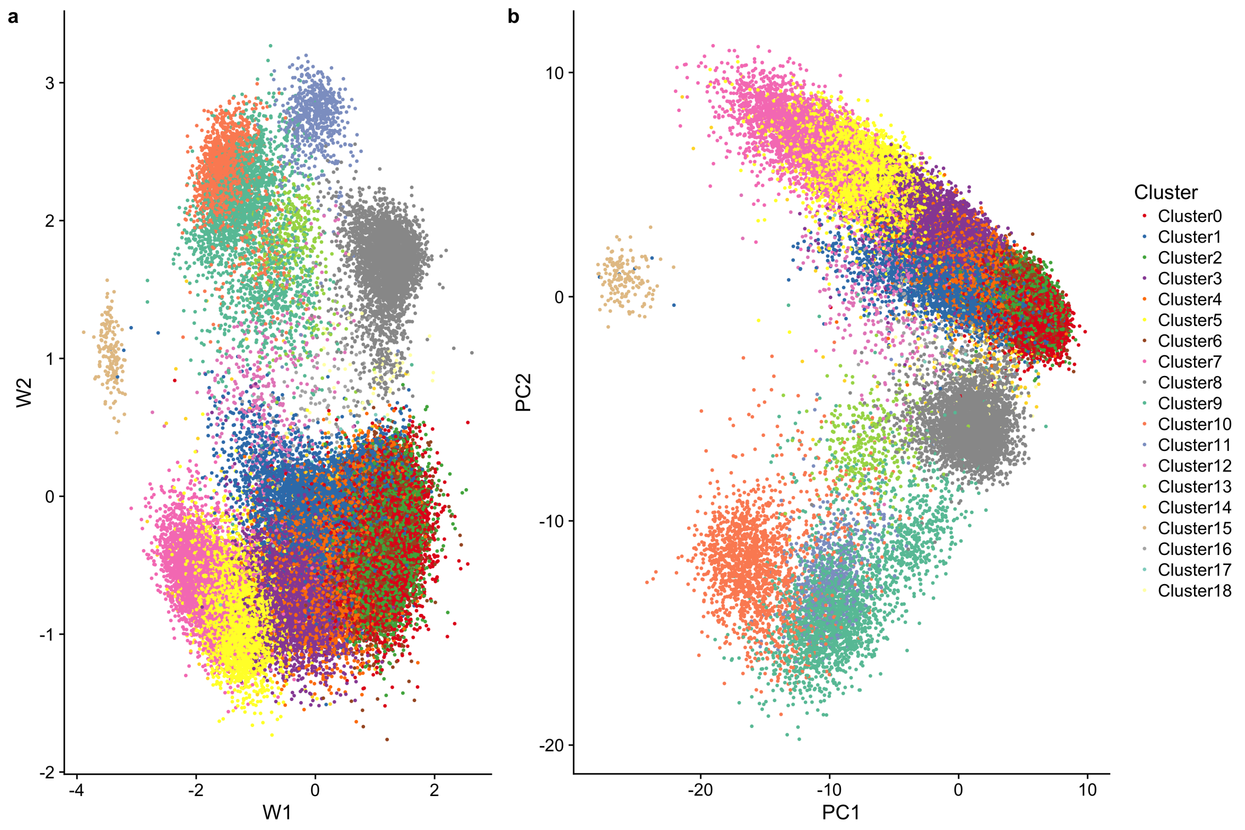


Figure 2.14: *Analysis of the 10x Genomics 68k PBMCs dataset.* **(a)** Two-dimensional signal inferred using ZINB-WaVE. **(b)** First two principal components. Cells are color-coded by sequential k-means clustering (see Methods for details on the clustering procedure).

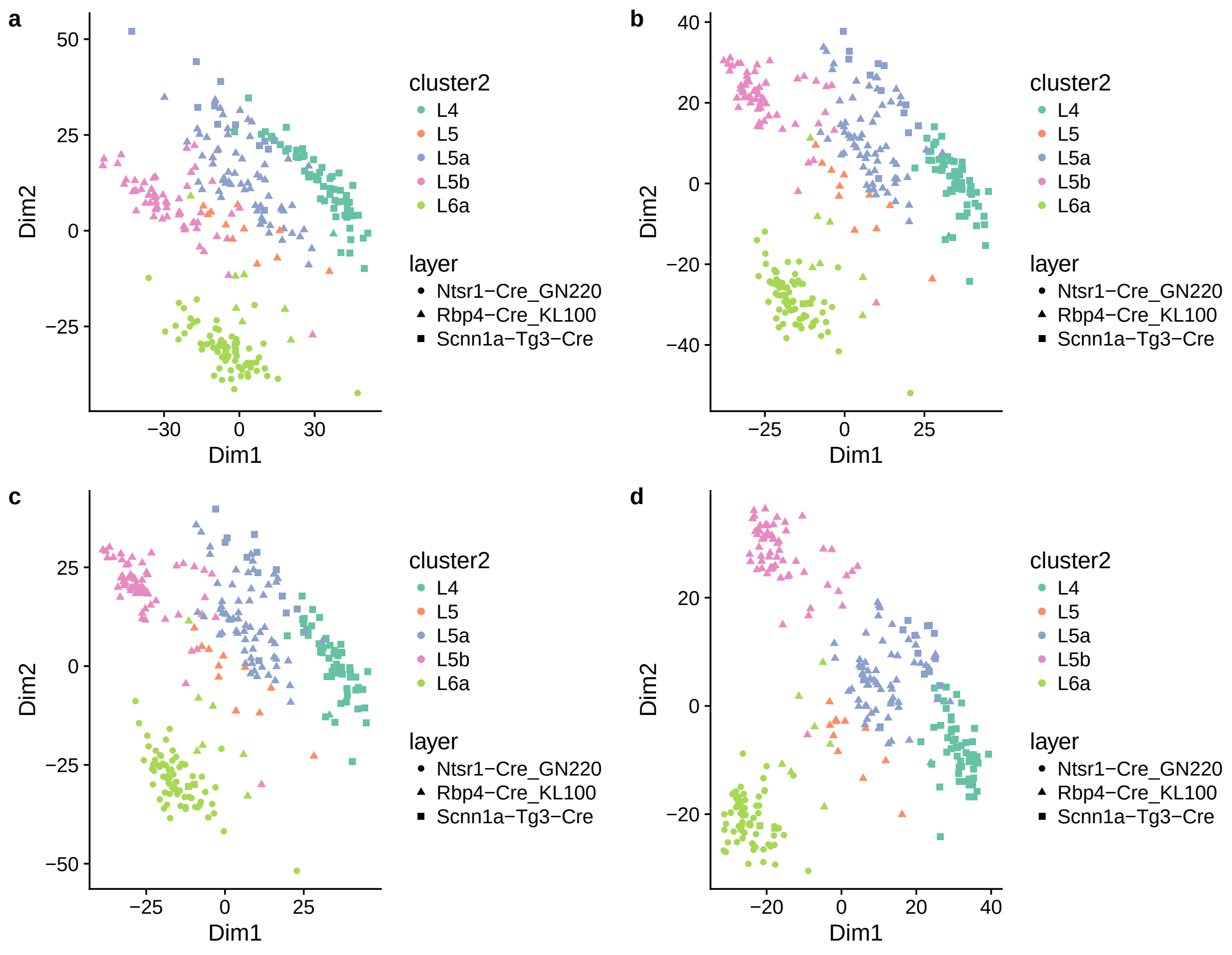


Figure 2.15: *Principal component analysis for V1 dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

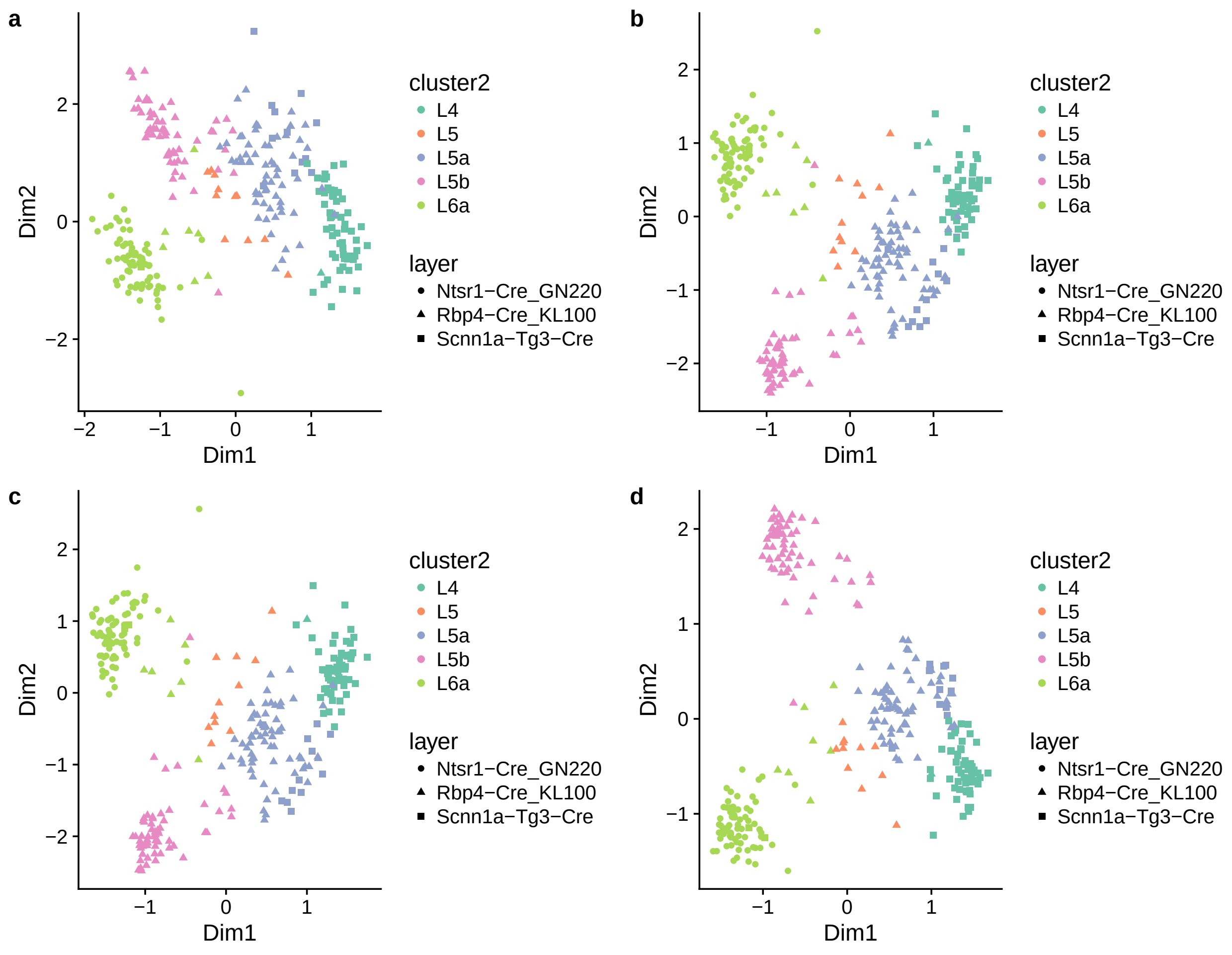


Figure 2.16: *Zero-inflated factor analysis for V1 dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

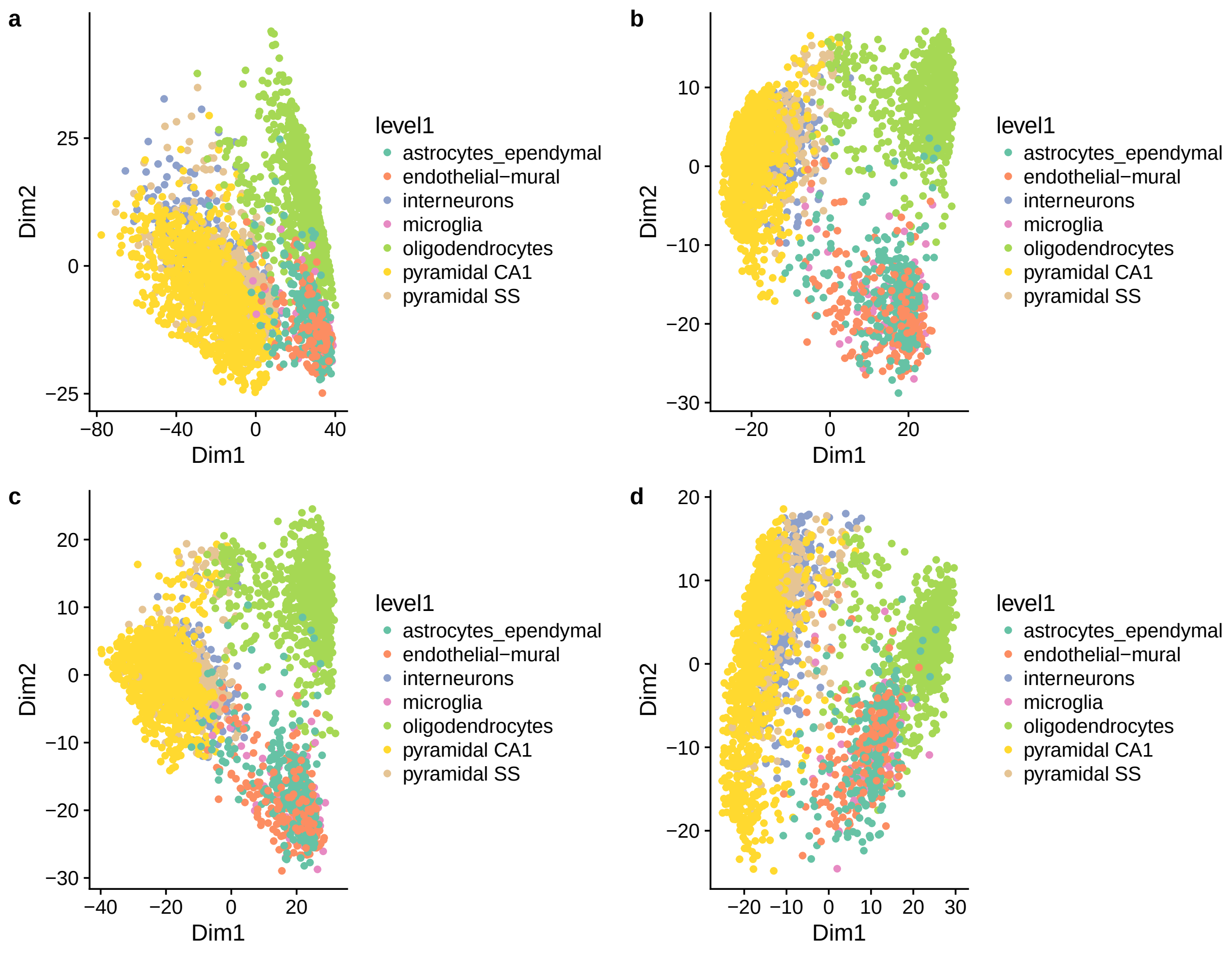


Figure 2.17: *Principal component analysis for S1/CA1 dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

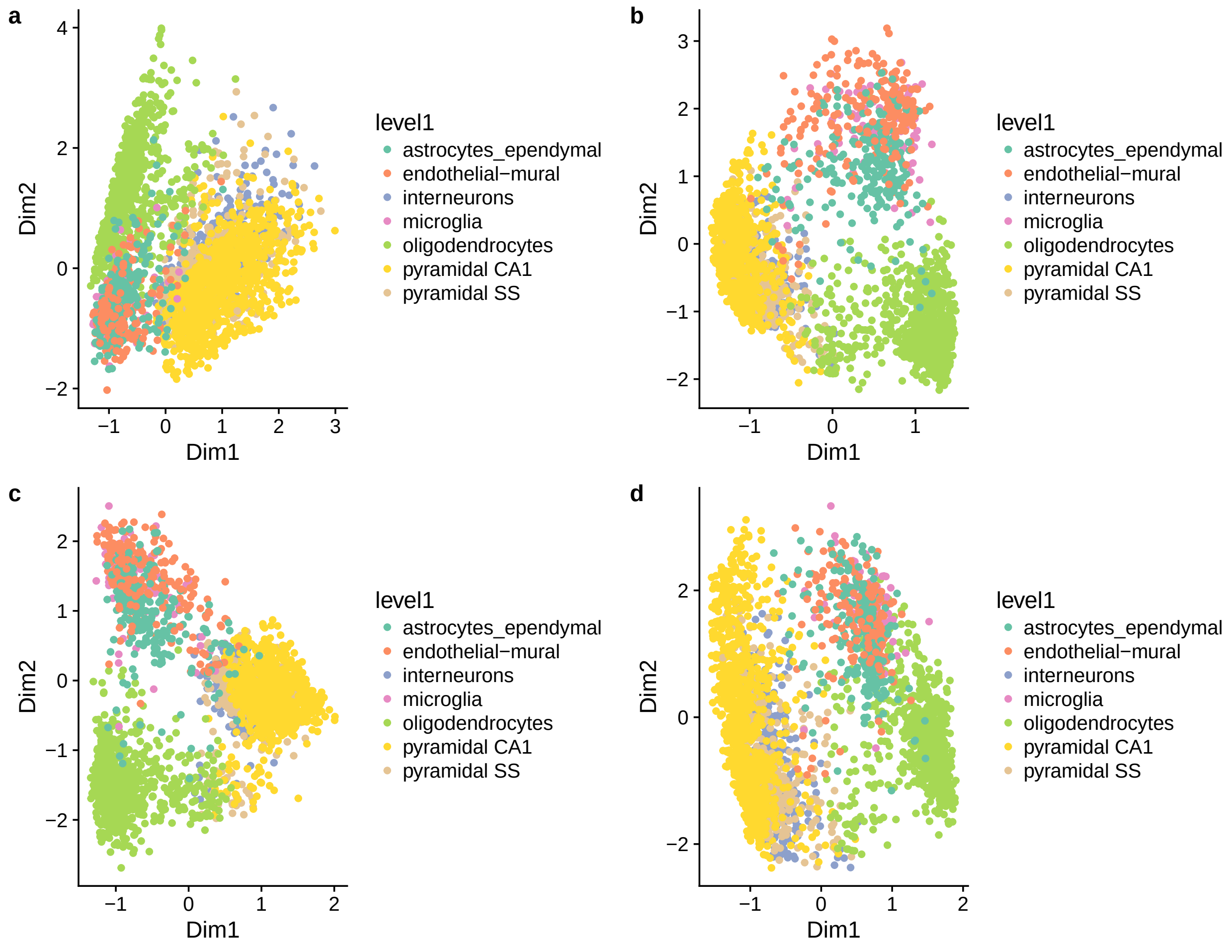


Figure 2.18: *Zero-inflated factor analysis for S1/CA1 dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

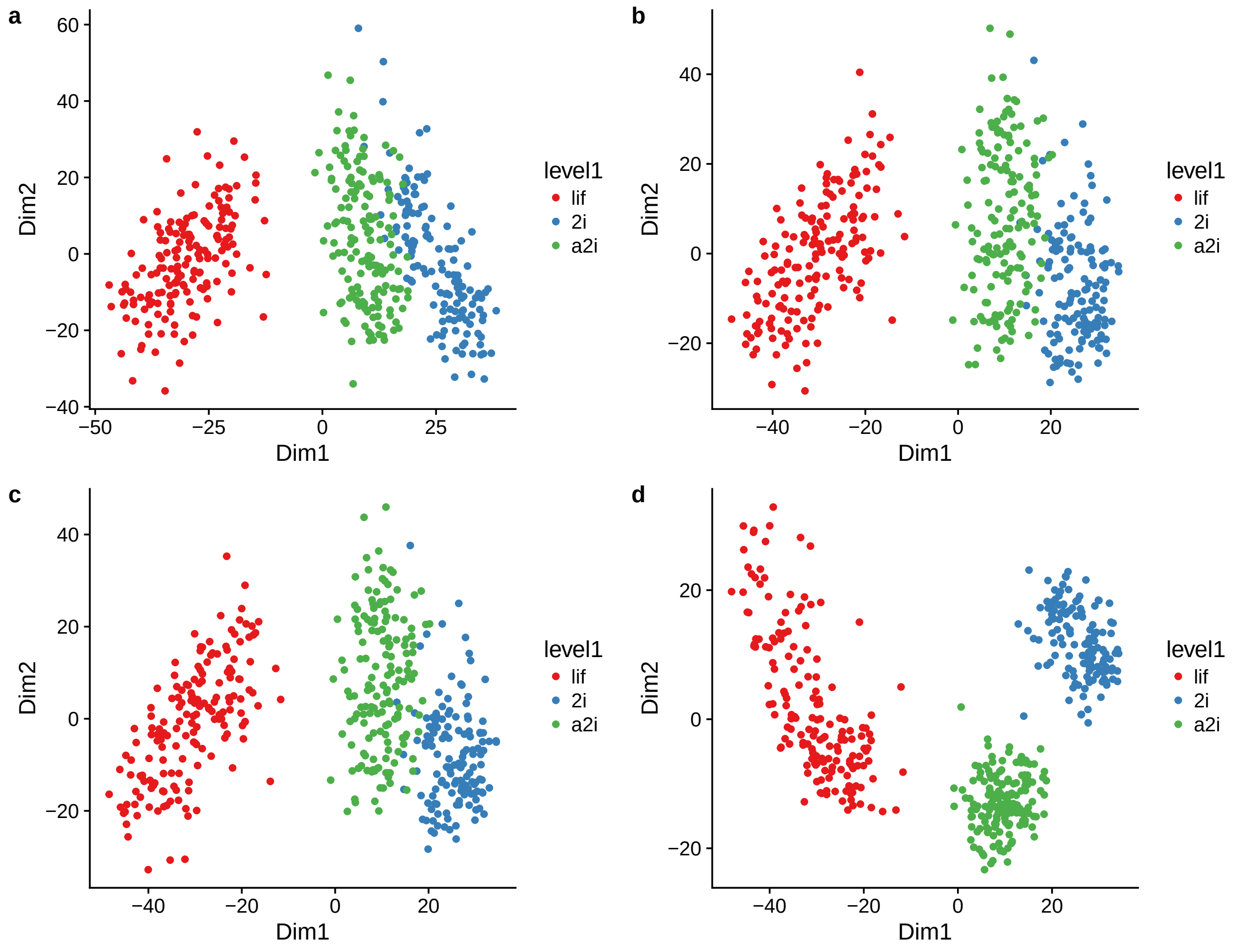


Figure 2.19: *Principal component analysis for mESC dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

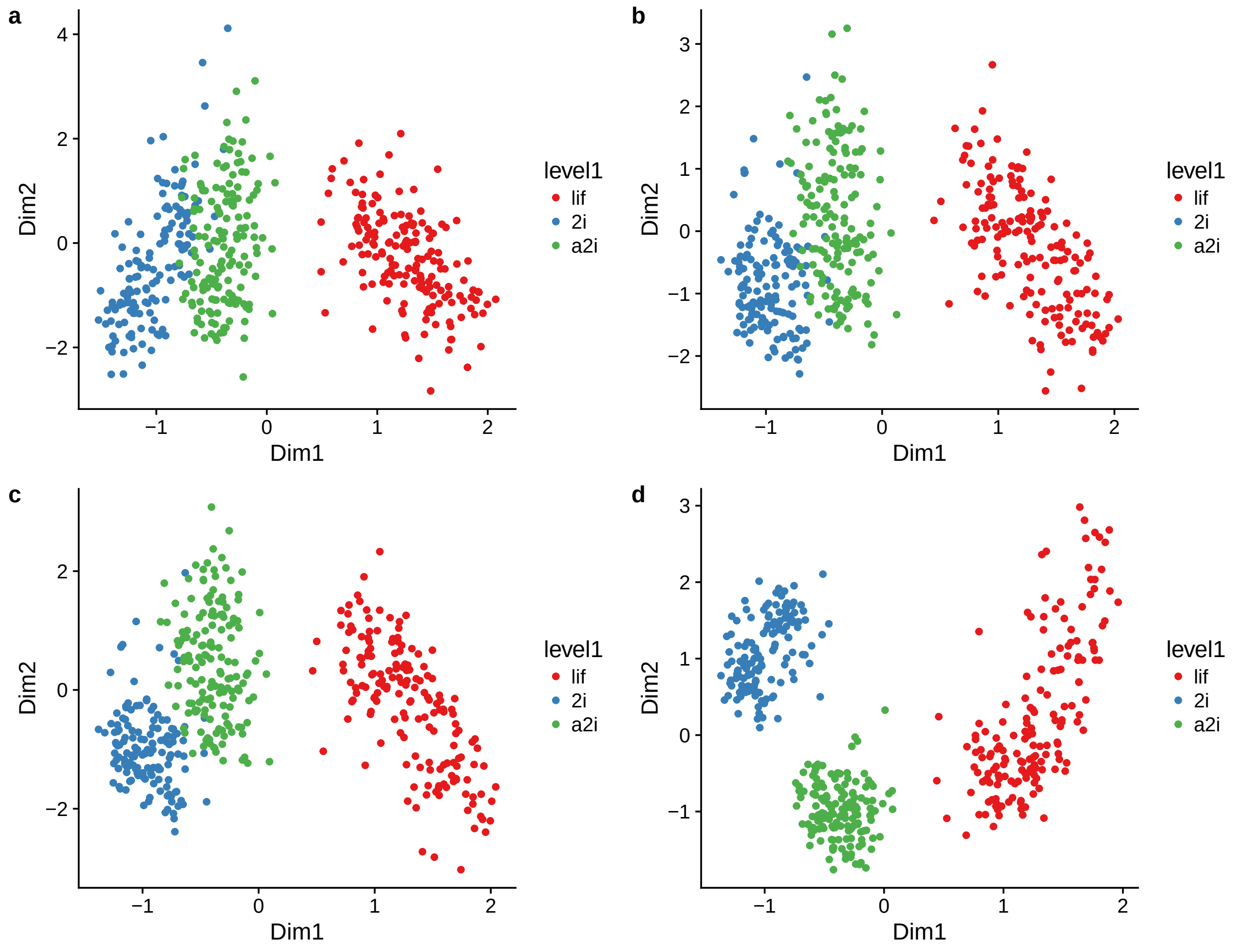


Figure 2.20: *Zero-inflated factor analysis for mESC dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

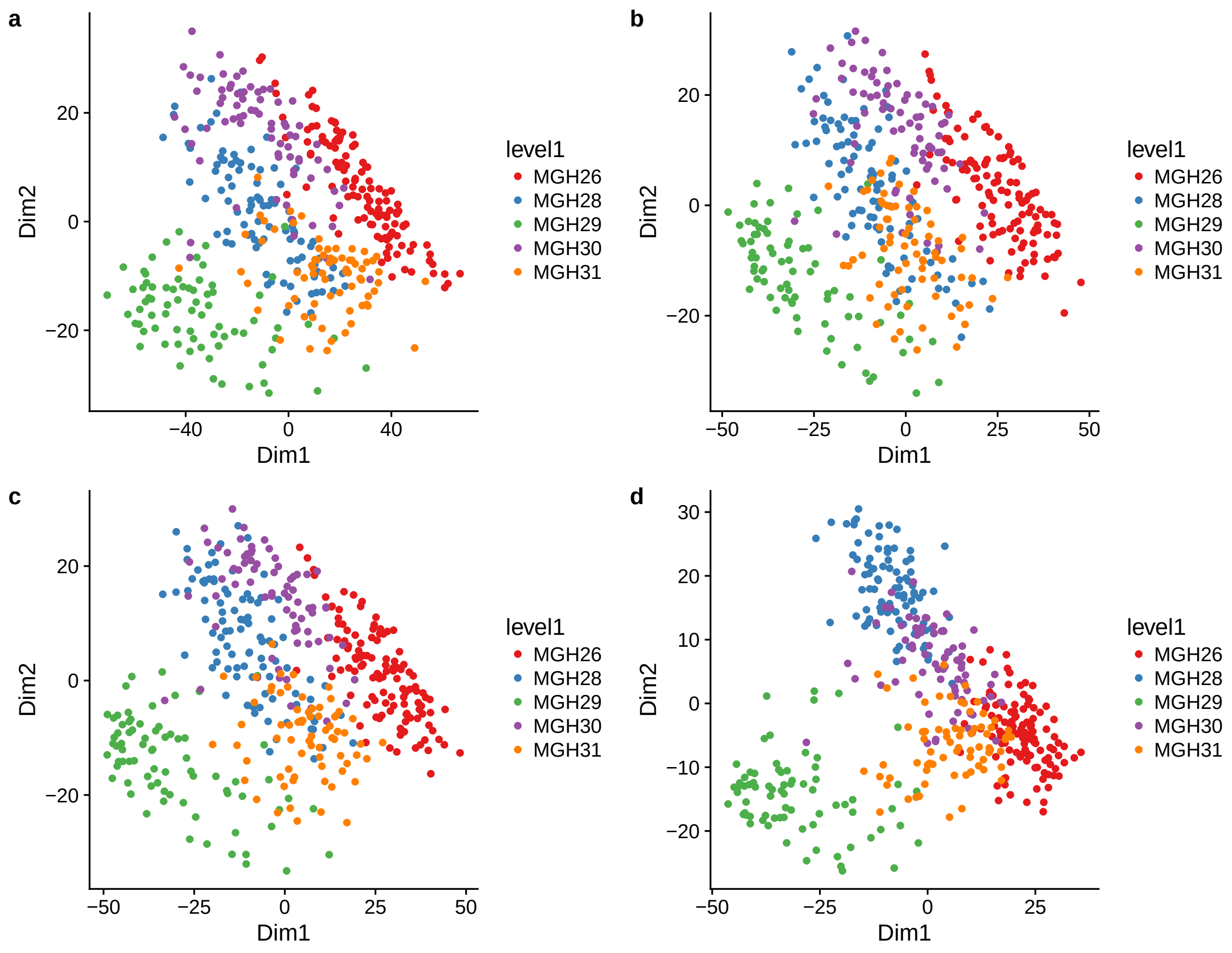


Figure 2.21: *Principal component analysis for Glioblastoma dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

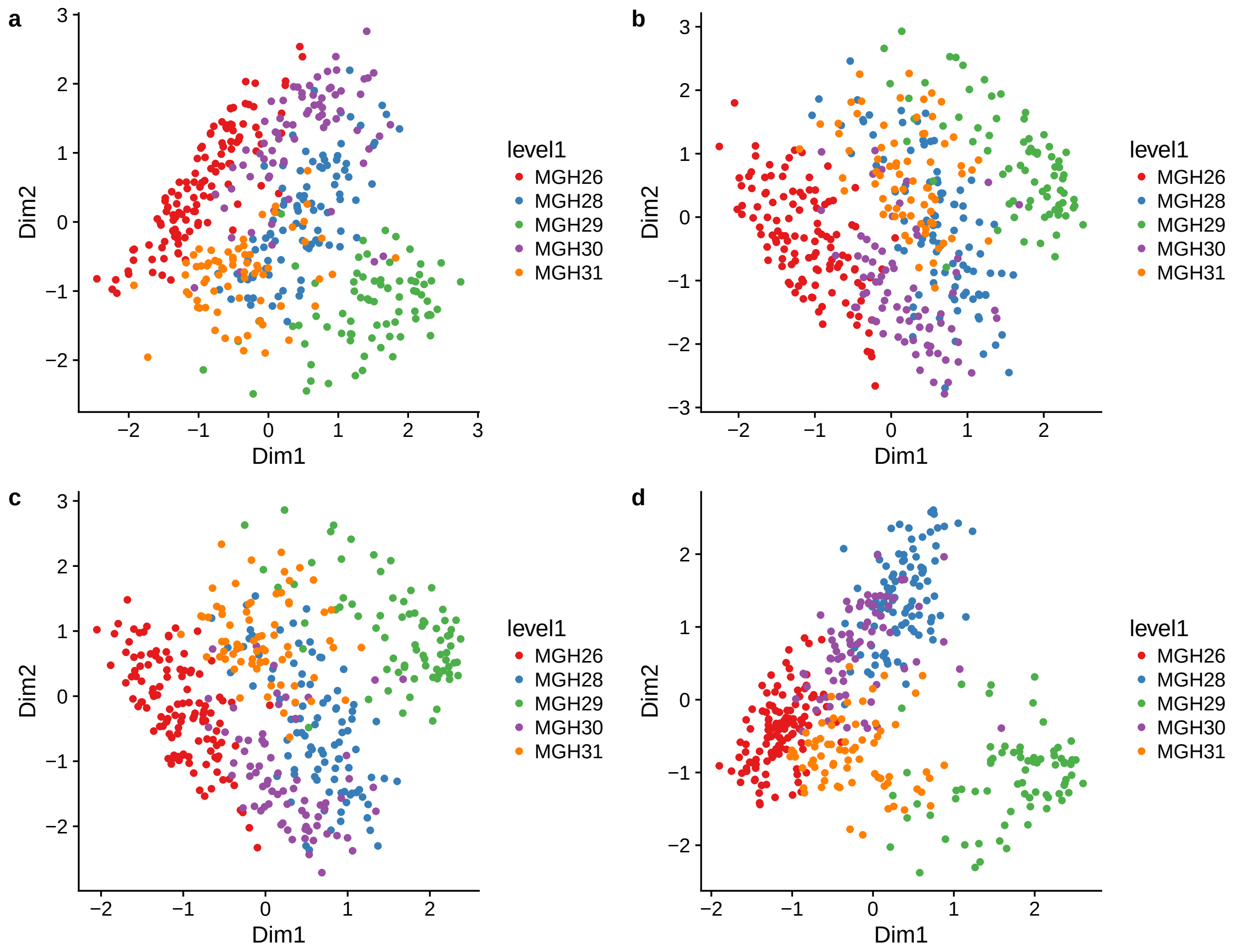


Figure 2.22: *Zero-inflated factor analysis for Glioblastoma dataset.* **(a)** No normalization; **(b)** TC normalization; **(c)** TMM normalization; **(d)** FQ normalization.

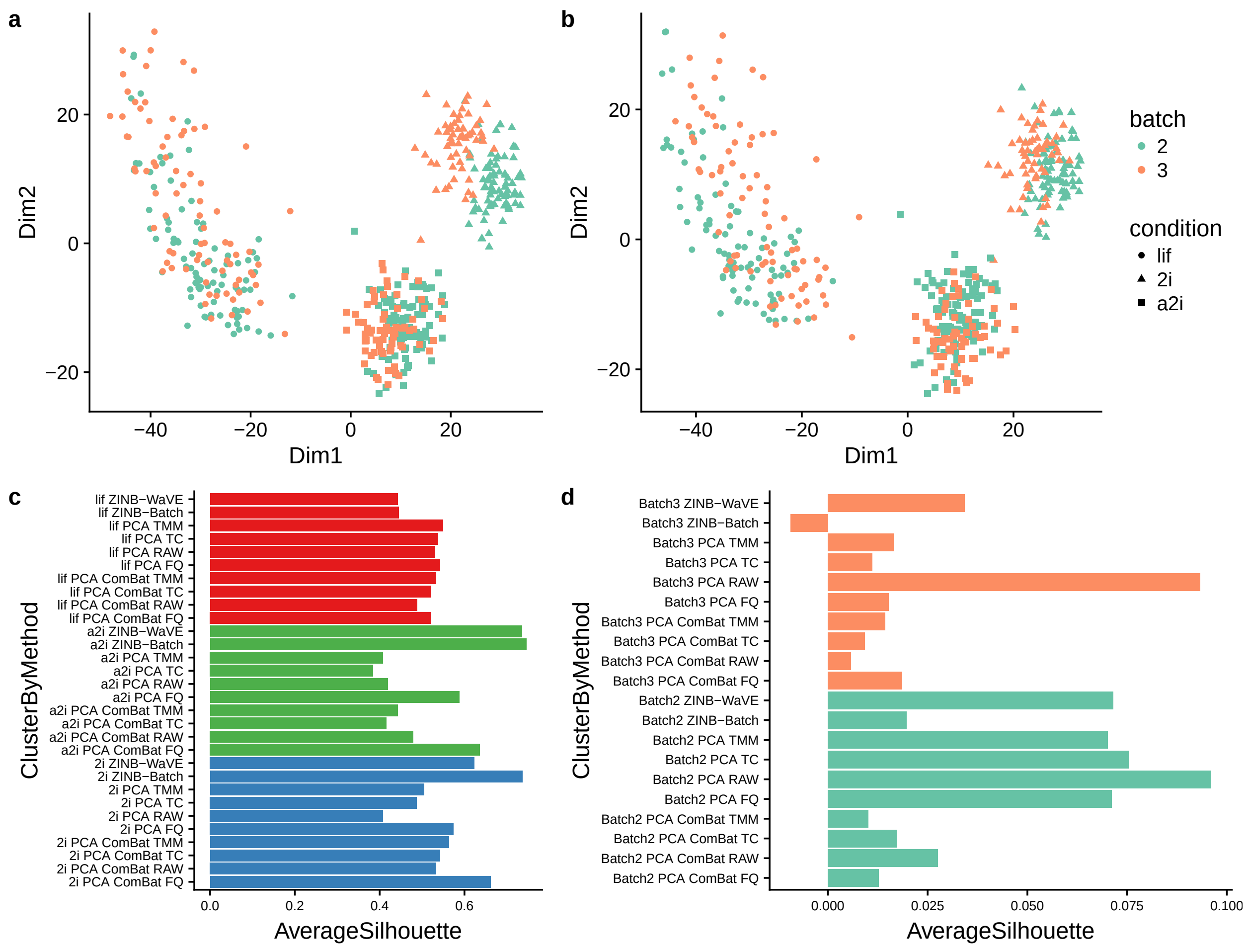


Figure 2.23: *ZINB-Wave and ComBat: mESC dataset.* Upper panels provide two-dimensional representations of the data, with cells color-coded by batch and shape reflecting culture conditions: **(a)** PCA on FQ-normalized counts; **(b)** PCA on ComBat-normalized counts. **(c)** Average silhouette widths by biological condition for ZINB-WaVE with and without batch covariate, PCA with and without applying ComBat on raw counts and TC, TMM, and FQ-normalized counts; **(d)** Average silhouette widths by batch for ZINB-WaVE with and without batch covariate, PCA with and without applying ComBat on raw counts and TC, TMM, and FQ-normalized counts.

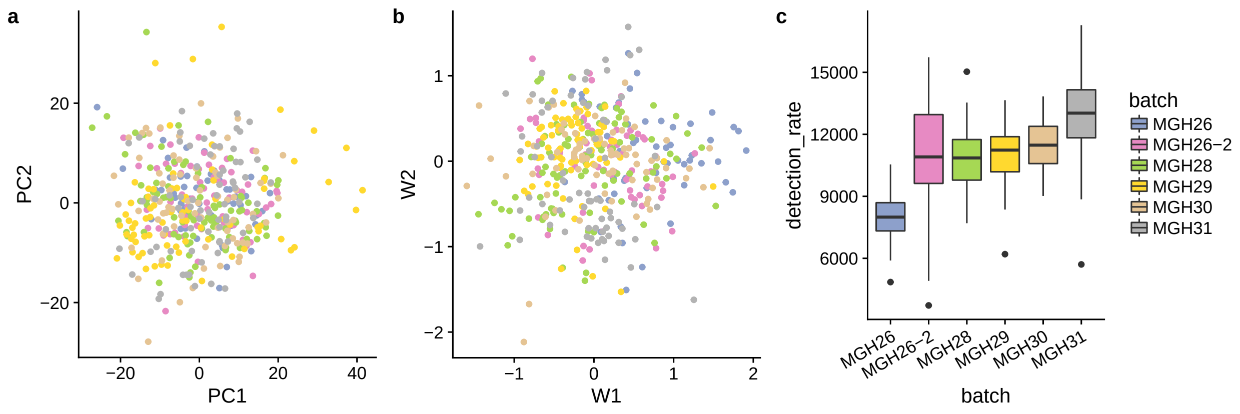


Figure 2.24: *ZINB-Wave and ComBat: Glioblastoma dataset.* **(a)** PCA on FQ + ComBat-normalized counts; **(b)** ZINB-WaVE with batch as sample covariate; **(c)** Boxplot of sample detection rate stratified by batch. Sample detection rate is defined as the total number of genes with at least one read in a given sample.

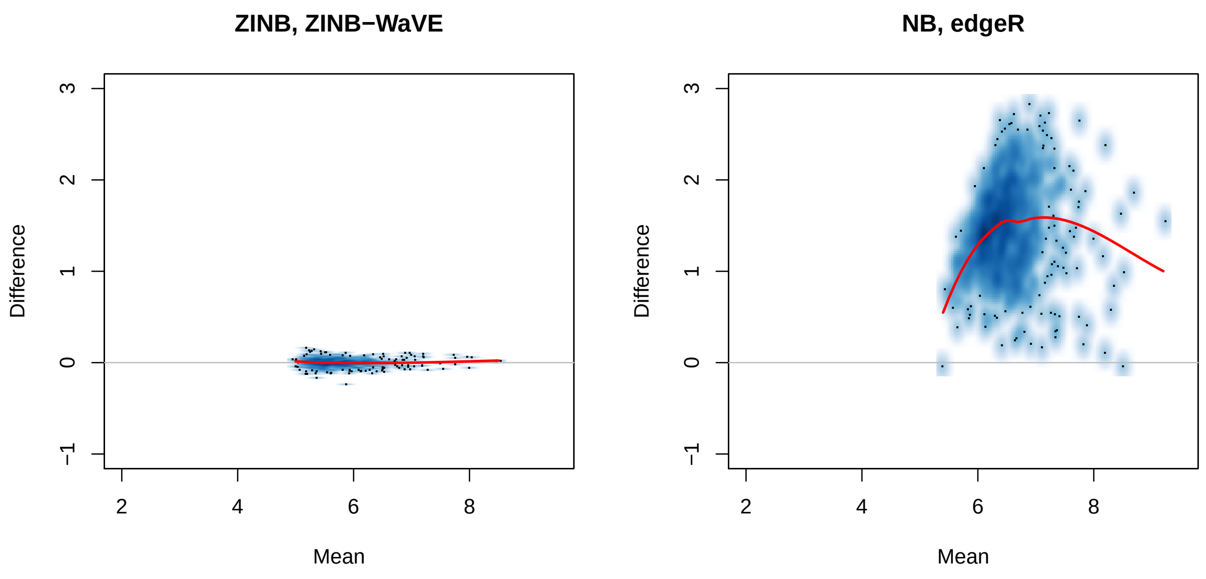


Figure 2.25: *Goodness-of-fit of ZINB-WaVE and NB models: Mean-difference plots of estimated vs. observed mean count for V1 dataset.* Left panel: ZINB-WaVE. Right panel: Negative binomial model fit using edgeR package. Observed and estimated mean counts were averaged over n cells. Counts were plotted on a log scale. See Methods for details.

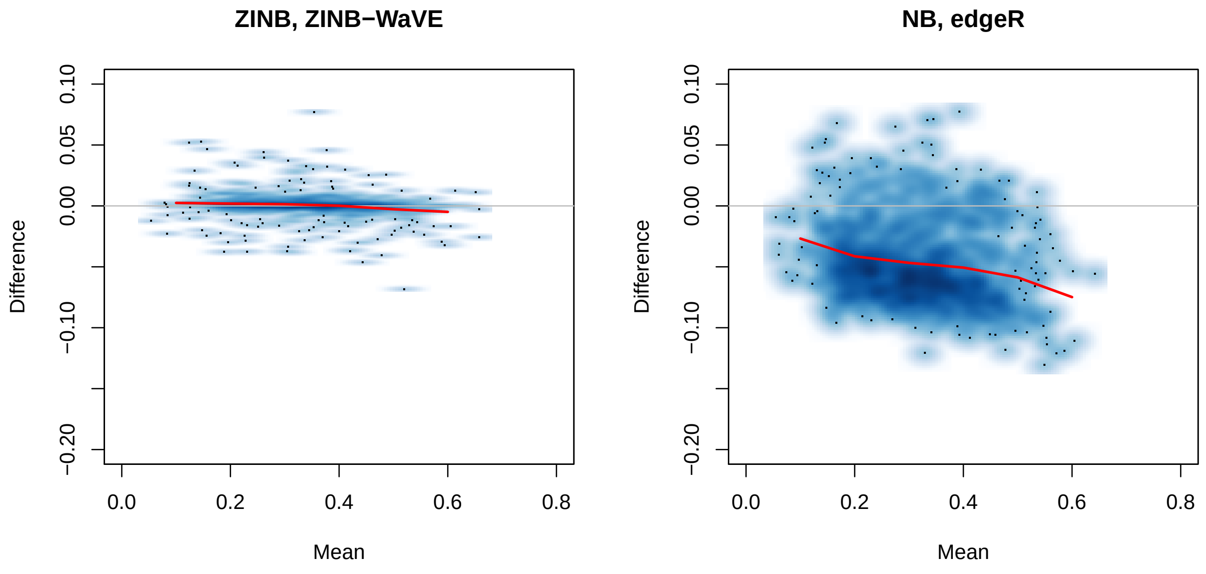


Figure 2.26: *Goodness-of-fit of ZINB-WaVE and NB models: Mean-difference plots of estimated vs. observed zero probability for V1 dataset.* Left panel: ZINB-WaVE. Right panel: Negative binomial model fit using edgeR package. Observed and estimated zero probabilities were averaged over n cells. See Methods for details.

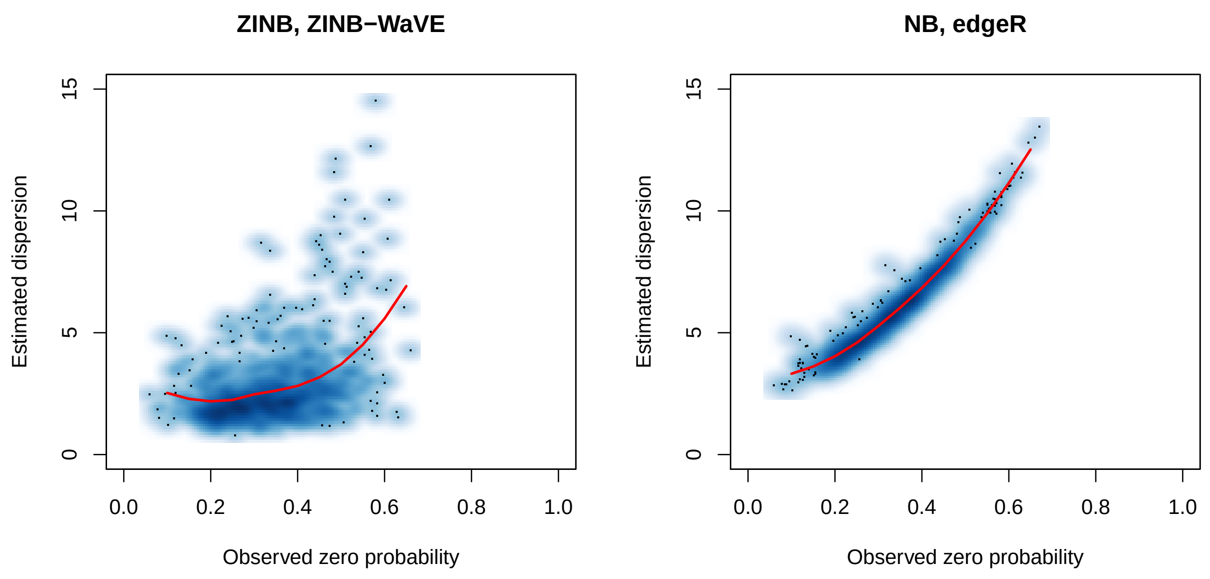


Figure 2.27: *Goodness-of-fit of ZINB-WaVE and NB models: Estimated dispersion parameter vs. observed proportion of zero counts for V1 dataset.* Left panel: Genewise dispersion parameters ϕ_j estimated using ZINB-WaVE. Right panel: Genewise dispersion parameters ϕ_j estimated using edgeR package. See Methods for details.

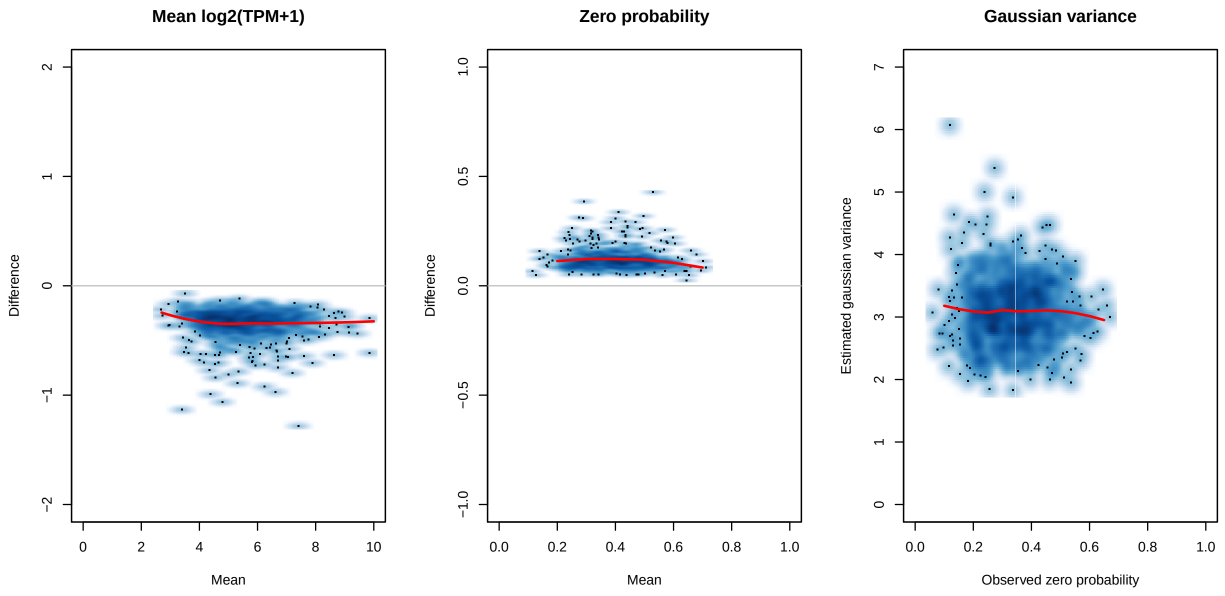


Figure 2.28: *Goodness-of-fit of MAST hurdle model for V1 dataset.* Left panel: Mean-difference plot of estimated vs. observed mean log2(TPM+1). Middle panel: Mean-difference plot of estimated vs. observed zero probability. Right panel: Estimated Gaussian variance parameter σ_j^2 vs. observed proportion of zero counts. For left and middle panels, observed and estimated mean log2(TPM+1) and zero probabilities were averaged over n cells. Parameters were estimated using the function `zlm` from the MAST package, with an intercept and a covariate for the cellular detection rate (as recommended in the MAST vignette for the MAIT data analysis) for both the discrete and continuous parts.

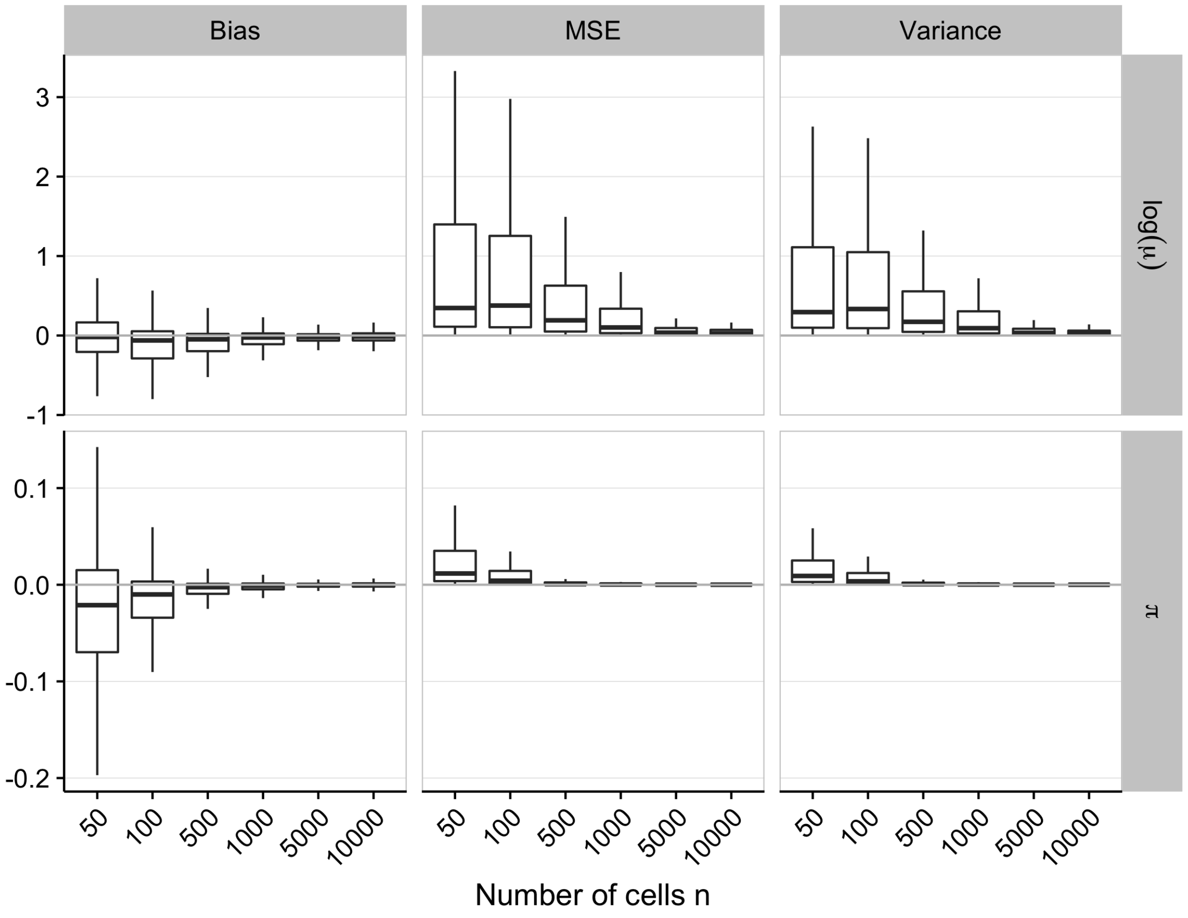


Figure 2.29: *Bias, MSE, and variance for ZINB-WaVE estimation procedure: ZINB-WaVE simulation model.* Boxplots of bias, MSE, and variance for $\ln(\mu)$ and π as a function of the number of cells n. For each gene and cell, bias, MSE, and variance were averaged over $B = 10$ datasets simulated from our ZINB-WaVE model, based on the S1/CA1 dataset and with $n \in \{50, 100, 500, 1,000, 5,000, 10,000\}$ cells, $J = 1,000$ genes, scaling of one for the ratio of within to between-cluster sums of squares ($b^2 = 1$), and zero fraction of about 80%. The following values were used for both simulating the data and fitting the ZINB-WaVE model to these data: $K = 2$ unknown factors, $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion.

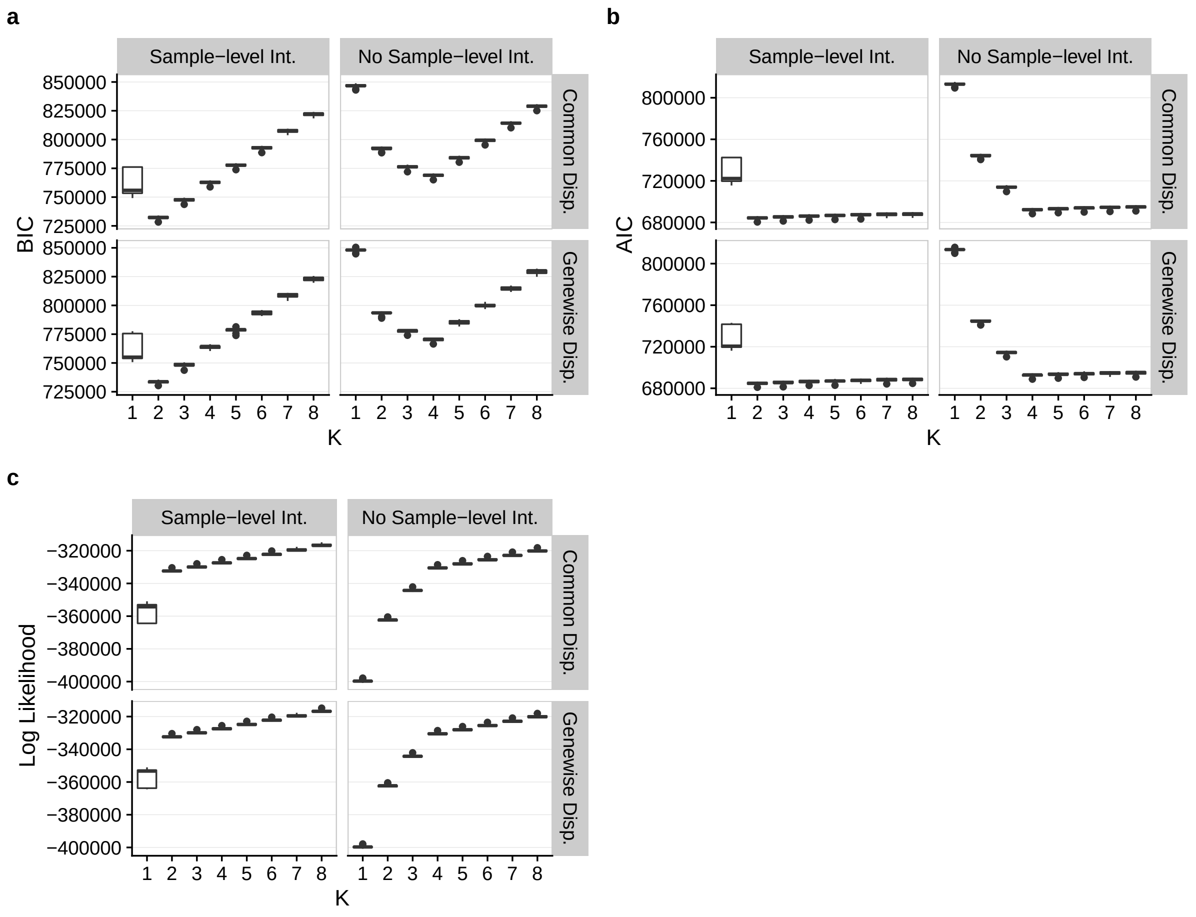


Figure 2.30: *BIC, AIC, and log-likelihood for ZINB-WaVE estimation procedure: ZINB-WaVE simulation model.* Panels show boxplots of **(a)** BIC, **(b)** AIC, and **(c)** log-likelihood for ZINB-WaVE estimation procedure, as a function of the number of unknown covariates K. ZINB-WaVE was fit with $X = \mathbf{1}_n$, common/genewise dispersion, and with/without sample-level intercept (i.e., column of ones in gene-level covariate matrix V). For each gene and cell, BIC, AIC, and log-likelihood were averaged over $B = 10$ datasets simulated from our ZINB-WaVE model, based on the S1/CA1 dataset and with $n = 1,000$ cells, $J = 1,000$ genes, scaling of one for the ratio of within to between-cluster sums of squares ($b^2 = 1$), $K = 2$ unknown factors, zero fraction of about 80%, $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion.

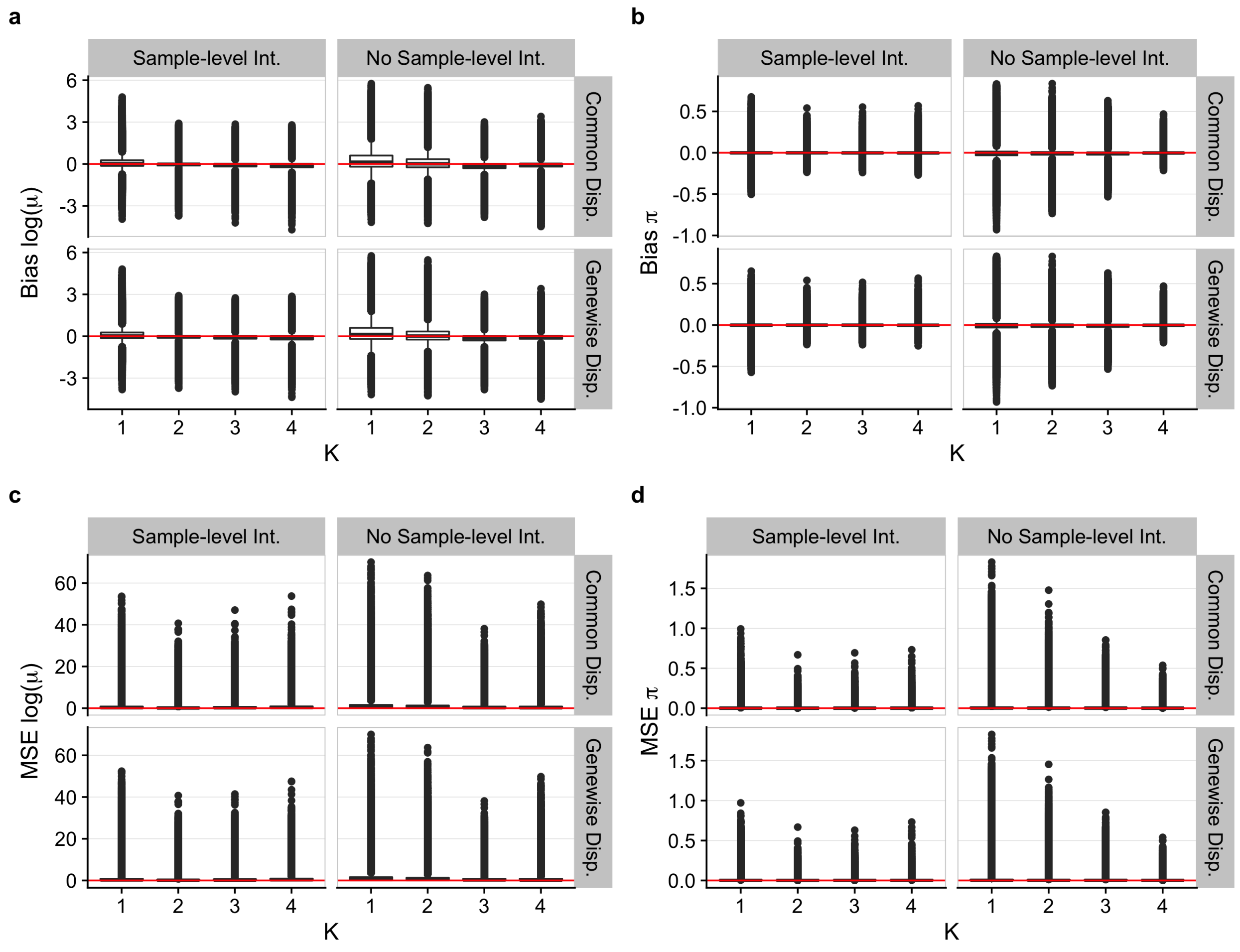


Figure 2.31: *Bias and MSE for ZINB-WaVE estimation procedure: ZINB-WaVE simulation model.* Same as Figure 2.6, but with outliers plotted individually (i.e., observations beyond the whiskers).

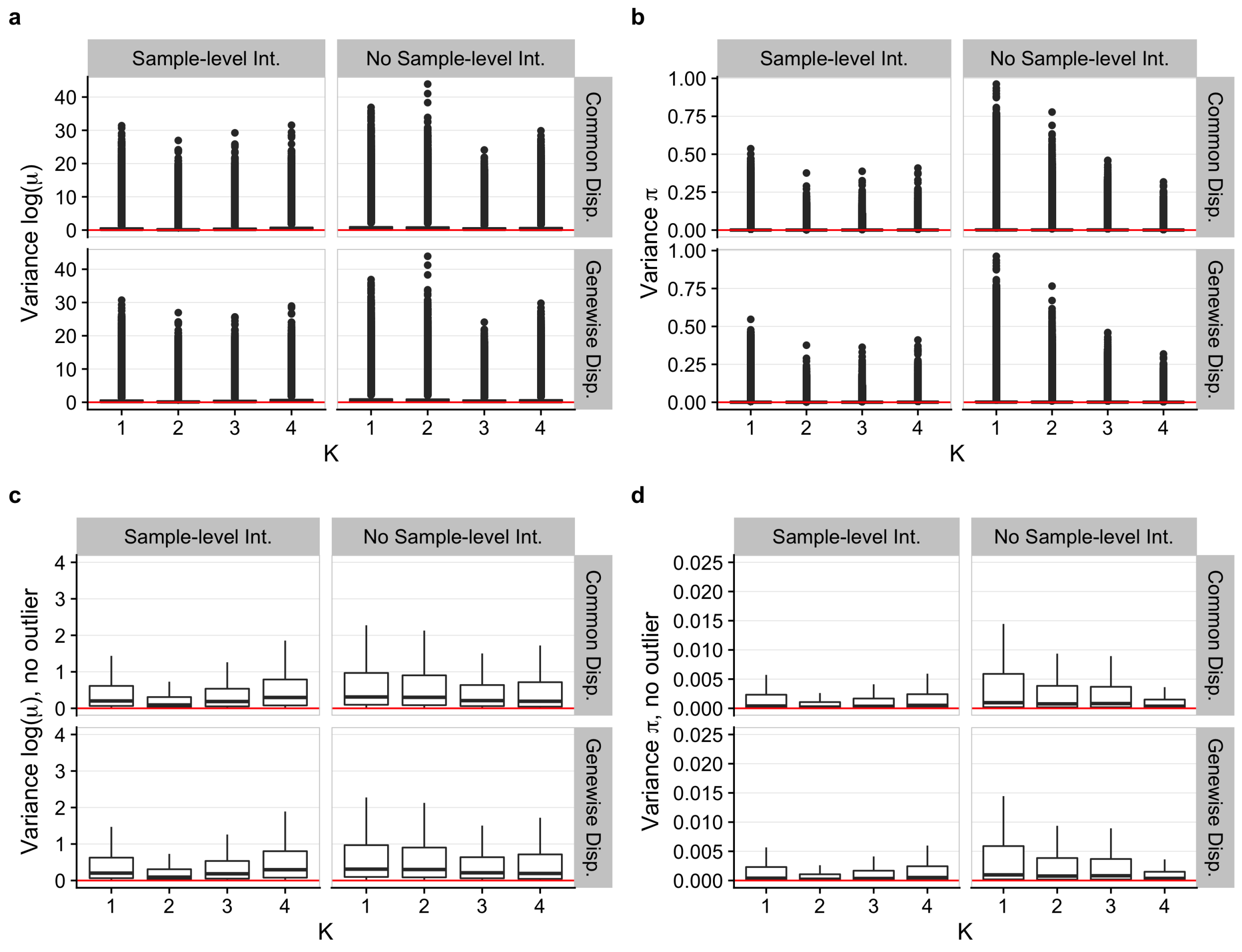
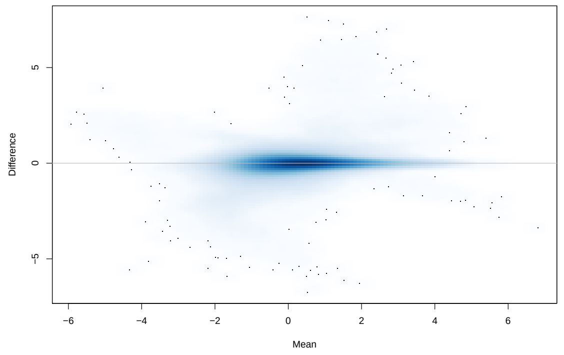
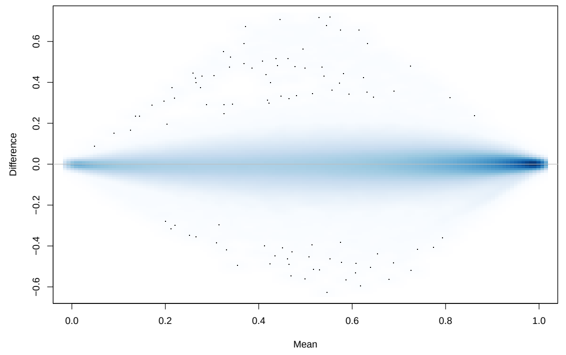


Figure 2.32: *Variance for ZINB-WaVE estimation procedure: ZINB-WaVE simulation model.* Panels show boxplots of variance (over $B = 10$ simulated datasets) for estimates of $\ln(\mu)$ **(a, c)** and π **(b, d)**. Outliers plotted in **(a, b)** and omitted in **(c, d)**. Simulation scenario as in Figure 2.6.



(a) $\ln(\mu)$



(b) π

Figure 2.33: *Bias for ZINB-WaVE estimation procedure: ZINB-WaVE simulation model.* **(a)** Mean-difference plot of estimated vs. true negative binomial mean (log scale), $\ln(\hat{\mu}) - \ln(\mu)$ vs. $(\ln(\mu) + \ln(\hat{\mu}))/2$. **(b)** Mean-difference plot of estimated vs. true zero inflation probability, $\hat{\pi} - \pi$ vs. $(\pi + \hat{\pi})/2$. The estimates are based on one of the $B = 10$ datasets simulated from our ZINB-WaVE model, based on the S1/CA1 dataset and with $n = 1,000$ cells, $J = 1,000$ genes, scaling of one for the ratio of within to between-cluster sums of squares ($b^2 = 1$), and zero fraction of about 80%. The following values were used for both simulating the data and fitting the ZINB-WaVE model to these data: $K = 2$ unknown factors, $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion (as in Figures 2.6, 2.31 and 2.32).

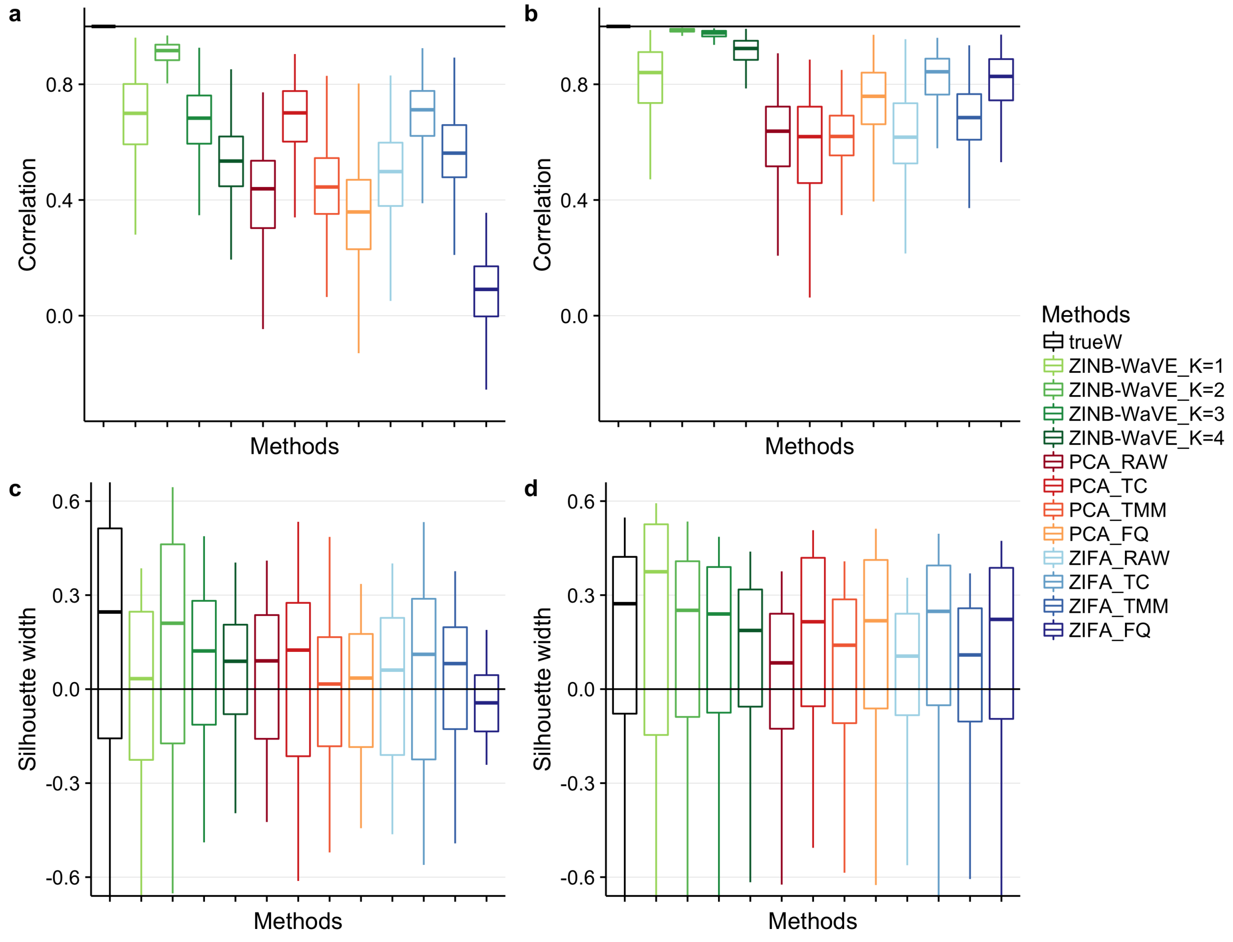


Figure 2.34: *Between-sample distances and silhouette widths for ZINB-WaVE, PCA, and ZIFA: ZINB-WaVE simulation model.* **(a)** Boxplots of correlations between between-sample distances based on true and estimated low-dimensional representations of the data for simulations based on the V1 dataset. **(b)** Same as **(a)** for simulations based on the S1/CA1 dataset. **(c)** Boxplots of silhouette widths for true clusters for simulations based on the V1 dataset. **(d)** Same as **(c)** for simulations based on the S1/CA1 dataset. All datasets were simulated from our ZINB-WaVE model with $n = 10,000$ cells, $J = 1,000$ genes, "harder" clustering ($b^2 = 5$), $K = 2$ unknown factors, zero fraction of about 80%, $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion. Each boxplot is based on n values corresponding to each of the n samples and defined as averages of correlations **(a, b)** or silhouette widths **(c, d)** over $B = 10$ simulations. Between-sample distance matrices and silhouette widths were based on W for ZINB-WaVE, the first two principal components for PCA, and the first two latent variables for ZIFA. ZINB-WaVE was applied with $X = \mathbf{1}_n$, $V = \mathbf{1}_J$, genewise dispersion, and $K \in \{1, 2, 3, 4\}$. For PCA and ZIFA, different normalization methods were used. Colors correspond to the different methods. See Figure 2.7a–d for the same scenario but with $n = 1,000$ cells and Figure 2.35 for additional scenarios.

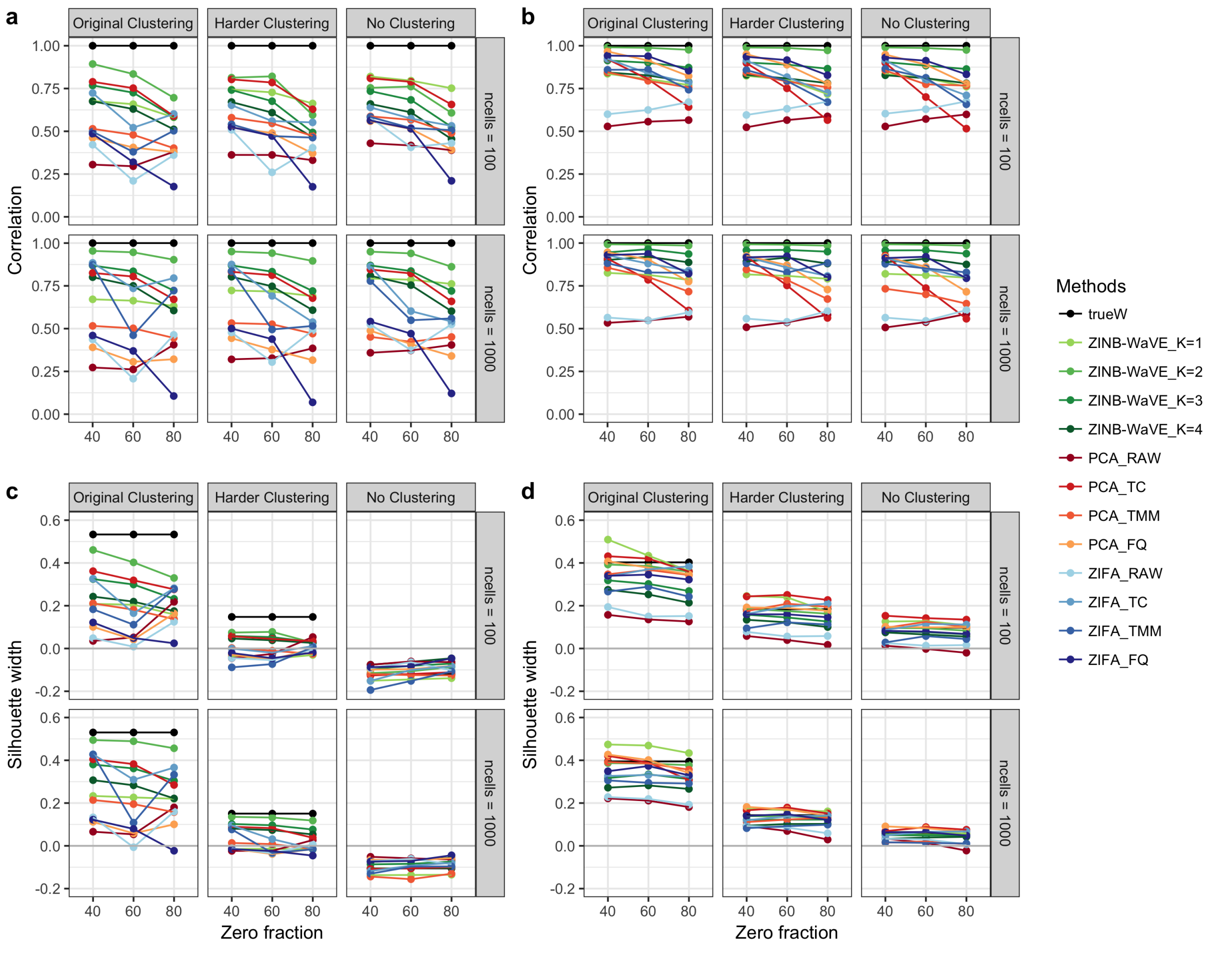


Figure 2.35: *Between-sample distances and silhouette widths for ZINB-WaVE, PCA, and ZIFA: ZINB-WaVE simulation model.* **(a)** Correlation between between-sample distances based on true and estimated low-dimensional representations of the data for simulations based on the V1 dataset. **(b)** Same as **(a)** for simulations based on the S1/CA1 dataset. **(c)** Silhouette width for true clusters for simulations based on the V1 dataset. **(d)** Same as **(c)** for simulations based on the S1/CA1 dataset. As in Figure 2.7, all datasets were simulated from our ZINB-WaVE model with $J = 1,000$ genes, $K = 2$ unknown factors, $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and genewise dispersion. Each point corresponds to a simulation scenario (zero fraction, clustering strength, sample size); correlations between true and estimated between-sample distances and silhouette widths are averaged over $B = 10$ simulated datasets and n cells. Column panels show three different clustering scenarios, where the scaling of the ratio of within to between-cluster sums of squares b^2 corresponds to the original clustering ($b^2 = 1$), a harder clustering ($b^2 = 5$), and no clustering ($b^2 = 10$). Row panels correspond to different numbers of cells ($n \in \{100, 1,000\}$). Between-sample distance matrices and silhouette widths were based on W for ZINB-WaVE, the first two principal components for PCA, and the first two latent variables for ZIFA. ZINB-WaVE was applied with $X = \mathbf{1}_n$, $V = \mathbf{1}_J$, genewise dispersion, and $K \in \{1, 2, 3, 4\}$. For PCA and ZIFA, different normalization methods were used. Colors correspond to the different methods.

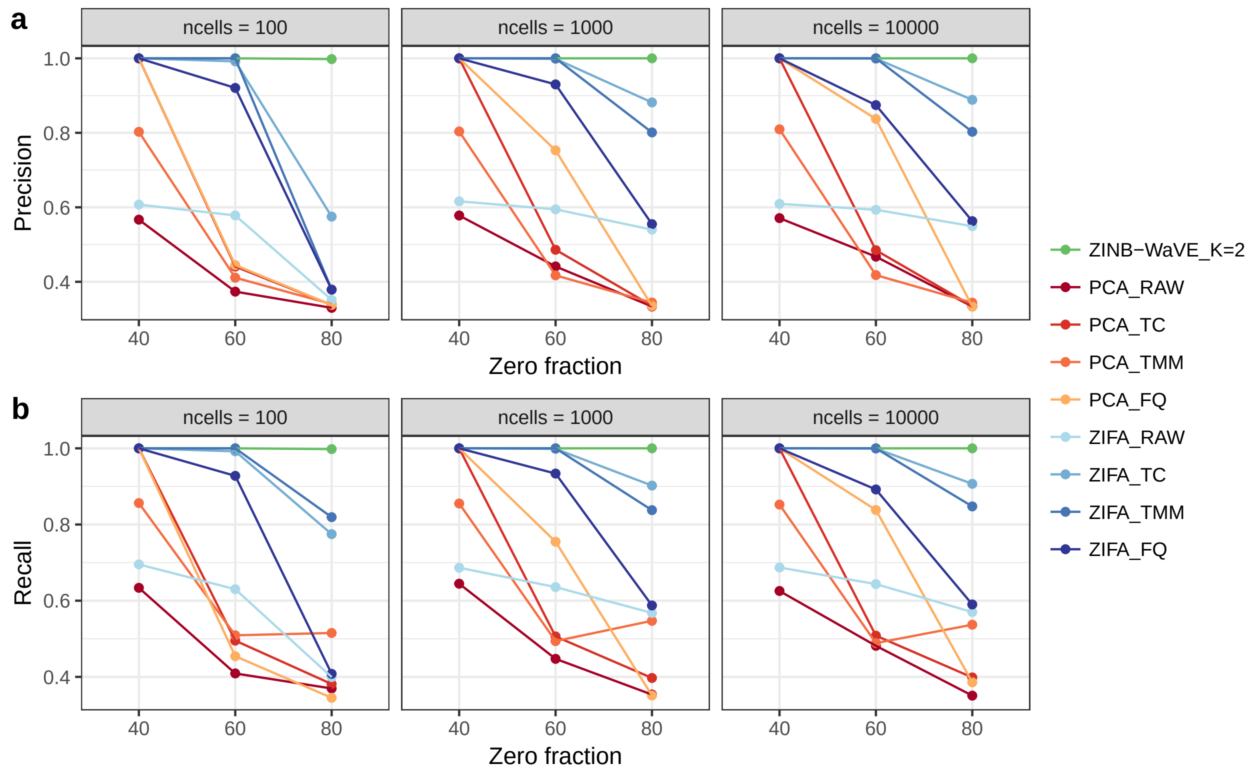


Figure 2.36: *Precision and recall for ZINB-WaVE, PCA, and ZIFA: Lun and Marioni [40] simulation model.* Average **(a)** precision coefficient and **(b)** recall coefficient (over n samples and $B = 10$ simulations) vs. zero fraction, for $n \in \{100, 1,000, 10,000\}$ cells, for datasets simulated from the Lun and Marioni [40] model, with $C = 3$ clusters and equal number of cells per cluster. Clustering was performed using k-means on W for ZINB-WaVE, the first two principal components for PCA, and the first two latent variables for ZIFA. ZINB-WaVE was applied with $X = \mathbf{1}_n$, $V = \mathbf{1}_J$, genewise dispersion, and $K = 2$. For PCA and ZIFA, different normalization methods were used. Colors correspond to the different methods. While ZINB-WaVE has a recall and precision of one for all sample sizes n and zero fractions, the performance of PCA and ZIFA decreases with larger zero fraction. See Methods for details on clustering procedure and precision and recall coefficients.

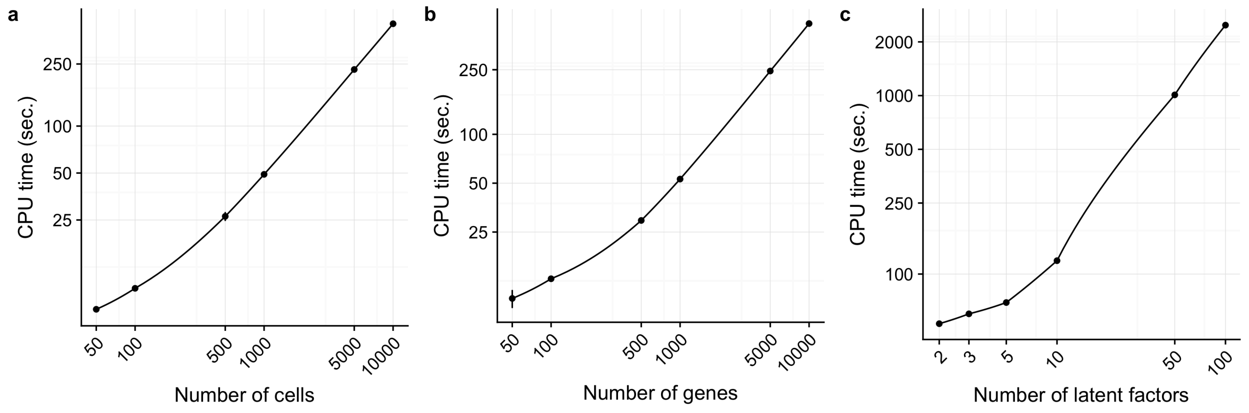


Figure 2.37: *CPU time for ZINB-WaVE estimation procedure.* Log-log scatterplot of mean CPU time (in seconds) vs. **(a)** sample size n, **(b)** number of genes J, and **(c)** number of latent factors K. For each panel $B = 10$ datasets were simulated from the Lun and Marioni [40] model with zero fraction of about 60%. The following specific values were used for each panel: **(a)** $n \in \{50, 100, 500, 1,000, 5,000, 10,000\}$ cells, $J = 1,000$ genes, $K = 2$ latent factors; **(b)** $J \in \{50, 100, 500, 1,000, 5,000, 10,000\}$ genes, $n = 1,000$ cells, $K = 2$ latent factors; **(c)** $n = 1,000$ cells, $J = 1,000$ genes, and $K \in \{2, 3, 5, 10, 50, 100\}$ latent factors. The following values were used to fit the ZINB-WaVE model: $X = \mathbf{1}_n$, cell-level intercept ($V = \mathbf{1}_J$), and common dispersion. CPU times were averaged over $B = 10$ simulated datasets and standard deviations are indicated by the vertical bars. Computations were done with 7 cores on an iMac with eight 4 GHz Intel Core i7 CPUs and 32 GB of RAM.

Chapter 3

Unlock bulk RNA-seq DE tools for zero-inflated and single-cell data

This chapter has been published in the journal Genome Biology with Koen Van den Berge as my first co-author. I specifically conducted the analysis of the real datasets while Koen Van den Berge focused on the simulations and analysis of simulated datasets.[1]

3.1 Background

Transcriptomics has become one of the standard tools in modern biology to unravel the molecular basis of biological processes and diseases. One of the most common applications of transcriptome profiling is the discovery of *differentially expressed* (DE) genes, which exhibit changes in expression levels across conditions [69, 70, 71]. Over the last decade, transcriptome sequencing (RNA-seq) has become the standard technology for transcriptome profiling, enabling researchers to study average gene expression over bulks of thousands of cells [72, 73]. The advent of single-cell RNA-seq (scRNA-seq) enables high-throughput transcriptome profiling at the resolution of single cells and allows, among other things, research on cell developmental trajectories, cell-to-cell heterogeneity, and the discovery of novel cell types [74, 75, 76, 77, 78, 79].

In scRNA-seq, individual cells are first captured, their RNA is then reverse-transcribed into cDNA, which is greatly amplified from the minute amount of starting material, and the resulting library is finally sequenced [80]. Transcript abundances are typically estimated by counts, which represent the number of sequencing reads mapping to an exon, transcript, or gene. Many scRNA-seq protocols have been published to conduct such core steps [81, 82, 83, 84, 85, 86], but despite these advances, scRNA-seq data remain inherently noisy.

[1]I would like to thank my first co-author, Koen Van den Berge, and co-authors, Charlotte Soneson, Michael I. Love, Davide Risso, Jean-Philippe Vert, Mark D. Robinson, Sandrine Dudoit, and Lieven Clement. Sandrine Dudoit and Lieven Clement are equally contributing authors of the paper.

Dropout events cause many transcripts to go undetected for technical reasons, such as inefficient cDNA polymerization, amplification bias, or low sequencing depth, leading to an excess of zero read counts as compared to bulk RNA-seq data [86, 87]. In addition, excess zeros can also occur for biological reasons, such as transcriptional bursting [88]. There are therefore two types of zeros in scRNA-seq data: *biological zeros*, when a gene is simply not expressed in the cell, and *technical zeros* (i.e., dropouts), when a gene is expressed in the cell but not detected. *Zero inflation*, i.e., excess zeros compared to standard count distributions (e.g., negative binomial) used in bulk RNA-seq, occurs for both biological and technical reasons and disentangling the two sources is non-trivial. In addition, scRNA-seq counts are inherently more variable than bulk RNA-seq counts because the transcriptional signal is not averaged across thousands of individual cells (Figure 3.7), making cell-to-cell heterogeneity, cell type mixtures, and stochastic expression bursts important contributors to between-sample variability [89, 75].

Typical scRNA-seq data analysis workflows often involve identifying cell types *in silico* using tailored clustering algorithms [90, 91] or ordering cells along developmental trajectories, where cell types are defined as terminal states of the developmental process [92, 93, 74, 94]. A natural subsequent step is the discovery of marker genes for the defined cell types by assessing differential gene expression between these groups. Another common setting is the identification of marker genes for *a priori* known cell types. Differential expression analysis between homogeneous cell populations as in the aforementioned scRNA-seq applications is the use case for our method.

Popular bulk RNA-seq DE tools, such as those implemented in the Bioconductor R packages edgeR [70] and DESeq2 [69], assume a negative binomial (NB) count distribution across biological replicates, while limma-voom [71] uses linear models for log-transformed counts and observation-level weights to account for the mean-variance relationship of the transformed count data. Such tools can also be applied for scRNA-seq DE analysis [95]. However, dropouts, transcriptional bursting, and high variability in scRNA-seq data raise concerns about their validity. These challenges have triggered the development of novel dedicated tools, which typically introduce an additional model component to account for the excess of zeros through, for example, zero-inflated (SCDE, Kharchenko, Silberstein, and Scadden [96]) or hurdle (MAST, Finak et al. [87]) models. However, Jaakkola et al. [97] and Soneson and Robinson [98] have recently shown that these bespoke tools do not provide systematic benefits over standard bulk RNA-seq tools in scRNA-seq applications.

We argue that standard bulk RNA-seq tools, however, still suffer in performance due to zero inflation with respect to the negative binomial distribution. It is illustrated in this paper using biological coefficient of variation (BCV) plots [99], which represent the mean-variance relationship of the counts. Note that the BCV plots of scRNA-seq data exhibit striped patterns (Figures 3.1a-b and 3.8 for scRNA-seq datasets subsampled to ten cells) that are indicative of genes with few positive counts (Figure 3.9) and very high dispersion estimates.

Randomly adding zeros to bulk RNA-seq data, likewise consisting of ten samples, also results in similar striped patterns (Figure 3.1c-d). Negative binomial models, as implemented in DESeq2 and edgeR, will thus accommodate excess zeros by overestimating the dispersion parameter, which jeopardizes the power to infer differential expression. However, by correctly identifying the excess zeros and downweighting them in the dispersion estimation and model-fitting, one reconstructs the original mean-variance relationship (Figure 3.1e), thus recovering the power to detect differential expression (Figure 3.1f). Hence, identifying and downweighting excess zeros provides the key to unlocking bulk RNA-seq tools for scRNA-seq differential expression analysis. Note that methods based on a zero-inflated negative binomial (ZINB) model naturally implement such an approach: Excess zeros are attributed weights through the zero inflation probability and inference can be focused on the mean of the negative binomial count component.

We therefore propose a weighting strategy based on ZINB models to unlock bulk RNA-seq tools for scRNA-seq DE analysis. In this manuscript, we build on the zero-inflated negative binomial-based wanted variation extraction (ZINB-WaVE) method of Risso et al. [91], designed specifically for scRNA-seq data. ZINB-WaVE efficiently identifies excess zeros and provides gene and cell-specific weights to unlock bulk RNA-seq pipelines for zero-inflated data. As most bulk RNA-seq DE methods are based on generalized linear models (GLM), which readily accommodate observation-level weights, our approach seamlessly integrates with standard pipelines (e.g., edgeR, DESeq2, limma). Our method is shown to outperform competing methods on simulated bulk and single-cell RNA-seq datasets. We also illustrate our method on two publicly available real datasets. As detailed in the "Software implementation" section, our approach is implemented in open-source Bioconductor R packages and the code for reproducing the analyses presented in this manuscript is provided in a GitHub repository.

3.2 Methods

3.2.1 ZINB-WaVE

Zero-inflated distributions. A major difference between single-cell and bulk RNA-seq data is arguably the high abundance of zero counts in the former. Traditionally, excess zeros are dealt with by the use of hurdle or zero-inflated models, as recently proposed by Finak et al. [87], Kharchenko, Silberstein, and Scadden [96], and Paulson et al. [103]. A zero-inflated count distribution is a two-component mixture distribution between a point mass at zero and a count distribution, in our case, the negative binomial distribution which has been used successfully for bulk RNA-seq [69, 70, 71, 104].

The probability mass function (PMF) f_{ZINB} for the zero-inflated negative binomial

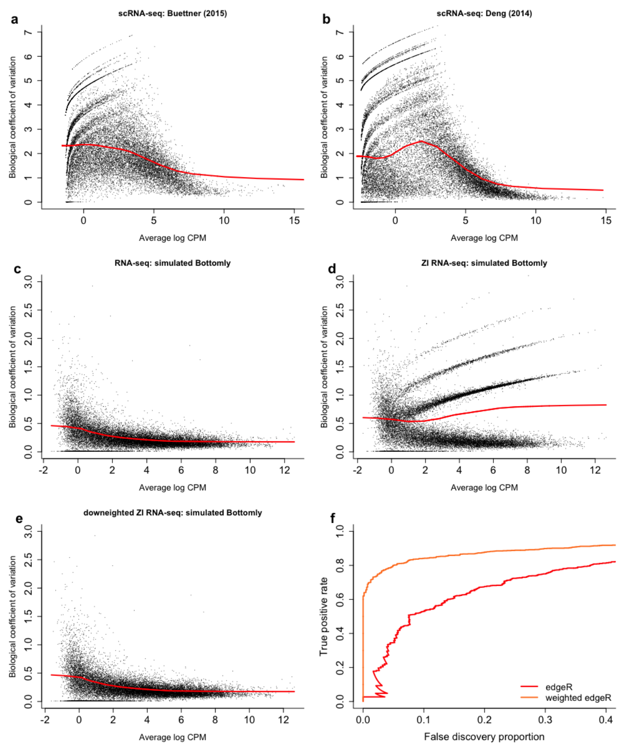


Figure 3.1: *Zero inflation results in overestimated dispersion and jeopardizes power to discover differentially expressed genes.* **(a–e)** Scatterplots of estimated biological coefficient of variation (BCV, defined as the square root of the negative binomial dispersion parameter ϕ) against average log count per million (CPM) computed using `edgeR`. **(a)** BCV plot for the real Buettner et al. [75] scRNA-seq dataset subsampled to $n = 10$ cells. **(b)** BCV plot for the real Deng et al. [100] scRNA-seq dataset subsampled to $n = 10$ cells. Both panels (a) and (b) show striped patterns in the BCV plot, which significantly distort the mean-variance relationship, as represented by the red curve. **(c)** BCV plot for a simulated bulk RNA-seq dataset ($n = 10$), obtained from the Bottomly et al. [101] dataset using the simulation framework of Zhou, Lindsay, and Robinson [102]. Dispersion estimates generally decrease smoothly as gene expression increases. **(d)** BCV plot for a simulated zero-inflated bulk RNA-seq dataset, obtained by randomly introducing 5% excess zero counts in the dataset from (c). Zero inflation leads to overestimated dispersion for the genes with excess zeros, resulting in striped patterns, as observed also for the real scRNA-seq data in panels (a) and (b). **(e)** BCV plot for simulated zero-inflated bulk RNA-seq dataset from (d), where excess zeros are downweighted in dispersion estimation (i.e., weights of 0 for excess zeros and 1 otherwise). Downweighting recovers the original mean-variance trend. **(f)** True positive rate vs. false discovery proportion (FDP-TPR curves) for simulated zero-inflated dataset of (d). The performance of `edgeR` (red curve) is deteriorated in a zero-inflated setting due to overestimation of the dispersion parameter. However, assigning the excess zeros a weight of zero in the dispersion estimation and model-fitting results in a dramatic performance boost (orange curve). Hence, downweighting excess zero counts is the key to unlocking bulk RNA-seq tools for zero inflation.

(ZINB) distribution is given by

$$f_{ZINB}(y;\mu,\theta,\pi) = \pi\delta_0(y) + (1-\pi)f_{NB}(y;\mu,\theta), \quad \forall y \in \mathbb{N}, \tag{3.1}$$

where $\pi \in [0,1]$ denotes the mixture probability for zero inflation, $f_{NB}(\cdot;\mu,\theta)$ the negative binomial (NB) PMF with mean μ and dispersion $\theta = 1/\phi$, and $\delta_0(\cdot)$ the Dirac function ($\delta_0(y) = +\infty$ when $y = 0$ and 0 otherwise and δ_0 integrates to one over $\mathbb{R}$, i.e., has cumulative distribution function equal to $I(y \geq 0)$). Here, π can be interpreted as the probability of an excess zero, i.e., inflated zero count, with respect to the NB distribution.

Under a ZINB model, the posterior probability that a given count y arises from the NB count component is given by Bayes' rule

$$w = \frac{(1-\pi)f_{NB}(y;\mu,\theta)}{f_{ZINB}(y;\mu,\theta,\pi)}.$$

As described below, such posterior probabilities can be used as weights in standard bulk RNA-seq workflows, for a suitable parameterization of the ZI probability and NB mean.

ZINB-WaVE model. Given n observations (typically, n single cells) and J features (typically, J genes) that can be counted for each observation, let Y_{ij} denote the count of feature j ($j = 1,\ldots,J$) for observation i ($i = 1,\ldots,n$). To account for various technical and biological effects frequent in single-cell sequencing technologies, we model Y_{ij} as a random variable following a ZINB distribution with parameters μ_{ij}, θ_{ij}, and π_{ij}, and consider the following regression models for these parameters:

$$\begin{aligned} \ln(\mu_{ij}) &= \left(X\beta_\mu + (V\gamma_\mu)^\top + W\alpha_\mu + O_\mu\right)_{ij}, \\ \text{logit}(\pi_{ij}) &= \left(X\beta_\pi + (V\gamma_\pi)^\top + W\alpha_\pi + O_\pi\right)_{ij}, \\ \ln(\theta_{ij}) &= \zeta_j\,. \end{aligned} \tag{3.2}$$

Both the NB mean expression level μ and the ZI probability π are modeled in terms of *observed sample-level and gene-level covariates* (X and V, respectively), as well as *unobserved sample-level covariates* (W) that need to be inferred from the data. O_μ and O_π are known matrices of offsets. The matrix X can include covariates that induce variation of interest, such as cell types, or covariates that induce unwanted variation, such as batch or quality control (QC) measures. It can also include a constant column of ones for an intercept that accounts for gene-specific global differences in mean expression level or dropout rate. The matrix V can include gene-level covariates, such as length or GC-content. It can also accommodate an intercept to account for cell-specific global effects, such as size factors representing differences in library sizes (i.e., total number of reads per sample). The unobserved matrix W contains unknown sample-level covariates, which could correspond to unwanted variation as in RUV [35, 36] (e.g., *a priori* unknown batch effects) or could be of interest as in cluster analysis (e.g., *a priori* unknown cell types). The model extends the RUV framework to the ZINB distribution (thus far, RUV had only been implemented for linear [35] and log-linear regression [36]). It differs, however, in interpretation from RUV in the $W\alpha$ term, which is not necessarily considered unwanted and generally refers to unknown low-dimensional variation. It is important to note that although W is the same, the matrices X and V could differ in the modeling of μ and π, if we assume that some known factors do not affect both.

As detailed in Risso et al. [91], the model is fit using a penalized maximum likelihood estimation procedure.

3.2.2 Using ZINB-WaVE weights in DE inference methods

We only consider statistical inference on the count component of the mixture distribution, that is, we are concerned with identifying genes whose expression levels are associated with covariates of interest as parameterized in the mean μ of the negative binomial component. Most popular bulk RNA-seq methods are based on the methodology of generalized linear models (GLM), which readily accommodates inference based on observation-level weights (R function `glm`), e.g., negative binomial model in Bioconductor R packages edgeR and DESeq2. Note that although the ZINB-WaVE weights are gene and cell-specific, the GLMs are fit gene by gene; hence, for given gene, the cell-specific weights are used as observation-specific weights in the GLMs.

edgeR. We extended the edgeR package [70, 104] by fitting a negative binomial model genewise, with ZINB-WaVE posterior probabilities as observation-level weights in the `weights` slot of an object of class *DGEList*, and estimating the dispersion parameter by the usual approximate empirical Bayes shrinkage. Downweighting is accounted for by adjusting the degrees of freedom of the null distribution of the test statistics. Specifically, we reintroduced the moderated F-test from a previous version of edgeR, where the denominator residual degrees of freedom df_j for a particular gene j are adjusted by the extent of zero inflation identified for this gene, i.e., $df_j = \sum_i w_{ij} - p$, where w_{ij} is the ZINB-WaVE weight for gene j in cell i and p the number of parameters estimated in the NB generalized linear model. This weighted F-test is implemented in the function `glmWeightedF` from the Bioconductor R package zinbwave.

DESeq2. We extended the DESeq2 package [69] to accommodate zero inflation by providing the option to use observation-level weights in the parameter estimation steps. In this case, the ZINB-WaVE weights are supplied in the `weight` slot of an object of class *DESeqDataSet*.

DESeq2's default normalization procedure is based on geometric means of counts, which are zero for genes with at least one zero count, greatly limiting the number of genes that can be used for normalization in scRNA-seq applications [19]. We therefore use the normalization method suggested in the phyloseq package [105], which calculates the geometric mean for a gene by only using its positive counts, so that genes with zero counts could still be used for normalization purposes. The phyloseq normalization procedure can now be applied by setting the argument `type` equal to `poscounts` in the DESeq2 function `estimateSizeFactors`. For single-cell UMI data, for which the expected counts may be very low, the likelihood ratio test implemented in `nbinomLRT` should be used. For other protocols (i.e., non-UMI), the Wald test in `nbinomWaldTest` can be used, with null distribution a t-distribution with degrees of freedom corrected for downweighting. In both cases, we recommend the minimum expected count to be set to a small value (`minmu=1e-6`). The Wald test in DESeq2 allows for testing contrasts of the coefficients.

limma-voom. For the limma-voom approach [71], implemented in the `voom` function from the limma package, heteroscedastic weights are estimated based on the mean-variance relationship of the log-transformed counts. We adapt limma-voom to zero-inflated situations by multiplying the heteroscedastic weights by the ZINB-WaVE weights and using the resulting weights in weighted linear regression. To account for the downweighting of zeros, the residual degrees of freedom of the linear model are adjusted as with edgeR before the empirical Bayes variance shrinkage and are therefore also propagated to the moderated statistical tests. Both the standard and ZINB-WaVE-weighted versions of limma-voom were considered in the simulation study; the latter was not considered for the real datasets, due to its poor performance in the simulation study.

Multiple testing. For the simulation study, in order to reduce the number of tests performed [106], we apply the independent filtering procedure implemented in the genefilter package and used in DESeq2 [69]. As in DESeq2, we exclude from the multiple testing correction any gene whose average expression strength (i.e., average of fitted values) is below a threshold chosen to maximize the number of differentially expressed genes. Note that the filtering procedure can affect each method differently, due to differences in fitted values and p-value distributions. Unless specified otherwise, the p-values for all methods are then adjusted using the Benjamini and Hochberg [107] procedure for controlling the false discovery rate (FDR).

Performance assessment. We assess performance based on scatterplots of the true positive rate (TPR) vs. the false discovery proportion (FDP), as well as receiver operating characteristic (ROC) curves of the true positive rate (TPR) vs. the false positive rate

(FPR), according to the following definitions

$$\begin{aligned} FDP &= \frac{FP}{\max(1, FP+TP)} \\ FPR &= \frac{FP}{FP+TN} \\ TPR &= \frac{TP}{TP+FN}, \end{aligned}$$

where FN, FP, TN, and TP denote, respectively, the numbers of false negative, false positives, true negatives, and true positives. FDP-TPR curves and ROC curves are implemented in the Bioconductor R package iCOBRA [108].

DE method comparison. We compared our weighted DE approach to state-of-the-art bulk RNA-seq methods implemented in the packages edgeR (v3.20.1) [70, 104], DESeq2 (v1.19.8) [69], and limma (v3.34.0) [71]. We also considered dedicated scRNA-seq tools from the packages scde (v2.6.0) [96], MAST (v1.4.0) [87], and NODES (v0.0.0.9010) [109], as well as metagenomeSeq (v1.18.0) [103] developed to account for zero inflation in metagenomics applications. A ZINB model is also implemented in ShrinkBayes [110], but the method does not scale to the typical sample sizes encountered in scRNA-seq and has many tuning parameters, which lead us to not include it in our comparison. In DESeq2, we disable the outlier imputation step and allow for shrinkage of fold-changes by default. In addition, for large 3′-end sequencing datasets like the Usoskin and 10x Genomics PBMC datasets, we set the minimum expected count estimated by DESeq2 to 10^{-6}, allowing the method to cope with large sample sizes and low counts. We use the recommended gene filtering procedures for NODES and MAST, except for computing time benchmarking, where no genes are filtered out to allow a fair comparison. For all other methods, arguments were set to their default values.

3.2.3 scRNA-seq data simulation

We extended the framework of Zhou, Lindsay, and Robinson [102] towards scRNA-seq applications and provide user-friendly R code to simulate scRNA-seq read counts in the GitHub repository linked to this manuscript (`https://github.com/statOmics/zinbwaveZinger`). The user can input a real scRNA-seq dataset to infer gene-level parameters for the read count distributions. Library sizes for the simulated samples are by default resampled from the real dataset, but can also be user-specified. The simulation paradigm randomly resamples parameters estimated from the original dataset, where all parameters of a given gene are resampled jointly in order to retain gene-specific characteristics present in the original dataset.

In scRNA-seq, dropouts and bursting lead to bias in parameter estimation. Our simulation framework alleviates this problem by using zero-truncated negative binomial (ZTNB) method-of-moments estimators [111, 112] on the positive counts to estimate the expression fraction $\lambda_j = E[Y_{ij}/N_i]$, with $N_i = \sum_j Y_{ij}$ the sequencing depth of cell i, and the negative binomial dispersion $\theta_j = 1/\phi_j$. Specifically, initial NB-based estimators are iteratively updated according to the ZTNB-based estimators provided by

$$\begin{aligned}
\hat{\lambda}_j^{new} &= \frac{\sum_i Y_{ij}\left(1 - f_{NB}(0; \hat{\lambda}_j N_i, \hat{\theta}_j)\right)}{\sum_i N_i}, \\
\hat{\theta}_j^{new} &= \frac{\sum_i (\hat{\lambda}_j N_i)^2}{\sum_i Y_{ij}^2\left(1 - f_{NB}(0; \hat{\lambda}_j N_i, \hat{\theta}_j)\right) - \sum_i (\hat{\lambda}_j N_i)^2 - \sum_i (\hat{\lambda}_j N_i)}.
\end{aligned} \tag{3.3}$$

Note that, when Y_{ij} is zero, it does not contribute to the estimators of λ_j and θ_j. These estimates are then used to simulate counts according to a negative binomial distribution.

We additionally simulate excess zeros by modeling the empirical zero abundance $p_{ij} = I(Y_{ij} = 0)$ as a function of an interaction between the gene-specific expression intensity, measured as average log count per million (CPM)

$$\hat{A}_j \approx \log_2 \frac{10^6}{n} \sum_{i=1}^{n} \frac{Y_{ij}}{N_i}$$

(as calculated using the `aveLogCPM` function from `edgeR`), and the cell-specific sequencing depth N_i, using a semi-parametric additive logistic regression model,

$$\begin{aligned}
p_{ij} &\sim B(\rho_{ij}), \\
\ln\left(\frac{\rho_{ij}}{1 - \rho_{ij}}\right) &= s(\hat{A}_j) + \ln(N_i) + s(\hat{A}_j) \times \ln(N_i),
\end{aligned} \tag{3.4}$$

where $B(\rho_{ij})$ denotes the Bernoulli distribution with parameter ρ_{ij} and $s(\cdot)$ a non-parametric thin-plate spline [113]. We then compare, for every gene, the estimated probability of zero counts based on the model in Equation (3.4) to the corresponding NB-based probability $f_{NB}(0; \hat{\mu}_{ij}, \hat{\theta}_j)$ with $\hat{\mu}_{ij} = \hat{\lambda}_j N_i$, and randomly add excess zeros whenever the former probability is higher than the latter. The model in Equation (3.4) is motivated by dataset-specific associations observed in real scRNA-seq datasets (Figures 3.10, 3.11).

This framework acknowledges both gene-specific characteristics as well as broad dataset-specific associations across all genes and provides realistic scRNA-seq data for method evaluation. We assessed performance of various DE methods using data simulated based on the Islam et al. [84] dataset, a subset of the Trapnell et al. [114] dataset, and a 10x Genomics PBMC dataset. See the "Real datasets" section for information on these datasets.

3.2.4 Gene set enrichment analysis

To identify cell types corresponding to the two CD4+ T-cell subclusters of the 10x Genomics PBMC dataset, we used gene set enrichment analysis (GSEA) with the function `fgsea` from the Bioconductor R package `fgsea` (v1.4.0) [115] and gene sets for 64 immune and stroma cell types from the R package `xCell` (v1.1.0) [116]. For each DE method, the input to `fgsea` is a list of genes ranked by a test statistic comparing expression in the two CD4+ T-cell subclusters.

To facilitate comparison between DE methods, the test statistic used here is a transformation of the unadjusted p-values (p) with the sign of the log-fold-change (lfc): $\Phi^{-1}(1-p/2) * \text{sign}(lfc)$, where $\Phi(\cdot)$ denotes the standard Gaussian cumulative distribution function. As suggested by `fgsea`, all genes were used for the analysis. To assess the enrichment/depletion of one cluster compared to the other cluster, we used the normalized enrichment score (NES). The enrichment score (ES) is the same as in the Broad GSEA implementation [117] and reflects the degree to which a gene set is overrepresented at the top or bottom of a ranked list of genes. Briefly, the ES is calculated by walking down the ranked list of genes, increasing a running-sum statistic when a gene is in the gene set and decreasing it when it is not. A positive ES indicates enrichment at the top of the ranked list; a negative ES indicates enrichment at the bottom of the ranked list. The enrichment score is then normalized by the mean enrichment of random samples of genes, where genes are permuted from the original ranked list ($10,000$ permutations were used).

3.2.5 Real datasets

Usoskin dataset. This dataset concerns mouse neuronal cells from the dorsal root ganglion, sequenced on either an Illumina Genome Analyzer IIx or HiSeq 2000 [79]. The cells were robotically picked in three separate sessions and the $5'$-end of the transcripts sequenced. The expression measures were downloaded from "Supplementary Data" accompanying the original manuscript (`http://linnarssonlab.org/drg/`). After quality control and sample filtering (removal of non-single cells and non-neuronal cells), the authors considered 622 cells which were classified into eleven neuronal cell type categories. Only the genes with more than 20 non-zero counts were retained, for a total of $12,132$ genes.

The authors acknowledge the existence of a batch effect related to the picking session for the cells. For differential expression analysis, the picking session was therefore included as a batch covariate in all models.

To mimic a null dataset with no differential expression, we created two groups of 45 cells each, where, for each group, 15 cells were sampled at random, without replacement (over all cell types) from each picking session. For each of 30 such mock null datasets, we considered seven methods to identify genes that are DE between the two groups and declared a gene DE if its nominal unadjusted p-value was less than or equal to 0.05. For these mock datasets, any gene declared DE between the two groups is a false positive. Thus, the *nominal* per-comparison error rate (PCER) of 0.05 for each method is compared to its *actual* PCER which is simply the proportion of genes declared DE.

10x Genomics PBMC dataset. We analyzed a dataset of peripheral blood mononuclear cells (PBMC) freely available from 10x Genomics (`https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/pbmc3k`). We downloaded the data, which correspond to $2,700$ single cells sequenced on the Illumina NextSeq 500 using unique molecular identifiers (UMI). We clustered cells following the tutorial available at `http://satijalab.org/seurat/pbmc3k_tutorial.html` and using the R package Seurat (v2.1.0) [118]. The major steps of the pipeline are quality control, data filtering, identification of high-variance genes, dimensionality reduction using the first ten components from principal component analysis (PCA), and graph-based clustering. To identify cluster markers, we used our ZINB-WaVE-weighted DE method instead of the method implemented in Seurat.

We created 30 mock null datasets and identified DE genes on these as for the Usoskin dataset, i.e., we created two groups of 45 cells each, by sampling at random, without replacement from the $2,700$ cells of the real dataset (no batch information available).

Islam dataset. The count table for the Islam et al. [84] dataset was downloaded from the Gene Expression Omnibus (GEO) with accession number GSE29087. The Islam dataset represents 44 embryonic fibroblasts and 48 embryonic stem cells in the mouse, sequenced on an Illumina Genome Analyzer II. Negative control wells were removed and only the $11,796$ genes with at least 5 positive counts were retained for analysis. For the simulation, we generated datasets with 2 groups of 40 cells each.

Trapnell dataset. The dataset from Trapnell et al. [114] was downloaded from the preprocessed single-cell data repository conquer (`http://imlspenticton.uzh.ch:3838/conquer`). Cells were sequenced on either an Illumina HiSeq 2000 or HiSeq 2500. We only used the subset of cells corresponding to the 48h and 72h timepoints of differentiating human myoblasts in order to generate two-group comparisons. Wells that do not contain one cell or that contain debris were removed. We used a more stringent gene filtering criterion than for the Islam dataset and retained the $24,576$ genes with at least 10 positive counts. The simulated datasets contain two conditions with 75 cells in each condition, thereby replicating the sample sizes of the Trapnell dataset.

3.2.6 Software implementation

An R software package for our novel scRNA-seq simulation framework is available on the GitHub repository for this manuscript (`https://github.com/statOmics/zinbwaveZinger`) and is soon to be submitted to the Bioconductor project (`http://www.bioconductor.org`). The ZINB-WaVE weight computation is implemented in the `computeObservationalWeights` function of the Bioconductor R package zinbwave. ZINB-WaVE-weighted edgeR can be implemented using the `glmWeightedF` function from the zinbwave package, while ZINB-WaVE-weighted DESeq2 can be implemented using the native `nbinomWaldTest` function from the DESeq2 package. More details on a ZINB-WaVE-weighted analysis can be found in the zinbwave vignette (`http://bioconductor.org/packages/zinbwave/`). Additionally, all analyses and figures reported in the manuscript can be reproduced using code provided in the GitHub repository (`https://github.com/statOmics/zinbwaveZinger`).

3.3 Results

3.3.1 ZINB-WaVE extends bulk RNA-seq tools to handle zero-inflated data

We argue that standard bulk RNA-seq methods for inferring differential gene expression suffer from zero inflation with respect to the assumed negative binomial distribution when applied to scRNA-seq data. We propose instead to model scRNA-seq data using a zero-inflated model and perform inference on the count component of the model, which is equivalent to standard NB regression where excess zeros are downweighted based on posterior probabilities (weights) inferred from a ZINB model. Such weights play a central role in many estimation approaches for ZINB models (e.g., [119]). In this contribution, we show that the weights can effectively unlock bulk RNA-seq methods for zero-inflated data, allowing us, in particular, to borrow strength across genes to estimate dispersion parameters. Here, we use weights derived from the zero-inflated negative binomial-based wanted variation extraction (ZINB-WaVE) method of Risso et al. [91], which is a general and flexible framework for the extraction of low-dimensional signal from scRNA-seq read counts, accounting for zero inflation (i.e., dropouts, bursting), over-dispersion, and the discrete nature of the data. Note that although we focus on ZINB-WaVE weights, our weighted DE approach is generic and researchers might choose to adopt their own weights.

A zero-inflated negative binomial (ZINB) distribution is a two-component mixture between a point mass at zero and a negative binomial distribution. Specifically, the density function for the ZINB-WaVE model is

$$f_{ZINB}(y_{ij};\mu_{ij},\theta_j,\pi_{ij}) = \pi_{ij}\delta_0(y_{ij}) + (1-\pi_{ij})f_{NB}(y_{ij};\mu_{ij},\theta_j), \tag{3.5}$$

where y_{ij} denotes the read count for cell i and gene j, π_{ij} the mixture probability for zero inflation, $f_{NB}(\cdot;\mu_{ij},\theta_j)$ the negative binomial probability mass function with mean μ_{ij} and dispersion θ_j, and δ_0 the Dirac delta function (see Equations (3.1) and (3.2)).

The ZINB-WaVE parameterization of the NB mean μ and ZI probability π in Equation (3.2) allows adjusting for both known (e.g., treatment, batch, quality control measures) and unknown (RUV) [35, 36] cell-level covariates, i.e., supervised and unsupervised normalization, respectively. It also allows adjusting for known gene-level covariates (e.g., length, GC-content). The ZINB-WaVE model and its associated penalized maximum likelihood estimation procedure are described more fully in the "Methods" section and in Risso et al. [91].

From the ZINB-WaVE density of Equation (3.5), one can readily derive the posterior probability that a count y_{ij} was generated from the negative binomial count component

$$w_{ij} = \frac{(1-\pi_{ij}) f_{NB}(y_{ij}; \mu_{ij}, \theta_j)}{f_{ZINB}(y_{ij}; \mu_{ij}, \theta_j, \pi_{ij})}. \tag{3.6}$$

We propose to use these probabilities as weights in bulk RNA-seq DE analysis methods, such as those implemented in the Bioconductor R packages edgeR, DESeq2, and limma (limma-voom method with `voom` function). All of these methods are based on the methodology of generalized linear models, which readily accommodates inference based on observation-level weights. Note that although the ZINB-WaVE weights are gene and cell-specific, the GLMs are fit gene by gene; hence, for a given gene, the cell-specific weights are used as observation-specific weights in the GLMs. The implementation of the weighting strategy for edgeR, DESeq2, and limma-voom is described in greater detail in the "Methods" section.

3.3.2 Impact of zero inflation on mean-variance relationship

We have already noted that adding zeros to bulk RNA-seq data results in an overestimation of the dispersion parameter leading to striped patterns in the BCV plot (Figure 3.2a), which are indicative of genes with many zeros (Figure 3.9) and very high dispersion estimates. Our ZINB-WaVE method, however, identifies many of the introduced excess zeros as such (Figure 3.2a-b), by classifying them in the zero-inflated component of the ZINB mixture distribution. Using our posterior probabilities as observation-level weights in edgeR recovers the original BCV plot and mean-variance trend (Figure 3.2c), illustrating the ability of our method to account for zero inflation. Hence, observation weights provide the key to unlocking standard bulk RNA-seq tools for zero-inflated data.

The BCV plot for the Islam et al. [84] scRNA-seq dataset (Figure 3.2d) shows similar striped patterns as for zero-inflated bulk RNA-seq data. Such patterns are observed in many single-cell datasets (Figure 3.8). ZINB-WaVE identifies many zeros to be excess for the Islam dataset. It also provides good classification power for excess zeros for data simulated from the Islam dataset (Figure 3.2e). Incorporating the ZINB-WaVE weights in an edgeR analysis removes the striped patterns and yields a BCV plot that is similar to that for bulk RNA-seq data (Figure 3.2f), suggesting that zero inflation was indeed present and accounted for.

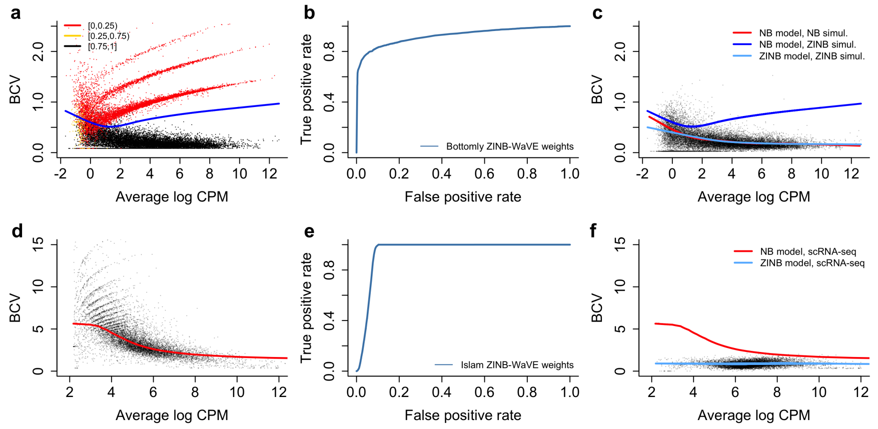


Figure 3.2: *Impact of zero inflation on mean-variance relationship for simulated bulk RNA-seq and Islam scRNA-seq datasets.* Zero inflation distorts the mean-variance trend in (sc)RNA-seq data, but is correctly identified by the ZINB-WaVE method. The top panels represent simulated data based on the Bottomly et al. [101] bulk RNA-seq dataset (as in Figure 3.1), for a two-group comparison with five samples in each group, where 5% of the counts were randomly replaced by zeros. The bottom panels represent the scRNA-seq dataset from Islam et al. [84]. **(a)** The BCV plot shows that randomly replacing 5% of the read counts with zeros induces zero inflation and distorts the mean-variance trend by leading to overestimated dispersion parameters. Points are color-coded according to the average ZINB-WaVE posterior probability for all zeros for a given gene and the blue line represents the mean-variance trend estimated with edgeR. **(b)** Receiver operating characteristic (ROC) curve for the identification of excess zeros by the ZINB-WaVE method. A very good classification precision is obtained. **(c)** Downweighting excess zeros using the ZINB-WaVE posterior probabilities recovers the original mean-variance trend (as indicated with the red line) and inference on the negative binomial count component will now no longer be biased because of zero inflation. The light blue line represents the estimated mean-variance trend for ZINB-WaVE-weighted edgeR. The blue line is the trend estimated by unweighted edgeR on zero-inflated data as in panel (a). **(d)** The BCV plot for the Islam et al. [84] dataset illustrates the higher variability of scRNA-seq data as compared to bulk RNA-seq data (note the difference in y-axis scales between (a) and (d)). As in (a), zero inflation induces striped patterns leading to an overestimation of the NB dispersion parameter. **(e)** ROC curve for the identification of excess zeros by the ZINB-WaVE method for scRNA-seq data simulated from the Islam dataset using the simulation framework described in "Methods". A good classification precision is obtained, but note the difference with bulk RNA-seq data: The noisier scRNA-seq dataset makes excess zero identification harder. **(f)** Using the ZINB-WaVE posterior probabilities as observation weights results in lower estimates of the dispersion parameter, unlocking powerful differential expression analysis with standard bulk RNA-seq DE methods. The red line is the mean-variance trend for unweighted edgeR, as in panel (d), and the light blue line is the mean-variance trend for ZINB-WaVE-weighted edgeR. A similar pattern is observed for the simulated Islam dataset (Figure 3.32).

3.3.3 High power and false positive control on simulated (sc)RNA-seq data

We provide a scRNA-seq data simulation paradigm that retains gene-specific characteristics as well as global associations across all genes (see "Methods" for details). More specifically, we first estimate dataset-specific associations between zero abundance, sequencing depth, and average log counts per million (CPM), and next explicitly account for these associations in our simulation model (Figures 3.10–3.11).

The scRNA-seq simulation study is based on three datasets: the Islam et al. [84] dataset, comparing 48 embryonic stem cells to 44 embryonic fibroblasts in the mouse; a subset of the Trapnell et al. [114] dataset, comparing differentiating human myoblasts at the 48h (85 cells) and 72h (64 cells) timepoints; and a 10x Genomics peripheral blood mononuclear cells (PBMC) dataset (see "Real datasets" section in "Methods" for details). The datasets differ in throughput, sequencing depth, and extent of zero inflation, e.g., Figure 3.12 shows a higher proportion of excess zeros in the Islam dataset as compared to the Trapnell dataset, an observation further supported by the fact that the Islam and Trapnell datasets contain $\sim 65\%$ and $\sim 48\%$ zeros, respectively. 10x Genomics datasets are known to contain even more zeros; the evaluated subset of the PBMC dataset contains $\sim 87\%$ zeros. The simulated datasets successfully mimic the characteristics of the original datasets, as evaluated with the R package countsimQC [120] (Additional Files 2–4). This diverse range of datasets is therefore representative of scRNA-seq datasets that occur in practice and a suitable basis for method evaluation and comparison.

We evaluate method performance in terms of sensitivity and false positive control using false discovery proportion - true positive rate (FDP-TPR) curves. Figure 3.3 (Figure 3.13) illustrates that many methods break down on the simulated Islam dataset due to a high degree of zero inflation. Surprisingly, even methods specifically developed to deal with excess zeros, like SCDE and metagenomeSeq, suffer from poor performances, with MAST being a notable exception. The DESeq2 methods, however, are able to cope with the high degree of zero inflation. Note that, in general, it is a good strategy to disable the outlier imputation step in DESeq2, since it deteriorates performance on scRNA-seq data (Figure 3.14). Seurat, limma-voom, and SCDE have very low sensitivity. The methods based on ZINB-WaVE weights dominate all competitors in terms of sensitivity and specificity, providing high power, good false discovery rate (FDR) control, and sensible p-value distributions (Figure 3.15). Note that the remaining methods also suffer from poor FDR control.

Since zero inflation is fairly modest for the Trapnell dataset, most methods perform better than for the Islam simulation (Figure 3.3). The ZINB-WaVE-based methods and DESeq2 outperform the remaining methods in terms of sensitivity and provide good FDR control. edgeR is their closest competitor and the remaining methods provide much lower sensitivity and/or very liberal FDR control. Note how bespoke scRNA-seq methods seem to break down on datasets with a lower degree of zero inflation, often providing too liberal or too conservative p-value distributions, while ZINB-WaVE-based methods in general show a reasonable p-value distribution, with an enrichment of low p-values and approximately uniformly distributed larger p-values (Figure 3.16).

Typical 10x Genomics datasets contain a high number of cells with shallow sequencing depth, due to the extreme multiplexing of libraries. As a result, counts and hence estimated NB means are lower, making zeros more plausible according to the NB distribution and excess zeros thus harder to identify. This is picked up by the simulation framework, where only $\sim 8\%$ of the genes were simulated to have at least one excess zero in $n = 1,200$ samples. Bulk RNA-seq methods can hence be expected to be among the top performers. Figure 3.4 shows FDP-TPR curves for the 10x Genomics simulation study, demonstrating a good performance of bulk RNA-seq methods edgeR and DESeq2. ZINB-WaVE edgeR and ZINB-WaVE DESeq2 are among the top performers, having comparable or slightly lower performance as compared to their unweighted counterparts. MAST is their closest competitor, providing good sensitivity and FDR control. SCDE, NODES, metagenomeSeq, and limma-voom have lower sensitivity and/or very liberal FDR control as compared to the dominating methods. These results suggest that, in a scenario of low counts or low degree of zero inflation, ZINB-WaVE-weighted edgeR/DESeq2 reduce to standard unweighted edgeR/DESeq2, while other bespoke scRNA-seq tools may deteriorate in performance. This notion is further supported by results on simulated bulk RNA-seq data, where ZINB-WaVE-weighted edgeR/DESeq2 have a similar performance as standard unweighted edgeR/DESeq2 in the absence of zero inflation (Figure 3.17). Hence, adopting ZINB-WaVE-based DE methods provides a performance boost in zero-inflated applications, while performance is not significantly deteriorated in the absence of zero inflation.

All analyses performed in this work are based on estimating one common dispersion parameter across all genes for the ZINB-WaVE model. ZINB-WaVE allows the estimation of genewise dispersion parameters, however, this approach is much more computationally intensive and can be an order of magnitude slower. Figures 3.13 and 3.18 show that estimating genewise dispersion parameters does not seem to be required for calculating the ZINB-WaVE weights, since no gain in performance is achieved when doing so. Note that genewise dispersions are still estimated by edgeR and DESeq2 in the final DE inference procedure.

3.3.4 False positive rate control

We compared our ZINB-WaVE-weight-based method to commonly-used DE methods for mock comparisons based on two publicly available real scRNA-seq datasets. We assessed performance based on the per-comparison error rate (PCER), defined as the proportion of false positives (i.e., Type I errors) among all genes being considered for DE, where a gene is

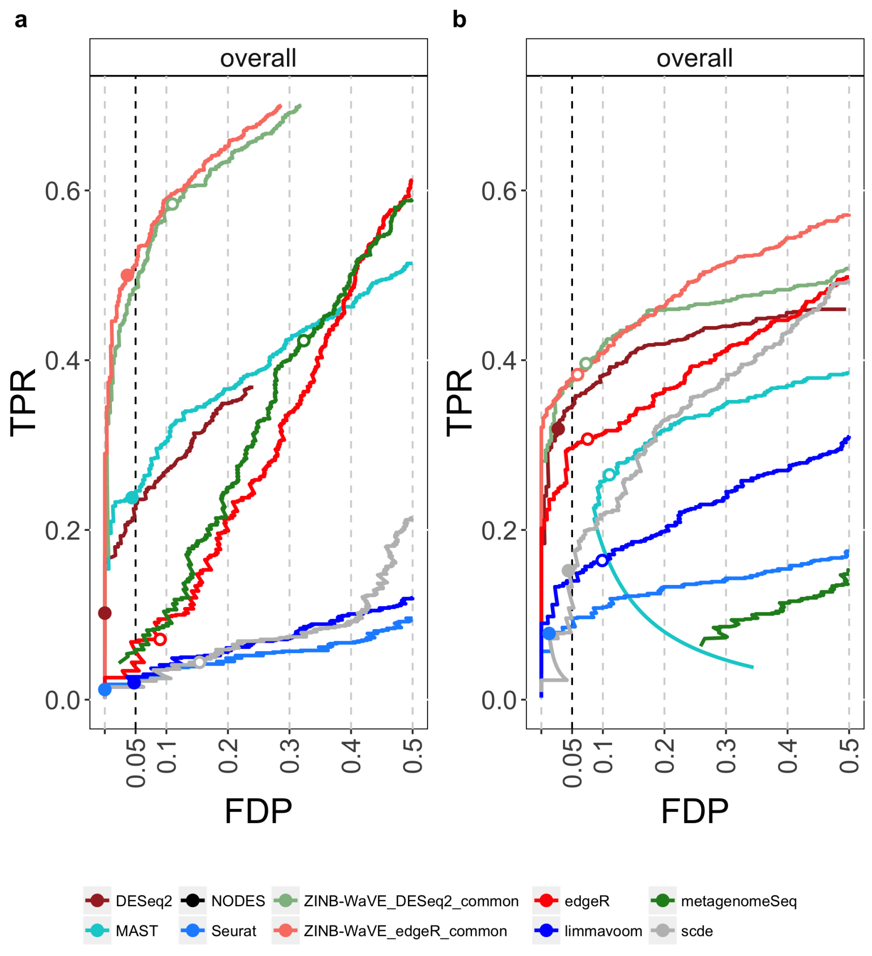


Figure 3.3: Comparison of DE methods on simulated scRNA-seq data. **(a)** scRNA-seq data simulated from Islam et al. [84] dataset ($n = 90$). **(b)** scRNA-seq data simulated from Trapnell et al. [114] dataset ($n = 150$). DE methods are compared based on scatterplots of the true positive rate (TPR) vs. the false discovery proportion (FDP); zoomed versions of the FDP-TPR curves are shown here, full curves are displayed in Figure 3.13. Circles represent working points on a nominal 5% FDR level and are filled if the empirical FDR (i.e., FDP) is below the nominal FDR. Methods based on ZINB-WaVE weights clearly outperform other methods for both simulated datasets. Note that the methods differ in performance between datasets, possibly because of a higher degree of zero inflation in the Islam dataset. The SCDE and metagenomeSeq methods, specifically developed to deal with excess zeros, are outperformed in both simulations by ZINB-WaVE-based methods and by DESeq2. The DESeq2 curve in panel (a) is cut off due to NA adjusted p-values resulting from independent filtering. The behavior in the lower half of the curve for MAST in (b) is due to a smooth increase in true positives with an identical number of false positives over a range of low FDR cut-offs. The curve for NODES is not visible on this figure, only in the full FDP-TPR curves.

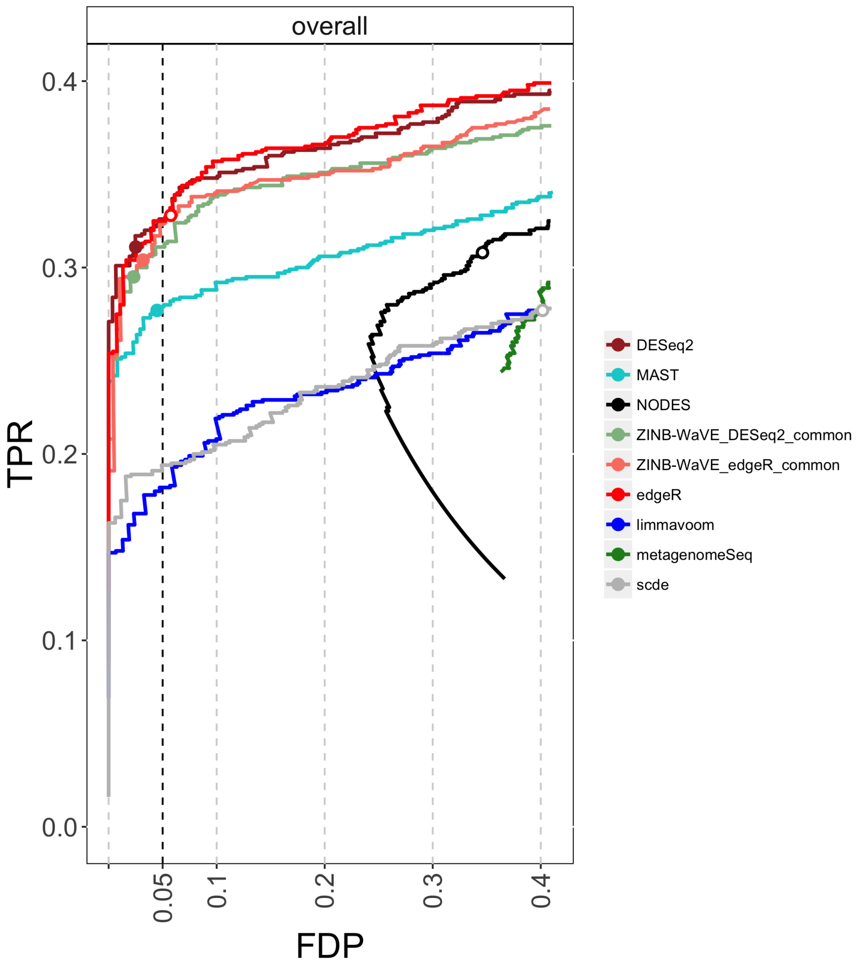


Figure 3.4: Comparison of DE methods on simulated scRNA-seq datasets. DE methods are compared based on FDP-TPR curves for data simulated from a 10x Genomics PBMC scRNA-seq dataset ($n = 1,200$); zoomed versions of the FDP-TPR curves are shown here, full curves are displayed in Figure 3.18. Circles represent working points on a nominal 5% FDR level and are filled if the empirical FDR (i.e., FDP) is below the nominal FDR. 10x Genomics sequencing typically involves high-throughput and massive multiplexing, resulting in very shallow sequencing depths and thus low counts, making it extremely difficult to identify excess zeros. Unweighted and ZINB-WaVE-weighted edgeR are tied for best performance, followed by ZINB-WaVE-weighted DESeq2. In general, bulk RNA-seq methods are performing well in this simulation, probably because the extremely high zero abundance in combination with low counts can be reasonably accommodated by the negative binomial distribution. The behavior in the lower half of the curve for NODES is due to a smooth increase in true positives with an identical number of false positives over a range of low FDR cut-offs.

declared DE if its nominal unadjusted p-value is less than or equal to 0.05.

The first dataset, referred to as Usoskin [79] dataset, concerns 622 mouse neuronal cells from the dorsal root ganglion, classified in eleven categories. The authors acknowledge the existence of a batch effect related to the picking session for the cells. We find that the batch effect is not only associated with expression measures, but also influences the relationship between sequencing depth and zero abundance (Figure 3.5a) [121]. The large differences in sequencing depths between batches attenuate the overall association with zero abundance when cells are pooled across batches (Figure 3.5a). We therefore added a covariate to account for the batch effect in both the negative binomial mean (μ) and the zero inflation probability (π) of the ZINB-WaVE model used to produce the weights for DE analysis. Adjusting for batch yields weights with a slightly higher mode near zero, suggesting a more informative discrimination between excess and NB zeros (Figure 3.5b). Although the batch effect is small in terms of the weights, it illustrates the generality and flexibility of our ZINB-WaVE weighting approach: through a suitable parameterization of both the NB mean and ZI probability one can adjust for effects that can bias the weights and hence the DE results.

For the Usoskin dataset, we assessed false positive control by comparing the *actual* vs. the *nominal* PCER for mock null datasets where none of the genes are expected to be differentially expressed. Specifically, we generated 30 mock datasets where, for each dataset, two groups of 45 cells each were created by sampling 15 cells at random, without replacement from each of the three picking sessions. Sampling cells within batch allows to control for potential confounding by the batch variable. For each of the 30 mock datasets, we considered seven methods to identify genes that are DE between the two groups and declared a gene DE if its nominal unadjusted p-value was less than or equal to 0.05. For these mock datasets, any gene declared DE between the two groups is a false positive. Thus, for each method, the nominal PCER of 0.05 is compared to the actual PCER which is simply the proportion of genes declared DE (Figure 3.5c–d).

The seven methods considered are: unweighted and ZINB-WaVE-weighted edgeR, unweighted and ZINB-WaVE-weighted DESeq2, unweighted limma-voom (ZINB-WaVE-weighted limma-voom was found to perform poorly in the simulation study and hence is not considered here), MAST, and SCDE (see “Methods” for details). edgeR and DESeq2 with ZINB-WaVE weights and unweighted edgeR controlled the PCER close to its nominal level (Figure 3.5c). The unweighted versions of DESeq2, MAST, and SCDE tended to be conservative, whereas limma-voom tended to be anti-conservative. In addition, the weighted versions of edgeR and DESeq2 and unweighted edgeR yielded near uniform p-value distributions (as expected under this complete null scenario), while unweighted DESeq2, MAST, and SCDE tended to yield conservative p-values (mode near 1) and limma-voom anti-conservative p-values (mode near 0) (Figure 3.5d).

We also replicated the original analysis of Usoskin et al. [79], by performing one-against-

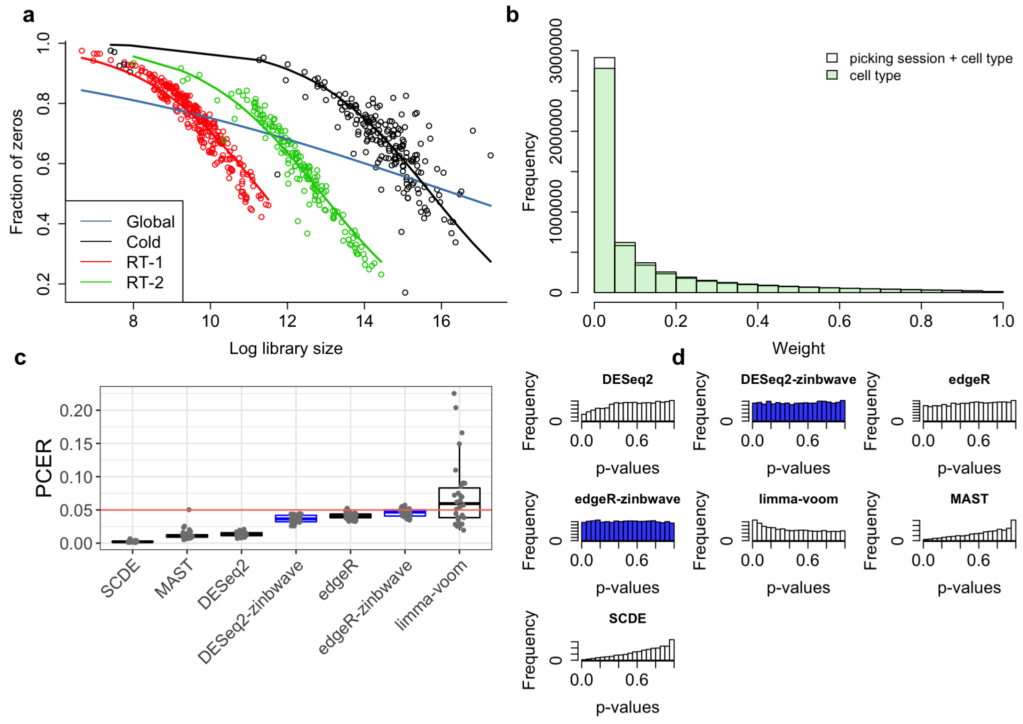


Figure 3.5: *False positive control on mock null Usoskin datasets (n = 622 cells).* **(a)** The scatterplot and GLM fits (R `glm` function with `family=binomial`), color-coded by batch (i.e., picking sessions Cold, RT-1, and RT-2), illustrate the association of zero abundance with sequencing depth. The three batches differ in their sequencing depths, causing an attenuated global relationship when pooling cells across batches (blue curve). Adjusting for the batch effect in the ZINB-WaVE model allows to properly account for the relationship between sequencing depth and zero abundance. **(b)** Histogram of ZINB-WaVE weights for zero counts for original Usoskin dataset, with (white) and without (green) including batch as a covariate in the ZINB-WaVE model. The higher mode near zero for batch adjustment indicates that more counts are being classified as dropouts, suggesting more informative discrimination between excess and NB zeros. **(c)** Boxplot of per-comparison error rate (PCER) for 30 mock null datasets for each of seven DE methods; ZINB-WaVE-weighted methods are highlighted in blue. **(d)** Histogram of unadjusted p-values for one of the datasets in (c). ZINB-WaVE was fit with intercept, cell type covariate (actual or mock), and batch covariate (unless specified otherwise) in X, $V = \mathbf{1}_J$, $K = 0$ for W, common dispersion, and $\epsilon = 10^{12}$.

all tests of DE for each cell type (Figure 3.19). limma-voom found a high number of DE genes, confirming our results from the mock evaluations where it was too liberal. The ZINB-WaVE methods tended to find a high number of DE genes, which is promising combined with the good PCER control seen in the mock comparisons. While introducing ZINB-WaVE weights in DESeq2 lead to a higher number of significant genes on average, the effect is less clear with edgeR and seems to depend on the contrast.

Similar results were observed for a 10x Genomics PBMC dataset comprising 2,700 single cells sequenced on the Illumina NextSeq 500 (Figure 3.20), with the distinction that we found a conservative p-value distribution for ZINB-WaVE-weighted DESeq2. Since no information was provided about potential batch effects, we did not consider batch covariates for this dataset.

Additionally, we examined the PCER and p-value distributions on mock comparisons while varying the regularization parameter (ϵ) for the ZINB-WaVE estimation procedure. Not surprisingly, we observed that the PCER decreases with increasing ϵ, i.e., as the parameters of the ZINB-WaVE model are subjected to more “shrinking” (Figures 3.21 and 3.22 for Usoskin and 10x Genomics PBMC datasets, respectively).

3.3.5 Biologically meaningful clustering and differential expression results

To analyze the 2,700 cells from the 10x Genomics PBMC dataset (see “Methods”), we followed the tutorial available at `http://satijalab.org/seurat/pbmc3k_tutorial.html` and used the R package Seurat [118]. The major steps of the pipeline were quality control, data filtering, identification of high-variance genes, dimensionality reduction using the first ten components from principal component analysis (PCA), and graph-based clustering. The final step of the pipeline was to identify genes that are differentially expressed between clusters, in order to derive cell type signatures. Two different parameterizations were used for the Seurat clustering. With one parameterization, a single cluster was identified for CD4+ T-cells, while with another, two CD4+ T-cell subclusters were identified, corresponding to CD4+ naive T-cells and CD4+ memory T-cells (gold and red clusters in Figure 3.6a, respectively). At the end of the tutorial, the authors concluded that the memory/naive split was weak and more cells would be needed to have a better separation between the two CD4+ T-cell subclusters.

In order to find DE genes between the two CD4+ T-cell subclusters, we used Seurat, unweighted edgeR, ZINB-WaVE-weighted edgeR, MAST, and limma-voom. We then sought to identify cell types using gene set enrichment analysis (GSEA), with the function `fgsea` from the Bioconductor R package fgsea [115] and gene sets for 64 immune and stroma cell types from the R package xCell [116]. While unweighted edgeR found that one cluster was enriched in both CD4+ memory and naive T-cells compared to the other cluster, our weighted-edgeR method as well as Seurat, and limma-voom found that the cluster was enriched in CD4+ T-effector memory, CD4+ T-central memory, and CD4+ memory T-cells, and depleted in CD4+ naive T-cells. MAST found that the cluster was depleted in CD4+ memory T-cells and CD4+ naive T-cells, but enriched in CD4+ T-effector memory and CD4+ T-central memory T-cells (see Figure 3.6b and Figure 3.23). These results suggest that our ZINB-WaVE weights can successfully unlock edgeR for zero-inflated data, leading to biologically meaningful DE genes.

While ZINB-WaVE can be used to compute weights in a supervised setting with *a priori* known cell types, it can also be used to perform dimensionality reduction in an unsupervised setting. To demonstrate the ability of our method to find biologically relevant clusters and DE genes, we performed dimensionality reduction using ZINB-WaVE with $K = 20$ unknown covariates (matrix W, see “Methods”), where $K = 20$ was chosen using the Akaike information criterion (AIC) (Figure 3.24). We then used W, instead of the first 10 components of PCA as in the Seurat tutorial, to cluster the cells using the Seurat graph-based clustering. We found similar clusters as the Seurat clusters, except for the NK-cell and B-cell clusters which were partitioned differently and the cluster with CD4+ T-cells (Figure 3.25). Using this new clustering, GSEA showed a better separation between CD4+ naive T-cells and CD4+ memory T-cells for all the methods, suggesting a biological meaningful clustering using ZINB-WaVE dimensionality reduction instead of PCA. The CD4+ T-effector memory, CD4+ T-central memory, and CD4+ memory cell types were enriched using limma-voom, unweighted edgeR, MAST, and Seurat, but only the CD4+ T-central memory cell type was depleted using our weighted edgeR method (Figures 3.6c and Figure 3.23). As we do not have prior knowledge about the cells in the different clusters, we are unable to say whether the cluster is more representative of the CD4+ T-effector memory cell type or if our method missed the enrichment in the CD4+ T-central memory cell type. However, it is interesting that using ZINB-WaVE to account for zero inflation in the clustering allowed edgeR to find results that seem more biologically meaningful than without accounting for zero inflation.

Finally, using a Benjamini and Hochberg [107] adjusted p-value cut-off of 0.05, limma-voom declared 433 and 194 DE genes and weighted-edgeR 371 and 151, for clustering based on, respectively, the first 10 PCs and W from ZINB-WaVE. We additionally showed on mock comparisons for the same 10x Genomics PBMC dataset that limma-voom had a greater actual PCER than weighted edgeR (Figure 3.20), suggesting that some of the DE genes found by limma-voom are likely to be false positives. This belief is reinforced by the skewed distribution of limma-voom p-values (Figure 3.26).

3.3.6 Alternative approaches to weight estimation

ZINB-WaVE is one particular approach to fit a ZINB model to single-cell RNA-seq data. However, our proposed data analysis strategy to unlock conventional RNA-seq tools with

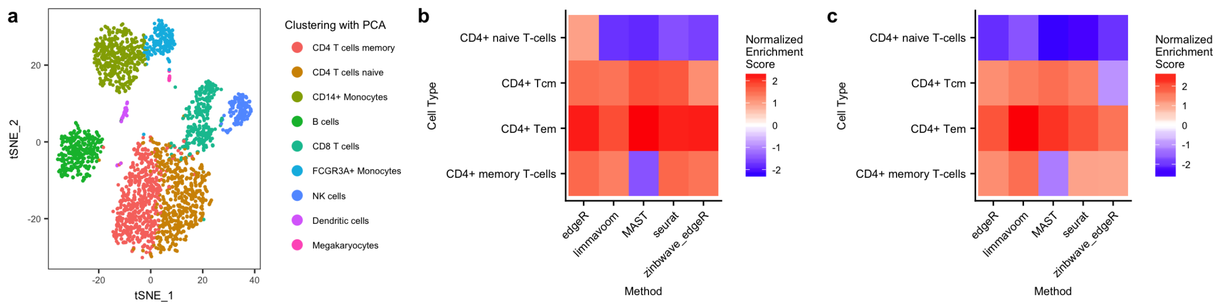


Figure 3.6: *Biologically meaningful DE results for 10x Genomics PBMC dataset.* **(a)** Scatterplot of the first two t-SNE dimensions obtained from the first 10 principal components. Cells are color-coded by clusters found using the Seurat graph-based clustering method on the first 10 principal components. Pseudo-color images on the right display normalized enrichment scores (NES) after gene set enrichment analysis (GSEA) for cell types related to CD4+ T-cells (see "Methods"), for clustering based on **(b)** the first 10 principal components and **(c)** W from ZINB-WaVE with $K = 20$. For dimensionality reduction, ZINB-WaVE was fit with $X = \mathbf{1}_n$, $V = \mathbf{1}_J$, $K = 20$ for W (based on AIC), common dispersion, and $\epsilon = 10^{12}$. To compute the weights for DE analysis, ZINB-WaVE was fit with intercept and cell type covariate in X, $V = \mathbf{1}_J$, $K = 0$ for W, common dispersion, and $\epsilon = 10^{12}$. NES for more cell types are shown in Figure 3.23.

ZINB observation-level weights is not restricted to ZINB-WaVE-based workflows. In particular, we illustrate the use of weights estimated by the zingeR method, an expectation-maximization (EM) algorithm which we developed earlier and that builds upon edgeR for estimating the NB parameters of the ZINB model [122]. The ZINB-WaVE and zingeR approaches differ in the following respects. The zingeR weights are based on a constant cell-specific excess zero probability π_i for each cell i, while the ZINB-WaVE excess zero probability π_{ij} is both cell and gene-specific, a strategy that was also advocated in recent methods [90, 87]. Secondly, the ZINB-WaVE negative binomial mean μ and zero inflation probability π are modeled in terms of both wanted and unwanted cell and gene-level covariates, allowing normalization for a variety of nuisance technical effects. Thirdly, different parameter estimation strategies are adopted: parameters from the zingeR model are estimated with an EM algorithm, whereas those from the ZINB-WaVE model are estimated using a penalized maximum likelihood approach. Finally, methods based on zingeR weights have the desirable property of converging to their unweighted counterparts in the absence of zero inflation.

In terms of performance, based on the simulation study on full-length protocols, zingeR workflows dominate both bulk RNA-seq and dedicated scRNA-seq methods, but were found to be inferior in terms of sensitivity to ZINB-WaVE workflows (Figure 3.27). However, for the Usoskin dataset, zingeR seems to find a higher number of DE genes than ZINB-WaVE and than its bulk RNA-seq counterparts (Figure 3.28), while also controlling the PCER in mock evaluations (Figure 3.29). However, the computational burden of the zingeR method prevented us from applying it to large-scale datasets, such as those from the 10x Genomics platform, thus limiting our comparison.

3.3.7 Computational time

The better performance of our ZINB-WaVE-weighted DE method comes at a computational cost, since we first fit ZINB-WaVE to the entire cells-by-genes matrix of read counts to compute the weights and then use a weighted version of DESeq2 or edgeR for inferring DE. To give the reader an idea of how different methods scale in terms of computation time, we benchmarked three different datasets: the Islam dataset (92 cells), one of the mock null Usoskin datasets used in Figure 3.5 (90 cells), and the CD4+ T-cell cluster of the 10x Genomics PBMC dataset ($1,151$ cells). For each dataset, $10,000$ genes were sampled at random and the two cell types were used as covariates. For the Usoskin dataset, batch was added as a covariate for all methods. For all datasets, the fastest method was limma-voom followed by edgeR (Figure 3.30). As DESeq2 was slower than edgeR, not surprisingly weighted-DESeq2 was also slower than weighted-edgeR, especially for the 10x Genomics PBMC dataset.

3.4 Discussion

This manuscript focused on adapting standard bulk RNA-seq differential expression tools to handle the severe zero inflation present in single-cell RNA-seq data. We proposed a simple and general approach that integrates seamlessly with a range of popular DE software packages, such as edgeR and DESeq2. The main idea is to use weights for zero inflation in the negative binomial model underlying bulk RNA-seq methods, where the weights are based on the ZINB-WaVE method of Risso et al. [91]. The general and flexible ZINB-WaVE framework allows to extract low-dimensional signal from scRNA-seq read counts, accounting for zero inflation (e.g., dropouts), over-dispersion, and the discrete nature of the data. In particular, the ZINB-WaVE model allows for read count normalization through an appropriate parameterization of the negative binomial means and zero inflation probabilities in terms of both gene and cell-level covariates.

Our results complement the findings of Jaakkola et al. [97] and Soneson and Robinson [98], that bespoke scRNA-seq tools do not systematically improve upon bulk RNA-seq tools. Although MAST, metagenomeSeq, and SCDE were explicitly developed to handle excess zeros, they suffer from poor performance in a high zero inflation setting, as demonstrated in the simulation study.

The value of our method was demonstrated for scRNA-seq protocols relying on both standard (Islam, Usoskin, and Trapnell datasets) and unique molecular identifier (UMI) (10x Genomics PBMC dataset) read counting. UMIs were recently proposed to reduce measurement variability across samples [83]. In UMI-based protocols, transcripts are labeled with a small random UMI barcode prior to amplification. After amplification and sequencing, one enumerates the unique UMIs found for every transcript, which correspond to individual sequenced UMI-labeled transcripts. There is some evidence in the literature that zero inflation is less of a problem for UMI-based than for full-length protocols and that UMI read counts could follow a negative binomial distribution [123, 124]. Hence, our method also provides good results for UMI-based data with limited zero inflation, demonstrating its broad applicability.

In the simulation study, power to detect DE was generally lower for 10x Genomics UMI datasets (Figure 3.4) than for full-length protocol datasets (Figure 3.3). While the 10x Genomics platform has the advantage of an extremely high throughput, allowing many cells to be characterized, the resulting datasets often have the disadvantage of low library sizes, a logical consequence of UMI counting and of the trade-off between sequencing depth and number of cells to be sequenced in one sequencing run. As a result, the sequencing depth of these datasets is much lower than that of bulk RNA-seq datasets, making it harder to identify excess zeros and assess differential expression, even in large sample size settings.

Although the 10x Genomics platform may be well suited for hypothesis generation, e.g., through cell type discovery or lineage trajectory studies, full-length protocols may be more appropriate for discovering marker genes between inferred cell types or trajectories, an approach that has also been adopted in previous studies [125].

We have used ZINB-WaVE in conjunction with either edgeR or DESeq2. However, the ZINB-WaVE posterior probabilities could be used as weights to unlock other standard RNA-seq workflows in zero inflation situations. Figure 3.13 shows that ZINB-WaVE weights combined with heteroscedastic weights in limma-voom also increase power in a scRNA-seq context, although this may be at the expense of Type I error control.

The ZINB-WaVE method penalizes the L2 norm of the parameter estimates for regularization purposes. It requires a penalty parameter, ϵ, that is rescaled differently for gene-specific parameters, cell-specific parameters, and dispersion parameters [91]. All analyses in this manuscript were performed with $\epsilon = 10^{12}$, to provide consistently comparable results. However, the optimal value of ϵ is dataset-specific and further research is needed to provide a data-driven approach for selecting an optimal ϵ. Indeed, based on our simulations, the value of the penalty parameter can have a profound influence on results (Figure 3.31), but we found $\epsilon = 10^{12}$ to have generally good performance.

ZINB-WaVE allows the option to infer latent variables W, which may correspond to either unmeasured confounding covariates or unmeasured covariates of interest. The observational weights were computed with the number of unknown covariates $K = 0$, i.e., no latent variables were inferred. For clustering of the real datasets, we inferred an optimal choice of K using the AIC (Figure 3.24). However, further investigation is needed to confirm that the AIC is appropriate for selecting K.

In principle, our proposed ZINB-WaVE model could also be used to identify DE genes both in terms of the negative binomial mean and the zero inflation probability, reflecting, respectively, a continuum in DE and a more binary (i.e., presence/absence) DE pattern. In this context, the parameters of interest are regression coefficients β corresponding to known sample-level covariates in the matrix X used in either μ or π (Equation (3.2)). Differentially expressed genes may be identified via likelihood ratio tests or Wald tests, with the standard errors of estimators of β obtained from the inverse of the Hessian matrix of the likelihood function. However, both types of tests would be computationally costly, as likelihood ratio tests would require refitting the entire model for each gene and Wald tests would require the Hessian matrix to be computed and inverted.

In this contribution, we have proposed to estimate the weights using ZINB-WaVE, but other approaches are possible. It is important to note that while methods such as ZINB-WaVE and zingeR can successfully identify excess zeros, they cannot however readily discriminate between their underlying causes, i.e., between technical (e.g., dropout) and biological (e.g., bursting) zeros. Although we cannot make this distinction with the weights, an increase in bursting rates between cell types, characterized by higher counts and more zeros [126], can however be picked up by the count component of the ZINB model.

3.5 Conclusions

In summary, we provide a realistic simulation framework for single-cell RNA-seq data and use the well-tested ZINB-WaVE method to successfully identify excess zeros and yield gene and cell-specific weights for differential expression analysis in scRNA-seq experiments. The tools we have developed allow an integrated workflow for normalization, dimensionality reduction, cell type discovery, and the identification of cell type marker genes. We confirmed that state-of-the-art scRNA-seq tools do not improve upon common bulk RNA-seq tools for differential expression analysis based on scRNA-seq data. Our workflow, however, outperforms current methods and has the merit of not deteriorating in performance in the absence of zero inflation. Inference of DE is focused on the count component of the ZINB model and our method produces posterior probabilities that can be used as observation-level weights by bulk RNA-seq tools. Hence, our approach unlocks widely-used bulk RNA-seq DE workflows for zero-inflated data and will assist researchers, data analysts, and developers in improving power to detect DE in the presence of excess zeros. The framework is general and applicable beyond scRNA-seq, to zero-inflated count data structures arising in applications such as metagenomics [103, 127].

3.6 Supplement

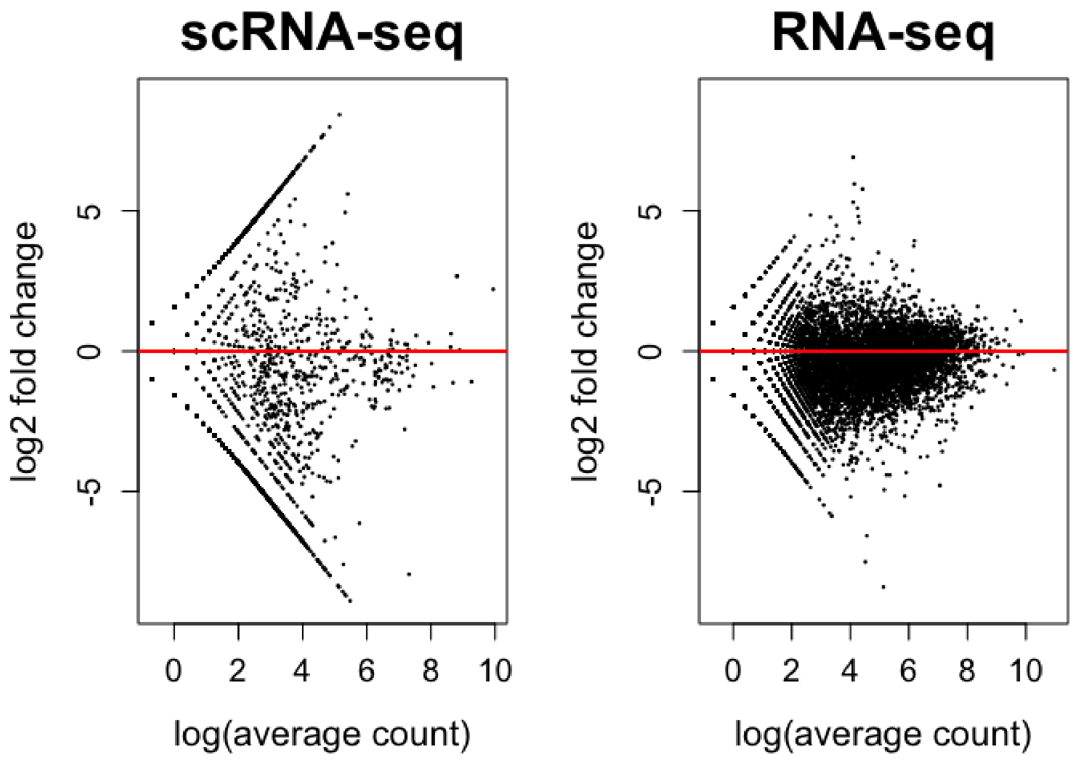


Figure 3.7: *Variability in bulk and single-cell RNA-seq data.* Mean-difference plots for two samples from the Islam et al. [84] scRNA-seq dataset (left panel) and two samples from the Pickrell et al. [128] bulk RNA-seq dataset (right panel). A higher variability in the scRNA-seq data is observed as compared to the bulk RNA-seq data.

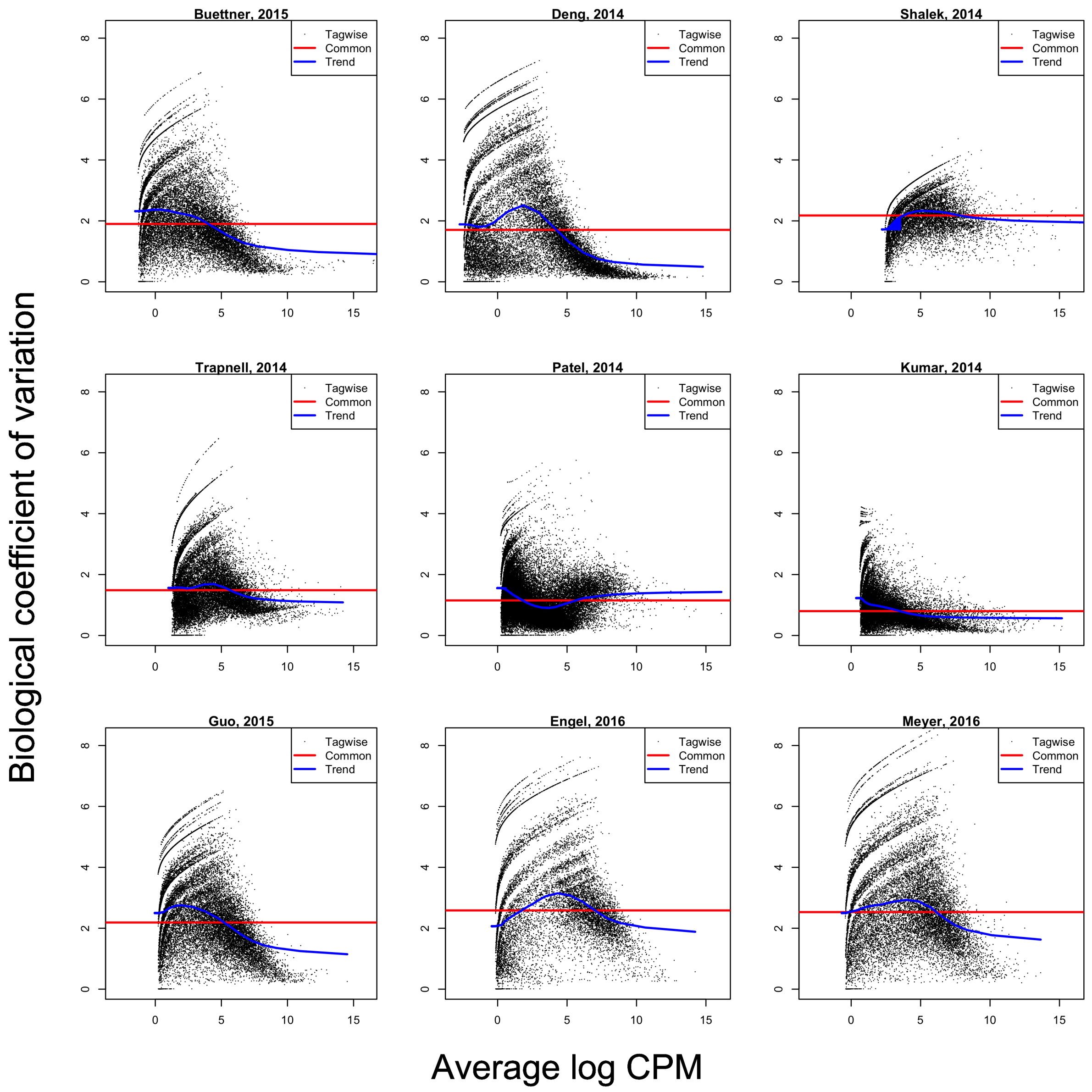


Figure 3.8: *BCV plots for conquer datasets.* Estimated biological coefficient of variation (BCV) vs. average log count per million (CPM), computed by edgeR, for conquer scRNA-seq datasets subsampled to $n = 10$ cells. The striped patterns reflect genes with many zero counts and high dispersion estimates, distorting the mean-variance relationship.

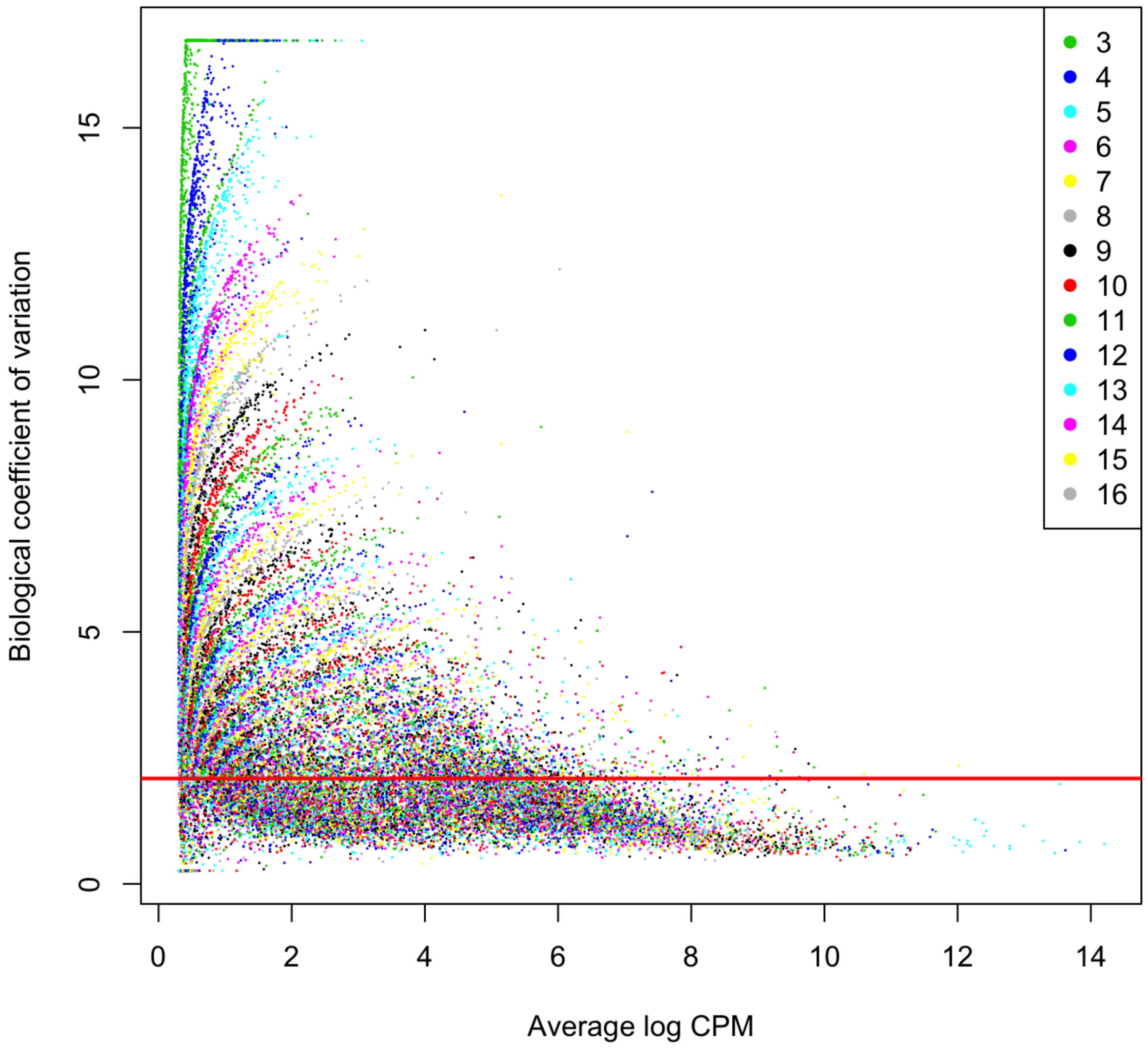


Figure 3.9: *BCV plot for Trapnell dataset.* Estimated biological coefficient of variation (BCV) vs. average log count per million (CPM), computed by edgeR, for the 72h subset of the Trapnell et al. [114] scRNA-seq dataset, where colors represent the total number of positive counts across cells. The striped patterns originate from genes with few positive counts. They are also present in the lower half of the BCV plot, but can only be noticed with the coloring. The red line indicates the common dispersion estimated with edgeR.

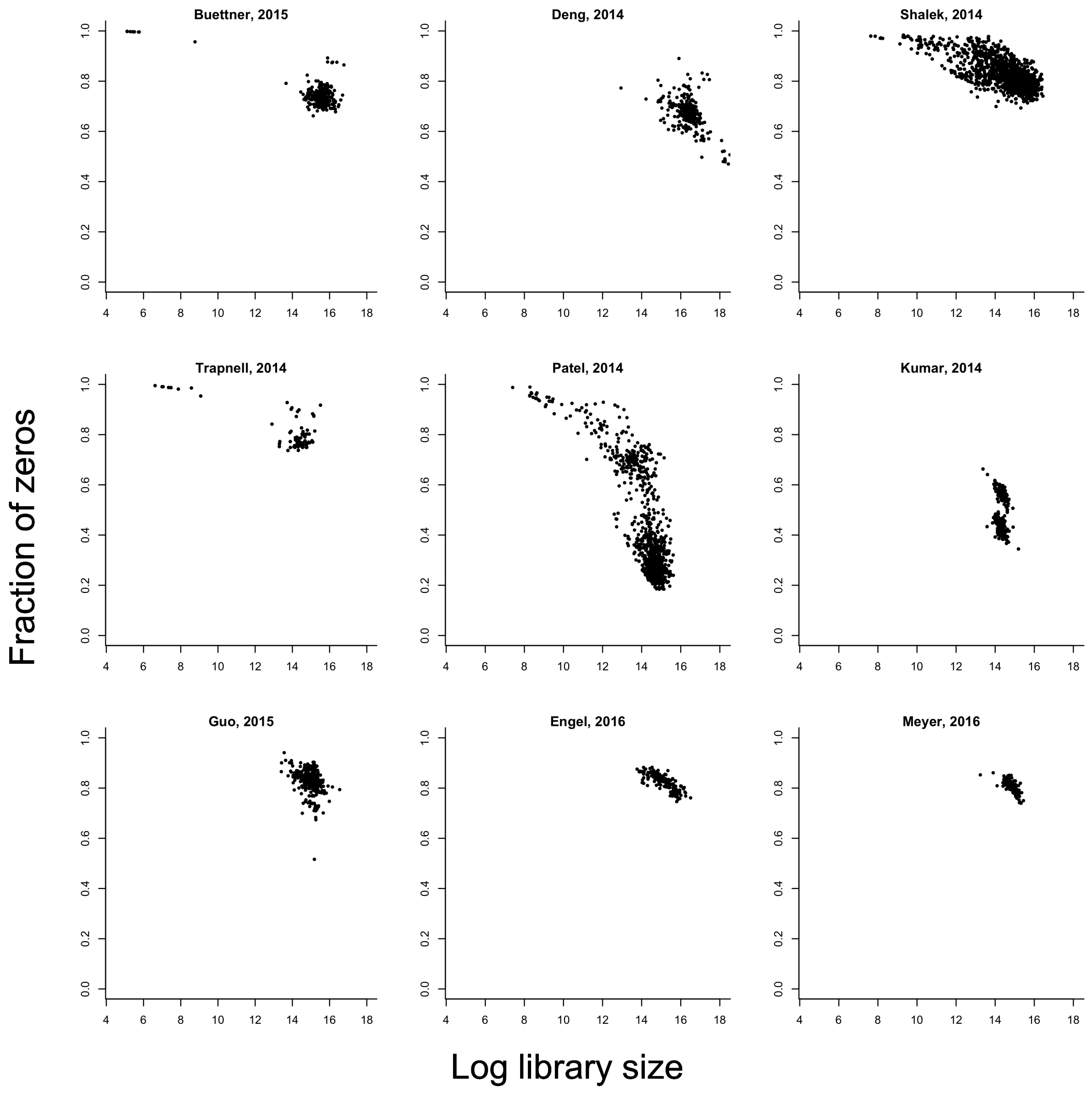


Figure 3.10: *Zero proportion vs. log library size for conquer datasets.* The fraction of zero counts for a cell is associated with library size, for scRNA-seq datasets downloaded from the conquer repository.

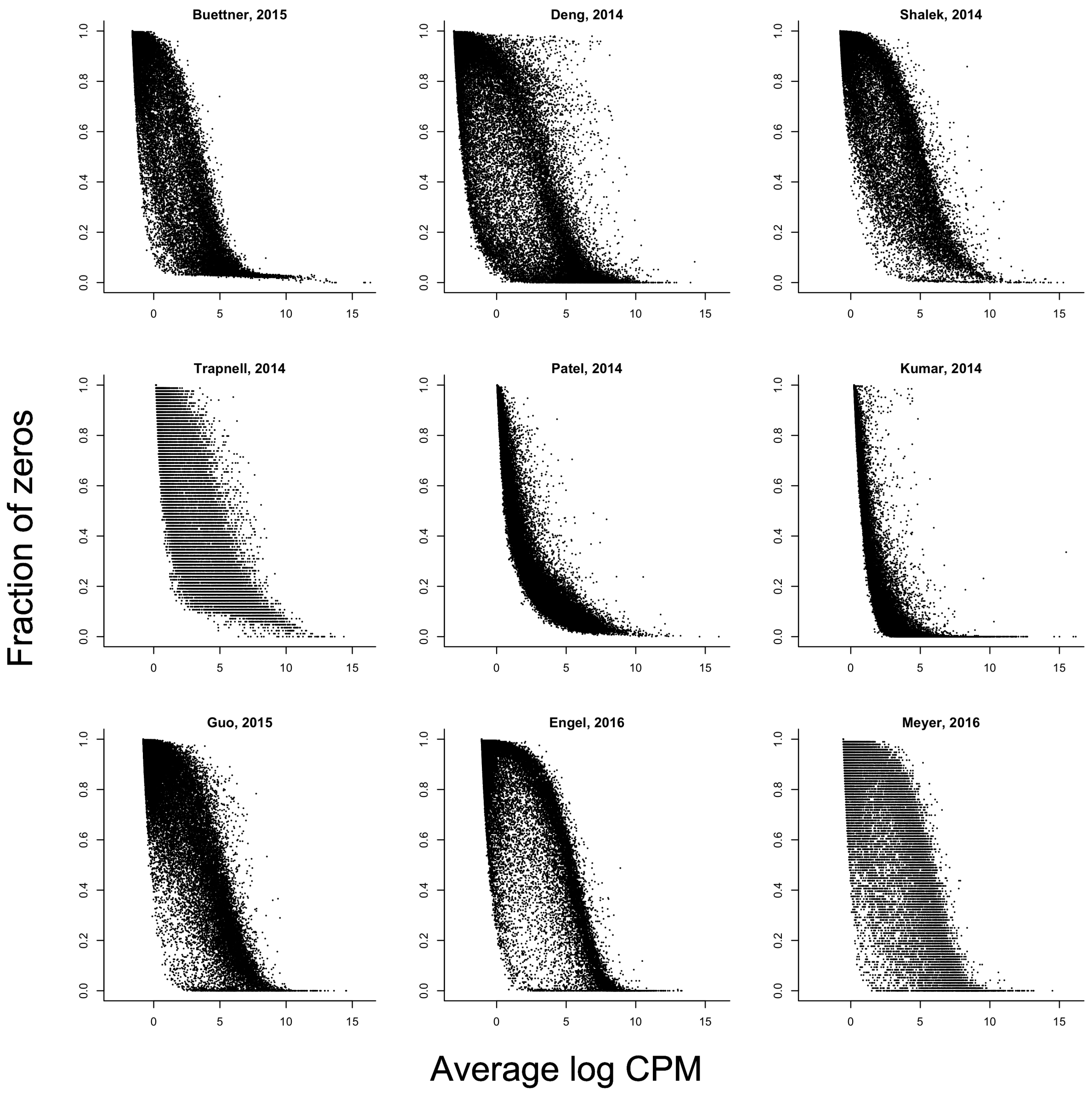


Figure 3.11: *Zero proportion vs. average log CPM for conquer datasets.* The fraction of zero counts for a gene is associated with its average expression, measured by the average log count per million (CPM), for scRNA-seq datasets downloaded from the conquer repository.

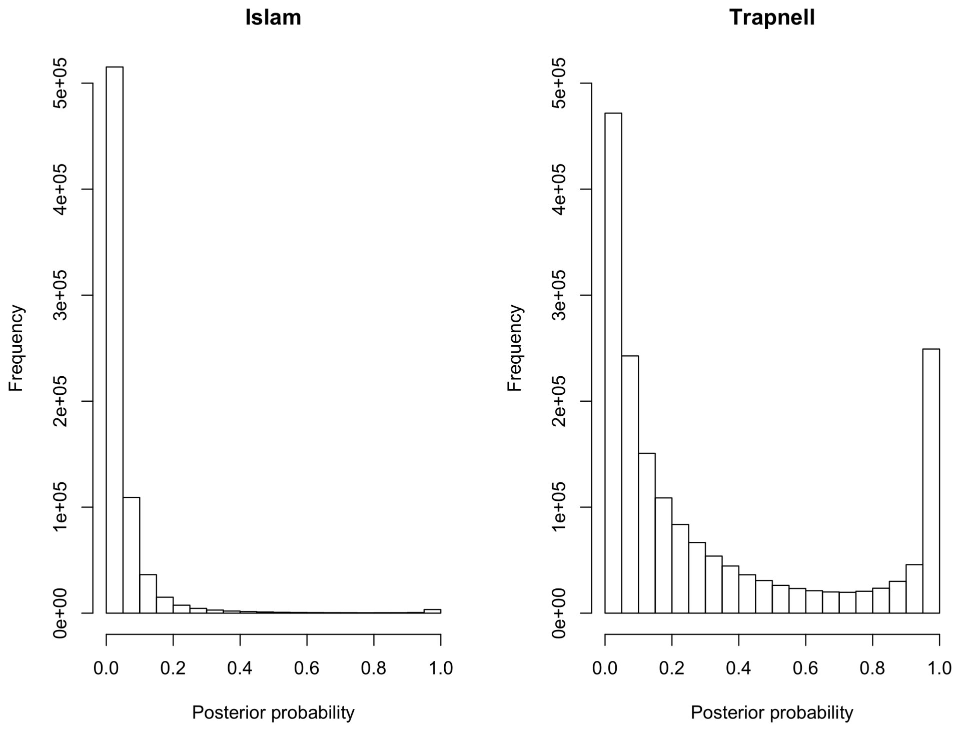


Figure 3.12: *ZINB-WaVE posterior probabilities on real Islam and Trapnell scRNA-seq datasets.* The histograms display the ZINB-WaVE estimated posterior probabilities of belonging to the negative binomial count component for all zeros in the Islam et al. [84] and Trapnell et al. [114] datasets. Many zeros are identified as excess zeros in the Islam dataset, while in the Trapnell dataset a reasonable proportion are estimated to be NB zeros.

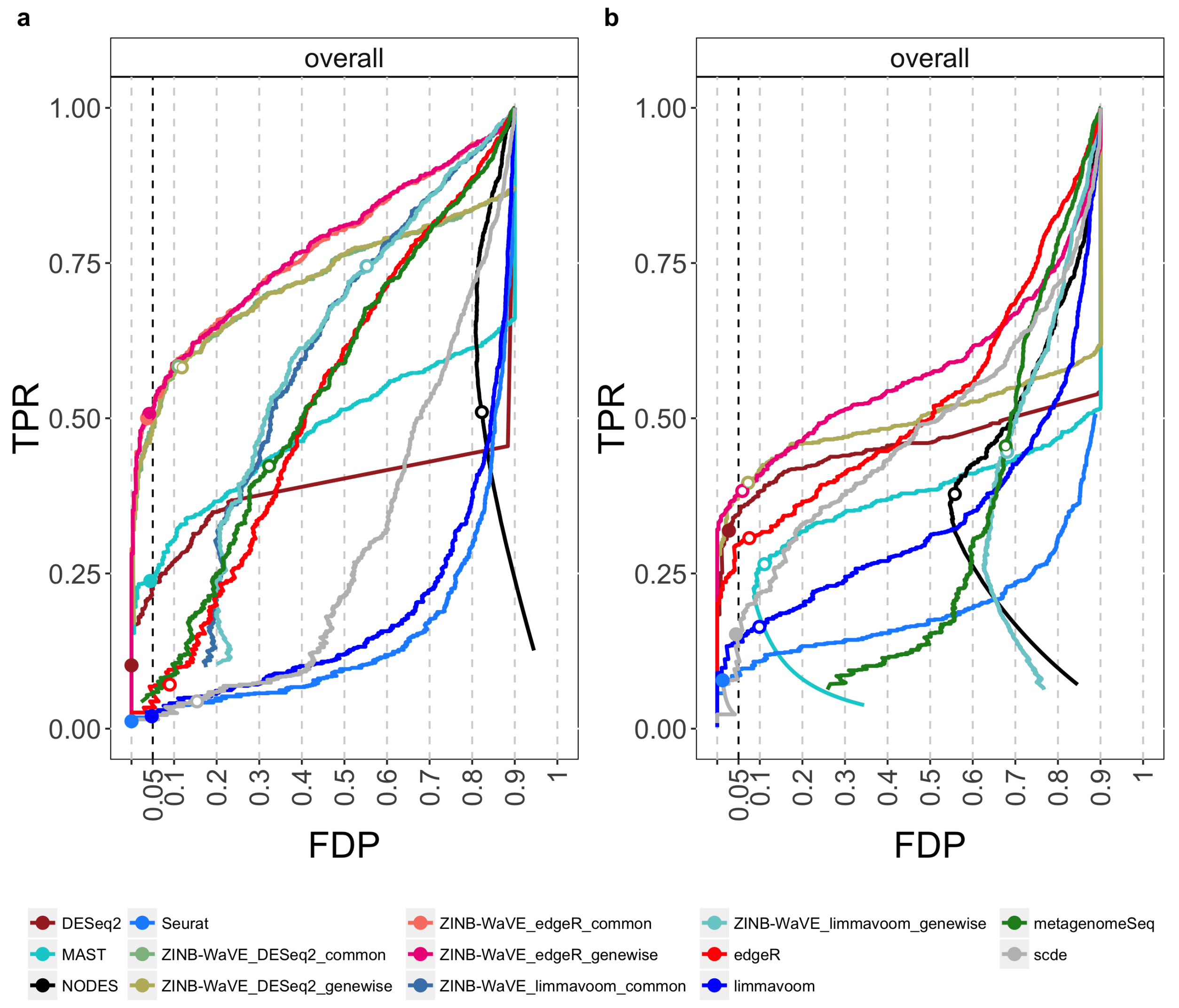


Figure 3.13: *Comparison of DE methods on simulated scRNA-seq datasets.* **(a)** scRNA-seq data simulated from Islam et al. [84] dataset ($n = 90$). **(b)** scRNA-seq data simulated from Trapnell et al. [114] dataset ($n = 150$). As in Figure 3.3, DE methods are compared based on scatterplots of the true positive rate (TPR) vs. the false discovery proportion (FDP). Circles represent working points on a nominal 5% FDR level and are filled if the empirical FDR (i.e., FDP) is below the nominal FDR. Methods based on ZINB-WaVE weights clearly outperform other methods for both simulated datasets. Note that the methods differ in performance between datasets, possibly because of a higher degree of zero inflation in the Islam dataset. The SCDE and metagenomeSeq methods, specifically developed to deal with excess zeros, are outperformed in both simulations by ZINB-WaVE-based methods and by DESeq2.

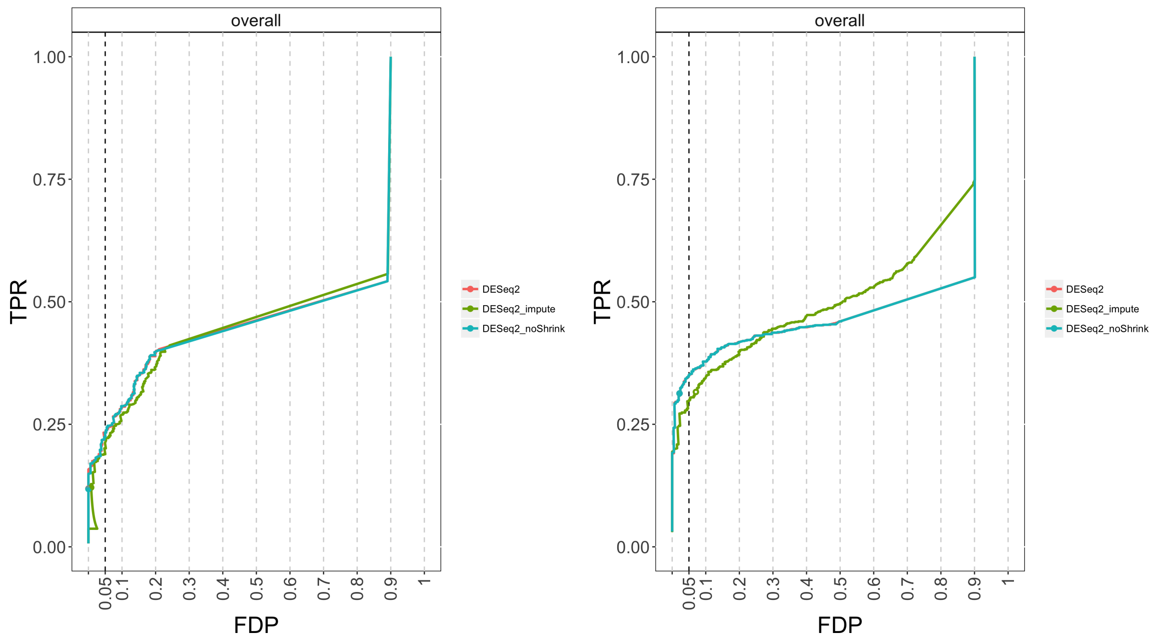


Figure 3.14: *Comparison of DESeq2 variants on simulated scRNA-seq data.* **(a)** scRNA-seq data simulated from Islam et al. [84] dataset ($n = 90$). **(b)** scRNA-seq data simulated from Trapnell et al. [114] dataset ($n = 150$). Three DESeq2 variants are compared based on scatterplots of the true positive rate (TPR) vs. the false discovery proportion (FDP). Circles represent working points on a nominal 5% FDR level and are filled if the empirical FDR (i.e., FDP) is below the nominal FDR. Enabling the imputation step in DESeq2 (method `DESeq2_impute`) results in a deterioration of performance on both datasets. Our default DESeq2 analysis (method `DESeq2`) has shrinkage of the fold-changes enabled, but disabling this option (method `DESeq2_noShrink`) does not seem to have a detrimental effect on performance. Note that the curve for `DESeq2_noShrink` is superimposed on the curve for `DESeq2` due to approximately identical performance.

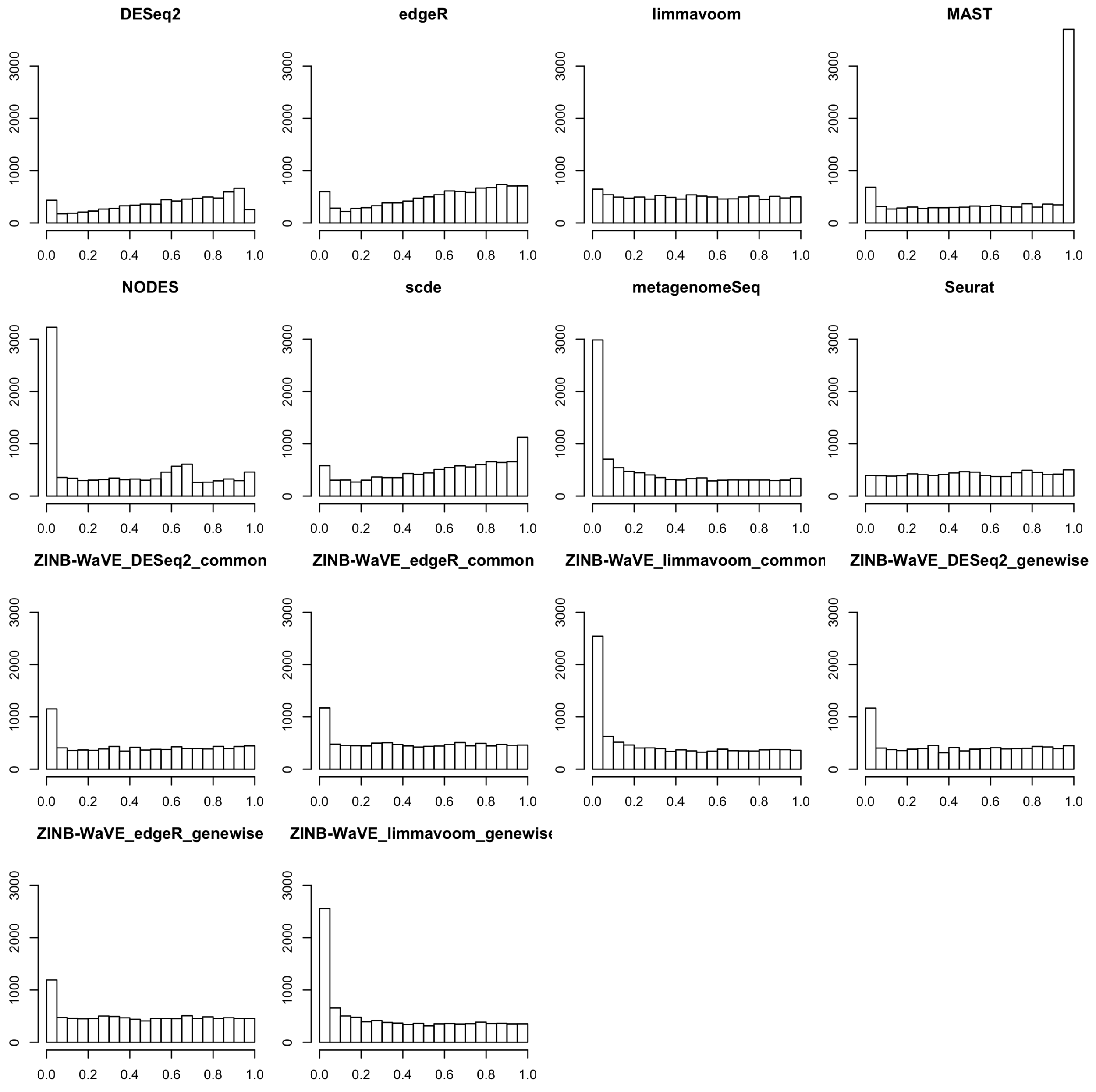


Figure 3.15: *p-value distributions on simulated Islam scRNA-seq dataset (n = 90).* The p-value distributions for methods based on ZINB-WaVE weights appear uniform for large p-values and enriched for low p-values, as would be expected from an appropriate statistical inference method on datasets with known effects. metagenomeSeq, NODES, and ZINB-WaVE-weighted limma-voom have anti-conservative p-value distributions, also evident in the FDP-TPR curves. SCDE and MAST have conservative p-value distributions.

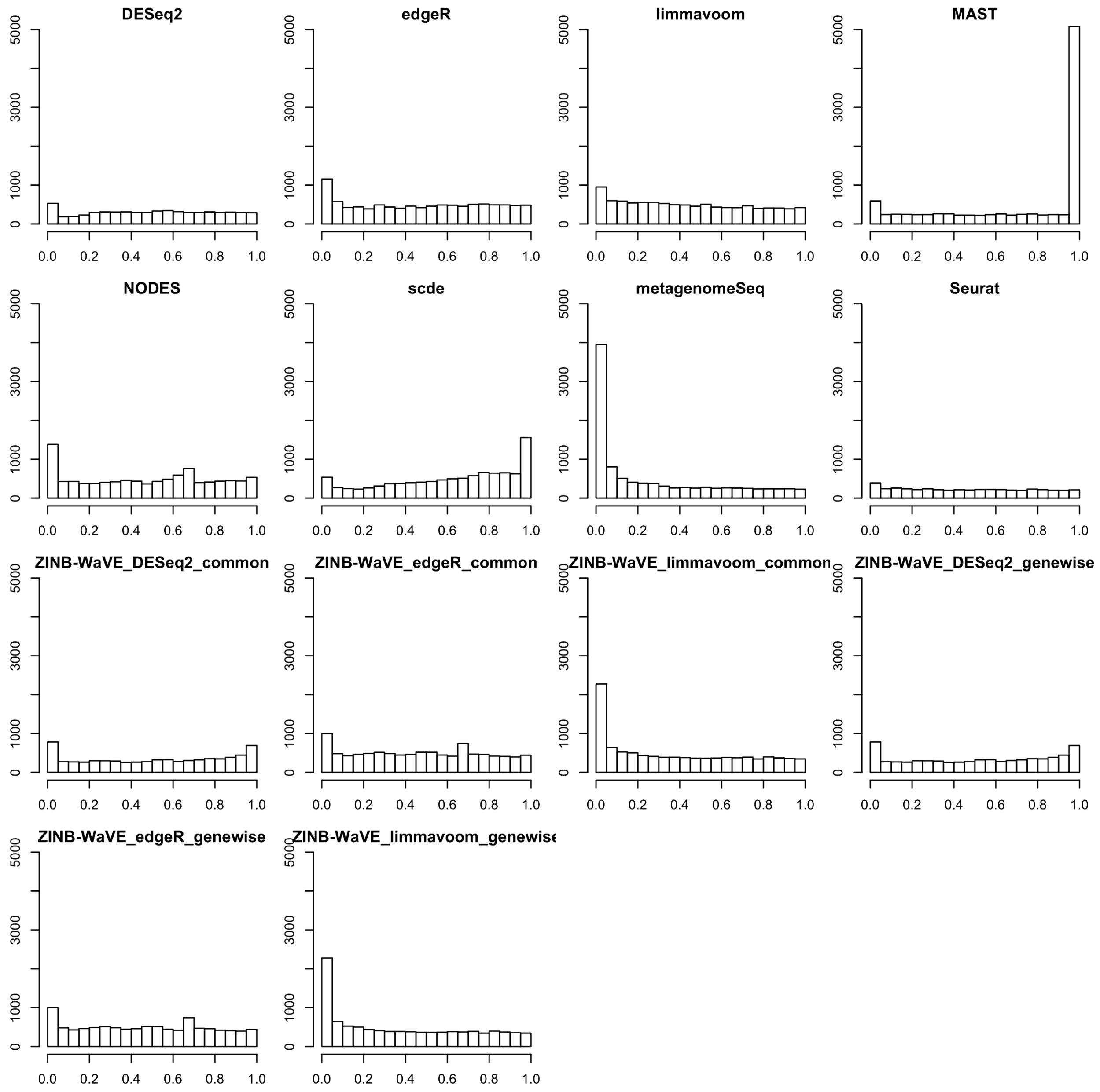


Figure 3.16: *p-value distributions on simulated Trapnell scRNA-seq dataset ($n = 150$).* The p-value distributions for methods based on ZINB-WaVE weights appear uniform for large p-values and enriched for low p-values, as would be expected from an appropriate statistical inference method on datasets with known effects. metagenomeSeq, NODES, and ZINB-WaVE-weighted limma-voom have anti-conservative p-value distributions, also evident in the FDP-TPR curves. SCDE and MAST have conservative p-value distributions.

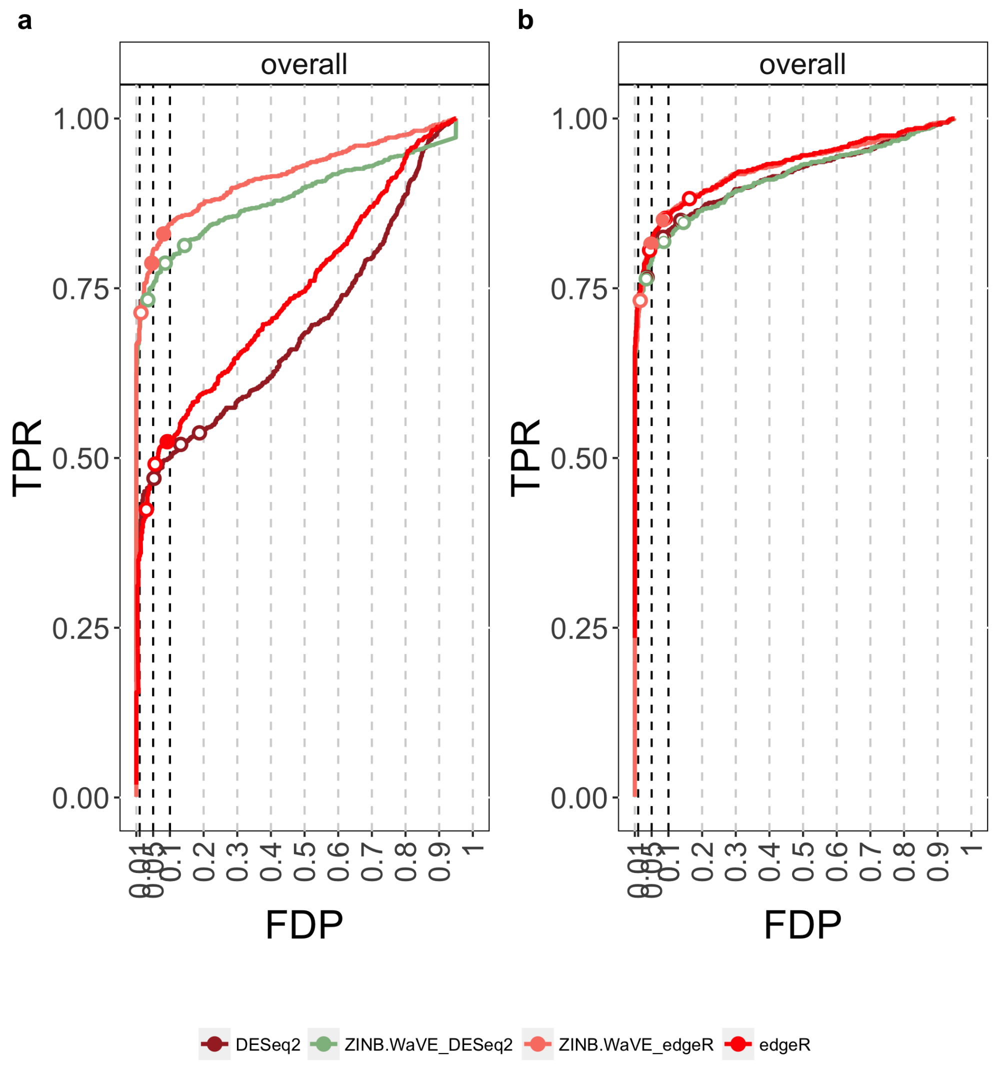


Figure 3.17: *Comparison of edgeR and DESeq2 with and without ZINB-WaVE weights on simulated bulk RNA-seq datasets ($n = 10$).* Bulk RNA-seq data were simulated from the Bottomly et al. [101] dataset using the simulation framework of Zhou, Lindsay, and Robinson [102]. DE methods are compared based on scatterplots of the true positive rate (TPR) vs. the false discovery proportion (FDP). Circles represent working points on a nominal 5% FDR level and are filled if the empirical FDR (i.e., FDP) is below the nominal FDR. **(a)** Zero-inflated bulk RNA-seq dataset, where 5% of all counts were randomly replaced by zeros. Methods based on ZINB-WaVE weights correctly identify excess zeros, while standard unweighted bulk RNA-seq tools break down in performance due to overestimation of the dispersion parameters. **(b)** Bulk RNA-seq dataset. Methods based on ZINB-WaVE weights have a similar performance to their unweighted counterparts, highlighting that in the absence of zero inflation, it is not detrimental to use the ZINB-WaVE weights for differential expression analysis.

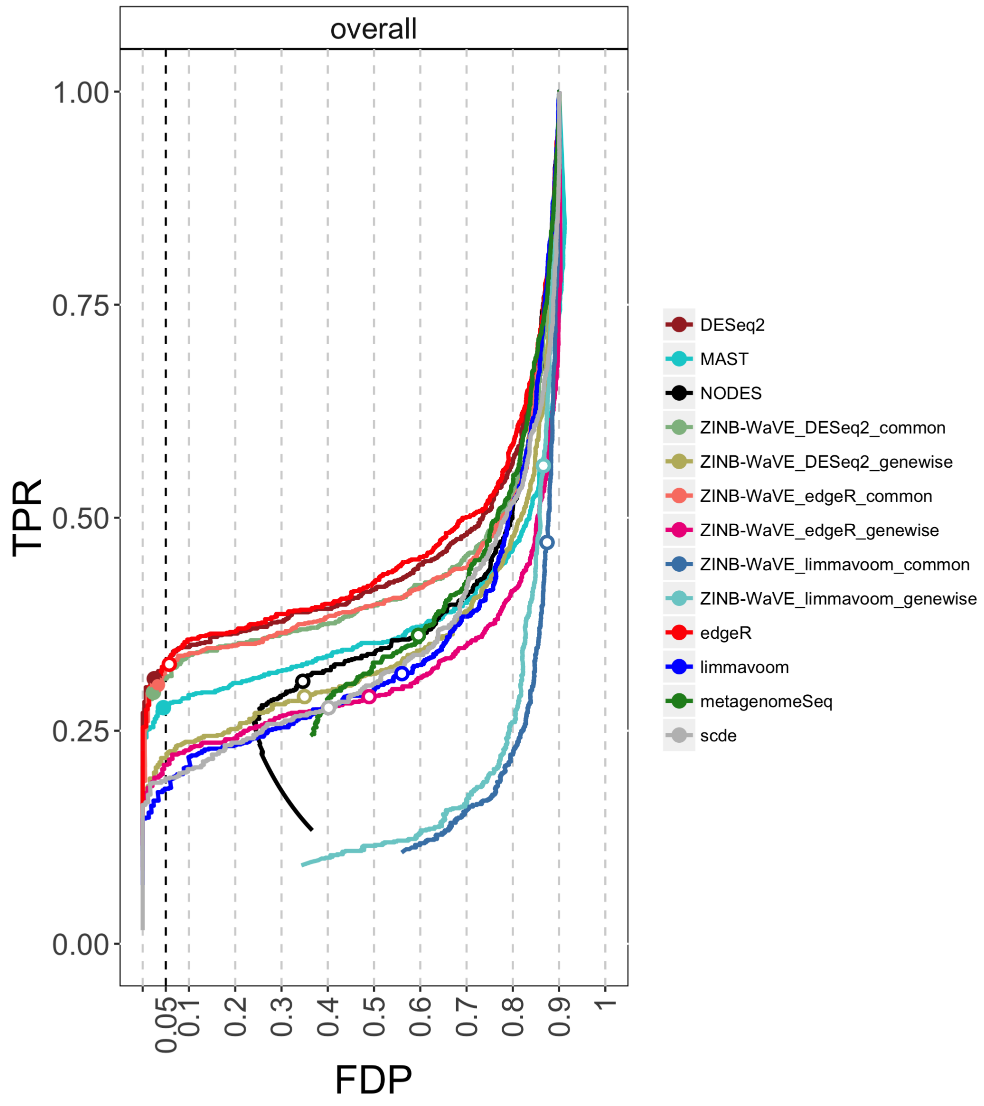


Figure 3.18: *Comparison of DE methods on simulated scRNA-seq datasets.* As in Figure 3.4, DE methods are compared based on FDP-TPR curves for data simulated from a 10x Genomics PBMC scRNA-seq dataset ($n = 1,200$). Circles represent working points on a nominal 5% FDR level and are filled if the empirical FDR (i.e., FDP) is below the nominal FDR. 10x Genomics sequencing typically involves high-throughput and massive multiplexing, resulting in very shallow sequencing depths and thus low counts, making it extremely difficult to identify excess zeros. Unweighted and ZINB-WaVE-weighted edgeR are tied for best performance, followed by ZINB-WaVE-weighted DESeq2. In general, bulk RNA-seq methods are performing well in this simulation, probably because the extremely high zero abundance in combination with low counts can be reasonably accommodated by the negative binomial distribution.

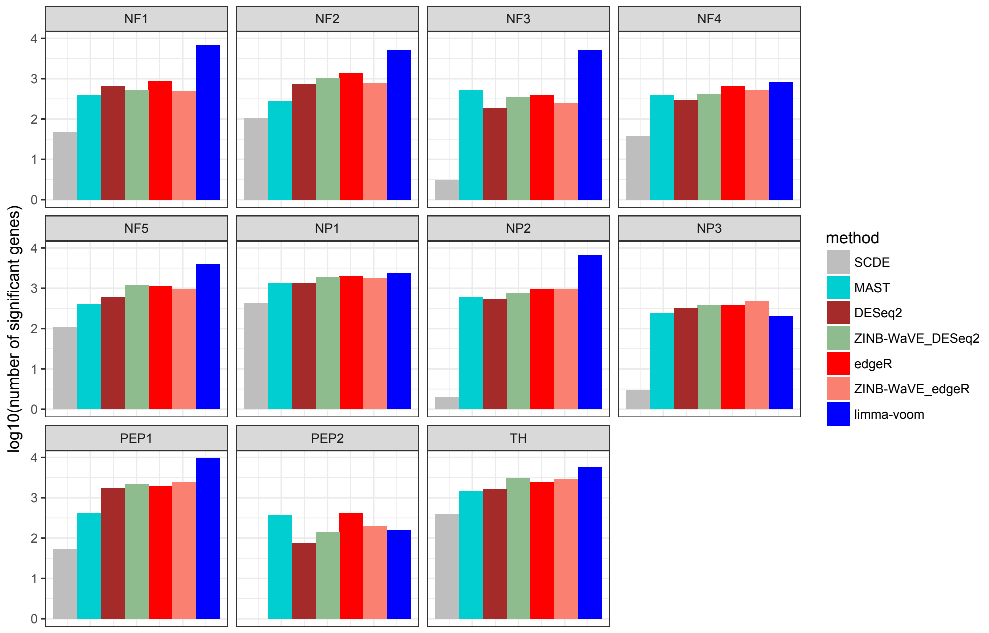


Figure 3.19: *Differential expression results for Usoskin scRNA-seq dataset.* The barplots provide the number of DE genes for the Usoskin et al. [79] dataset, based on 7 DE methods comparing each cell type (panels) to all other cell types combined. The results for SCDE were obtained by assessing the number of genes with $|Z_{adj}| \geq 1.96$ (see supporting information for the original manuscript; `http://pklab.med.harvard.edu/scde/sensory.html`). All other methods are evaluated on a 5% nominal FDR level.

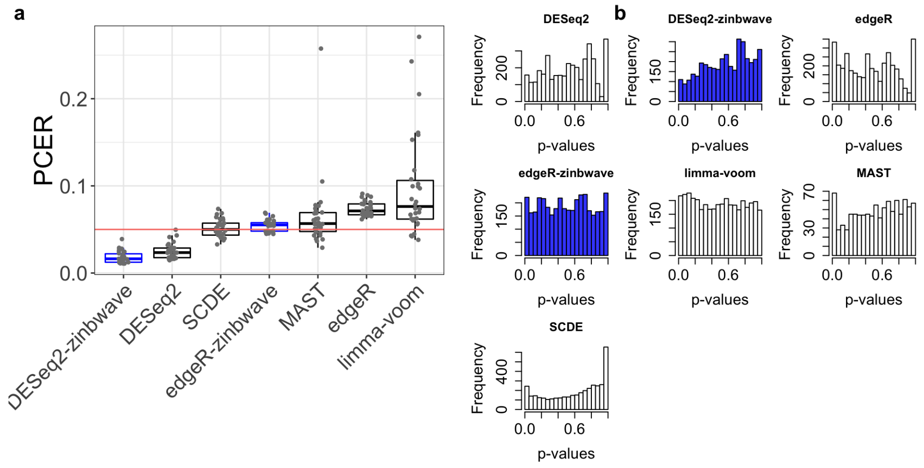


Figure 3.20: *False positive control on mock null 10x Genomics PBMC datasets ($n = 2,700$ cells).* **(a)** Boxplot of per-comparison error rate (PCER) for 30 mock null datasets for each of seven DE methods; ZINB-WaVE-weighted methods are highlighted in blue. **(b)** Histograms of unadjusted p-values for one of the datasets in (a). ZINB-WaVE was fit with intercept and mock cell type covariate in X, $V = \mathbf{1}_J$, $K = 0$ for W, common dispersion, and $\epsilon = 10^{12}$.

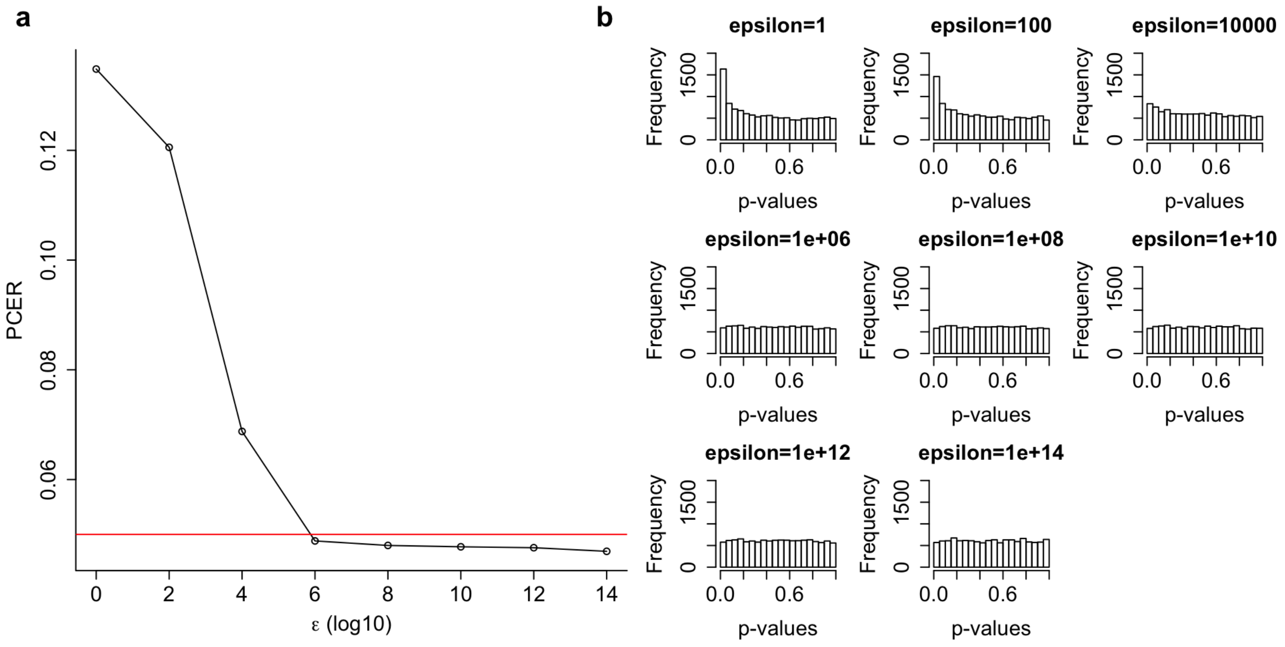


Figure 3.21: *Impact of ZINB-WaVE regularization parameter for one mock null Usoskin dataset.* **(a)** PCER as a function of the ZINB-WaVE regularization parameter ϵ. **(b)** Histograms of unadjusted p-values for different values of ϵ. DE genes for one mock null dataset are identified based on ZINB-WaVE-weighted edgeR, with an unadjusted p-value cut-off of 0.05. ZINB-WaVE was fit with intercept, mock cell type covariate, and batch covariate in X, $V = \mathbf{1}_J$, $K = 0$ for W, and common dispersion.

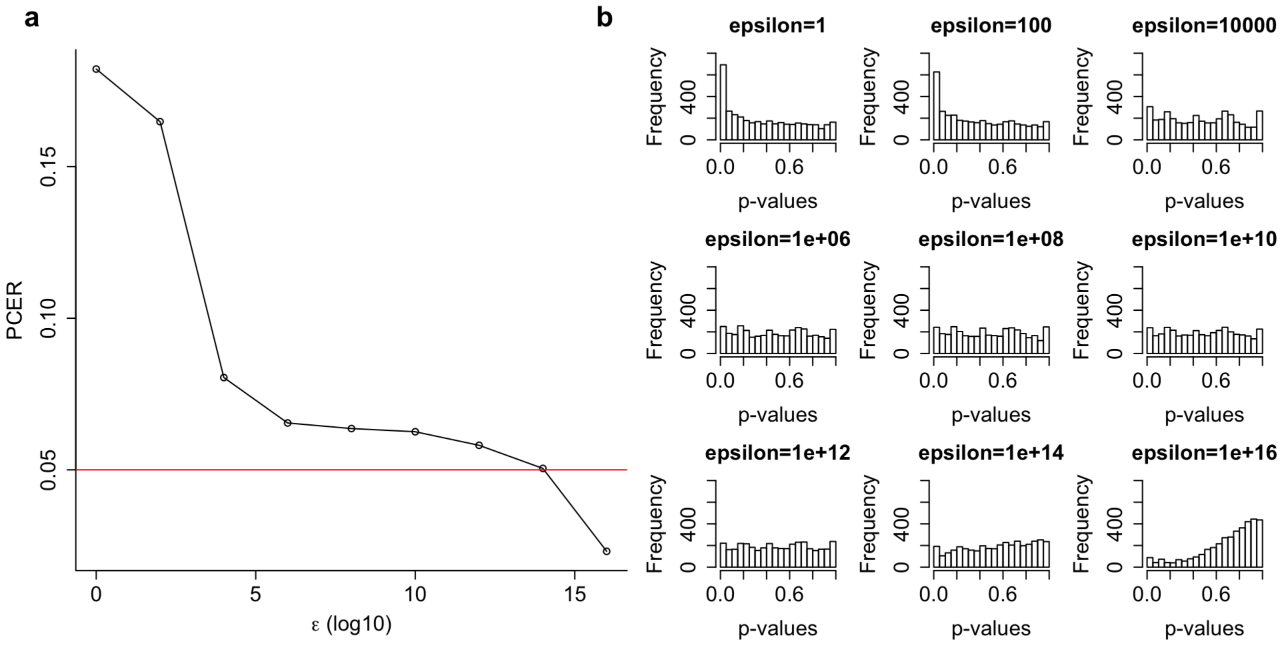


Figure 3.22: *Impact of ZINB-WaVE regularization parameter for one mock null 10x Genomics PBMC dataset.* **(a)** PCER as a function of the ZINB-WaVE regularization parameter ϵ. **(b)** Histograms of unadjusted p-values for different values of ϵ. DE genes for one mock null dataset are identified based on ZINB-WaVE-weighted edgeR, with an unadjusted p-value cut-off of 0.05. ZINB-WaVE was fit with intercept and mock cell type covariate in X, $V = \mathbf{1}_J$, $K = 0$ for W, and common dispersion.

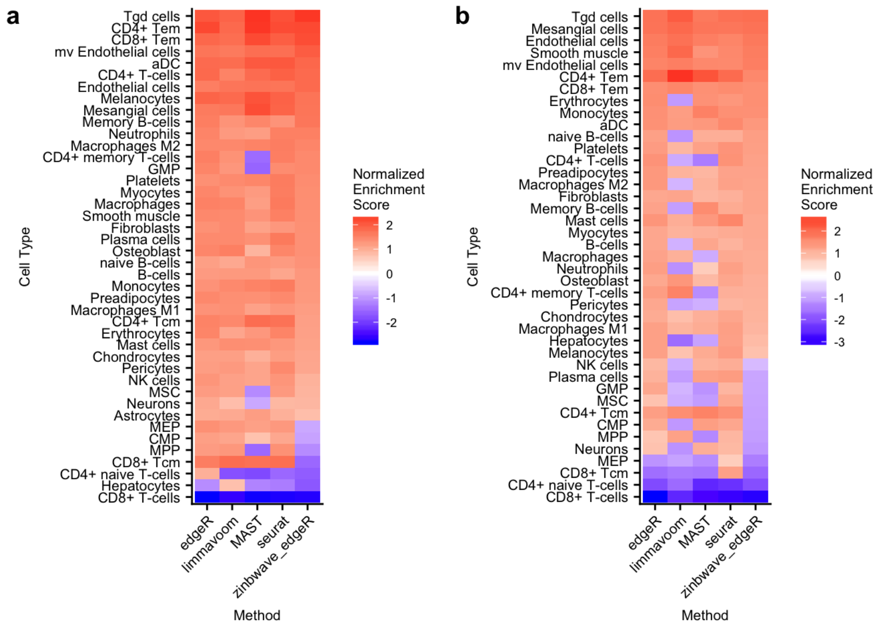


Figure 3.23: *Gene set enrichment analysis based on PCA and ZINB-WaVE for 10x Genomics PBMC dataset.* Pseudo-color images of normalized enrichment scores for gene set enrichment analysis, for differential expression between Seurat subclusters (CD4+ naive T-cells and CD4+ memory T-cells) based on **(a)** the first 10 principal components and **(b)** W from ZINB-WaVE with $K = 20$. ZINB-WaVE and GSEA parameters are as in Figure 3.6

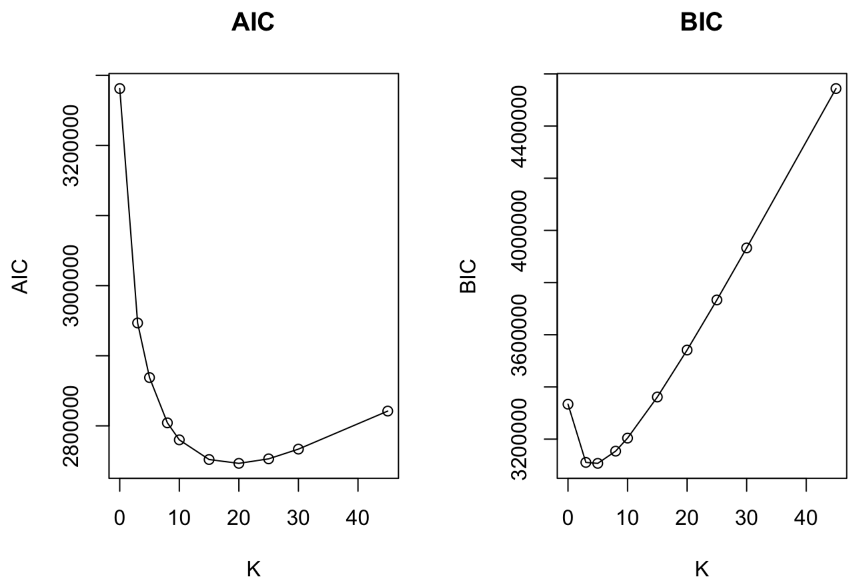


Figure 3.24: *AIC and BIC for selecting the number of unknown cell-level covariates in ZINB-WaVE for 10x Genomics PBMC dataset.* Panels show **(a)** the Akaike information criterion (AIC) and **(b)** the Bayesian information criterion (BIC) as a function of the number of unknown cell-level covariates K in the W matrix from the ZINB-WaVE model of Equations (3.1) and (3.2). ZINB-WaVE parameters are as in Figure 3.6.

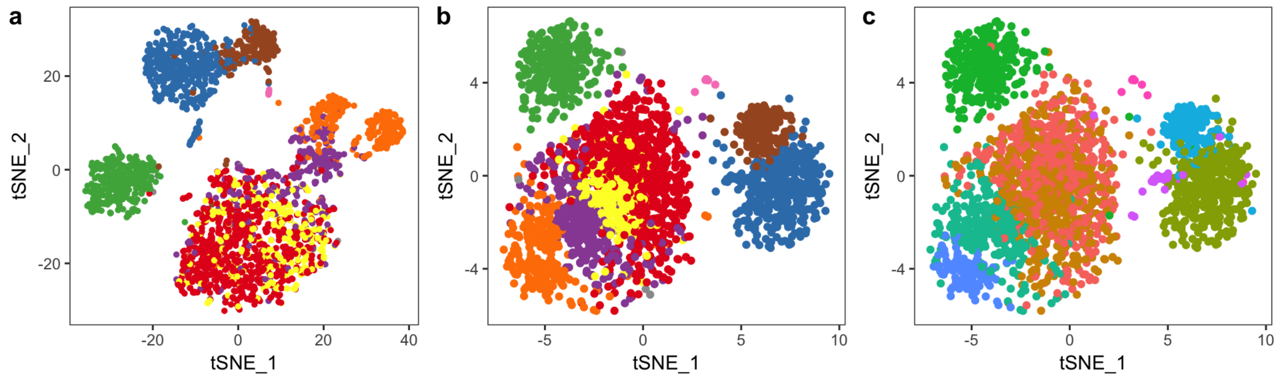


Figure 3.25: *Clustering based on PCA and ZINB-WaVE for 10x Genomics PBMC dataset.* **(a)** Scatterplot of first two t-SNE dimensions obtained from the first 10 principal components; cells are color-coded by Seurat graph-based clustering on ZINB-WaVE W ($K = 20$). **(b)** Scatterplot of first two t-SNE dimensions obtained from ZINB-WaVE W with $K = 20$; cells are color-coded by Seurat graph-based clustering on ZINB-WaVE W ($K = 20$). **(c)**. Scatterplot of first two t-SNE dimensions obtained from ZINB-WaVE W with $K = 20$; cells are color-coded by Seurat graph-based clustering on the first 10 principal components. In panels (a) and (b), CD4+ naive T-cells and CD4+ memory T-cells are, respectively, in yellow and red. Colors in panel (c) are the same as in Figure 3.6, where CD4+ naive T-cells and CD4+ memory T-cells are, respectively, in gold and red. ZINB-WaVE parameters are as in Figure 3.6.

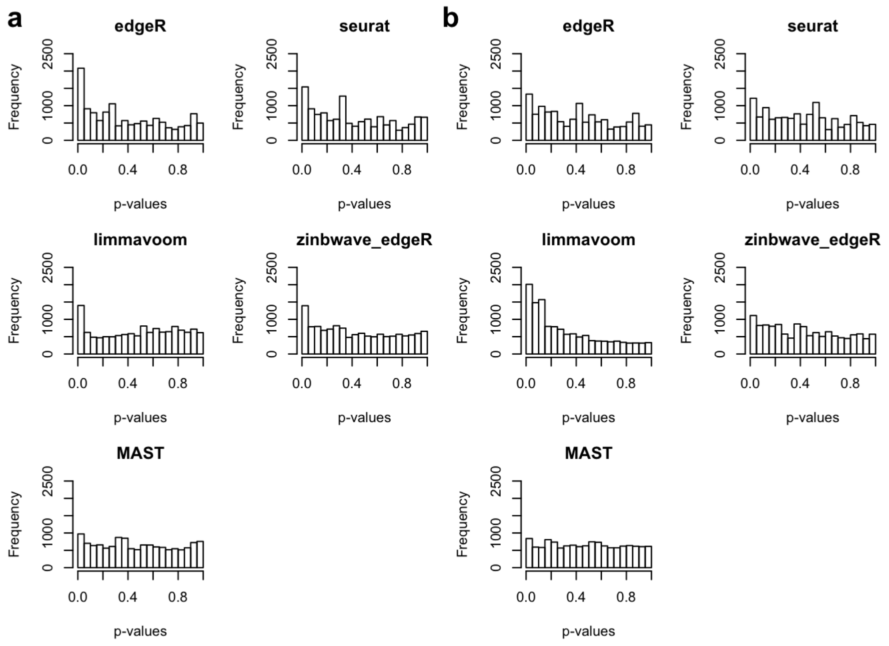


Figure 3.26: *Differential expression between clusters based on PCA and ZINB-WaVE for 10x Genomics PBMC dataset.* Histograms of unadjusted p-values from five methods for differential expression between Seurat subclusters (CD4+ naive T-cells and CD4+ memory T-cells) based on **(a)** the first 10 principal components and **(b)** W from ZINB-WaVE with $K = 20$. ZINB-WaVE parameters are as in Figure 3.6.

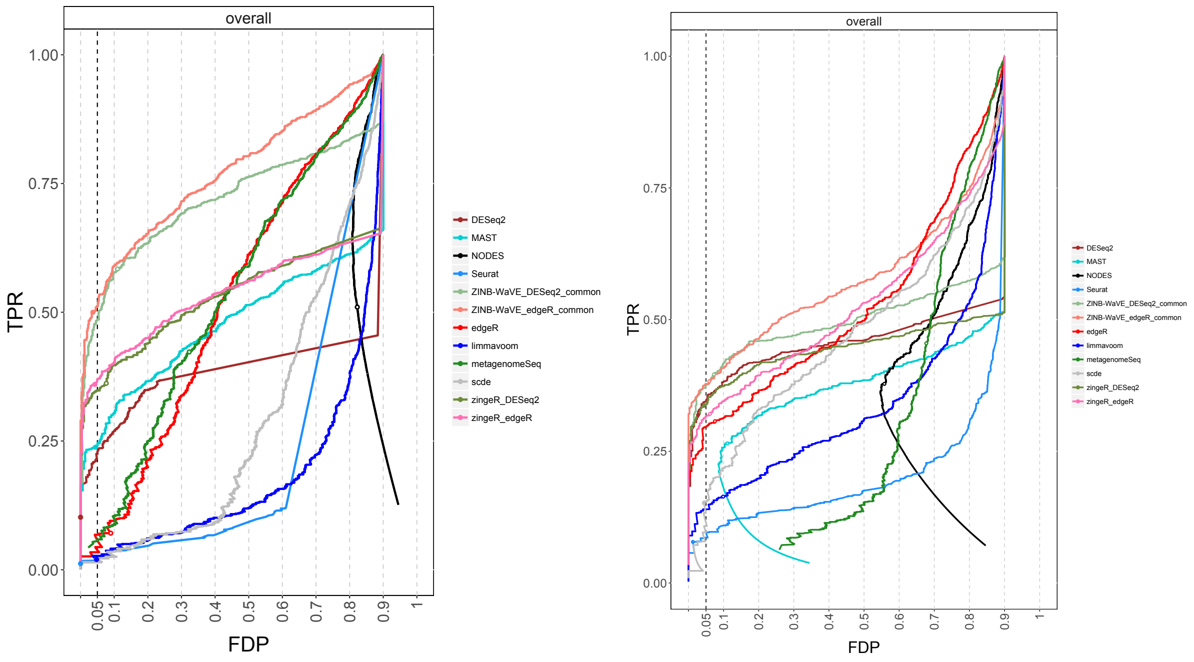


Figure 3.27: *Comparison of DE methods on simulated scRNA-seq data, including the* zingeR *methods.* Left panel: scRNA-seq data simulated from Islam et al. [84] dataset ($n = 80$). Right panel: scRNA-seq data simulated from Trapnell et al. [114] dataset ($n = 150$). DE methods are compared based on scatter plots of the true positive rate (TPR) vs. the false discovery proportion (FDP). Circles represent working points on a nominal 5% FDR level and are filled if the empirical FDR (i.e., FDP) is below the nominal FDR. The zingeR methods outperform all competing methods, except for the ZINB-WaVE methods. The DESeq2 curve in the left panel is cut off due to NA p-values resulting from independent filtering. The behavior in the lower half of the curve for MAST in the right panel is due to an extrapolation between two low working points.

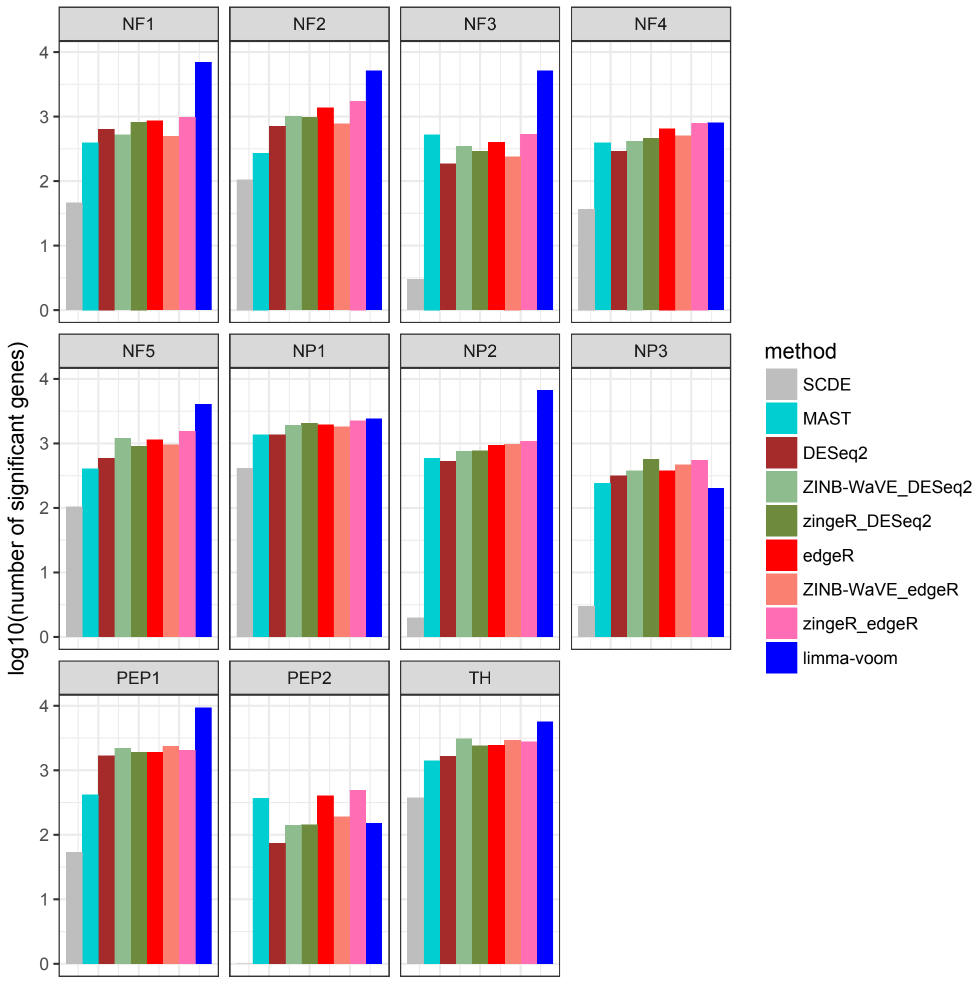


Figure 3.28: *Differential expression results for Usoskin scRNA-seq dataset.* The barplots provide the number of DE genes for the Usoskin et al. [79] dataset, based on 9 DE methods comparing each cell type (panels) to all other cell types combined. The zingeR methods consistently have a higher number of DE genes as compared to their unweighted counterparts. The results for SCDE were obtained by assessing the number of genes with $|Z_{adj}| \geq 1.96$ (see supporting information for the original manuscript; `http://pklab.med.harvard.edu/scde/sensory.html`). All other methods are evaluated on a 5% nominal FDR level.

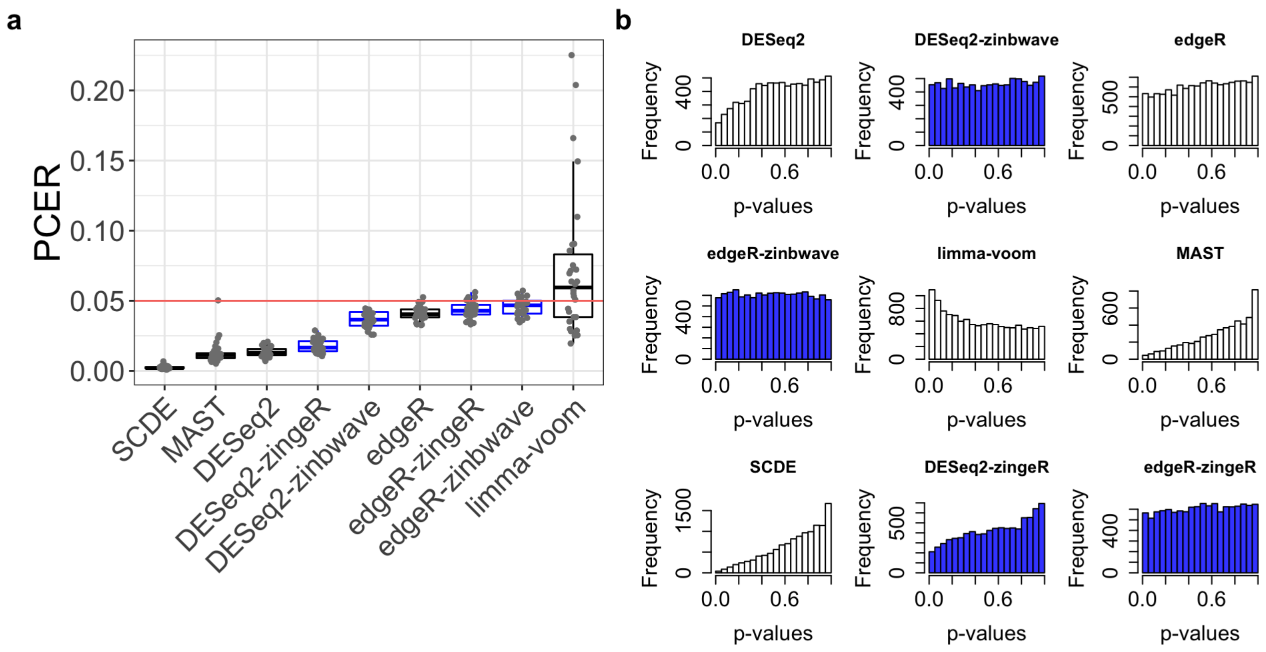


Figure 3.29: *False positive control on mock null Usoskin datasets (n = 622 cells).* **(a)** Boxplot of per-comparison error rate (PCER) for 30 mock null datasets for each of seven DE methods; ZINB-WaVE-weighted and zingeR-weighted methods are highlighted in blue. **(b)** Histogram of unadjusted p-values for one of the datasets in (a). ZINB-WaVE was fit with intercept, cell type covariate (actual or mock), and batch covariate (unless specified otherwise) in X, $V = \mathbf{1}_J$, $K = 0$ for W, common dispersion, and $\epsilon = 10^{12}$.

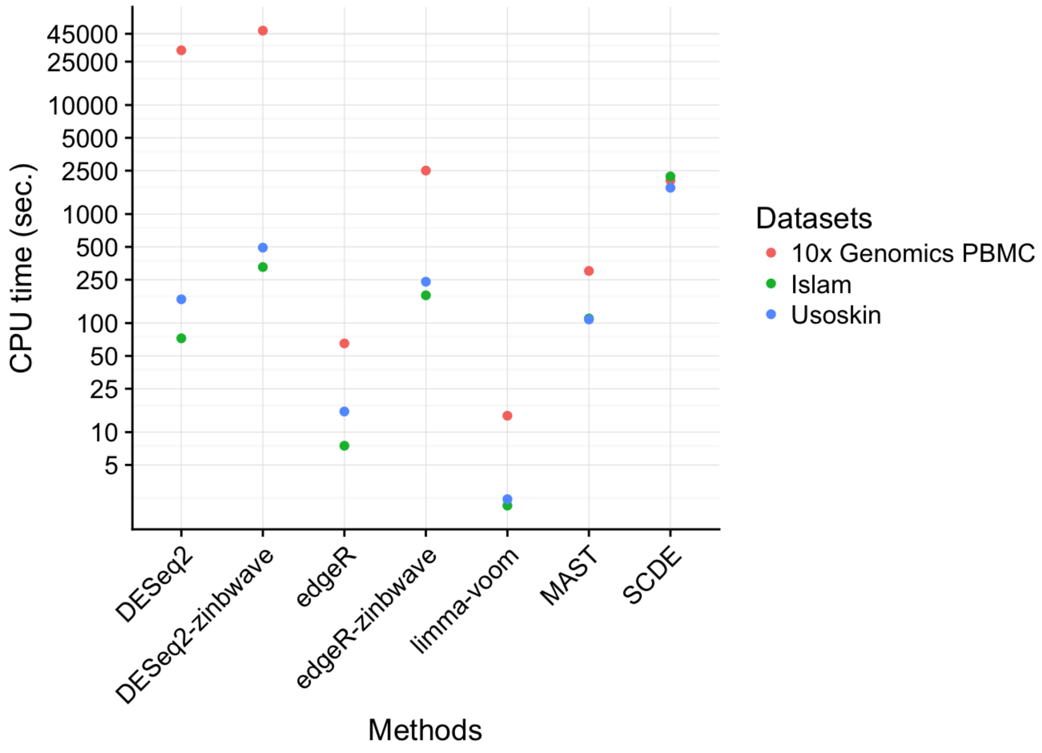


Figure 3.30: *CPU time for different DE methods and scRNA-seq datasets.* Mean CPU time (in seconds and on the log scale) for seven DE methods applied to three scRNA-seq datasets. For each method, the same parameters as in section *** "False positive rate control" were used. Colors correspond to different datasets. Islam dataset: Actual read counts for $n = 92$ cells. Usoskin dataset: Read counts for $n = 90$ cells from one of the mock null datasets used in Figure 3.5; batch was included as a covariate for all methods. 10x Genomics PBMC dataset: Only the $n = 1,151$ cells in the CD4+ T-cells clusters were used. $10,000$ genes were sampled at random for each dataset. Computations were done on a MacBook Pro with four 2.7 GHz Intel Core i5 CPUs and 8 GB of RAM. Although some methods allow the use of multiple cores, only one core was used here for comparison purposes. ZINB-WaVE was fit with intercept and cell type covariate (and batch covariate for Usoskin dataset) in X, $V = \mathbf{1}_J$, $K = 0$ for W, common dispersion, and $\epsilon = 10^{12}$.

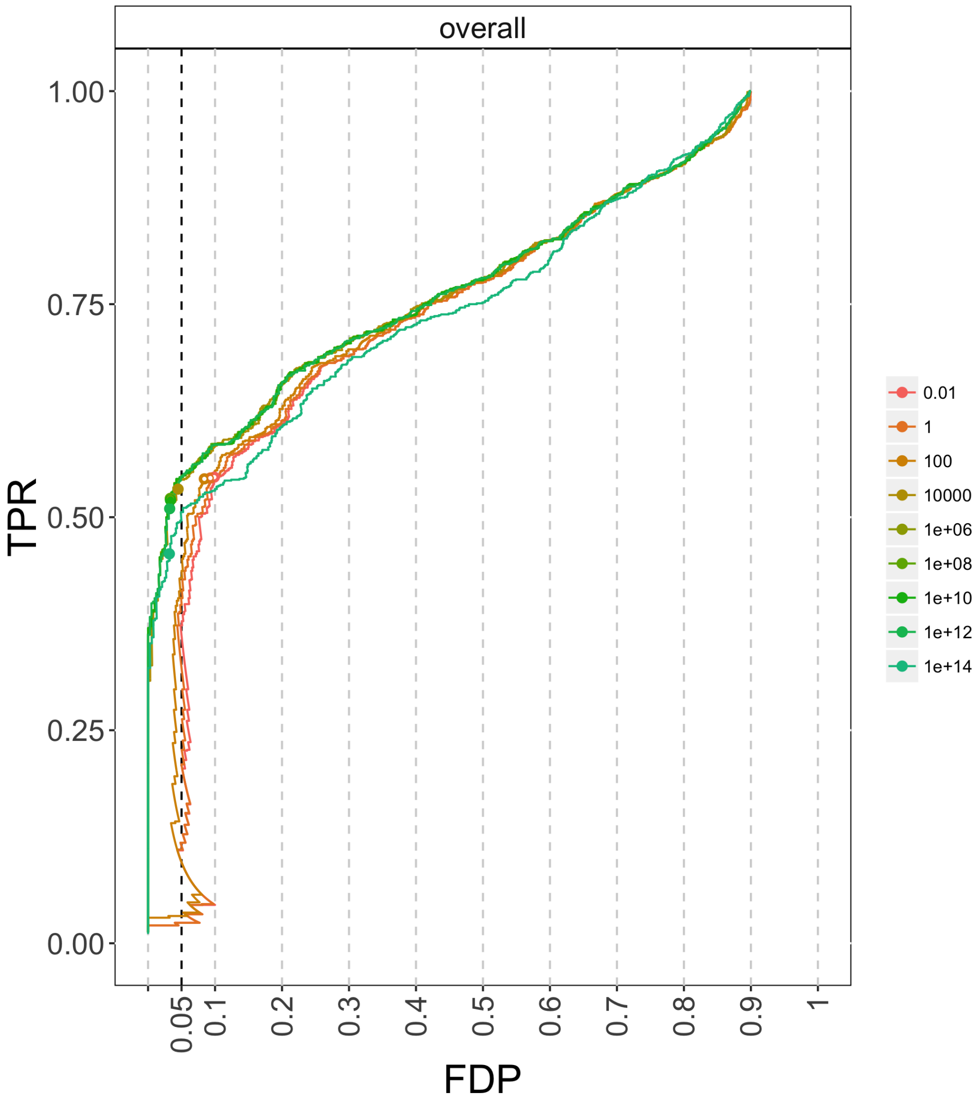


Figure 3.31: *Impact of ZINB-WaVE regularization parameter on ZINB-WaVE-weighted edgeR for simulated Islam dataset.* The different FDP-TPR curves correspond to ZINB-WaVE-weighted edgeR analyses with varying values for the ZINB-WaVE regularization parameter ϵ, as specified in the legend. The regularization parameter ϵ has a big influence on performance due to the different degrees of shrinkage applied to the ZINB-WaVE parameter estimates. We have found that setting $\epsilon = 10^{12}$ works well in general, but further research is needed to select optimal values of the penalty parameter.

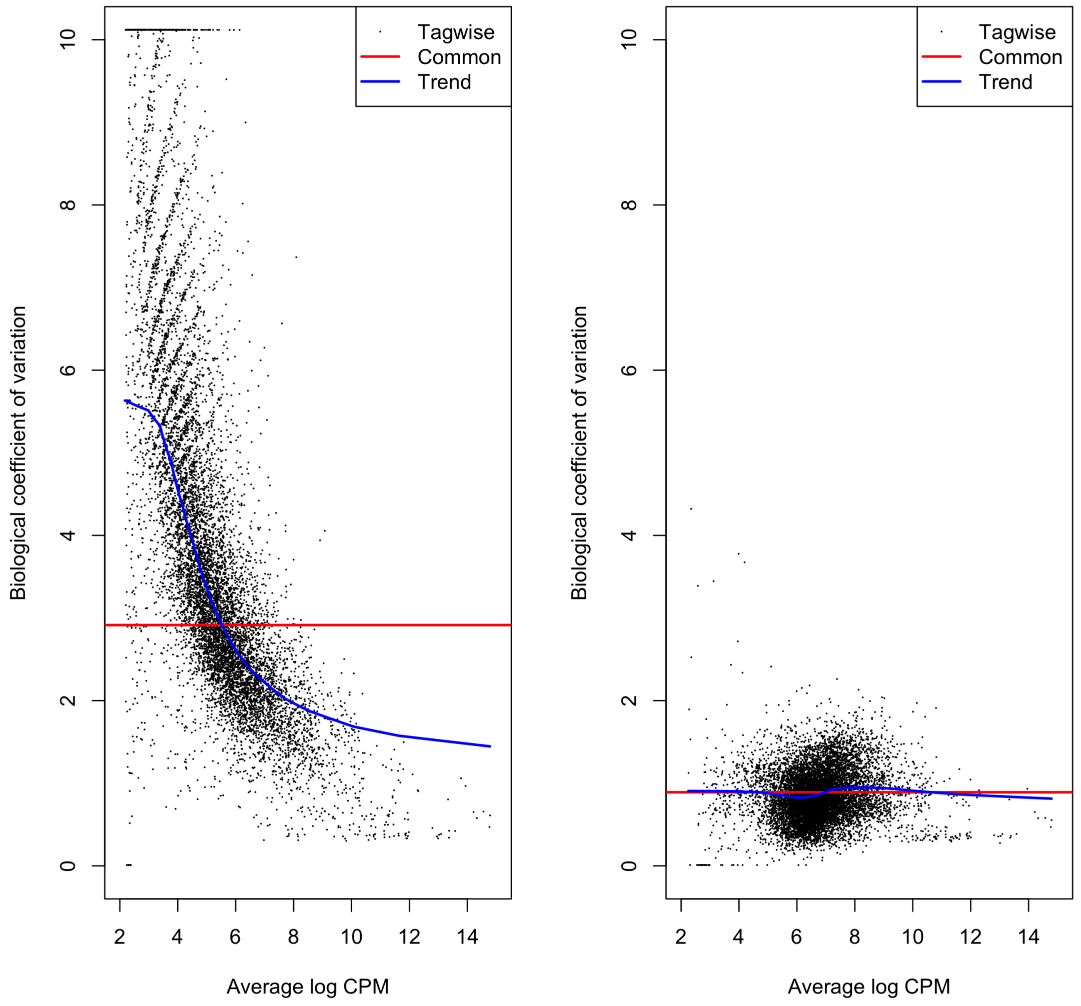


Figure 3.32: *Effect of weighting on the mean-variance relationship for the simulated Islam scRNA-seq dataset.* Estimated biological coefficient of variation (BCV) vs. average log count per million (CPM), computed by edgeR. Left panel: The BCV plot based on unweighted observations shows a strong mean-variance relationship, with very high dispersion estimates for lowly-expressed genes. Right panel: The BCV plot on the same dataset, where excess zeros identified by ZINB-WaVE are downweighted, shows much lower dispersion estimates.

Chapter 4

Estimation of bacterial abundance in microbial samples

This chapter is available on bioRxiv.[1]

4.1 Introduction

Our ability to rapidly and accurately sequence DNA has improved exponentially over the last 30 years. With this improvement, we are beginning to understand the diverse bacterial communities that surround us and how these ecosystems impact biological processes like energy harvest and vitamin synthesis. While we have appreciated bacteria's capacity to cause infection and illness for over a century, with the introduction of very high-throughput DNA sequencing, we are beginning to understand how the host's microbiome affects the course of viral and bacterial infections. Beyond these effects, the microbiome plays a key role in proper immune development and has been implicated in immune-mediated disorders like inflammatory bowel diseases and atopic dermatitis.

To study microbial ecosystems, by far, the most popular method is the sequencing of 16S rDNA derived from a polymerase chain reaction (PCR) amplification reaction targeting universal priming sites. Because this method has been used extensively to study a variety of communities, large databases exist with information connecting 16S sequences to taxonomic identifiers and subsequent annotation therein [129].

Beyond the identification of species and the estimation of their abundances in any given biological sample, we would additionally like to understand and characterize the entire genomic repertoire of a microbiome. While the 16S rRNA gene may provide clear taxonomic

[1]I would like to thank my co-authors, Jim Bullard and Sandrine Dudoit. I am grateful to Joey McMurdie and Christian Sieber of Whole Biome for providing valuable feedback on the manuscript and the whole amazing team for the great working atmosphere for this project.

resolution for a sequence, it may not be able to predict the genomic repertoire of a sample because of issues like, horizontal gene transfer, database incompleteness and bias towards culturable organisms, primer mismatch and bias [130] [131], and inherent ambiguity in the mapping between 16S and a single genome. Despite these limitations, the identification of a strain from a marker gene sequence is a valuable method.

A first step to the construction of a reliable mapping between 16S sequences and genetic capabilities is to produce estimates of a species or strain's abundance in a sample from 16S sequences. Here, our goal is to define and characterize the performance of a bacterial abundance estimation procedure. Importantly, we seek procedures that provide estimates of uncertainty when estimating the abundance. We want to ensure that uncertainty in assignment is propagated in downstream analyses.

4.1.1 16S rRNA Gene Profiling

4.1.1.1 Motivation

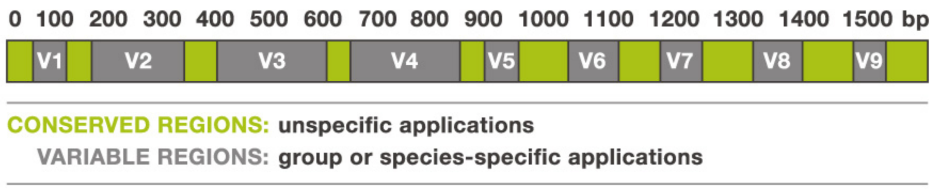

0 100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 bp
V1
V2
V3
V4
V5
V6
V7
V8
V9
CONSERVED REGIONS: unspecific applications
VARIABLE REGIONS: group or species-specific applications

Figure 4.1: *Mosaic structure of the 16S rRNA gene.* Conserved regions of the gene are identical for all bacteria, while variable regions contain specific sites unique to individual bacteria. Variable regions enable taxonomic positioning and identification of bacteria, while conserved regions are used to unselectively amplify all bacterial DNA present in a sample [132].

The 16S rRNA gene is the most commonly used genetic marker for classifying bacteria, since it is conserved in all bacteria and has a length large enough (about $1,500$ base-pairs or bp) for discriminative purposes [133]. The distinctive feature of a 16S rRNA gene, which makes it a suitable genetic marker, is its mosaic structure, i.e., alternating conserved and variable regions. Conserved regions of the gene are nearly identical across the majority of characterized bacteria, while variable regions (V1–V9) contain specific sites unique to individual bacterial taxa. Variable regions enable taxonomic positioning and identification of bacteria, while conserved regions are used to non-specifically amplify bacterial DNA present in a sample [132] (see Figure 4.1). Additionally, large databases of known 16S rRNA gene sequences and primers exist (e.g., Ribosomal Database Project [134], SILVA [135], Greengenes [136]), currently comprising more than three million gene sequences [135].

4.1.1.2 Issues

Biologists group organisms into taxa, which are assigned a taxonomic rank, whereby groups of a given rank can be aggregated to form a super group of higher rank, thus creating a taxonomic hierarchy. There are commonly eight ranks in biological taxonomy: Domain, kingdom, phylum or division, class, order, family, genus, and species (Figure 4.2). The ultimate goal of 16S rRNA classification is to identify bacteria at the lowest taxonomic level possible, e.g., species. Although 16S rRNA gene sequencing has been effective at bacterial classification at higher taxonomic rank, e.g., phylum or family, it can perform poorly when attempting to classify at the species level. The degradation in performance for finer grained distinctions occurs for the following three principal reasons:

- **Inter-species similarity.** Two bacteria from different species or even genera can have very similar 16S rRNA genes. For example, Yassin et al. [137] reported that *Nocardia brevicatena* and *Nocardia paucivorans* show 99.6% 16S rRNA gene sequence similarity. However, the results of DNA-DNA hybridization studies [2] and phenotypic testing indicate that they are distinct species. This is not a novel finding, as Fox [138] remarked in their 1992 publication that 16S rRNA sequence identity may not be sufficient to guarantee species identity, especially for recently diverged species.
- **Intra-species heterogeneity.** One bacterium can have between 1 and 15 different 16S genes, i.e., loci with different 16S gene sequences or multiple copies of a given gene sequence. For instance, for the University of Michigan Ribosomal RNA Database, the median and mode for the number of 16S genes for the bacteria in the database are respectively 4 and 2 [139]. Even if there is a strong pressure to maintain a high level of conservation for the 16S rRNA gene, intra-genomic heterogeneity exists, where the average number of nucleotides that differ between any pair of 16S rRNA genes within a genome is 2.91 (standard deviation 4.78) and the corresponding average similarity is 99.81% (standard deviation 0.31) [140]. This intra-species heterogeneity is especially problematic for bacterial classification, as the various sequences of the 16S rRNA gene in a single organism can be more different from each other than those in different organisms.
- **Genotype doesn't correspond to phenotype.** Phylogenetic classification using 16S rRNA gene sequences largely relies on the assumption that 16S rRNA genes are

[2]The DNA of one organism is labeled, then mixed with the unlabeled DNA of another organism to be compared against. The mixture is incubated to allow DNA strands to dissociate and reassemble, forming hybrid double-stranded DNA. Hybridized sequences with a high degree of similarity will bind more firmly and require more energy (i.e., higher temperature) to be separated.

only vertically inherited from parent to offspring and thus belong exclusively to a given species. However, microbial genomic analysis has revealed that some bacteria contain 16S rRNA genes that are mosaics of sequences from multiple species (this phenomenon is called horizontal gene transfer), suggesting that 16S rRNA can be transferred between different species [141]. Assuming that these mosaics are not in fact chimeras (i.e. artifacts made during the PCR process), classification of bacteria based on the 16S rRNA gene should be carefully interpreted. Additionally, virulence or phenotypically relevant genes are probably horizontally transferred so if these horizontally transferred genes are the drivers of phenotype, then using a vertically inherited gene would be problematic.

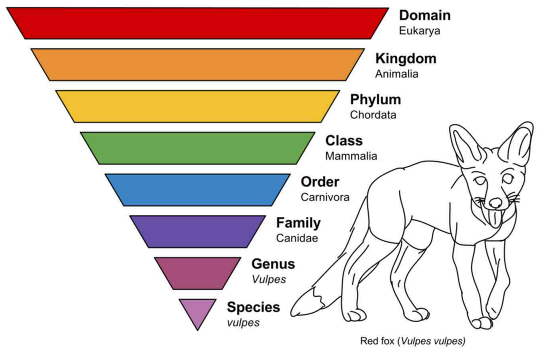


Figure 4.2: *Biological classification.* Main taxonomic ranks: Domain, kingdom, phylum, class, order, family, genus, and species. In this example, taxonomic ranking is used to classify animals and earlier life forms related to the red fox, *Vulpes vulpes* [142].

4.1.2 16S rRNA Gene Sequencing

4.1.2.1 Sequencing technologies

In early bacterial community studies, near full-length 16S rRNA genes were sequenced using the Sanger technology. This approach, though informative, was time-consuming, expensive, and provided a limited depth of sequencing, which was insufficient to uncover the complete bacterial diversity present in a complex sample [143] [3] (Table 4.1). Moreover, the Sanger technology is not capable of accurately sequencing mixtures, but rather demands an isogenic population of molecules, which is incompatible with 16S rRNA studies.

In contrast to Sanger sequencing, second-generation sequencing technologies, e.g., Illumina, provide much higher sequencing depth and allow sequencing of complex mixtures [144]. The latter have been widely used to assess the composition of microbial communities, enabling the completion of high-profile microbiome projects, such as the Human Microbiome

	First-generation (e.g., Sanger)	Second-generation (e.g., Illumina MiSeq, Roche 454)	Third-generation (e.g., PacBio)
Read scale	near full-length	variable region(s)	> full-length
High throughput	no	yes	yes
Sequencing error rate	low	low	high

Table 4.1: *Overview of the three generations of sequencing technologies in the context of 16S rRNA gene sequencing.* First-generation sequencing allows sequencing of near full-length 16S rRNA gene sequences. However, low throughput limits power to analyze microbial communities. Second-generation sequencing is currently the most used technology, as it has high throughput and low sequencing error rate. However, the short length of the sequencing reads makes it necessary to select variable regions as informative markers to identify taxa, limiting phylogenetic power. Third-generation sequencing seems promising for increasing phylogenetic resolution, as reads spanning the entire length of 16S rRNA genes can be sequenced. However, third-generation technologies have high sequencing error rates.

Project (HMP) [145] [146]. However, the higher throughput comes at the expense of read length; current technologies produce only partial sequences of 16S rRNA genes, with length varying from 250 (Illumina MiSeq) to 500 base-pairs (Roche 454). Therefore, most studies using second-generation sequencing technologies have to select the most informative variable regions (V1–V9, see Figure 4.1) as phylogenetically informative markers to identify taxa. Such partial 16S rRNA gene sequencing can bias estimates of diversity, since nucleotide differences are not evenly distributed along 16S rRNA genes [147].

Thus, the recent development of third-generation sequencing technologies offers a promising approach for high-resolution analysis of microbial communities (Table 4.1). By virtue of sequencing single contiguous molecules, third-generation technologies are capable of producing long reads. Unfortunately, with current technology, gains in length can come at the expense of accuracy. In particular, the basecall error rate for the current PacBio RS sequencer is around 10%, whereas second-generation technologies like Illumina produce basecalls with error rates around 0.02%. Since our aim is to distinguish between species with sequence similarity around 99%, the high sequencing error rate is a problem. To solve this problem, we use high-coverage single-molecule consensus reads as input to our classification algorithm. This classifier is trained off-line on a training set where reads have been annotated. It means that for each read, we known from which bacteria it has been sequenced.

4.1.2.2 Pacific Biosciences' sequencing

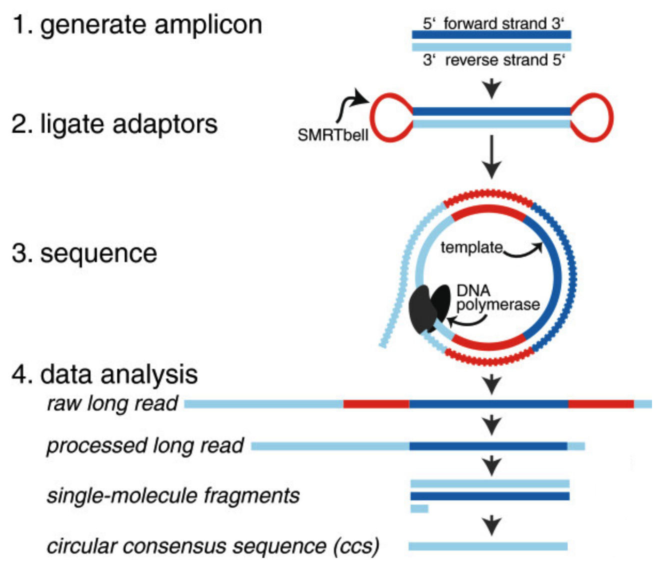


Figure 4.3: *Illustration of PacBio's single-molecule, real-time circular consensus sequencing.* Hairpin adaptors are ligated to double-stranded DNA, e.g., PCR amplicons or shotgun library. This construct is known as a SMRTbell and a library of these will be loaded onto a single chip. The PacBio RS II observes each polymerase as it sequences copies of the SMRTbell. By circularizing the DNA when constructing the library, the instrument is able to collect many measurements of each base in its forward and reverse complement context. After combining these individual base-pair measurements, the RS II software produces a consensus sequence, i.e., best estimate of the original double-stranded DNA molecule [148].

The first step in Pacific Biosciences' (PacBio) single-molecule, real-time (SMRT) circular consensus sequencing (CCS) process is to amplify full-length 16S rRNA genes from metagenomic DNA samples using polymerase chain reaction, with 16S rRNA gene-specific primers. The resulting PCR products, called amplicons, are then converted into a circular form, called SMRTbell, by ligation of hairpin adaptors to both ends of the double-stranded linear amplicons. A hairpin adaptor contains a sequence complementary to a primer, where a DNA polymerase can bind forming a sequencing-productive complex [149].

Essentially, the sequencing platform takes a video of the polymerase adding fluorescent nucleotides to the template. Each nucleotide $\{A, C, G, T\}$ is associated with a different fluorescent color, allowing the optical system to distinguish between different nucleotides. The sequencing process stops when the circular template either falls off or is killed as a side-effect of the fluorescent excitation, or when the polymerase dies. If the amplicon is short enough, the polymerase will have time to go around the hairpin multiple times, producing a sequencing read with multiple observations for each base. These multiple observations can then be used to generate high-accuracy consensus sequences for each 16S rRNA gene.

Circular consensus sequencing (CCS), enabling a consensus sequence to be obtained from multiple passes on a single template, overcomes the high single-pass error rate of the PacBio SMRT technology. While the overall single-pass sequencing error rate is estimated at about 15% [150], in the circular consensus sequencing mode, the error rate depends on the number of passes of the polymerase on the template (i.e., the number of sequenced nucleotides for each unique base in the template). With the PacBio RS II sequencer and the P4/C2 chemistry, the mean length of the sequencing reads is of about $6,900$ bp. As 16S rRNA genes are about $1,500$ bp long, most of the raw long reads are sequenced from at least 3 passes of the polymerase on the template. This results in an overall sequencing error rate of about 2% for a typical dataset for 16S rRNA genes.

4.1.3 Review of Current Statistical Methods

4.1.3.1 Cluster-based methods

Most current statistical methods used to analyze microbial communities are based on clustering and comprise two steps.

Step 1. Clustering 16S rRNA gene sequencing reads.

OTUs.
In the first step, the reads sequenced from a 16S rRNA gene are clustered according to their sequence similarity. The idea is to compare all pairs of sequences and group similar sequences together. As the number of reads is large, performing all pairwise comparisons is intractable, thus greedy algorithms are used instead. Specifically, a sequence is selected to constitute the seed of the first cluster using some criterion (e.g., the longest sequence of the dataset is selected). Next, remaining sequences are compared with the seed. If its similarity with the seed meets a predefined cutoff (often 97%), a sequence is grouped into that cluster; otherwise, it becomes the seed of a new cluster. The process is repeated until all reads are clustered. Examples of commonly-used algorithms include UPARSE [151], CD-HIT [152], UCLUST [153], and DNACLUST [154].

The resulting clusters, called operational taxonomic units (OTU), collapse the complete set of reads into a smaller collection of representative sequences (one for each OTU), where reads within an OTU have a similarity higher than a fixed threshold. The commonly-used threshold is 97% and would correspond to clustering at the species level. However, it remains debatable how well this method recapitulates a true microbial phylogeny, as it both overestimates diversity when there are more sequencing errors than the OTU-defining threshold and cannot resolve real diversity at a scale finer than that threshold [155].

Exact-sequence methods.
Recently, new methods have been developed that resolve amplicon sequence variants (ASVs) from amplicon data without imposing the arbitrary similarity threshold that define OTUs. These methods are also referred as exact-sequence methods. ASVs are inferred by a *de novo* process in which biological sequences are discriminated from errors on the basis of, in part, the expectation that biological sequences are more likely to be repeatedly observed than are error-containing sequences [156]. The most commonly used tools for exact-sequence methods are DADA2 [157] (offered in QIIME2 [158]), UNOISE2 [159] (offered in USEARCH [153]), and Deblur [160] (offered in QIIME2 [158]).

Step 2. Labeling clusters using a reference database

In the second step, a sequence per cluster (e.g., the seed used during the clustering step) can be defined to be the cluster representative and corresponding abundances are computed based on the number of reads falling within each cluster. Then, the sequence of the cluster representative is compared to all the 16S rRNA sequences present in a reference database [3], such as the Ribosomal Database Project, Greengenes, or SILVA, where each sequence in these reference databases is annotated with its taxonomic rank. Finally, each cluster representative is assigned the same taxonomic rank as its most similar sequence in the reference database. The most commonly used tools are the RDP-Classifier, which uses a naive Bayesian classifier [161], UTAX [162], and PyNAST [163].

Pipelines Several pipelines can be used to perform the above two steps [164]. Commonly-used pipelines include QIIME [165], `Mothur` [166], and MG-RAST [167]. For example, QIIME uses UCLUST as the default algorithm to cluster reads into OTUs, then the most abundant read in each OTU is selected as the representative sequence, and the script `Assign_taxonomy.py` is used for the classification of each of the representative sequences.

4.1.3.2 Closed-reference methods

The methods developed to label cluster representatives using a reference database can also be used to label each read in a dataset. The benefit of such an approach is that there is no need to select an arbitrary similarity cutoff (often 97%) to build OTUs. However, such a computation can be time-consuming and require a lot of memory, especially when the method has not been designed to handle long reads. For example, to build a reference database using our own reference sequences (about $80,000$ sequences, about $1,500$bp-long each) using the software UTAX, with the command *usearch* with argument *makeudb_tax*, requires more than 4Gb of memory. Unfortunately, the free 32-bit version of UTAX cannot handle more than 4Gb of memory; a 64-bit version is available with a paid license. Thus, it was impossible for us to use the free version of this tool to classify long reads.

	Cluster-based	Closed reference-based	Proposed method
Read length	short	short/long	long
Initial step	pre-clustering of the reads into OTUs	none	pre-filtering of reference sequences
Labeling	cluster representatives	reads	reads

Table 4.2: *Comparison between current commonly-used statistical methods and our method.* Most studies of microbial communities based on 16S rRNA genes use second-generation sequencing to generate short reads spanning only portions of a 16S rRNA gene and consist of two steps. In the first step, similar reads are grouped into clusters, called Operational Taxonomic Units (OTUs), while in the second step each cluster is labeled with taxonomic rank. In our method, reads spanning the entire length of a 16S rRNA gene are sequenced using PacBio's single-molecule, real-time circular consensus sequencing. Each read is then assigned a taxonomic rank by comparison with annotated sequences in a pre-filtered reference database.

Recently though, a few tools have been designed for long reads. For example, oneCodex [168] and SImple Non-Bayesian TAXonomy (SINTAX) [169] were released, respectively, in 2015 and 2016. OneCodex is fast and easy to use. However, we could not apply this method to our data, as it does not allow the use of a reference database different from either the NCBI RefSeq database or their own in-house reference database. On the other hand, with SINTAX it is easy to provide a reference database and there is no need to train the algorithm (while training is required when using UTAX or RDP-Classifier). However, SINTAX has a low sensitivity, that is, some bacteria known to be present in the sample are not detected (see Section 4.4).

4.1.4 Our Approach

Most studies using the 16S rRNA gene to analyze microbial communities rely on second-generation sequencing. However, the short reads yielded by these technologies only allow consideration of a few variable regions as phylogenetically informative markers to identify taxa. Such partial 16S rRNA gene sequencing can bias estimates of diversity, since nucleotide differences are not evenly distributed along the 16S rRNA gene. Instead, we use third-generation sequencing, more specifically PacBio's single-molecule, real-time circular consensus sequencing, to sequence reads spanning the entire length of the 16S rRNA gene

(Table 4.2). As standard tools were designed for short reads, there is a need to develop new statistical methods for long reads.

Sequencing full-length 16S rRNA genes has the potential to provide a higher phylogenetic resolution than short-read sequencing (at a finer level than the 97% threshold commonly used to define OTU), as it is no longer necessary to target specific variable regions. To take full advantage of long reads, our method does not group reads into OTUs; instead, each read is assigned the same taxonomic level as its most similar sequence in a reference database. As the number of sequences in a reference database is typically on the order of millions and the number of sequencing reads on the order of twenty thousand, such a computation is intractable. To reduce the computation, reference sequences with a small probability of having generated the reads in a dataset are eliminated in a pre-filtering step. Then, the probability that each read could have been sequenced from the remaining sequences in the reduced reference database is computed. Finally, the read is assigned the same taxonomic rank as the reference sequence with which it has the greatest probability. This probabilistic framework, using sequencing reads spanning the entire length of 16S rRNA genes, allows us to determine accurately bacterial relative abundances at the species (or subspecies) level.

4.1.5 Statistical Inference Framework

4.1.5.1 Population and parameters of interest

Consider a population of M ($M \simeq 10^{12}$) bacteria (e.g., a patient's gut microbiome [170]), where the bacteria are of K different types, $\mathcal{B} = \{b_k : k = 1, \dots, K\}$. Let $\pi = (\pi_k : k = 1, \dots, K)$ denote the population frequencies for each of the K bacteria in $\mathcal{B}$. Our goal is to estimate the parameter $\pi = (\pi_k : k = 1, \dots, K)$. In some cases, we may also wish to estimate a function of π or test hypotheses about π (e.g., test for each k whether $\pi_k > \epsilon$, where $\epsilon > 0$ represents the detection limit of the system).

Although beyond the scope of this report, a problem of great interest is the comparison of two bacterial populations, i.e., the identification of bacteria k which are present at different frequencies in the two populations (cf. differential expression in high-throughput microarray and sequencing assays). This involves testing for each k the null hypothesis that $\pi^1_k = \pi^2_k$, where π^1 and π^2 denote, respectively, the bacterial frequencies in the first and second population.

4.1.5.2 Reference database

We rely on an in-house annotated reference database

$$\mathcal{R} = \{r_j : j = 1, \dots, J\}, \text{ with } r_j \in \{A, C, G, T\}^{l(r_j)},$$

where l is the function mapping a sequence of nucleotides to its length (see Section 4.2.2). The J ($J \simeq 1.4 \times 10^6$) reference sequences in $\mathcal{R}$ are all the 16S genes extracted from the bacteria in $\mathcal{B}$, i.e., it is assumed that all bacteria in $\mathcal{B}$ are represented in $\mathcal{R}$. Note that one bacterium can have 16S genes at different loci in its genome, with either identical sequences or slightly different sequences (at only a handful of bases). In other words, a given bacterium can have, at different loci, either multiple identical *copies* of a 16S gene or multiple *variants* of a 16S gene. Moreover, two different bacteria can have the same 16S gene. Given this setting, we define a $K \times J$ matrix C, where C_{kj} designates the number of copies of reference sequence r_j in bacterium b_k. In particular, we let the mapping $b(r_j)$ denote the set of all bacteria containing at least one copy of r_j, i.e.,

$$b(r_j) \equiv \{b_k \in \mathcal{B} : C_{kj} > 0\}. \tag{4.1}$$

4.1.5.3 Data generation model

In order to infer the bacterial population frequencies $\pi = (\pi_k : k = 1, \ldots, K)$ (or functions of these frequencies), we adopt the following three-step data generation model.

- **Step 1. Sampling bacteria.** Sample m (a few thousand) bacteria at random (without replacement) from the population of interest and denote by $\mathcal{Z} = \{Z_i : Z_i \in \mathcal{B}, i = 1, \ldots, m\}$ the resulting set of bacteria (Figure 4.4). For simplicity, we ignore the presence of eukaryotic and viral cells and we assume that these cells have no effect on subsequent steps. For large M and small m compared to M, sampling without replacement can be treated as sampling with replacement, so that the sample frequencies for each of the K bacteria in $\mathcal{B}$ follow a multinomial distribution, that is,

  $$\left(\sum_{i=1}^{m} \mathbb{1}(Z_i = b_k) : k = 1, \ldots, K\right) \sim \text{Multinomial}(m, \pi),$$

  where $\mathbb{1}(\cdot)$ is the indicator function, equal to one if its argument is true and zero otherwise. In particular, $\Pr(Z_i = b_k) = \pi_k$.

- **Step 2. Sampling 16S amplicons.** For each sampled bacterium in $\mathcal{Z}$, extract all of its 16S genes and amplify them using polymerase chain reaction (PCR). In what follows, we refer to the amplified 16S gene sequences as *amplicons*. Select at random (without replacement) n of these amplicons (pooled from all m sampled bacteria), $\mathcal{X} = \{X_i : X_i \in \mathcal{R}, i = 1, \ldots, n\}$ (Figure 4.5). As cell lysis performance varies from one bacterium to another, we define $e(b_k)$ as the chance that DNA is extracted from bacterium b_k. Additionally, as different sequences are amplified with varying efficiencies, not all 16S genes have the same chance of being sampled. For a sequence r_j with PCR efficiency $e(r_j)$, c PCR cycles yield approximately $(2e(r_j))^c$ copies of the sequence. Hence, the probability of selecting amplicon $X = r_j$ for bacterium $Z = b_k$ is

  $$\rho_{kj} \equiv \Pr(X = r_j | Z = b_k) = e(b_k) \frac{C_{kj}(2e(r_j))^c}{\sum_{j=1}^{J} C_{kj}(2e(r_j))^c}. \tag{4.2}$$

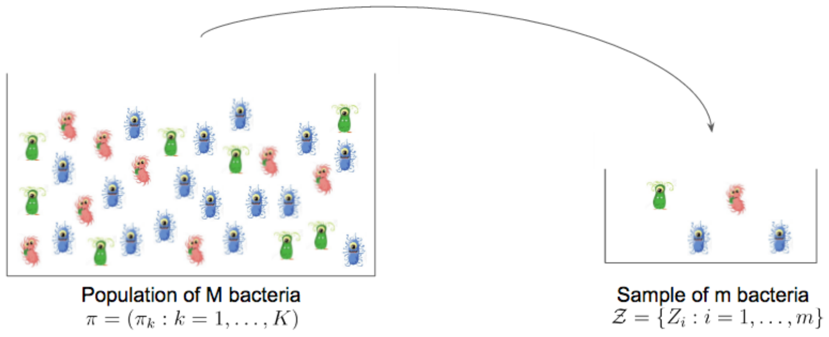

Population of M bacteria
$\pi = (\pi_k : k = 1, \ldots, K)$
Sample of m bacteria
$\mathcal{Z} = \{Z_i : i = 1, \ldots, m\}$

Figure 4.4: *Step 1. Sampling bacteria.* Simple random sample of m bacteria from a population of M bacteria. For simplicity, we ignore the presence of eukaryotic and viral cells in the figure. We assume that these cells have no effect on subsequent steps.

Here, we will assume that PCR makes no errors. Without this assumption, the amplicons may not belong to $\mathcal{R}$. Additionally, we assume that each sampled bacterium contains all copies of its 16S genes represented in $\mathcal{R}$, i.e., all C_{kj} copies of $r_j \in \mathcal{R}$ when $Z = b_k$. DNA extraction and PCR efficiencies will be estimated in an upcoming experiment at Whole Biome.

- **Step 3. Sequencing 16S amplicons.** Sequence each of the 16S amplicons in $\mathcal{X}$, using the PacBio SMRT platform, to obtain a set of n reads

$$\mathcal{Y} = \left\{ Y_i : Y_i \in \{A, C, G, T\}^{l(Y_i)}, i = 1, \ldots, n \right\},$$

where Y_i is the sequencing read corresponding to amplicon X_i and $l(Y_i) \simeq 1,500$ bp.

If amplicons in $\mathcal{X}$ were sequenced without error, each read in $\mathcal{Y}$ could be matched to a reference sequence in $\mathcal{R}$. However, errors occur during the sequencing process, introducing noise in the data. Essentially, the sequencing platform takes a video of a polymerase adding fluorescent nucleotides to a template. Each nucleotide $\{A, C, G, T\}$ is associated with a different fluorescent color, allowing the optical system to distinguish between different nucleotides. As this process happens quickly – in real time, actually – the imaging system might randomly make mistakes (i.e., mismatches), skip, or add a nucleotide. Each sequence of nucleotides Y we get to observe can then be different from the true sequence of nucleotides X. Viewing the error generation as stochastic, we can consider the probability $\Pr(Y = y | X = x)$ that a noisy read Y was generated from a true sequence X. We will estimate such probabilities using the generalized pair hidden Markov model (GPHMM) presented in Section 4.3.3.1 and we further assume that errors are introduced independently between reads [171].

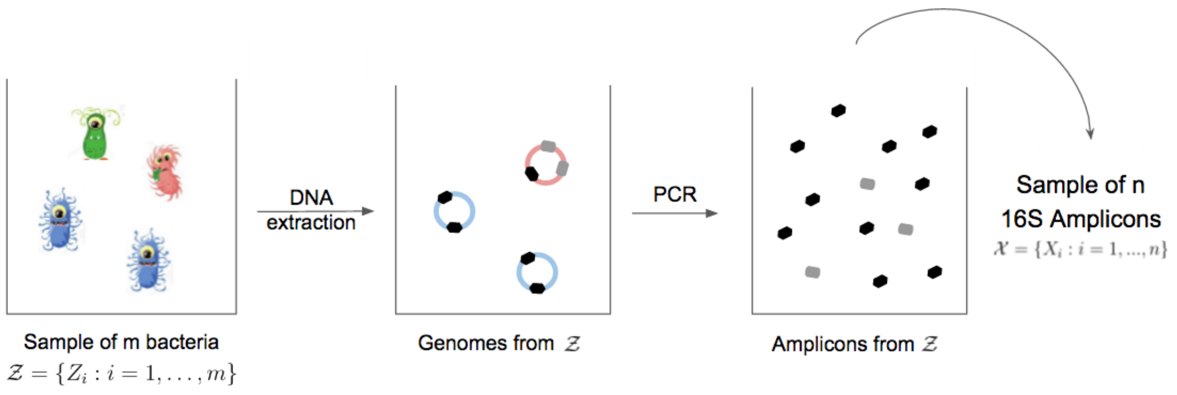

DNA extraction
PCR
Sample of n 16S Amplicons
$\mathcal{X} = \{X_i : i = 1, \ldots, n\}$
Sample of m bacteria
$\mathcal{Z} = \{Z_i : i = 1, \ldots, m\}$
Genomes from $\mathcal{Z}$
Amplicons from $\mathcal{Z}$

Figure 4.5: *Step 2. Sampling 16S amplicons.* Sample n amplicons at random (without replacement) from the amplicons of the bacteria in $\mathcal{Z}$. Here, four bacteria were sampled (two blue, one green, and one pink). DNA extraction did not work for the green bacterium. The blue bacterium has two identical copies of its 16S gene (black), while the pink bacterium has two different variants of its 16S gene (black and gray), with one variant (gray) having two identical copies. Hence, for the pink bacterium, there are three distinct loci where amplification of the 16S gene can occur. The blue and pink bacteria share the same 16S gene sequence (black variant). One cycle of PCR is represented and PCR worked only partially for the gray variant.

Our three-step data generation model is equivalent to the graphical model in Figure 4.6, whereby, for each $i = 1, \ldots, n$,

$$\Pr(Y_i, X_i, Z_i; \pi) = \Pr(Y_i|X_i) \Pr(X_i|Z_i) \Pr(Z_i; \pi). \tag{4.3}$$

However, only the reads Y are observed and the probability of a read Y_i is

$$\begin{aligned}\Pr(Y_i = y_i; \pi) &= \sum_{k=1}^{K} \sum_{j=1}^{J} \Pr(Y_i = y_i|X_i = r_j) \Pr(X_i = r_j|Z_i = b_k) \Pr(Z_i = b_k; \pi) \\ &= \sum_{k=1}^{K} \pi_k \sum_{j=1}^{J} \Pr(Y_i = y_i|X_i = r_j)\rho_{kj}. \end{aligned} \tag{4.4}$$

Given that each read Y_i in $\mathcal{Y}$ is generated independently and from Equation (4.4), the log-likelihood of the data $\mathcal{Y}$ is

$$\ell(\pi; \mathcal{Y}) \equiv \log \Pr(\mathcal{Y}; \pi) = \sum_{i=1}^{n} \log \Pr(Y_i = y_i; \pi) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} \pi_k \sum_{j=1}^{J} \Pr(Y_i = y_i|X_i = r_j)\rho_{kj} \right). \tag{4.5}$$

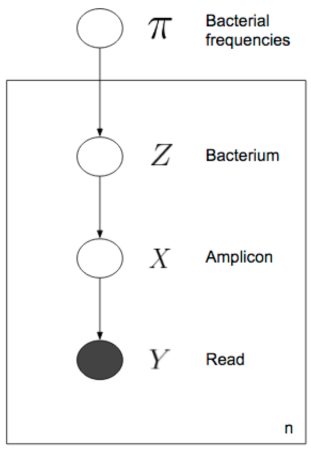


Figure 4.6: *Data generation model.* Graphical model representation of generative model for read dataset $\mathcal{Y}$. To generate each read $Y \in \mathcal{Y}$, sample a bacterium Z at random from a microbiomic population of interest with bacterial frequencies π. For each sampled bacterium Z, extract all of its 16S genes, amplify them using PCR, and select at random one of these 16S amplicons, X. Finally, sequence amplicon X to generate read Y. The process is repeated independently for each of the n reads in $\mathcal{Y}$. Only the shaded node is observed.

A natural estimator of the bacterial frequencies π is the maximum likelihood estimator (MLE), defined as

$$\hat{\pi}^{MLE} \equiv \arg\max_{\pi} \ell(\pi; \mathcal{Y}), \text{ s.t. } \sum_{k=1}^{K} \pi_k = 1, \ 0 \leq \pi_k \leq 1. \tag{4.6}$$

The naive way to compute the log-likelihood $\ell(\pi; \mathcal{Y})$ would be to compute $\Pr(Y_i = y_i | X_i = r_j)$ for each of the n reads Y_i in $\mathcal{Y}$ and each of the J reference sequences r_j in $\mathcal{R}$. The number of reference sequences in $\mathcal{R}$ being on the order of 1.4 million and the number of reads in $\mathcal{Y}$ on the order of the thousands, such a computation is intractable. Thus, our reference database $\mathcal{R}$ is first reduced to a smaller database of a few thousand sequences (the exact number depending on the dataset $\mathcal{Y}$), by eliminating the reference sequences with a small probability of having generated the reads in $\mathcal{Y}$. This step is performed using the alignment tool Bowtie2 [172] (Section 4.3.2). Secondly, the probabilities $\Pr(Y_i = y_i | X_i = r_j)$, for a reduced version of the database $\mathcal{R}$, are computed using the Viterbi algorithm for a generalized pair hidden Markov model (Section 4.3.3). Given estimates of $\Pr(Y_i = y_i | X_i = r_j)$ and ρ_{kj}, Section 4.3.4 describes various approaches for estimating π. This process is summarized in Figure 4.7 and, in greater detail, in the workflow of Figure 4.15.

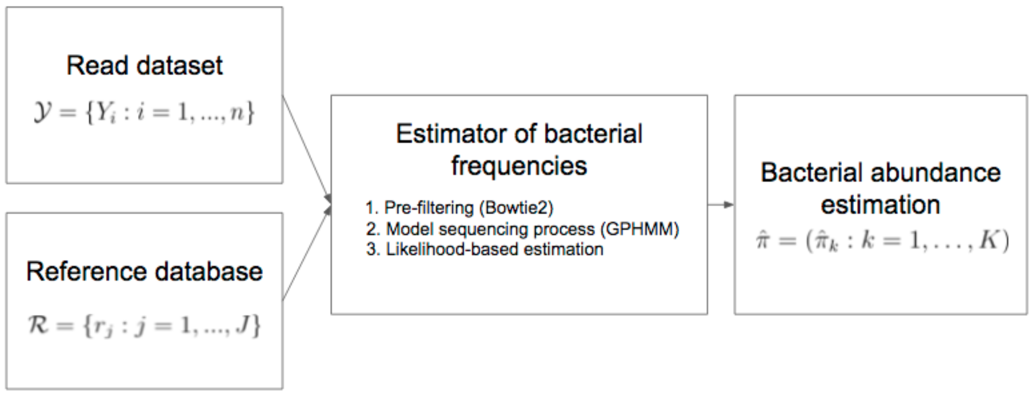

Read dataset
$\mathcal{Y} = \{Y_i : i = 1, ..., n\}$
Reference database
$\mathcal{R} = \{r_j : j = 1, ..., J\}$
Estimator of bacterial frequencies
1. Pre-filtering (Bowtie2)
2. Model sequencing process (GPHMM)
3. Likelihood-based estimation
Bacterial abundance estimation
$\hat{\pi} = (\hat{\pi}_k : k = 1, \ldots, K)$

Figure 4.7: *Bacterial composition estimation procedure.* The read dataset $\mathcal{Y}$ and the reference database $\mathcal{R}$ are used to estimate the bacterial composition of the population $\mathcal{Y}$ was derived from.

4.2 Data

4.2.1 Sequencing Reads

4.2.1.1 Simulated dataset

The goal is to simulate realistic datasets of 16S rRNA reads. In this section, we focus primarily on specifying bacterial frequencies $\pi = (\pi_k : k = 1, \ldots, K)$; given π, we then use the software package `PBSIM` [173] to generate read datasets $\mathcal{Y}$. The simulation process is summarized in Figure 4.8.

Different levels of richness (i.e., number of distinct species) and evenness (i.e., uniformity of species frequencies) can be observed in microbial communities. Reduced richness and/or imbalances in the gut microbiome have been associated with a variety of diseases, including obesity [174], inflammatory bowel diseases [175], type II diabetes [176], and fatty liver disease [177]. As detailed in the remainder of this section, to span the vast range of possible bacterial compositions that have been reported in the literature, we use gut and vaginal samples from the Human Microbiome Project (HMP) [178] to generate bacterial frequencies π. Specifically, to reflect different levels of richness and evenness, we start from four HMP microbial communities (two gut and two vaginal) and, for each such community, simulate bacterial frequencies using six different variability parameters. Additionally, to test the robustness of our method, bacterial communities $\mathcal{Z}$ are simulated using three different sampling distributions: the multinomial used in our data generation model and two other distributions, a Poisson distribution and a negative binomial distribution allowing over-dispersion. Overall, this yielded 72 simulated read datasets $\mathcal{Y}$, corresponding to four microbial communities (two gut and two vaginal), six variability parameters, and three sampling distributions.

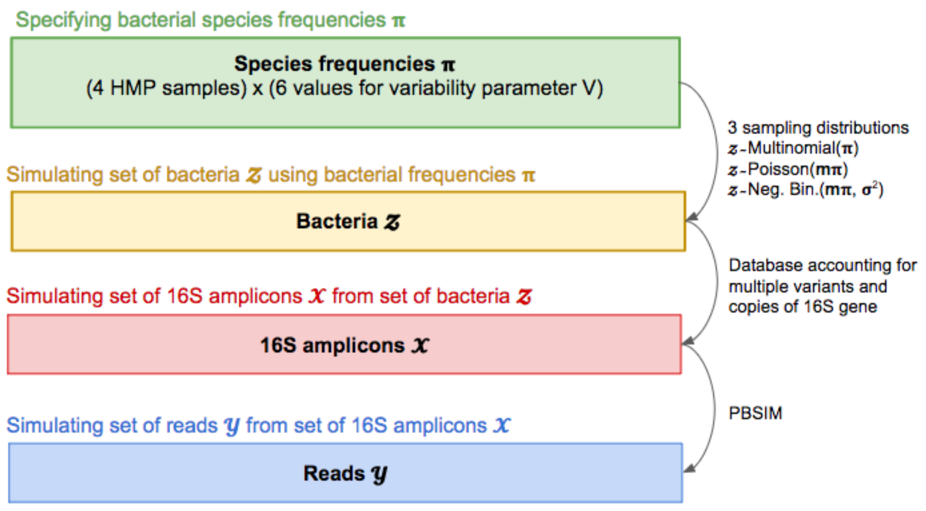

Specifying bacterial species frequencies π
Species frequencies π
(4 HMP samples) x (6 values for variability parameter V)
3 sampling distributions
𝒵-Multinomial(π)
𝒵-Poisson(mπ)
𝒵-Neg. Bin.(mπ, σ²)
Simulating set of bacteria 𝒵 using bacterial frequencies π
Bacteria 𝒵
Database accounting for multiple variants and copies of 16S gene
Simulating set of 16S amplicons 𝒳 from set of bacteria 𝒵
16S amplicons 𝒳
PBSIM
Simulating set of reads 𝒴 from set of 16S amplicons 𝒳
Reads 𝒴

Figure 4.8: *Simulation process.*

On average, about $n = 3,200$ reads from $m = 1,000$ bacteria were simulated for each of the 72 datasets. Figure 4.9 displays an overview of the 24 sets of generative bacterial frequencies $\pi = (\pi_k : k = 1, \ldots, K)$ for the 72 simulated microbial communities: Number of distinct species $\sum_k \mathbb{1}(\pi_k > 0)$, Shannon diversity $-\sum_k \pi_k \log \pi_k$ summarizing richness and evenness, and minimum, median, and maximum of the bacterial frequencies π_k. Figure 4.11 represents the distributions of read lengths and quality scores for each of the 72 simulated datasets. Figure 4.12 displays the bacterial frequencies π_k for simulated community *Gut 2* for the two most extreme variability parameters ($V = 0.1$ and $V = 10,000$).

Specifying bacterial species frequencies π. The first step of the simulation is concerned with obtaining the bacterial species frequencies π used to simulate bacterial communities $\mathcal{Z}$. The results of an analysis performed as part of the Human Microbiome Project [178] [179] were downloaded from the HMP website (`http://hmpdacc.org/`, dataset `hmp1.v13.hq`). In this analysis, 16S variable regions 1–3 of about 4,000 bacterial samples from five different body sites (oral, airways, skin, gut, vagina) were sequenced using the Roche-454 FLX Titanium platform. The software package `Mothur` was used to classify the sequencing reads at the genus level. See details of the analysis in [180]. From these 4,000 samples, we randomly selected four samples, two from the gut body site and two from the vaginal body site, yielding four bacterial communities: *Gut 1*, *Gut 2*, *Vaginal 1*, and *Vaginal 2*. Then, for each community, we computed the proportion of reads assigned to each genus using files `hmp1.v13.hq.phylotype.counts` and `hmp1.v13.hq.phylotype.lookup` downloaded from the HMP website. The bacterial genera frequencies for each simulated community were set to be equal to the HMP genera frequencies.

However, as we want to evaluate the ability of our method to estimate bacterial frequencies at the species/sub-species level and the HMP dataset only provides information at the genus level, we use our reference database $\mathcal{R}$ to identify species and their associated 16S sequences within each genus. Specifically, for each genus, the number of species κ included in the simulation is the number of distinct species found in $\mathcal{R}$ for this genus. For each species, a reference sequence is then randomly sampled from $\mathcal{R}$ and its frequency (for simulation purposes) is sampled from a normal distribution with mean $\mu = \frac{1}{\kappa}$ and variance μV, where $V \in \{0.1, 1, 10, 100, 1,000, 10,000\}$. A small variability parameter V yields a balanced microbial community, whereas a high variability parameter corresponds to a less diverse community. Sampled species frequencies greater than 1 or smaller than 0 are set to 0, resulting in a less rich microbial community, especially when the variability parameter is large. Finally, for each genus, species frequencies are scaled to sum to the HMP genus frequency.

This sampling process yields frequencies $\pi = (\pi_k : k = 1, \ldots, K)$ for each of K bacterial species. For example, for the simulation scenario of the left panel of Figure 4.12, the genus *Alistides*, with frequency of 0.09 in the HMP dataset, has six distinct species, each with a single reference sequence, according to the reference database $\mathcal{R}$. This genus is represented in the simulation with six unique sequences, with different frequencies (0.0162, 0.0141, 0.0164, 0.0187, 0.0136, 0.011).

Simulating set of bacteria $\mathcal{Z}$ using bacterial frequencies π. Following Step 1 of our data generation model (see Figure 4.4), a set of $m = 1,000$ bacteria $\mathcal{Z} = \{Z_i : Z_i \in \mathcal{B}, i = 1, \ldots, m\}$ is simulated from the Multinomial(m, π) distribution. To test the robustness of our method, we additionally used two other distributions to simulate $\mathcal{Z}$: A Poisson distribution, where, for each bacterium b_k, Z_k is sampled from Poisson$(m\pi_k)$, and a negative binomial distribution, where Z_k is sampled from Negative Binomial$(\mu = m\pi_k, \sigma^2 = \mu + \phi\mu^2)$ with $\phi = 1/2$ (here, μ is the mean, σ^2 the variance, and ϕ the dispersion parameter, corresponding to the inverse of the `size` argument of the R function `rnbinom`).

Simulating set of 16S amplicons $\mathcal{X}$ from set of bacteria $\mathcal{Z}$. For Step 2 of our data generation model (see Figure 4.5), DNA extraction and PCR efficiencies are set to one (i.e., $e(b_k) = 1\ \forall k \in \{1, \ldots, K\}$ and $e(r_j) = 1\ \forall j \in \{1, \ldots, J\}$) and the number of PCR cycles is set to $c = 1$.

Ideally, our referene database $\mathcal{R}$ would provide all possible variants and copies of a 16S rRNA gene, i.e., the number C_{kj} of copies of variant r_j for bacterium b_k would be known $\forall k \in \{1, \ldots, K\}$ and $\forall j \in \{1, \ldots, J\}$). Then, an amplicon dataset $\mathcal{X}$ could be simulated using the sequences of each variant and the expected number of amplicons for variant with reference sequence r_j from bacterium b_k would be $m\pi_k C_{kj}$.

However, not all variants of a 16S gene are usually available in our database $\mathcal{R}$ and C_{kj} is unknown. Still, to account for the possible multiple variants and copies of a 16S rRNA gene, we estimated the total number of loci at which a 16S gene could be found in the genome of each bacterium b_k (i.e., $\sum_{j=1}^{J} C_{kj}$) using database rrnDB [181]. Bacteria for which the number of 16S genes is not available at the species level are assigned the number of 16S genes of their lowest taxonomic rank for which the number of 16S genes is available. If bacteria at this taxonomic level have different numbers of 16S genes, the median is used as an approximation for $\sum_{j=1}^{J} C_{kj}$. Then, for each bacterium b_k, one variant of its 16S gene with reference sequence r_j is randomly sampled from our reference database $\mathcal{R}$. The number of amplicons for bacterium b_k is therefore expected to be $m\pi_k \sum_{j=1}^{J} C_{kj}$.

Simulating set of reads $\mathcal{Y}$ from set of 16S amplicons $\mathcal{X}$. A simulated read dataset $\mathcal{Y}$ is obtained by subjecting each amplicon in the simulated dataset $\mathcal{X}$ to a sequencing error process. Following the PacBio CCS error model, deletions, insertions, and mismatches are simulated in the amplicon sequences using the `PBSIM` software [173]. Specifically, errors are introduced independently at each position of a sequence according to the following deletion, insertion, and mismatch probabilities, P_D, P_I, and P_{Mis}, respectively. The deletion probability P_D is assumed constant at each position of a simulated read and defined as

$$\begin{aligned} P_D &= \Pr(\text{Deletion}|\text{Error}) \Pr(\text{Error}) \\ &= \frac{R_D}{R_D + R_I + R_{Mis}} \mu_{error}, \end{aligned}$$

where μ_{error} is a user-supplied error probability for the read set and R_D, R_I, and R_{Mis} are, respectively, user-supplied ratios of deletions, insertions, and mismatches. The insertion and mismatch probabilities are computed for each position of a simulated read from the quality score Q of the nucleotide at that position:

$$\begin{aligned} P_I(Q) &= \Pr(\text{Insertion}|\text{Error}) \Pr(\text{Error}, Q) \\ &= \frac{R_I}{R_D + R_I + R_{Mis}} 10^{-Q/10}, \\ P_{Mis}(Q) &= \Pr(\text{Mismatch}|\text{Error}) \Pr(\text{Error}, Q) \\ &= \frac{R_{Mis}}{R_D + R_I + R_{Mis}} 10^{-Q/10}, \end{aligned}$$

where $\Pr(\text{Error}, Q) = 10^{-Q/10}$ is the error probability of nucleotides with quality score Q and Q is sampled from an in-house FASTQ file containing previously sequenced PacBio CCS reads for 16S genes. Moreover, mismatches are simulated by using a uniform distribution over the four nucleotides $\{A, C, G, T\}$, while half of inserted nucleotides are chosen to be the same as their following nucleotide and the other half selected from a uniform distribution over $\{A, C, G, T\}$. For our simulation, default parameters of `PBSIM` are used, that is, $\mu_{error} = 0.02$, $R_D = 0.73$, $R_I = 0.21$, and $R_{Mis} = 0.06$.

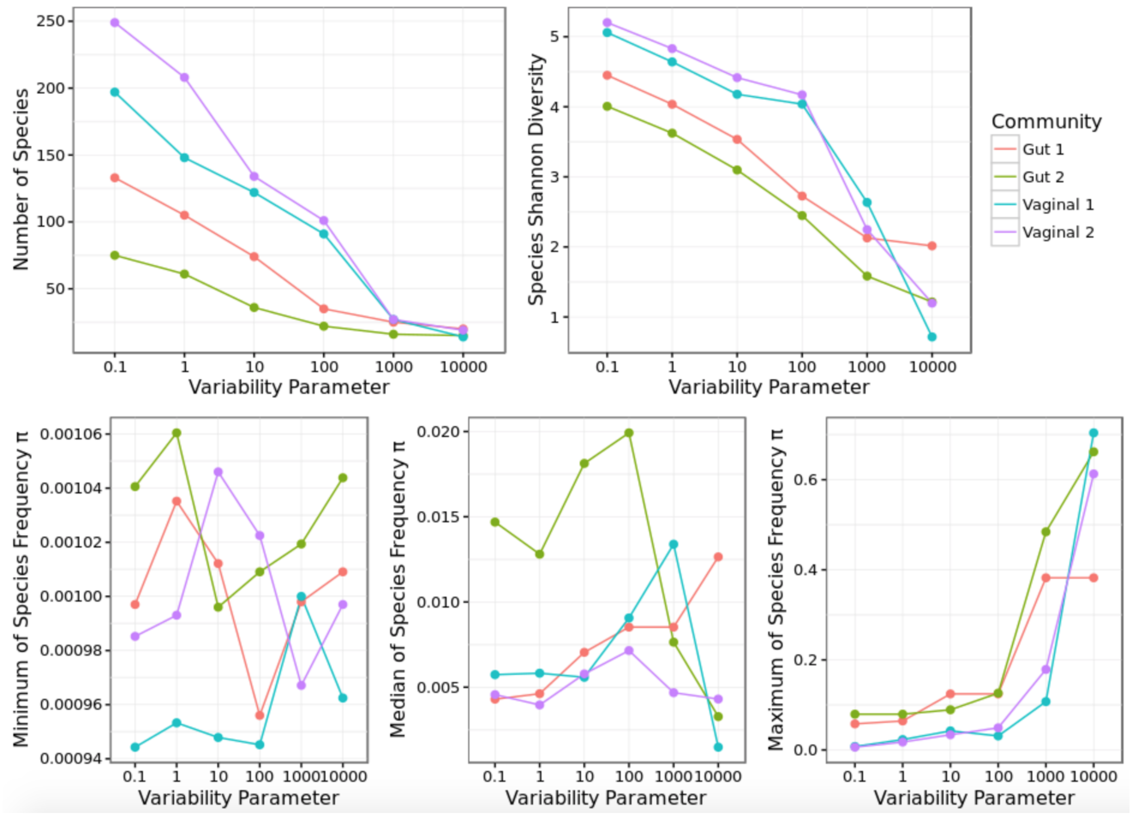


Figure 4.9: *Overview of bacterial frequencies for the simulated datasets.* Read datasets were simulated using 24 different sets of bacterial frequencies $\pi = (\pi_k : k = 1, \ldots, K)$ (at the species level). The frequencies π correspond to four HMP communities (two gut and two vaginal) and six different variability parameters ($V \in \{0.1, 1, 10, 100, 1,000, 10,000\}$) for the sampling of species from each of these communities. Top-left panel: Number of distinct species $\sum_k \mathbb{1}(\pi_k > 0)$. Top-right panel: Shannon diversity of bacterial frequencies $-\sum_k \pi_k \log \pi_k$, providing a measure of richness and evenness of a microbial community. Bottom panels, from left to right: Minimum, median, and maximum of the bacterial frequencies π_k. While minimum and median bacterial frequencies tend to be similar across simulated datasets, the maximum bacterial frequency is higher when the variability parameter is larger, introducing imbalance in the microbial community. Note that the y-axis scales are different for the graphs in the bottom panels.

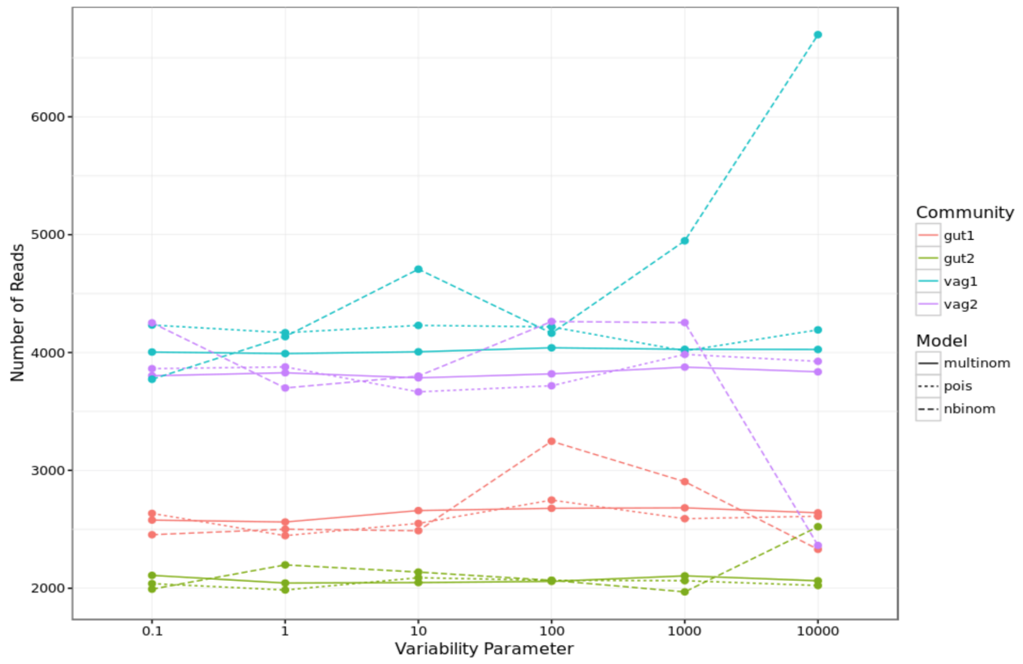
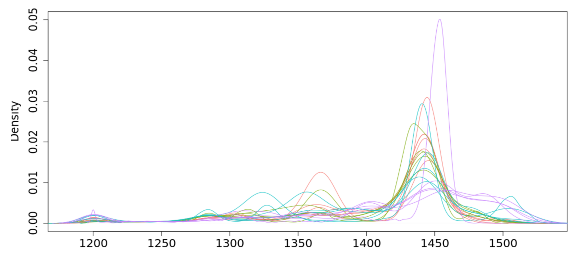
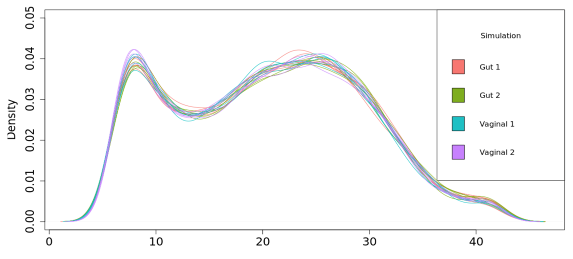


Figure 4.10: *Number of reads in simulated datasets.* The total number of bacteria in a simulated community $\mathcal{Z}$ is $m = 1,000$. If there was only one 16S gene per bacterium, the number of reads would be equal to the number of bacteria. In practice, however, there are multiples variants and copies of a 16S gene for a given bacterium, so the number of reads n is greater than m. The number of reads is bigger for the vaginal simulations than for the gut simulations because there are more 16S genes in the genome of the species picked for the vaginal simulations. Not surprisingly, the number of reads is more variable for the negative binomial model as the number of bacteria simulated for each species b_k is more variable. It is especially true when π_k is big. For example, for *Vaginal 1*, $V = 10,000$, and model negative binomial, the number of reads is big. In this simulated dataset, the two most abundant strains are from species *Lactobacillus vaccinostercus* ($\pi = 0.70$) and *Lactobacillus sanfranciscensis* ($\pi = 0.27$). Therefore, the means of the negative binomial were respectively $0.7 * 1000 = 700$ and $0.27 * 1000 = 270$, but because of the over-dispersion of the negative binomial, the number of bacteria simulated were respectively 991 and 644. Both of the strains have four 16S genes in their genome, explaining the about $6,500$ reads in this dataset.



Panel (a): Length.



Panel (b): Quality scores.

Figure 4.11: Panel (a): Distribution of read lengths for each of the 72 simulated datasets. Panel (b): Distribution of Phred read quality scores for each of the 72 simulated datasets (extracted from FASTQ file generated by software `PBSIM`).

4.2.1.2 Microbial mock community

We estimate the performance of our methods on a mock community composed of an even distribution of genomic DNA from 21 bacterial strains: Acinetobacter baumannii ATCC 17978, Actinomyces odontolyticus ATCC 17982, Bacillus cereus ATCC 10987, Bacteroides vulgatus ATCC 8482, Clostridium beijerinckiiATCC 51743, Deinococcus radiodurans ATCC 13939, Enterococcus faecalis ATCC 47077, Escherichia coli ATCC 70096, Helicobacter pylori ATCC 700392, Lactobacillus gasseri ATCC33323, Listeria monocytogenes ATCC BAA-679, Neisseria meningitidis ATCC BAA-335, Porphyromonas gingivalis ATCC 33277, Propionibacterium acnes DSM 16379, Pseudomonasaeruginosa ATCC 47085, Rhodobacter sphaeroides ATCC 17023, Staphylococcus aureusATCC BAA-1718, Staphylococcus epidermidis ATCC 12228, Streptococcus agalactiae ATCCBAA-611, Streptococcus mutans ATCC 700610, and Streptococcus pneumoniae ATCC BAA-334. The data used here are a subset of the data used in [182] (v3.1, HM-278D) and were downloaded from the Sequence Read Archive at NCBI under accession SRP051686 associated with BioProject PRJNA271568.

Library generation, sequencing, and pre-processing to generate the downloaded reads are described in [182]. Briefly, we used reads sequenced from the V1-V9 variable region (full length 16S rRNA gene) sequenced by Pacific Biosciences using the P6-C4 chemistry on a PacBio RS II SMRT DNA Sequencing System with MagBead loading. We filtered out reads with length smaller than 1,300 and greater than 1,600 bp resulting in a dataset with 66,450 reads. See the distribution of the length of the reads in Figure 4.13. For this mock community, the individual DNA extracts were mixed based on the genome size and the number of different loci where 16S rDNA genes are in each genome to have equal-molar 16S rDNA copies for each species [183]. So, as opposed to section 4.1.5.2, we did not account for the multiple variants and copies of the 16S gene for the different strains.

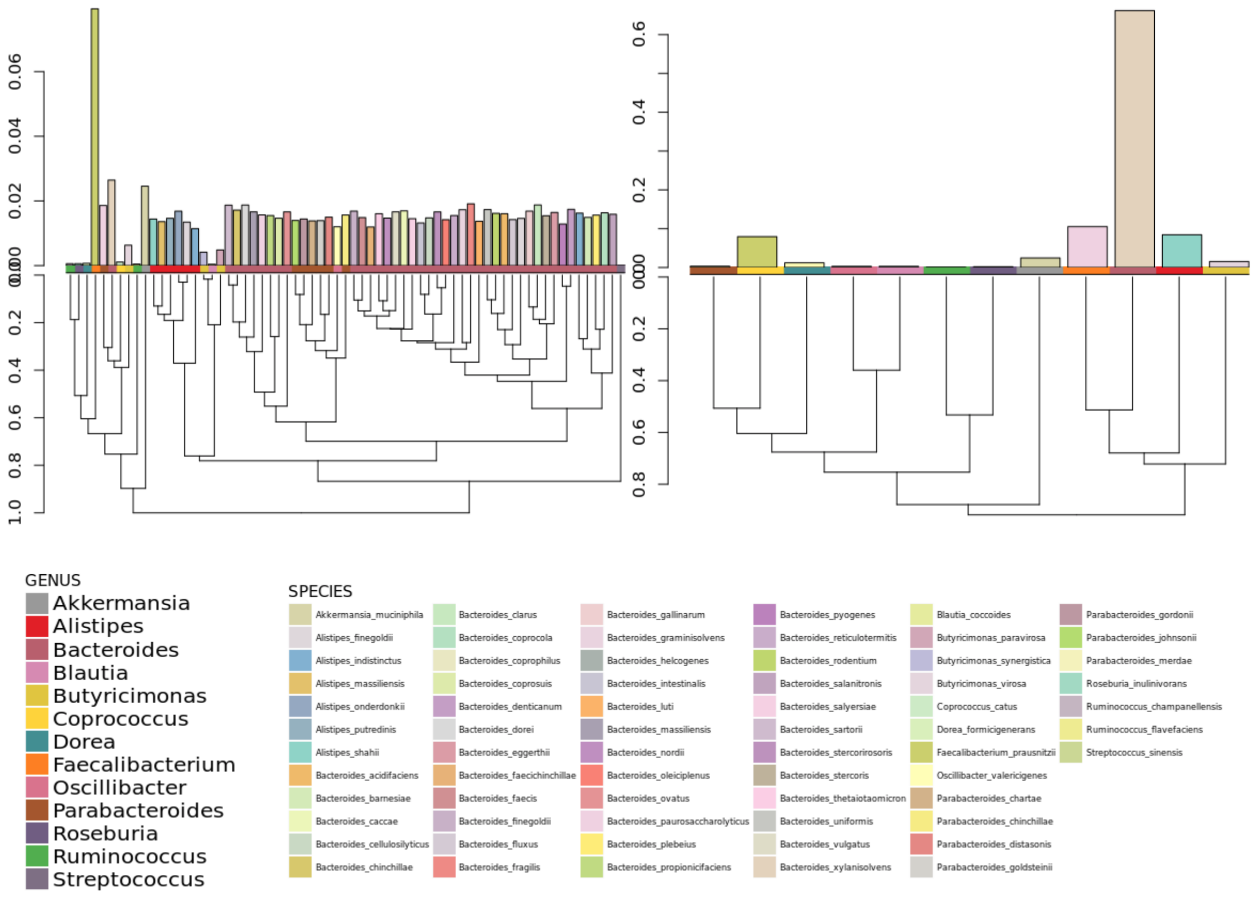


Figure 4.12: *Phylogenetic tree and frequencies of bacterial species π in two simulation scenarios.* Barplots in the top-left and top-right panels represent bacterial frequencies π_k for community *Gut 2* with the two most extreme variability parameters, respectively, $V = 0.1$ and $V = 10,000$. The smallest variability parameter yields a rich and balanced microbial community, whereas the greatest variability parameter results in a less rich and more imbalanced community. The phylogenetic trees in the bottom panels represent the similarity between the bacterial species, where the distance between two species is the Levenshtein distance between their consensus 16S gene sequences. The horizontal colored bars above the phylogenetic trees indicate the genus of each species.

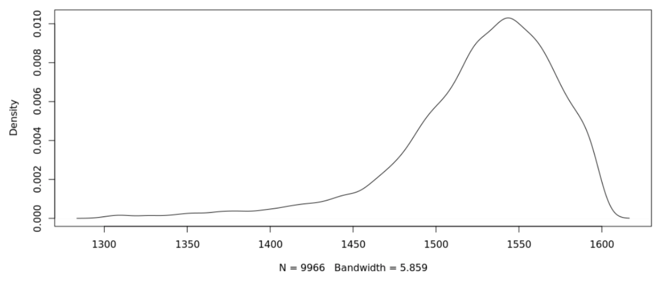


Figure 4.13: Distribution of the length of the reads in the filtered dataset for the mock community.

4.2.2 16S Database

It is essential to use a reference database with high confident taxonomic classifications. In our method, each read is assigned the same taxonomic level as its most similar sequence in a reference database. Then, if the database has annotation errors, a prediction could be wrong not because of incorrect assignment of the read to a reference sequence, but because of incorrect annotations of the reference sequences in the database. Among the different available databases (e.g., SILVA, Greengenes), we decided to use the full RDP database because it is exhaustive, i.e., it contains most of the sequences found in the other databases.

First, we downloaded the full RDP database, version 11.4, with 3,070,243 16S gene sequences. After filtering out sequences with length outside of the range 1,300bp to 1,600bp, sequences with ambiguous bases (i.e., nucleotides that are not A,C,G,T), sequences with identical DNA sequences (Bowtie2 does not allow duplicated sequences), and sequences with entirely redundant annotation (i.e., annotation where the only difference is the RDP identifier), 1,362,820 16S gene sequences remained. To allow the possibility that novel reference sequences could enter the database, we used 32-byte md5 hash values as the sequences identifiers. Then, when we want to add a sequence to the database, we compute its md5-hash value, determine if it exists in our dataset, if so, simply add the annotation to the corresponding reference sequence, otherwise construct a new entry in our database.

The last step is to add lineage when it is not specified in the filtered RDP database. The taxonomies in the full RDP database were predicted by the RDP Classifier which has a high rate of over-classification errors (i.e., novel taxa are incorrectly predicted to have known names) on full-length sequences [169]. To get a database with only well-annotated reference sequences, we filtered out the filtered RDP database to keep only reference sequences with taxonomic annotations at the species level, often added manually to the database, thus highly accurate annotations. It resulted in a species level annotated RDP database with 81,088 16S gene sequences. We then used this species level annotated database to aligned the entire set of sequences in the filtered RDP database. If alignments surpass 99% of simi-

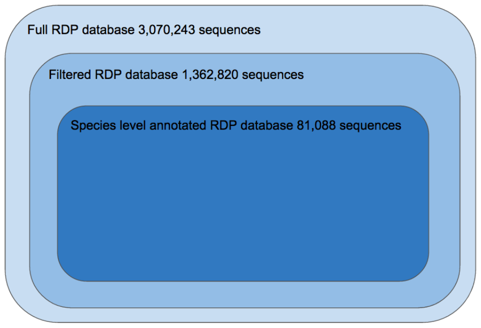

Full RDP database 3,070,243 sequences
Filtered RDP database 1,362,820 sequences
Species level annotated RDP database 81,088 sequences

Figure 4.14: *16S databases.* The full RDP database, version 11.4, with 3,070,243 16S gene sequences was downloaded and filtered out to get a filtered RDP database with 1,362,820 16S gene sequences. To create a database with only well-annotated reference sequences, we filtered out the filtered RDP database to keep only reference sequences with taxonomic annotations at the species level, resulting in a reduced RDP database with 81,088 16S gene sequences. Both the filtered and species level annotated RDP databases were used to classify the reads.

larity, we re-annotated the reference sequence with the best matching reference's annotation.

Both filtered and species level annotated databases were used to classify reads. See Figure 4.14. The former is used to increase the sensitivity (i.e. decrease the number of false negatives, that is make sure we do not miss strains) at the risk of assigning reads to sequences with incorrect annotations. The later is used to increase the specificity (i.e. decrease the number of false positives, that is make sure we do not incorrectly call a strain present while it is absent) at the risk of missing strains that have been filtered out from the database. In this report, we only show results when the species level annotated RDP database is used.

4.3 Methods

4.3.1 Training and Validation of Estimation Procedure

In order to train our bacterial composition estimation procedure, i.e., select optimal Bowtie2 tuning parameters and estimate the parameters of the HMM, and evaluate its overall accuracy, we dispose of an annotated dataset $\widetilde{\mathcal{Y}}$ for which we know, for each read Y_i, its corresponding true 16S gene sequence $\tilde{Y}_i$,

$$\tilde{\mathcal{Y}} = \{(Y_i, \tilde{Y}_i) : i = 1, \ldots, n\} \text{ with } Y_i \in \{A, C, G, T\}^{l(Y_i)} \text{ and } \tilde{Y}_i \in \{A, C, G, T\}^{l(\tilde{Y}_i)}.$$

We divide the annotated learning dataset $\widetilde{\mathcal{Y}}$ at random into two datasets:

- a training set $\widetilde{\mathcal{Y}}_0$, containing n_0 reads ($n_0 = 5,000$), used only to select optimal Bowtie2 tuning parameters and estimate the parameters of the HMM,

- a validation set $\widetilde{\mathcal{Y}}_1$, containing $n_1 = n - n_0$ reads ($n_1 = 5,000$), used only to assess the overall accuracy of the procedure trained using $\widetilde{\mathcal{Y}}_0$.

By construction, reads in $\widetilde{\mathcal{Y}}$ are generated from 16S gene sequences present in our database $\mathcal{R}$, that is, $\tilde{Y}_i \in \mathcal{R}$ for each $i = 1, \ldots, n$, so that one can identify for each pair $(Y_i, \tilde{Y}_i)$ the bacteria of origin $b(\tilde{Y}_i)$.

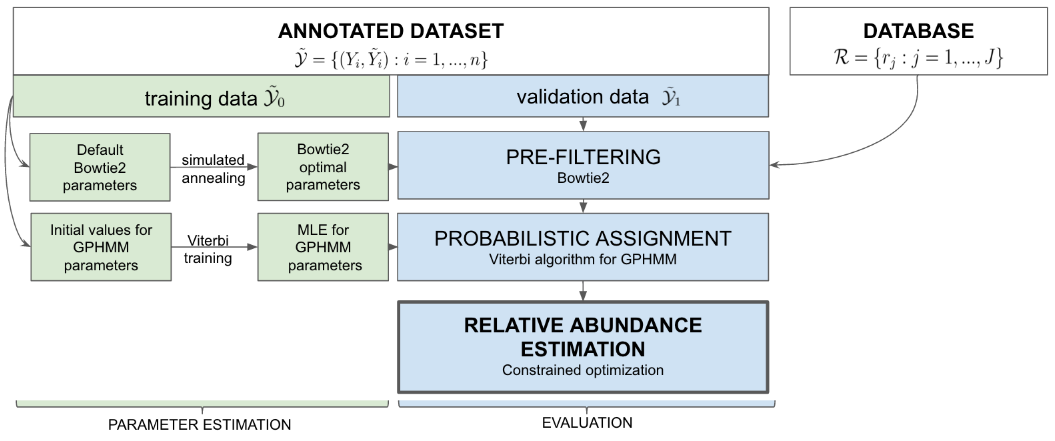

ANNOTATED DATASET
$\tilde{\mathcal{Y}} = \{(Y_i, \tilde{Y}_i) : i = 1, \ldots, n\}$
DATABASE
$\mathcal{R} = \{r_j : j = 1, \ldots, J\}$
training data $\tilde{\mathcal{Y}}_0$
validation data $\tilde{\mathcal{Y}}_1$
Default Bowtie2 parameters
simulated annealing
Bowtie2 optimal parameters
PRE-FILTERING
Bowtie2
Initial values for GPHMM parameters
Viterbi training
MLE for GPHMM parameters
PROBABILISTIC ASSIGNMENT
Viterbi algorithm for GPHMM
RELATIVE ABUNDANCE ESTIMATION
Constrained optimization
PARAMETER ESTIMATION
EVALUATION

Figure 4.15: *Pipeline for training and validation of bacterial composition estimation procedure.*

4.3.2 Pre-filtering

4.3.2.1 Bowtie2

To estimate the bacterial composition of a microbiomic sample, we need to compute the likelihood $\Pr(\mathcal{Y}; \pi)$, meaning that we need to compute $\Pr(Y_i = y_i | X_i = r_j)$ for each read y_i in $\mathcal{Y}$ and each reference sequence r_j in $\mathcal{R}$ (Section 4.1.5.3). The number of reference sequences in $\mathcal{R}$ being on the order of 1.4 million and the number of reads in $\mathcal{Y}$ on the order of the thousands, such a computation is intractable. To reduce the number of computations, we use the alignment tool Bowtie2 to eliminate reference sequences with small probabilities of having generated the reads in $\mathcal{Y}$.

Bowtie2 first indexes the set of reference sequences in our database $\mathcal{R}$ using a scheme based on the Burrows-Wheeler transform [184] and FM-index [185], which allows compression of the database while still permitting fast substring queries. Then, each read in $\mathcal{Y}$ is divided into substrings – called seeds – and only reference sequences matching almost perfectly the seed substrings are kept. Seed substrings are then extended to the entire length of the read and a score is given to each alignment between the read and the selected reference sequences. This alignment score (AS) quantifies how similar a read is to a reference sequence it is aligned to; the higher the score, the higher the chance that the read could have been generated from the reference sequence. The score is calculated by adding a "bonus" for each match and subtracting a "penalty" for each difference (mismatch, insertion, or deletion) between the read and the reference sequence. Only reference sequences with an AS higher than a minimum alignment score (min-score) are selected. That is, for each read Y, the set of selected reference sequences is given by the mapping

$$\text{bowtie}(Y) \equiv \{r_j \in \mathcal{R} : \text{AS}(Y, r_j) \geq \text{min-score}\}. \tag{4.7}$$

Unlike other index-based alignment tools, such as, Burrows-Wheeler Aligner (BWA) [186], BWA's Smith-Waterman alignment (BWA-SW) [187], and short oligonucleotide alignment program 2 (SOAP2) [188], Bowtie2 has a rich set of tuning parameters, that have a large influence on the selection of the candidate reference sequences and are highly dependent on read length. As reads in our dataset are long reads, spanning the entire length of the 16S rRNA genes ($\simeq 1,500$ bp), and Bowtie2 default parameters are tuned for shorter reads, it is essential to select appropriate parameters for our setting:

- L, the length of the seed substrings,
- i, the interval between seed substrings,
- R, the maximum number of times Bowtie2 re-seeds reads with repetitive seeds,
- D, the number of consecutive seed extension attempts that can fail before Bowtie2 moves on,
- k, the maximum number of candidate reference sequences that can be selected during the seed search,
- min-score, the minimum alignment score needed for an alignment to be good enough to be selected,
- the parameters configuring the alignment scores: match bonus (ma), mismatch penalty (mp), deletion penalty (rdg), and insertion penalty (rfg).

For example, if for read Y the maximum number of candidates k allowed during the seed search is set low and there are many more candidate reference sequences with an AS higher than min-score, then Bowtie2 can select k candidate reference sequences matching the seeds in Y before finding the correct reference sequence. See the Bowtie2 manual for more details on tuning parameters.

Our goal is to find the optimal set of Bowtie2 tuning parameters, $\theta = \{$L, i, R, D, k, min-score, ma, mp, rdg, rfg$\}$, so that the true sequence for read Y is among the candidate reference sequences found by Bowtie2. For our training set $\tilde{\mathcal{Y}}_0$ (Section 4.3.1), we know that each observed noisy read Y_i was generated from a true sequence $\tilde{Y}_i$. Then, we can define an objective function L to be maximized over the set of Bowtie2 tuning parameters θ:

$$L(\theta; \widetilde{\mathcal{Y}}_0) \equiv \frac{1}{n_0} \sum_{i=1}^{n_0} \mathbb{1}(\tilde{Y}_i \in \text{bowtie}_\theta(Y_i)), \tag{4.8}$$

where bowtie_θ is the function mapping each read Y_i to the candidate reference sequences selected by Bowtie2 with tuning parameters θ.

4.3.2.2 Selecting optimal Bowtie2 parameters

Simulated annealing (SA) [189] is used to maximize the objective function $L(\theta; \widetilde{\mathcal{Y}}_0)$. SA is a powerful technique for approximating global optimization in a large search space and is known to perform well when the search space is discrete – which is the case here. For each individual parameter in θ, a search domain with lower and upper bounds is specified and random initial estimates of the parameters are chosen within the search domains. At each iteration i, the SA heuristic considers some neighboring parameter θ_{i+1} of the current parameter θ_i. The probability of making the transition from the current θ_i to the candidate new parameter θ_{i+1} (acceptance probability) is then calculated. The system moves ultimately to sets of parameters with higher objective function, the process being repeated until a stopping criterion is reached.

More precisely, at each step i, reads in our training set $\widetilde{\mathcal{Y}}_0$ are aligned to our database $\mathcal{R}$ using Bowtie2 with parameter θ_i and an acceptance probability is calculated as follows

$$\exp\left(\frac{L(\theta_{i+1}; \widetilde{\mathcal{Y}}_0) - L(\theta_i; \widetilde{\mathcal{Y}}_0)}{T}\right), \tag{4.9}$$

with T a global time-varying parameter called temperature. If the acceptance probability is larger than a value sampled uniformly between zero and one, then θ is set to the new value θ_{i+1}, otherwise it stays as θ_i. The search ends when an acceptable solution is found, that is, the objective function is higher than a threshold, or the maximum number of iterations is reached. The pseudocode corresponding to the description above is provided in Algorithm 1.

Algorithm 1: Simulated annealing

Data: Training set $\widetilde{\mathcal{Y}}_0$.

Result: Bowtie2 tuning parameters $\hat{\theta}$ maximizing objective function L

$$L(\theta; \widetilde{\mathcal{Y}}_0) = \frac{1}{n_0} \sum_{i=1}^{n_0} \mathbb{1}(\tilde{Y}_i \in \mathrm{bowtie}_\theta(Y_i)).$$

```
1 $\theta \leftarrow \theta_1$ initial random tuning parameters within specified domains;
2 **while** *stopping criteria not reached* **do**
3     Select a neighboring set of tuning parameters $\theta_{i+1}$;
4     **If** $\exp\left(\frac{L(\theta_{i+1};\widetilde{\mathcal{Y}}_0) - L(\theta_i;\widetilde{\mathcal{Y}}_0)}{T}\right) > \mathrm{U}([0,1])$;
5     **then** $\theta \leftarrow \theta_{i+1}$.
6 **end**
```

4.3.3 Modeling the Sequencing Process

We want to compute the probability $\Pr(Y|X)$ that an observed noisy read Y in our dataset $\mathcal{Y}$ was generated from an unobserved 16S gene sequence X. To do so, we need to consider the set $\mathcal{Q}$ of all possible alignments between X and Y, i.e., the ways X can be construed to have generated Y,

$$\Pr(Y|X) = \sum_{Q \in \mathcal{Q}} \Pr(Y, Q|X). \tag{4.10}$$

Specifically, three types of errors can occur in the sequencing of a given gene X:

- mismatches, which substitute a nucleotide in X with another nucleotide in Y,
- insertions, which add nucleotides in Y compared to X,
- deletions, which delete nucleotides in Y compared to X.

An alignment Q between two sequences X and Y therefore consists of a sequence of matches, mismatches, insertions, and deletions.

Using Equation (4.10), the probability $\Pr(Y, Q|X)$ would have to be computed for each alignment Q in $\mathcal{Q}$. The number of possible alignments between X and Y being the product of the lengths of X and Y ($l(X)l(Y) \simeq 1,500^2 = 2.25 \times 10^6$ for our problem), the computation is prohibitive. It has been shown that computing the probability of the optimal alignment between X and Y, instead of summing over the probabilities of all possible alignments, is substantially more computationally efficient, with no impact on accuracy [ref, PacBio suppl. paper, other ref]. Thus, we compute

$$\max_{Q \in \mathcal{Q}} \Pr(Y, Q|X) = \Pr(Y, Q^\star|X), \tag{4.11}$$

where $Q^\star \equiv \arg\max_{Q \in \mathcal{Q}} \Pr(Y, Q|X)$ is the most probable alignment between X and Y.

As detailed next, we model the sequencing process relating Y to X using a generalized pair hidden Markov model and use the Viterbi algorithm to infer the optimal alignment $Q^\star$. Although the true sequence X is unobserved, in what follows, X is treated as observed while the alignment Q is unobserved.

4.3.3.1 Generalized pair hidden Markov model

Instead of observing a single sequence of random variables, as is usually the case for a hidden Markov model (HMM), here, we have a pair of sequences of random variables, namely, the unaligned nucleotide sequences for a read Y and the putative 16S gene X it was sequenced from. We therefore adopt a generalized pair hidden Markov model (GPHMM) to model the sequencing process, where each hidden state emits a pair of sequences. As seen below, our model is a special case of a generalized pair HMM, in the sense that the durations are deterministic given the hidden states.

Our GPHMM is defined as follows, where, for the rest of Section 4.3.3, we modify the notation adopted above for an observed read Y and the corresponding unobserved 16S gene sequence X.

1. **Hidden states and durations.** Let $\{X_t : t = 1, \dots, T\}$ denote the hidden states corresponding to a particular pairwise alignment between two sequences $\bar{Y}^1$ and $\bar{Y}^2$, representing, respectively, a 16S gene and its corresponding sequencing read. Three hidden states $\mathcal{S} = \{M, Ins, Del\}$ are required to represent the sequence of matches/mismatches, insertions, and deletions corresponding to a particular pairwise alignment,

   - M, the state for matches/mismatches,
   - Ins, the state for insertions,
   - Del, the state for deletions.

   To each hidden state $X_t \in \mathcal{S}$, we associate a pair of also hidden random durations $D_t = (D_t^1, D_t^2) \in \{0, 1, \dots\}^2$, generated according to the conditional distribution $p_i(d) \equiv \Pr(D_t = d | X_t = i)$, and denoting the number of nucleotides emitted for the 16S gene sequence and the read, respectively. In our application, we consider a special case where durations are either 0 or 1 and are deterministic given hidden states, i.e., $D_t = (D_t^1, D_t^2) \in \{(0,1), (1,0), (1,1)\}$ and

$$\begin{aligned} \Pr(D_t = (1,1) | X_t = M) &= 1, \\ \Pr(D_t = (0,1) | X_t = Ins) &= 1, \\ \Pr(D_t = (1,0) | X_t = Del) &= 1. \end{aligned} \tag{4.12}$$

   Let $L_t^1 \equiv \sum_{s=1}^t D_s^1$ and $L_t^2 \equiv \sum_{s=1}^t D_s^2$ denote the cumulative durations up to time t.

2. **Emitted sequences.** Each hidden state X_t emits a pair of sequences $\bar{Y}_t = (\bar{Y}_t^1, \bar{Y}_t^2)$, where $\bar{Y}_t^h \equiv Y^h_{L^h_{t-1}+1:L^h_t} = (Y_s^h : s = L_{t-1}^h + 1, \dots, L_t^h)$, $h = 1, 2$. In our special case, $D_t^h \in \{0, 1\}$ and

   - if $D_t^h = 1$, then one nucleotide $\bar{Y}_t^h = Y^h_{L_t^h} \in \{A, C, G, T\}$ is emitted at location L_t^h,
   - if $D_t^h = 0$, then no nucleotide is emitted and $\bar{Y}_t^h$ is the empty string $\emptyset$.

   Then, the unobserved emitted aligned pair of sequences is

$$\left\{ \bar{Y}_t = (\bar{Y}_t^1, \bar{Y}_t^2) : t = 1, \dots, T \right\}$$

   and yields an observed unaligned pair of sequences

$$(\bar{Y}^1, \bar{Y}^2) = (Y^1_{1:L_T^1}, Y^2_{1:L_T^2}),$$

   where $L_T^1 = \sum_{s=1}^T D_s^1 = l(\bar{Y}^1)$ and $L_T^2 = \sum_{s=1}^T D_s^2 = l(\bar{Y}^2)$ are the lengths of the true 16S gene sequence and the read, respectively. Note that, in this GPHMM, only $Y^1_{1:L_T^1}$ and $Y^2_{1:L_T^2}$ are observed, that is, X_t, D_t, $\bar{Y}_t$, and T are hidden.

3. **State transition probability distribution.** We denote the state transition probability matrix by $A = (A_{ij} : i, j \in \mathcal{S})$. To simplify our model, we assume that transitions from state Ins to state Del and from Del to Ins are not allowed. We also define the following probabilities

   - γ_I the transition probability from state M to state Ins,
   - γ_D the transition probability from state M to state Del,
   - ϵ_I the probability of staying in state Ins,
   - ϵ_D the probability of staying in state Del.

   The state transition probability matrix A can then be written as

$$A = \begin{array}{c} \\ M \\ Ins \\ Del \end{array} \begin{array}{c} \begin{array}{ccc} M & Ins & Del \end{array} \\ \begin{pmatrix} 1-\gamma_I-\gamma_D & \gamma_I & \gamma_D \\ 1-\epsilon_I & \epsilon_I & 0 \\ 1-\epsilon_D & 0 & \epsilon_D \end{pmatrix} \end{array}, \tag{4.13}$$

   where the other entries of A result from noting that A is a stochastic matrix, i.e., its rows sum to one. For example, the probability to stay in state M is $1 - \gamma_I - \gamma_D$.

4. **Initial state distribution.** The initial state distribution $a = (a_i : i \in \mathcal{S})$ is the same as the first row of the transition matrix A, that is,

$$a_i \equiv \Pr(X_1 = i) = A_{M,i}, \ \forall i \in \mathcal{S}. \tag{4.14}$$

5. **Emission probability distribution.**

   Emitted sequences $\bar{Y}_t$ are generated according to the conditional joint distribution

$$B_{i,d}(\bar{Y}_t) \equiv \Pr(\bar{Y}_t^1, \bar{Y}_t^2 | X_t = i, D_t = d), \tag{4.15}$$

   which depends only on the current hidden state $X_t = i \in \mathcal{S}$ and pair of durations $D_t = d \in \{(0,1),(1,0),(1,1)\}$ and is represented in the matrix in Figure 4.16.

For convenience, we denote by $\lambda \equiv (A, B)$ the entire parameter set of our model.

A pair of sequences can be generated as follows from our GPHMM, for a given value of the parameter λ.

1. Generate the first hidden state $X_1 \in \mathcal{S}$ according to initial state distribution a in Equation (4.14). Given state X_1, generate durations $D_1 = (D_1^1, D_1^2)$ according to Equation (4.12). Given state X_1 and durations D_1, emit a pair of sequences $\bar{Y}_1 = (\bar{Y}_1^1, \bar{Y}_1^2)$ according to the emission distribution B. For example, in Figure 4.17, $X_1 = M$, thus $p_M(1,1) = 1$ and $D_1 = (1,1)$. The emitted pair of sequences is then chosen according to $B_{M,(1,1)}$ and is, for instance, $(\bar{Y}_1^1, \bar{Y}_1^2) = (A, A)$.

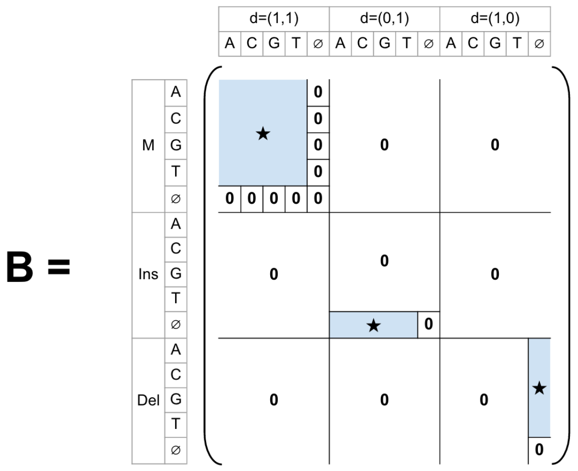


Figure 4.16: *Emission probability matrix.* Stars represent non-null entries.

2. Given X_1, select the next hidden state $X_2 \in \mathcal{S}$ according to the transition probability matrix A. For example, in Figure 4.17, $X_2 = Del$, thus $p_{Del}(1,0) = 1$ and $D_2 = (1,0)$. The emitted pair of sequences is then chosen according to $B_{Del,(1,0)}$ and is, for instance, $(\bar{Y}^1_2, \bar{Y}^2_2) = (C, \emptyset)$.

3. Repeat Step 2 until $L^1_t = l(\bar{Y}^1)$ and $L^2_t = l(\bar{Y}^2)$.

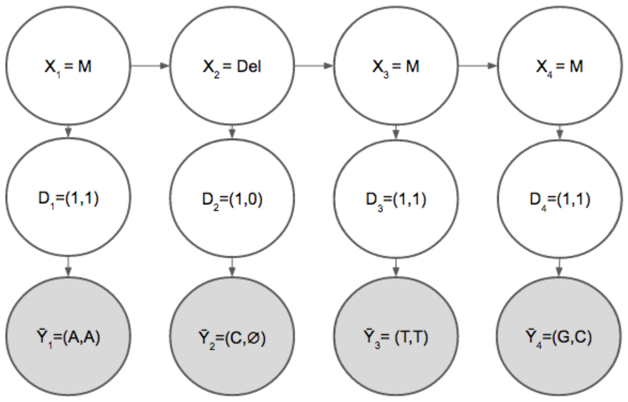
4.3.3.2 Most probable alignment: Viterbi algorithm

We want to find the sequence of hidden states X_t that describes best the fact that the read $\bar{Y}^2$ could have been sequenced from the 16S gene $\bar{Y}^1$, where our definition of best means maximizing the conditional probability of the sequences of hidden states $\bar{X} = (X_t : t = 1, \ldots, T)$ and pairs of durations $\bar{D} = (D_t : t = 1, \ldots, T)$ given the read and its corresponding true sequence. That is, we want to maximize $\Pr(\bar{X}, \bar{D}|\bar{Y}^1, \bar{Y}^2)$, which is equivalent to maximizing $\Pr(\bar{X}, \bar{D}, \bar{Y}^1, \bar{Y}^2)$, over $\{T, \bar{X}, \bar{D}\}$.

We use the dynamic programming method called the Viterbi algorithm, which involves computing the function $\delta(i, (d_1, d_2), (l_1, l_2))$, defined as the highest probability of an emitted pair of sequences of lengths (l_1, l_2) and corresponding sequences of hidden states and durations, ending in hidden state $X_t = i$ with durations $D_t = (d_1, d_2)$,

Additionally, to retrieve the optimal sequences of hidden states and durations for the alignment, we need to keep track of the arguments i and (d_1, d_2) that maximize $\delta(i, (d_1, d_2), (l_1, l_2))$ for each pair of emitted sequence lengths (l_1, l_2). To do so, we use a function $\psi(i, (d_1, d_2), (l_1, l_2))$.

More efficient implementation of the Viterbi algorithm The emission probability matrix in Figure 4.16 being sparse, it is possible to reduce the number of iterations in the re-



t	t = 1	t = 2	t = 3	t = 4
$Y^1_{1:L^1_t}$	A	AC	ACT	ACTG
$Y^2_{1:L^2_t}$	A	A	AT	ATC

Figure 4.17: *Generalized pair hidden Markov model.* Generation of a 16S gene sequence and its corresponding sequencing read using our generalized pair HMM. Upper white circles represent hidden states, intermediate white circles durations, and lower grey circles pairs of nucleotide sequences emitted at each state. The table below the graph represents the unaligned pairs of sequences up to time $t = 1, \dots, T = 4$. Only the 16S gene $Y^1_{1:4} = ACTG$ and the read $Y^2_{1:3} = ATC$ are observed.

cursion of Equation (**??**). Durbin et al. [190] propose a faster implementation of the Viterbi algorithm, where *Begin* and *End* states are added for, respectively, the initialization and termination of a pairwise alignment and where one does not make use of durations D_t.

There are no emissions from the *Begin* and *End* states. The transition probabilities from any hidden state in $\{M, Ins, Del\}$ to the *End* state are set to τ and the transition probabilities from the *Begin* state to any hidden state in $\{M, Ins, Del\}$ are given by the first row of the transition probability matrix A in Equation (4.13). Transitions from and to the *Begin* and *End* states are designated by dashed lines in Figure 4.19. For simplicity, it is assumed that an alignment starts in the M state. With Y^1_i denoting the nucleotide at location i in the true sequence $\bar{Y}_1$ and Y^2_j the nucleotide at location j in the read $\bar{Y}_2$, the algorithm is as follows.

1. **Initialization.** For $i \in \{1, \dots, l(\bar{Y}_1)\}$, $j \in \{1, \dots, l(\bar{Y}_2)\}$, and $k \in \{M, Ins, Del\}$, let

$$\begin{aligned} \delta^k(i,0) &= 0, \\ \delta^k(0,j) &= 0, \\ \delta^M(1,1) &= 1 \end{aligned} \tag{4.16}$$

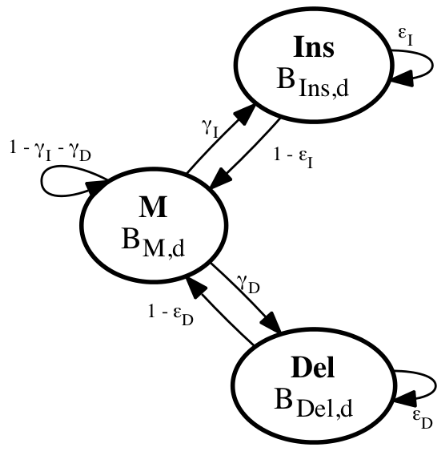


Figure 4.18: *Generalized pair hidden Markov model.* Hidden states and transition and emission probability distributions.

and

$$\psi^k(i,0) = \psi^k(0,j) = \psi^M(1,1) = 0.$$

2. **Recursion.** For $(i,j) \in \{1,\ldots,l(\bar{Y}_1)\} \times \{1,\ldots,l(\bar{Y}_2)\} \setminus \{(1,1)\}$,

$$\begin{aligned}
\delta^M(i,j) &= B_M(Y_i^1, Y_j^2) \max \begin{cases} (1-\gamma_I-\gamma_D-\tau)\delta^M(i-1,j-1) \\ (1-\epsilon_I-\tau)\delta^{Ins}(i-1,j-1) \\ (1-\epsilon_D-\tau)\delta^{Del}(i-1,j-1) \end{cases}, \quad (4.17) \\
\delta^{Ins}(i,j) &= B_{Ins}(\emptyset, Y_j^2) \max \begin{cases} \gamma_I \delta^M(i,j-1) \\ \epsilon_I \delta^{Ins}(i,j-1) \end{cases}, \\
\delta^{Del}(i,j) &= B_{Del}(Y_i^1, \emptyset) \max \begin{cases} \gamma_D \delta^M(i-1,j) \\ \epsilon_D \delta^{Del}(i-1,j) \end{cases}
\end{aligned}$$

and

$$\begin{aligned}
\psi^M(i,j) &= \arg\max_{k\in\{M,Ins,Del\}} \delta^k(i-1,j-1)\Big(\mathbb{1}(k=M)(1-\gamma_I-\gamma_D-\tau)+\\
&\qquad \mathbb{1}(k=Ins)(1-\epsilon_I-\tau)+\\
&\qquad \mathbb{1}(k=Del)(1-\epsilon_D-\tau)\Big),\\
\psi^{Ins}(i,j) &= \arg\max_{k\in\{M,Ins,Del\}} \delta^k(i,j-1)\Big(\mathbb{1}(k=M)\gamma_I+\mathbb{1}(k=Ins)\epsilon_I\Big),\\
\psi^{Del}(i,j) &= \arg\max_{k\in\{M,Ins,Del\}} \delta^k(i-1,j)\Big(\mathbb{1}(k=M)\gamma_D+\mathbb{1}(k=Del)\epsilon_D\Big).
\end{aligned}$$

3. **Termination.**

$$\begin{aligned}
L^\star &= (l(\bar{Y}_1), l(\bar{Y}_2)), \\
X_1^\star &= \arg\max_k \delta^k(l(\bar{Y}_1), l(\bar{Y}_2)).
\end{aligned} \tag{4.18}$$

4. **Backtracking.** Set $t=1$. While $L^\star \neq (0,0)$,

$$\begin{aligned}
X_{t+1}^\star &= \psi^{X_t^\star}(L^\star), \\
L^\star &= \begin{cases} L^\star-(1,1), & \text{if } X_t^\star = M \\ L^\star-(0,1), & \text{if } X_t^\star = Ins \\ L^\star-(1,0), & \text{if } X_t^\star = Del \end{cases}, \\
t &= t+1.
\end{aligned} \tag{4.19}$$

At the end of the loop, $t = T-1$. The sequence of hidden states can then be obtained by reversing the order of the elements of $X^\star$.

The notation for the emission probabilities B and the δ and ψ functions has been simplified as a result of not using the durations D_t.

4.3.3.3 Parameter estimation: Viterbi training algorithm

To compute the probability that an observed noisy read Y was generated from a true 16S gene sequence $\tilde{Y}$ using the Viterbi algorithm, we need to estimate the parameter $\lambda = (A, B)$. For this purpose, we rely on our training set $\widetilde{\mathcal{Y}}_0$ and maximize the conditional likelihood of the reads in $\widetilde{\mathcal{Y}}_0$ given their corresponding true sequences. Specifically, assuming that errors are introduced independently between reads, we need to compute

$$\arg\max_\lambda \ \Pr(Y_1,\ldots,Y_{n_0}|\tilde{Y}_1,\ldots,\tilde{Y}_{n_0};\lambda) = \arg\max_\lambda \prod_{i=1}^{n_0} \Pr(Y_i|\tilde{Y}_i;\lambda). \tag{4.20}$$

If the pairwise alignments between the reads and their corresponding true sequences were known, that is, if we knew a priori the sequence of matches/mismatches, insertions,

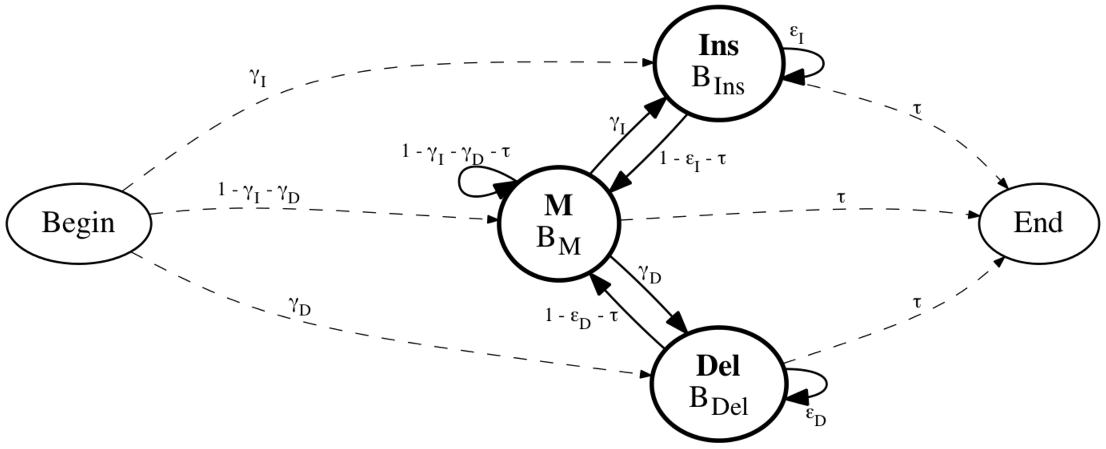


Figure 4.19: *Generalized pair hidden Markov model with Begin and End states added.* Hidden states and transition and emission probability distributions. Transitions from and to the *Begin* and *End* states are designated by dashed lines.

and deletions that occurred during the sequencing, we could estimate the transition and emission probabilities by counting the occurrences of each particular event in the training set $\widetilde{\mathcal{Y}}_0$. The maximum likelihood estimators of A and B would then be given by

$$\begin{aligned} \hat{A}_{ij} &= \frac{N^A_{ij}}{\sum_{j' \in \mathcal{S}} N^A_{ij'}}, \\ \hat{B}_{i,d}(\tilde{y}, y) &= \frac{N^B_{i,d}(\tilde{y}, y)}{\sum_{\tilde{y}', y'} N^B_{i,d}(\tilde{y}', y')}, \end{aligned} \tag{4.21}$$

with N^A_{ij} the number of transitions from hidden state i to state j and $N^B_{i,d}(\tilde{y}, y)$ the number of emissions of the aligned pair of nucleotides $(\tilde{y}, y) \in \{A, C, G, T, \emptyset\}^2$ from hidden state i with pair of durations d.

However, alignments are not known a priori and an iterative procedure must be used. There are two standard methods: the Baum-Welch and the Viterbi training algorithms. The Baum-Welch algorithm is a special case of the Expectation-Maximization (EM) algorithm. The most probable alignment needs to be found for each read in $\widetilde{\mathcal{Y}}_0$ using initial estimates for the transition and emissions probabilities. Then, new values for these probabilities are calculated. The procedure is iterated until some stopping criterion is met. The Baum-Welch algorithm is computationnaly expensive because the forward and backward probabilities need to be computed at each nucleotide in the sequence and for each read in $\widetilde{\mathcal{Y}}_0$ in order to find the most probable alignment. We use the Viterbi training algorithm instead. In this approach, the most probable alignment is computed for each read in our training set using the Viterbi algorithm, with initial values for the parameter λ. Then, Equation (4.21) is used to estimate the transition and emission probabilities.

Note that unlike the Baum-Welch algorithm, the Viterbi training algorithm does not maximize the conditional likelihood as in Equation (4.20). Instead, it seeks the value of λ that maximizes the contribution of the most probable alignments to the likelihood

$$\arg\max_{\lambda}\ \Pr(Y_1, \ldots, Y_{n_0}, Q_1^{\star}, \ldots, Q_{n_0}^{\star} | \tilde{Y}_1, \ldots, \tilde{Y}_{n_0}; \lambda), \tag{4.22}$$

where $Q_i^{\star}$ denotes the most probable alignment between a read Y_i and a true 16S gene sequence $\tilde{Y}_i$. Probably for this reason, the Viterbi training algorithm is known to perform less well in general than the Baum-Welch algorithm. However, as we use the Viterbi algorithm to find the most probable alignment $Q_i^{\star}$ instead of computing the likelihood over all possible alignments (Equation (4.11)), it is satisfactory for our purpose to estimate the parameter λ with the Viterbi training algorithm instead of the Baum-Welch algorithm [ref].

Incorporating quality values in transition probabilities Often, DNA-sequences obtained from high-throughput sequencing instruments come with base- and read-level quality values. These values are estimates of a base- or read-level error rate. In a PacBio CCS read, we make multiple observations of the same base, in both its forward and reverse-complement context. For each of these observations, an estimate of the base-error probability is available. A CCS-base error rate is computed by averaging the base-error estimates from each observation. The read-level error rate is an estimate of the reads accuracy when aligned to the true template that it was sequenced from. We use the CCS-base Phred quality values (QV) during estimation of transition probabilities where the Phred quality value is defined as $-10log_{10}($ CCS-base error rate $)$.

We let the GPHMM transition probabilities from state M to state Ins and Del, δ_I and δ_D, respectively, depend on the QV of each read in $\mathcal{Y}_0$. The procedure described below for δ_I can also be used for δ_D.

At each iteration of the standard Viterbi training algorithm, an estimator of δ_I is computed as

$$\hat{\delta}_I = \hat{A}_{M,Ins} = \frac{N^A_{M,Ins}}{\sum_{j\in\mathcal{S}} N^A_{M,j}},$$

where $N^A_{M,j}$ is the number of transitions from state M to state $j \in \mathcal{S}$ occurring in the entire training set $\tilde{\mathcal{Y}}_0$. As we want to model δ_I as a function of QV and each read Y_i has a different QV, QV_i, then an insertion probability $\delta_I(Y_i)$ now has to be estimated for each read separately.

We assume that the number of insertions $N^A_{M,Ins}(Y_i)$ in read Y_i has the binomial distribution

$$N^A_{M,Ins}(Y_i) \sim \text{Binomial}\left(\sum_{j\in\mathcal{S}} N^A_{M,j}(Y_i), \delta_I(Y_i)\right),$$

where the insertion probability $\delta_I(Y_i)$ is modeled using a generalized linear model (GLM) with logit link function [191] [192],

$$\operatorname{logit}\delta_I(Y_i) = \log\left(\frac{\delta_I(Y_i)}{1-\delta_I(Y_i)}\right) = \beta_0 + \beta_1 QV_i. \tag{4.23}$$

The maximum likelihood estimators of β_0 and β_1 can be obtained using the R function `glm` with `family = binomial()`, thus yielding an estimator $\hat{\delta}_I(Y_i)$ of the insertion probability for each read.

Consensus sequence To estimate the transition and emission probabilities in λ, we need to know the exact true sequence $\tilde{Y}$ that generated each read Y in $\tilde{\mathcal{Y}}_0$ in order to determine the number $N^A_{i,j}$ of transitions from state i to state j and the number $N^B_{i,d}(\tilde{y}, y)$ of emissions of the aligned pair of nucleotides $(\tilde{y}, y)$. However, we actually do not know the corresponding exact true sequence $\tilde{Y}$ of a read Y, but only the bacterium $Z \in \mathcal{B}$ it was generated from. Then, we need to consider two cases.

1. When bacterium Z has only one reference sequence r_j in the database $\mathcal{R}$, we assume that the true sequence $\tilde{Y}$ is the same as the reference sequence r_j, $\tilde{Y} = r_j$. Note that as r_j can belong to several bacteria, we have $Z \in b(r_j)$.

2. When bacterium Z has several reference sequences in $\mathcal{R}$, we need to compute a consensus sequence for the different 16S genes. In our database, sequences from the same bacterium are very similar. When at a given base all sequences have the same nucleotide, the consensus sequence is assigned that nucleotide. For locations at which nucleotides differ between sequences, an "N" is incorporated in the consensus sequence to indicate ambiguity. The function `consensusString` from the R package `msa` [193] is used to perform a multiple sequence alignment using ClustalW (with default parameters) and compute a consensus sequence for each bacterium in our training set $\tilde{\mathcal{Y}}_0$. Below is an example of a consensus sequence built from three different reference sequences $\{r_1, r_2, r_3\}$.

r_1	A	T	$-$	G	A	C
r_2	A	$-$	T	G	A	C
r_3	A	T	A	G	A	T
Consensus	A	N	N	G	A	N

4.3.3.4 Computation

Computation of a single probability that a read Y was generated from a true sequence $\tilde{Y}$ using the Viterbi algorithm involves two nested 'for' loops, with the number of iterations being the length of the read times the length of the true sequence, $l(Y)l(\tilde{Y}) \simeq 1,500^2 = 2.25 \times 10^6$. As expected, this computation is slow in R. To reduce computation time, the Viterbi and Viterbi training algorithms were coded using R package Rcpp. The computation was also parallelized with sixteen cores using the R function `mclappply` from the package parallel. Finally, to avoid underflow problems during the computation, the logarithm of the quantities in Equation (**??**) was used.

We implemented the Viterbi and Viterbi training algorithms in the R package gphmm available on CRAN. Up-to-date code is also available on github at `https://github.com/fperraudeau/gphmm`.

4.3.4 Estimation of Bacterial Abundances

4.3.4.1 Maximum likelihood estimator

As mentioned in Section 4.1.5.3, a natural estimator of the bacterial frequencies π is the maximum likelihood estimator (MLE), defined as

$$\hat{\pi}^{MLE} \equiv \arg\max_\pi \ell(\pi; \mathcal{Y}), \text{ s.t. } \sum_{k=1}^{K} \pi_k = 1,\ 0 \leq \pi_k \leq 1.$$

However, due to the log of sums in Equation (4.5), it is clear that no closed form exists for the MLE of π. Adapted numerical optimization techniques may be used, by noting that the log-likelihood function has the following general form:

$$\ell(\pi; \mathcal{Y}) = \sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} F_{ik}\pi_k\right) = \sum_{i=1}^{n} \log\left(F_{i\cdot}\pi\right), \tag{4.24}$$

where F is an $n \times K$ matrix with $F_{ik} \equiv \sum_{j=1}^{J} \Pr(Y_i = y_i | X_i = r_j)\rho_{kj}$ and ith row denoted by $F_{i\cdot}$.

Setting partial derivatives over π equal to zero, one has

$$0 = \frac{\partial\, \ell(\pi; \mathcal{Y})}{\partial \pi_k} = \sum_{i=1}^{n} \frac{F_{ik}}{F_{i\cdot}\pi} = \sum_{i=1}^{n} \frac{F_{ik}}{F_{i\cdot}\pi} \frac{\prod_{\substack{i'=1,\\ i' \neq i}}^{n} F_{i'\cdot}\pi}{\prod_{\substack{i'=1,\\ i' \neq i}}^{n} F_{i'\cdot}\pi},$$

so that

$$\sum_{i=1}^{n} F_{ik} \prod_{\substack{i'=1,\\ i' \neq i}}^{n} F_{i'\cdot}\pi = 0,\ \forall\, k \in \{1, \ldots, K\}. \tag{4.25}$$

Obtaining the MLE of π therefore involves solving a linear system of K equations in $K-1$ unknowns, summarized by $G\pi = 0$, where G is a $K \times K$ matrix with $G_{kk'} \equiv \sum_i \prod_{i' \neq i} F_{ik}F_{i'k'}$. As $det(G) \neq 0$, this system has a non-trivial solution. Numerical optimization techniques can then be used to estimate π. Here, we use the R function `solnp` from the package Rsolnp [194].

4.3.4.2 Penalized estimator

The MLE of π yields many false positives (i.e., $\hat{\pi}_k^{MLE} > 0$ when bacterium b_k is not present in the sample, $b_k \notin \mathcal{Z}$). To decrease the number of false positives, we consider estimators based on a penalized log-likelihood function

$$\hat{\pi}^{Pen} = \arg\max_\pi \left(\ell(\pi;\mathcal{Y}) - \lambda s(\pi)\right), \text{ s.t. } \sum_{k=1}^{K} \pi_k = 1,\ 0 \leq \pi_k \leq 1, \tag{4.26}$$

where s is a penalty function and λ a tuning parameter controlling the strength of the penalty term (i.e., the shrinking of the estimates towards zero). Note that when $\lambda = 0$, $\hat{\pi}^{Pen} = \hat{\pi}^{MLE}$.

We focus on two classes of penalty functions:

- Ridge penalty function: $s(\pi) = \sum_{k=1}^{K} \pi_k^2$. Ridge penalty is intuitive here as we want to decrease the number of false positives, so shrink estimates towards zero. While we know ridge penalized estimation introduces bias, there is a trade-off between bias and variance, so that the root mean square error (RMSE) may overall be reduced.

- Group LASSO [195] penalty function: $s(\pi) = \sum_{l=1}^{L} \sqrt{p_l}||\pi_l||_2$, where $||.||_2$ is the L_2 norm, π_l is the sum of the species frequencies for bacteria from genus l, and p_l is the number of reference sequences with genus l in our reduced reference database. This penalty should help manage over-representation of certain taxa in our database. For a given taxonomic rank, e.g., genus, certain taxa are vastly over-represented in comparison to other taxa, e.g., *Staphylococcus* is the most represented genus in our database (with $114,146$ reference sequences) and is well known to encompass several medically significant pathogens. As the difference in representation has entirely to do with human interests, e.g., studying pathogens, we would like to find a penalty that can perform equally well when assigning to a taxa of 10 versus $100,000$ members.

Note that the usual L_1 norm penalty function from the LASSO [196] is not useful here and simply returns the MLE, as, by definition, $\sum_k \pi_k = 1$ and $0 \leq \pi_k$.

The penalty parameter λ is selected by Monte-Carlo cross-validation (MCCV), where 90% of the reads are randomly sampled (without replacement) to form a training set and the remaining 10% form a validation set. This process is repeated independently $B = 100$ times, to yield training sets $\mathcal{Y}_b^{train}$ and validation sets $\mathcal{Y}_b^{valid}$, $b = 1, \ldots, B$. Note that since reads are partitioned independently for each CV fold, the same read can appear in multiple validation sets.

For each penalty parameter λ and each fold b, bacterial frequencies π are estimated on the training set $\mathcal{Y}_b^{train}$ as

$$\hat{\pi}_b^{Pen} = \arg\max_\pi \left(\ell(\pi; \mathcal{Y}_b^{train}) - \lambda s(\pi)\right), \text{ s.t. } \sum_{k=1}^{K} \pi_k = 1, \ 0 \leq \pi_k \leq 1 \tag{4.27}$$

and the log-likelihood evaluated on the validation set $\mathcal{Y}_b^{valid}$

$$\ell(\hat{\pi}_b^{Pen}; \mathcal{Y}_b^{valid}).$$

The cross-validated penalty parameter λ is then defined as the maximizer of the average validation set log-likelihood over the B different folds

$$\hat{\lambda} = \arg\max_\lambda \sum_{b=1}^{B} \ell(\hat{\pi}_b^{Pen}; \mathcal{Y}_b^{valid}). \tag{4.28}$$

4.3.4.3 Naive estimator

Additionally, using the same model as described in Section 4.1.5.3, we propose a naive estimator based on the heuristic that the frequency of bacterium b_k should be proportional to the average probability that the reads in dataset $\mathcal{Y}$ could have been sequenced from bacterium b_k, i.e.,

$$\begin{aligned}
\hat{\pi}_k^{Naive} &\propto \sum_{i=1}^{n} \Pr(Y_i = y_i | Z_i = b_k) \qquad (4.29)\\
&= \sum_{i=1}^{n} \sum_{j=1}^{J} \Pr(Y_i = y_i, X_i = r_j | Z_i = b_k)\\
&= \sum_{i=1}^{n} \sum_{j=1}^{J} \Pr(Y_i = y_i | X_i = r_j) \Pr(X_i = r_j | Z_i = b_k)\\
&= \sum_{j=1}^{J} \rho_{kj} \sum_{i=1}^{n} \Pr(Y_i = y_i | X_i = r_j).
\end{aligned}$$

The naive estimators $\hat{\pi}_k^{Naive}$ are then normalized to sum to one.

4.4 Results

4.4.1 Simulations

4.4.1.1 GPHMM read alignment probabilities

As explained in Section 4.3.2, to reduce the number of computations when calculating the log-likelihood $\ell(\pi; \mathcal{Y})$, we eliminate reference sequences in $\mathcal{R}$ with small probabilities of having generated the reads in a dataset $\mathcal{Y}$ using the alignment tool Bowtie2. More precisely, reference sequence r_j is eliminated if it has an alignment score lower than the selected minimum alignment score (min-score) for all the reads in $\mathcal{Y}$, i.e.,

$$\max_{i\in\{1,\ldots,n\}} \mathrm{AS}(Y_i, r_j) \leq \text{min-score}.$$

On average, for our 72 simulated datasets $\mathcal{Y}$, $1,180,537$ reference sequences were eliminated, leaving on average 687 sequences (standard deviation 680) in the reduced versions of the database $\mathcal{R}$.

Then, for each reference sequence r_j in a reduced database and each read y_i in a simulated dataset $\mathcal{Y}$, the probability $\Pr(Y = y_i|X = r_j)$ that read y_i could have been sequenced from reference sequence r_j is computed using the Viterbi algorithm for a generalized pair hidden Markov model (see Section 4.3.3).

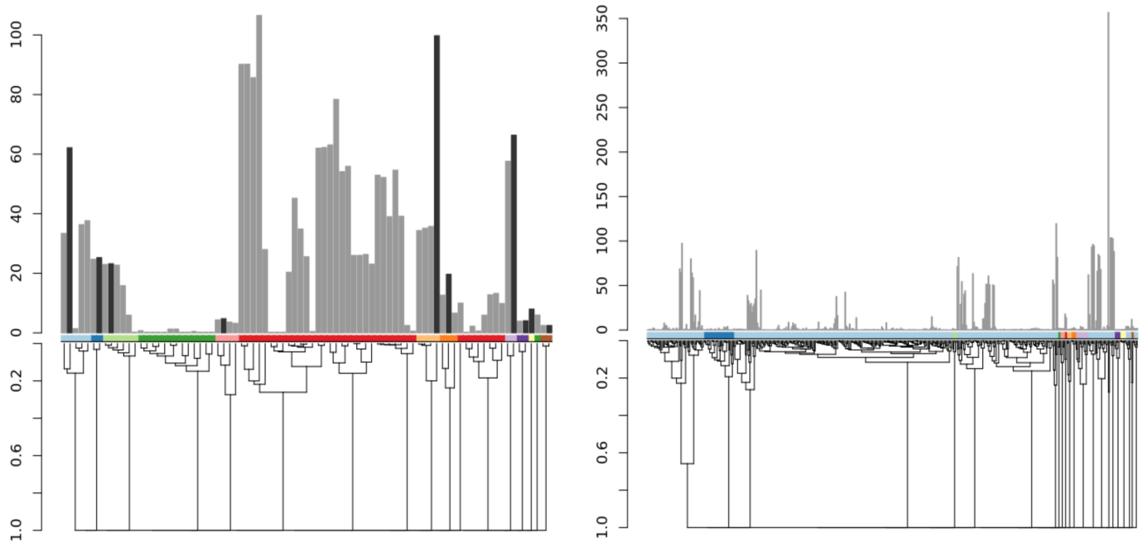
Figure 4.20 shows, for two simulated datasets $\mathcal{Y}$, the sum of these probabilities over the n reads for each reference sequence r_j, i.e.,

$$\sum_{i=1}^{n} \Pr(Y = y_i|X = r_j).$$

Note that if the reads were sequenced without error, i.e., $\Pr(Y = y_i|X = r_j) \in \{0, 1\}$, this quantity would simply be the number of reads aligned to reference sequence r_j. The two simulated microbial communities in Figure 4.20 are *Gut 2* and *Vaginal 1* with variability parameter $V = 100$ and $V = 1,000$ respectively, yielding similar Shannon diversity $-\sum_k \pi_k \log \pi_k$ (2.4 and 2.6, respectively) and number of distinct simulated species $\sum_{k=1}^{K} \mathbb{1}(\pi_k > 0)$ (24 and 27, respectively). Even if the two communities have similar Shannon diversity and number of distinct simulated species (see Figure 4.9), Figure 4.20 shows that it is harder to estimate the species frequencies for community *Vaginal 1* than for community *Gut 2*, as the reference sequences in the reduced version of the database are more similar (i.e., the phylogenetics is tighter) for community *Vaginal 1* than for community *Gut 2*. Probably for the same reason, the reference sequences that generated the reads (i.e., simulation frequency is greater than zero for these reference sequences) were included in the reduced version of the database for community *Gut 2* whereas none were included for community *Vaginal 1*. Similar results stand for all our simulated datasets (data not shown), it is harder to match the reference sequences that generated the reads for our simulated *Vaginal* communities than for our simulated *Gut* communities.

4.4.1.2 Comparison with SINTAX

The SImple Non-Bayesian TAXonomy (SINTAX) [169] algorithm predicts the taxonomic rank of the reads in a dataset $\mathcal{Y}$ by using k-mer similarity to identify the top hit in a reference



Panel (a): *Gut 2*. Panel (b): *Vaginal 1*.

Figure 4.20: *GPHMM read alignment probabilities for reference sequences.* Panel (a): Simulated microbial community *Gut 2* with Shannon diversity 2.4 and 24 distinct species. Panel (b): Simulated microbial community *Vaginal 1* with Shannon diversity 2.6 and 27 distinct species. The multinomial sampling distribution was used for both panels. The phylogenetic trees in the bottom panels represent pairwise similarity (based on the Levenshtein distance) between the reference sequences aligning to the reads in a simulated dataset $\mathcal{Y}$. The horizontal colored bars above the phylogenetic trees indicate the genus of each reference sequence, where the same color may correspond to different genera for panels (a) and (b). Each bar in the barplots corresponds to a reference sequence r_j from the phylogenetic tree, its color indicates whether the simulation frequency is greater than zero and its height represents the sum over the reads in $\mathcal{Y}$ of the probabilities that each read could have been sequenced from r_j, i.e., $\sum_{i=1}^{n} \Pr(Y = y_i|X = r_j)$. Note that if the reads were sequenced without error, i.e., $\Pr(Y = y_i|X = r_j) \in \{0, 1\}$, the height of the bar would simply be the number of reads aligned to that reference sequence. Note that the y-axes are on different scales for the two panels.

database. SINTAX does not require training and, unlike UTAX [197], can be used with full-length 16S reads without running out of memory. Additionally, the SINTAX algorithm provides bootstrap confidence measures for each read and all ranks in the prediction by repeatedly (default 100 iterations) sampling with replacement a set of 32 k-mers (default $k = 8$) from the overall set of k-mers of each read. At each iteration, the reference sequence with the greatest number of k-mers in common with the read is identified and its taxonomy is reported. Then, for each taxonomic rank, the name that occurs most often is identified and its frequency is reported as its bootstrap confidence measure. We compared our estimators to SINTAX results with default parameters and two different bootstrap cutoffs, 0.4 and 0.8. The default bootstrap cutoff is 0.8 and has not as good a performance as a bootstrap cutoff of 0.4.

4.4.1.3 Performance of bacterial frequency estimators

Let $\pi = (\pi_k : k = 1, \ldots, K)$ and $\hat{\pi} = (\hat{\pi}_k : k = 1, \ldots, K)$, where π_k denotes the true population frequency of bacterium b_k and $\hat{\pi}_k$ an estimator of this parameter, $k = 1, \ldots, K$. The performance of the estimator $\hat{\pi}$ can be assessed using the errors $\hat{\pi}_k - \pi_k$ (cf. bias) and the root mean squared error

$$\text{RMSE} = \sqrt{\frac{1}{K}\sum_{k=1}^{K}(\hat{\pi}_k - \pi_k)^2}.$$

We simulated our read datasets using a variability parameter V inversely related to the Shannon diversity, a metric commonly used to measure the richness and evenness of microbial communities. Indeed, as indicated in the top-right panel of Figure 4.9, when V is large, the Shannon diversity is low, whereas when V is small, the Shannon diversity is high. We compared the accuracy and robustness of our estimators as a function of the Shannon diversity instead of the variability parameter V as, unlike the artificial variability parameter, the Shannon diversity can be estimated for real datasets.

We also produced receiver operating characteristic (ROC) curves, where the true positive rate (TPR) is plotted against the false positive rate (FPR), with

$$TPR \equiv \frac{TP}{TP + FN},$$

$$FPR \equiv \frac{FP}{FP + TN},$$

where TP, FN, FP, and TN stand, respectively, for true positive, false negative, false positive, and true negative, as defined in Figure 4.23. Each point on an ROC curve corresponds to a different value for the detection threshold ϵ. The greater the area under curve (AUC), the better the estimator.

General results

Similar robustness for all estimators. To test the robustness of our method, bacterial communities were simulated using three different sampling distributions: the multinomial used in our data generation model and two other distributions, a Poisson distribution and a negative binomial distribution allowing over-dispersion. All our estimators showed similar robustness (see Supplementary Figure 4.27). Not surprisingly, microbial communities simulated from the multinomial and Poisson distributions showed more similar RMSE and sensitivity than the ones simulated from the negative binomial distribution.

Abundance estimation is harder for medium Shannon diversity. The Shannon diversity increases with both the richness (i.e., the number of species) and the evenness (i.e., similar relative abundances) of a microbial community. Figure 4.21 shows that bacterial abundance estimation is harder when the Shannon diversity is neither low nor high. When the Shannon diversity is high, the number of species is high, but the bacterial frequencies are similar. This makes the estimation easier, especially for the ridge and group LASSO estimators which give the same weight to, respectively, all bacteria and bacteria from the same genus. When the Shannon diversity is low, a few bacteria have much higher frequencies than other bacteria, but the number of bacteria is also smaller, which probably eases estimation. On the contrary, when a sample has intermediate Shannon diversity, the imbalance in the bacterial frequencies is not compensated by a small number of bacteria, making the estimation harder.

Comparison with SINTAX

Accuracy versus sensitivity. Except for our *Vaginal* simulated communities with medium Shannon diversity, SINTAX was superior with respect to RMSE as it had a lower or similar RMSE than our estimators (MLE, Ridge, and group LASSO) (see Figure 4.21). However, our estimators consistently showed higher sensitivity for a false positive rate greater than 0.05%, meaning that, while SINTAX was unable to detect some bacteria we knew were present in the sample (i.e., simulation frequency π_k was greater than zero for these bacteria), our estimators showed a smaller number of false negatives (i.e., $\hat{\pi_k} \geq \epsilon$ while $\pi_k = 0$), especially for the simulated gut samples. See ROC curves (Figure 4.22 and Supplementary Figure 4.28 and), and mean-difference plots (Supplementary Figures 4.29 to 4.30).

Flexibility. Additionally, our estimators allowed more flexibility than SINTAX estimators to trade false positives against false negatives. When the detection threshold ϵ is decreased, the number of true and false positives increases while the number of false negatives decreases. ROC curves (Figure 4.22 and Supplementary Figure 4.28) show that the trade-off is smoother for our estimators than for SINTAX estimators.

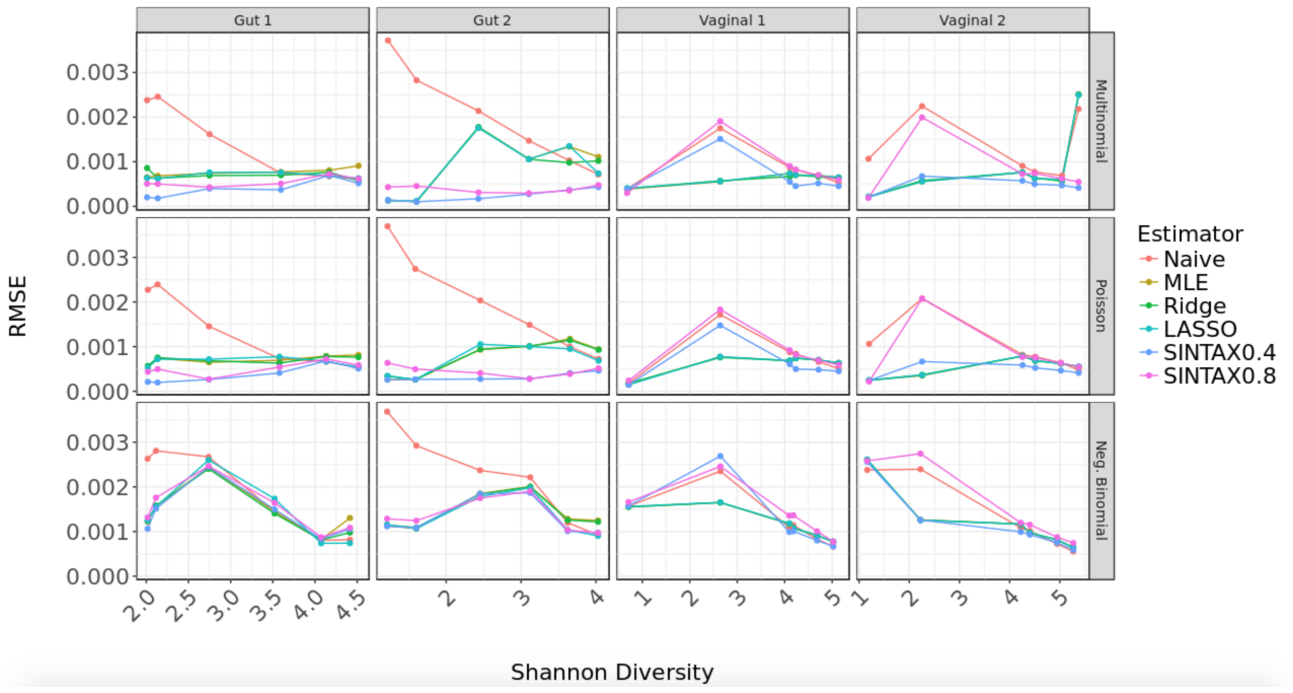


Figure 4.21: *RMSE vs. Shannon diversity.* Each panel displays root mean squared error (RMSE) as a function of Shannon diversity (see top-right panel of Figure 4.9) for a given community and sampling distribution. The colors correspond to our four different estimators, namely, Naive, MLE, ridge, and group LASSO, and to two estimators using SINTAX with bootstrap cutoffs 0.4 and 0.8. For low diversity (Shannon diversity smaller than 3), the MLE tends to have the smallest RMSE. For high diversity (Shannon diversity greater than 3), the MLE, ridge, and group LASSO estimators have similar RMSE. The naive estimator typically has the largest RMSE at all levels of diversity. SINTAX estimators have small RMSE for simulations from the gut communities.

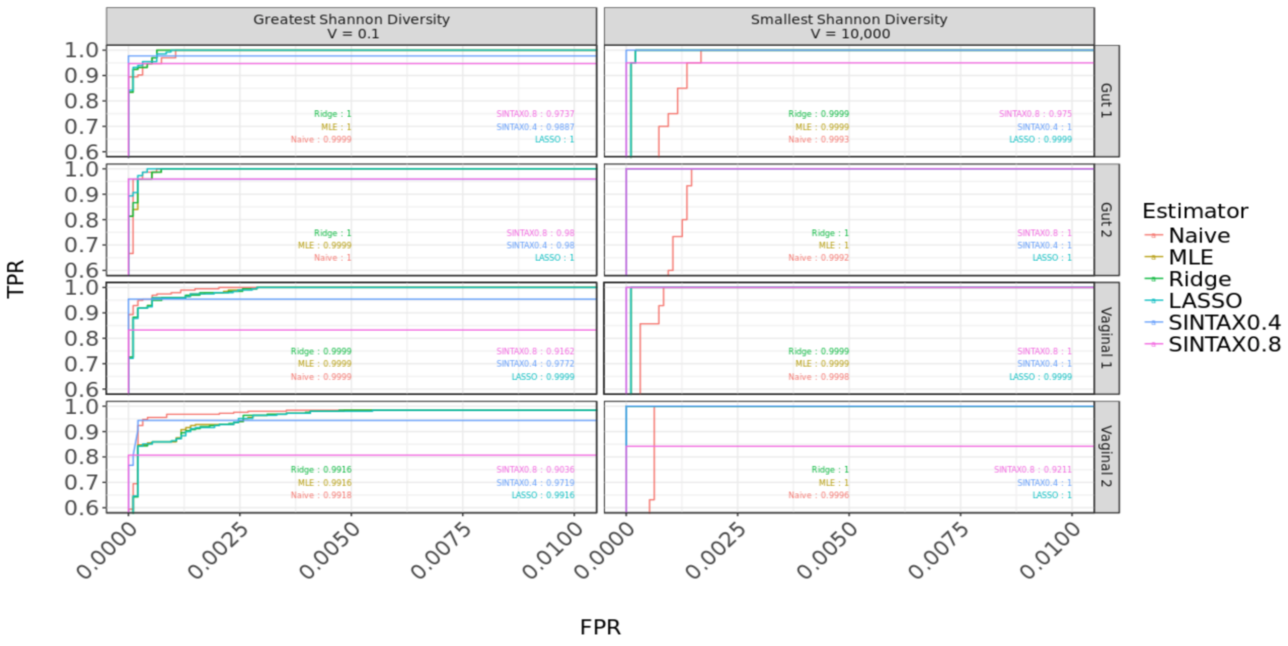


Figure 4.22: *Receiver operating characteristic (ROC) curve.* The true positive rate ($TPR = \frac{TP}{TP+FN}$) is plotted against the false positive rate ($FPR = \frac{FP}{FP+TN}$) at various levels of detection ϵ, where TP, FN, FP, and TN stand, respectively, for true positive, false negative, false positive, and true negative, as defined in Figure 4.23. Each panel corresponds to a different variability parameter V and community (*Gut 1*, *Gut 2*, *Vaginal 1*, and *Vaginal 2*) with Poisson sampling distribution. Colors correspond to different estimators.

		Truth		
		0	1	
Detected	0	True Negative $\hat{\pi}_k < \epsilon$ $\pi_k = 0$	False Negative $\hat{\pi}_k < \epsilon$ $\pi_k > 0$	Not Detected
	1	False Positive $\hat{\pi}_k \geq \epsilon$ $\pi_k = 0$	True Positive $\hat{\pi}_k \geq \epsilon$ $\pi_k > 0$	Detected
		Not Present	Present	

Figure 4.23: *True/false positives and negatives for the different estimators.* The table provides the definition of detected/not detected bacteria, present/not present bacteria ("Truth"), and resulting true/false positives and negatives.

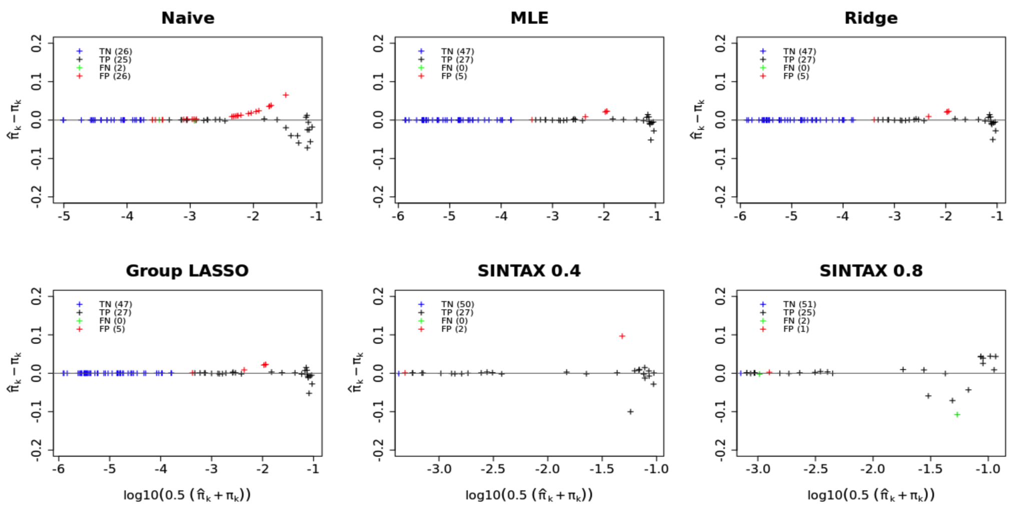


Figure 4.24: *Mean-difference plot of estimated vs. true bacterial frequencies for high Shannon diversity.* For simulated dataset *Vag1* with variability parameter $V = 1000$ and Poisson sampling distribution, scatterplots of the differences between estimated bacterial frequencies $\hat{\pi}_k$ and true bacterial frequencies π_k versus the logarithm of the mean of the two values, for the naive, MLE, ridge, group LASSO, and SINTAX estimators. Red points show false positives (i.e., $\hat{\pi}_k > \epsilon$ and $\pi_k = 0$), green points false negatives (i.e., $\hat{\pi}_k < \epsilon$ and $\pi_k > 0$), and black points true positives (i.e., $\hat{\pi}_k > \epsilon$ and $\pi_k > 0$) with the number of false positives (FP), false negatives (FN), and true positives (TP) in parenthesis on the upper-left corner (Figure 4.23). Note that the x-axes are on different scales.

4.4.2 Microbial Mock Community

4.4.2.1 GPHMM model evaluation

For all the sequencing reads from the mock community, Bowtie2 alignment scores were positively correlated with the length of the alignment whereas GPHMM probabilities were not (see Supplementary Figure 4.32). It suggests that GPHMM probabilities would be more appropriate for our application than Bowtie2 alignment scores as we want an alignment score or probability to depend on the accuracy of the alignment but not on the length of the reference sequence or the length of the alignment, for alignment lengths varying in a range of the length of 16S gene sequences ($1,300$ to $1,600$ bp).

For example, reads sequenced from strains *Helicobacter pylori* and *Lactobacillus gasseri* (part of our mock community) had similar read accuracy distributions although the length of their 16S genes (respectively $1,498$ bp and $1,585$ bp) is different (see Figure 4.25). When using Bowtie2 however, the distribution of the alignment scores for strain *Helicobacter pylori* (i.e., the strain with the shortest 16S gene) was shifted to the left compared to the distribution of the Bowtie2 alignment scores for strain *Lactobacillus gasseri* (i.e., the strain with the longest 16S gene). On the contrary, the distributions of the GPHMM probabilities were similar for the two strains showing that for this example length bias was corrected when GPHMM probabilities were used instead of Bowtie2 alignment scores. See Figure 4.25 and Supplementary Figure 4.31.

4.4.2.2 Classification performance of our estimators

We evaluated the performance of our estimators using precision-recall curves and a mock community where by definition the true positives (i.e., the strains present in the sample, that is $\pi_i > 0$ for the strains in the mock community) are known. At the genus level, RDP classifier had the greatest area under the curve (AUC) and precision at any recall level compared to our methods and SINTAX. See Figure 4.26 and Supplementary Figure 4.33. Additionally, all our methods had greater precision than SINTAX at any bootstrap cutoff for a recall between 0.96 and 1. Among SINTAX methods, SINTAX with bootstrap cutoffs greater than 0.8 had better performance than when SINTAX was used with a bootstrap cutoff of 0.5. At the species level, our methods and SINTAX with the bootstrap cutoff of 0.5 had high precision and recall. All of our methods had similar results with MLE having slightly smaller precision than naive, Ridge, and LASSO. As explicitly specified in the FAQ section of the RDP wiki page, RDP classifier was not designed to work at the species level. Thus, we did not compare RDP classifier to the other methods for the species level classification. Overall while at the genus level, RDP Classifier had better performance than our methods and SINTAX, it is not possible to use RDP Classifier to classify bacteria at the species level, and all our methods and SINTAX with a bootstrap cutoff of 0.5 had high precision and recall at the species level.

Note that to evaluate the classification performance of our estimators, we did not use the receiver operating characteristic (ROC) curves which depend on the number of true negatives (i.e., the strains not present in the mock community, that is $\pi_i = 0$ for these strains) which is unknown in this setting. The number of true negatives could be defined as the total number of bacteria present in the universe but absent from the mock community. As the total number of bacteria in the universe is unknown, the true negatives are often set to be the strains present in the database but absent in the mock community. This number is completely subjective to the database used, thus we preferred using the precision-recall curves, independent of the database.

While it was possible to evaluate the classification performances of our method using the mock community, it was impossible to evaluate our performances on quantification as the true bacteria frequencies in the mock community were unknown. The mock community was generated by mixing DNA extracts in equal frequencies (equimolar mixture) before PCR amplification therefore eliminating the DNA extraction bias, but not the PCR amplification bias. It has been studied for decades that PCR amplification efficiencies vary drastically from one bacterium to another introducing bias to estimate bacteria frequencies in microbial communities [198] [199] [200]. To evaluate the performances of our methods on quantification, we would need to create a dataset where true bacterial frequencies are known (see Discussion).

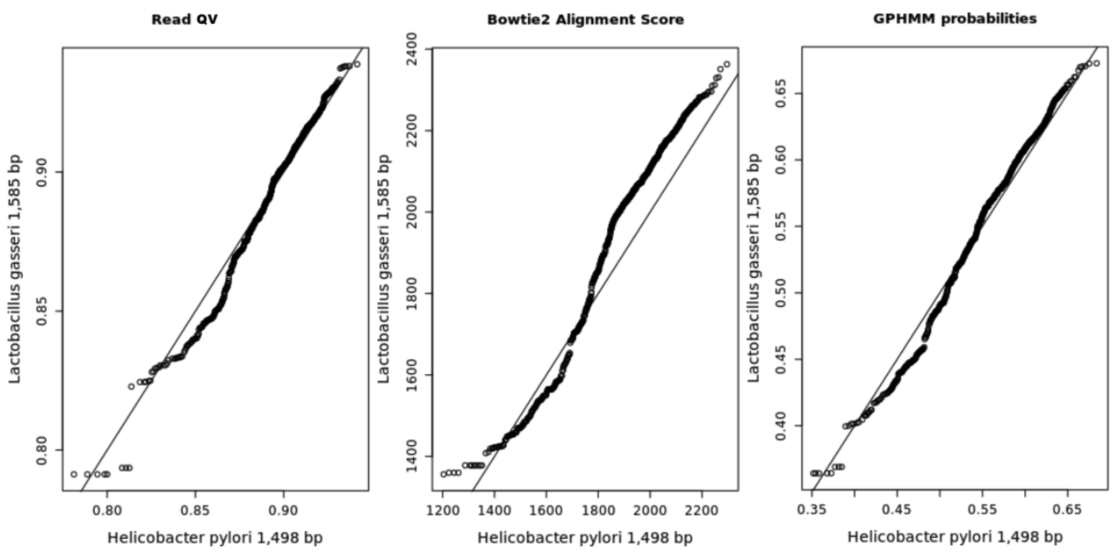


Figure 4.25: *QQplots of expected quality values for the sequencing reads, Bowtie2 alignment scores, GPHMM probabilities for reads sequenced from strains Helicobacter pylori (x-axis) and Lactobacillus gasseri (y-axis).* While reads quality values (QV) have similar distributions for strains *Lactobacillus gasseri* and *Helicobacter pylori*, Bowtie2 alignment scores are greater for strain *Lactobacillus gasseri* than for strain *Helicobacter pylori* introducing a length bias. On the contrary, the distribution of GPHMM probabilities are similar for both strains, showing that using GPHMM probabilities can eliminate the length bias. The length of the 16S genes for strains *Helicobacter pylori* and *Lactobacillus gasseri* are respectively 1, 498 and 1, 585 bp.

4.5 Discussion

4.5.1 Limitations to Accurate Quantification

The number of copies and variants of the 16S gene in bacteria genomes varies from one to fifteen [181], but the exact number is usually unknown or uncertain limiting accurate frequencies estimators in microbial samples. The rrnDB database provides estimates of the total number of loci at which a 16S gene could be found in the genome of each bacterium.

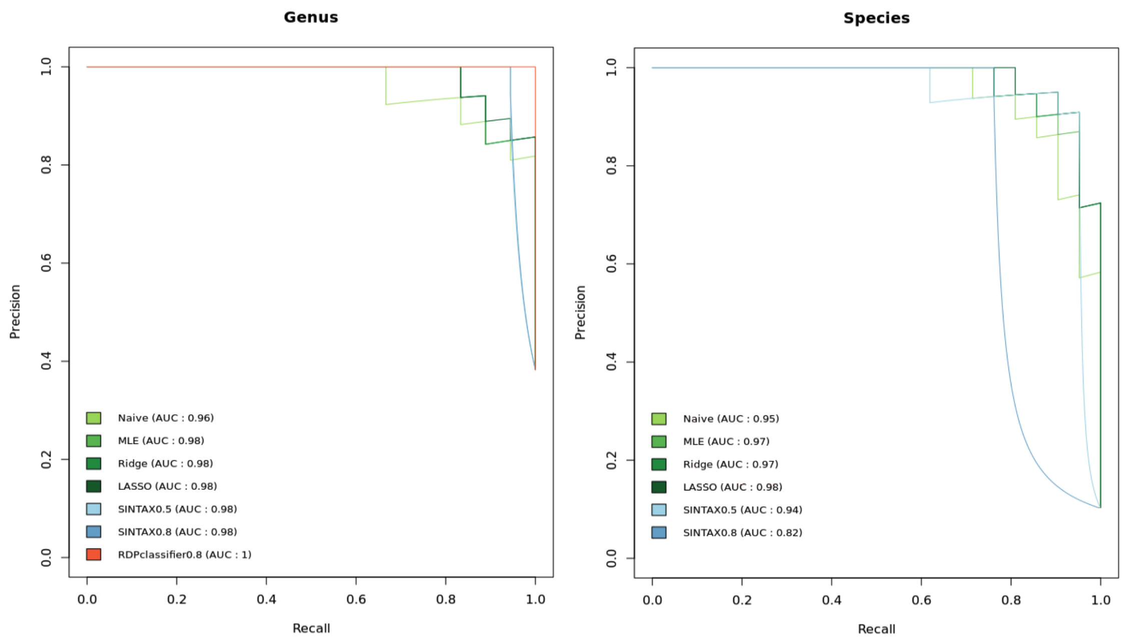


Figure 4.26: *Precision-Recall curve for SINTAX, RDP classifier, and our methods.* Left panel shows genus level classification and right panel species level classification. SINTAX is used with cutoff bootstraps of 0.5 and 0.8, and RDP classifier with a cutoff bootstrap of 0.8.

See section 4.1.5.2. Although the rrnDB database is a valuable resource, estimates often have a high standard deviation (total standard deviation for the rrnDB database is 2.9) and are not available at the species level. Several single copy housekeeping genes, e.g., those coding for RNA polymerase, concatenated ribosomal proteins, amino-acyl synthetases, or the 60 kDa chaperonin have been proposed as potential phylogenetic markers [201] [202], theoretically avoiding the problems with multiple and variable 16S rRNA copies within bacterial genomes. Protein-coding single copy genes were also successfully used for the assignment of metagenomic sequences, with a better taxonomic resolution than 16S rRNA genes [203]. However, the use of these genes for the analysis of microbial amplicons is limited because the degeneracy of protein sequences makes the design of universal primers difficult. Additionally, the number of 16S rRNA gene sequences in the sequence databases greatly exceeds those of other bacterial genes, which still makes its use preferable by increasing the probability of finding a close hit for taxonomic identification.

Another limitation to accurate quantification is the fact that DNA extraction efficiency varies from one bacterium to another introducing bias in the quantification estimators, because final sequence read counts may not accurately represent the relative abundance of original DNA fragments. For example, gram-positive bacteria wall have greater strength and rigidity, making it harder to extract the DNA from these bacteria [204]. A solution to estimate DNA extraction efficiencies is to perform an experiment where a known number of cells are lysed and the amount of extracted DNA is measured, using for example fluorometric quantitation. This experiment can be done for a few strains at a time (e.g. pathogens of interest), but it is infeasible to estimate the DNA extraction efficiencies for all the strains in a database. The field of microbiology would benefit from databases where the DNA extraction efficiency is recorded for each strain.

As DNA extraction, PCR amplification efficiencies varies from one bacterium to another affecting quantification accuracy. While this problem has been studied for decades [198] [199] [200], researchers have recently made progress by incorporating random molecular barcodes into PCR primer design. The concept of molecular barcodes is that each original DNA fragment, within the same sample, is attached to a unique sequence barcode which is designed as a string of totally random nucleotides. As a result, sequence reads that have different molecular barcodes represent different original DNA fragments, while reads that have the same barcodes are the result of PCR amplification from the same original DNA fragment. Thus, random molecular barcodes allow to estimate PCR amplification bias by aggregating reads having the same molecular barcode [205] [206].

4.5.2 Limitations Imposed by the Database

In the manuscript, we made the assumption that all bateria in the sample are represented in the annotated reference database. See section 4.1.5.2. It is obviously a (too) stringent assumption. In [169], Robert C. Edgar accounts for the bacteria not present in the database and estimates two types of false positive error: misclassifications, where a false positive means that a read was assigned to a wrong strain while the true strain is in the database, and over-classifications, where a false positive means that a read was assigned to a strain while the true strain is not in the database. We envision a future report where we would evaluate the over-classification error of our method on different databases.

Another limitation imposed by the use of a database is that bacteria represented in the database are sometimes incorrectly annotated. An accurate method to annotate novel strains would be to use DNA-DNA hybridization (see section 4.1.1.2) but as this method is time-consuming, labor-intensive, and expensive to perform, fewer and fewer laboratories worldwide perform such assays. Instead, as sequencing cost drastically decreased over the last decade, many studies describing new species are solely based upon their 16S gene sequences. For example, many annotations in the full RDP database are based on the RDP classifier which has a high rate of over-classification errors on full-length sequences (see section 4.2.2) resulting in incorrect annotations or annotations at low taxonomic levels (e.g., only about 2.6% of the reference sequences are annotated at the species level in the full RDP database). Additionally, some of the reference sequences are probably chimeras (i.e. artifacts made during the PCR process) [207] spuriously increasing the taxonomic diversity of the reference database. To accurately identify bacteria at the species level, it is essential to meticulously choose a complete reference database with correct annotations.

4.5.3 Shotgun Metagenomics

In the past few years, shotgun metagenomics has gained a growing interest in the microbiology field. In shotgun metagenomics, DNA is extracted not only from the 16S gene(s) but from the whole genomes for all the cells in a microbial sample, resulting in a lot of advantages compared to 16S studies. First, there is no need to use specific primers as the whole genomes are sequenced, therefore PCR amplification bias is eliminated. Second, amplicon sequencing typically only provides insight into the taxonomic composition of the microbial community and offers a limited functional resolution [208]. Some methods (e.g., PICRUSt [209]) exists to directly predict the functional profiles of microbial communities using 16S rRNA gene sequences, but is highly dependent on the quality and completeness of the reference database. Third, amplicon sequencing is limited to the analysis of taxa for which taxonomically informative genetic markers are known and can be amplified. Novel or highly diverged bacteria are difficult to study using this approach.

Although shotgun metagenomics indisputably offers a greater potential for identification of strains at the strain level and to perform functional analysis, it presents some challenges [210]. To start with, whole genomes are sequenced instead of just one gene, thus a lot of genomic information need to be sampled from a community. It results in the need of high sequencing depth and an increasing cost to store and analyze the data. Then, while contamination is a challenge general to environmental sequencing studies, the identification and removal of metagenomic sequence contaminants is especially problematic as many of the strains in the microbial samples are unknown [211] [212]. Finally, even if tools have been developed to assemble individual reads into scaffolds or genomes (e.g., Megahit [213], Celera Assembler [214], Omega [215], Spades [216]), classify reads to known genomes (e.g., Kraken [217], Kallisto [218]), bin reads or scaffolds to genome bins (e.g., MaxBin [219], MetaBAT [220]), or even with some additional curation reconstruct closed genomes from metagenomes (e.g. methods described in [221] [222]), these tools have some limitations. Specific exemplary problems include the presence of genes with low evolutionary divergence between organisms or repetitive genomic regions that are larger than a sequencing read [223]. In this context, new statistical and computational tools need to be developed to fully exploit the potential of shotgun metagenomics.

4.6 Supplement

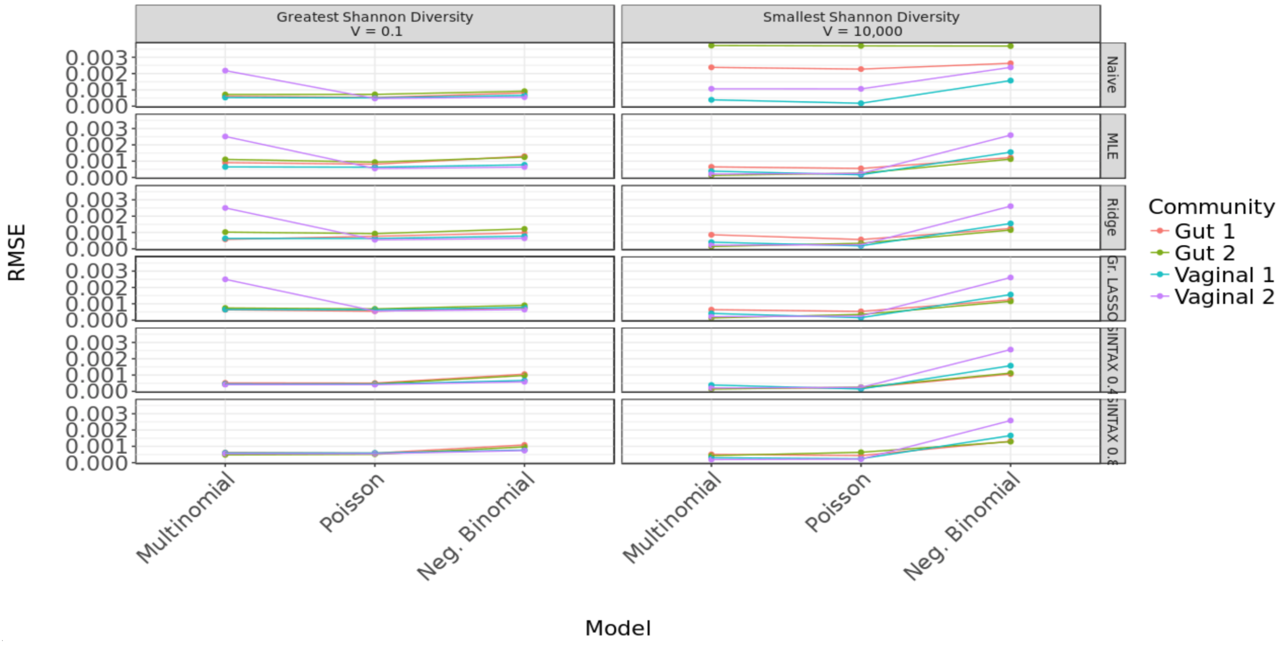


Figure 4.27: *Robustness of estimators.* Each panel displays RMSE for each of the three sampling distributions, for a given Shannon diversity and estimator. RMSE is higher for low Shannon diversity (i.e., $V = 10,000$) than for high Shannon diversity (i.e., $V = 0.1$). The naive estimator is the most robust, as RMSE is similar for the three distributions. The MLE, ridge, group LASSO, and SINTAX estimators are less robust, especially for low Shannon diversity.

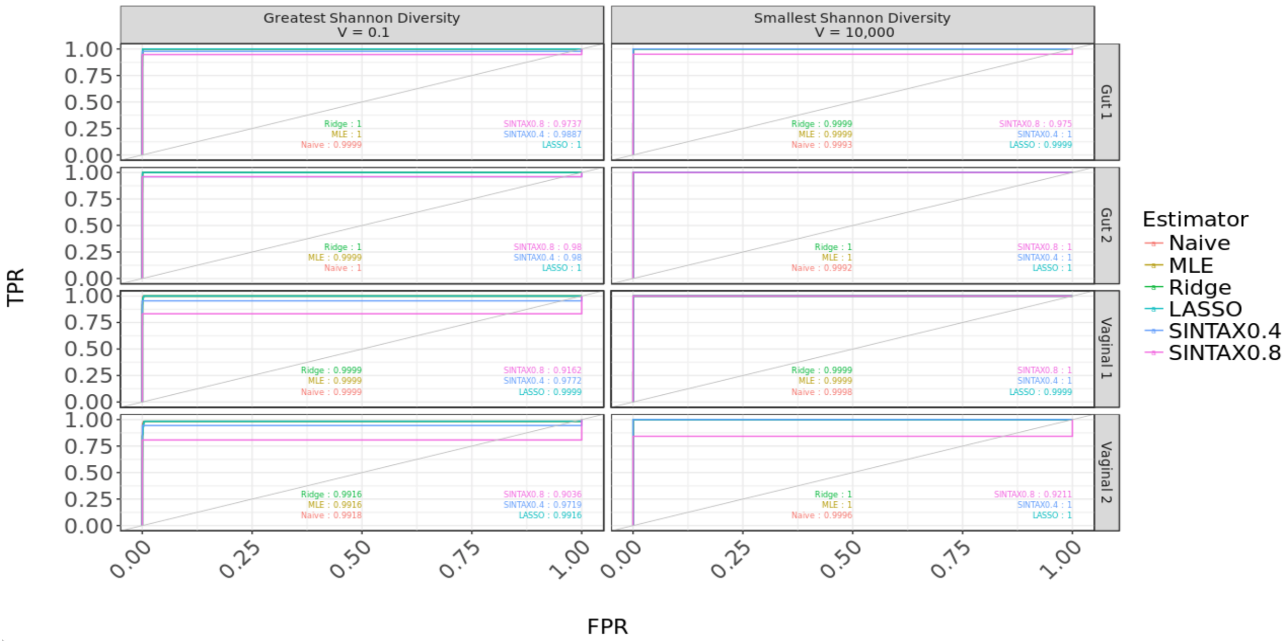


Figure 4.28: *Receiver operating characteristic (ROC) curve.* The true positive rate ($TPR = \frac{TP}{TP+FN}$) is plotted against the false positive rate ($FPR = \frac{FP}{FP+TN}$) at various levels of detection ϵ, where TP, FN, FP, and TN stand, respectively, for true positive, false negative, false positive, and true negative, as defined in Figure 4.23. Each panel corresponds to a different variability parameter V and community (*Gut 1*, *Gut 2*, *Vaginal 1*, and *Vaginal 2*) with Poisson sampling distribution. Colors correspond to different estimators.

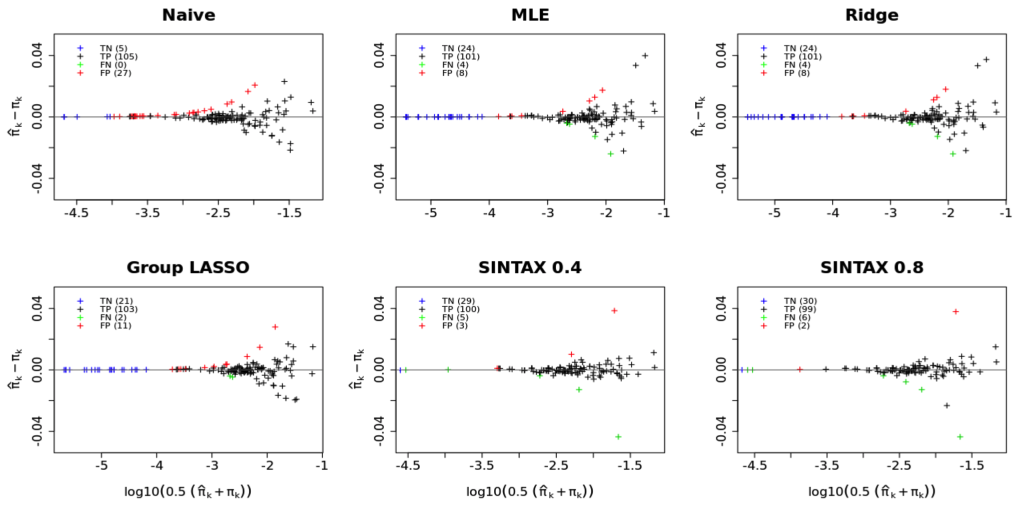


Figure 4.29: *Mean-difference plot of estimated vs. true bacterial frequencies for high Shannon diversity.* For simulated dataset *Gut1* with variability parameter $V = 10$ and Poisson sampling distribution, scatterplots of the differences between estimated bacterial frequencies $\hat{\pi}_k$ and true bacterial frequencies π_k versus the logarithm of the mean of the two values, for the naive, MLE, ridge, group LASSO, and SINTAX estimators. Red points show false positives (i.e., $\hat{\pi}_k > \epsilon$ and $\pi_k = 0$), green points false negatives (i.e., $\hat{\pi}_k < \epsilon$ and $\pi_k > 0$), and black points true positives (i.e., $\hat{\pi}_k > \epsilon$ and $\pi_k > 0$) with the number of false positives (FP), false negatives (FN), and true positives (TP) in parenthesis on the upper-left corner (Figure 4.23). Note that the x-axes are on different scales.

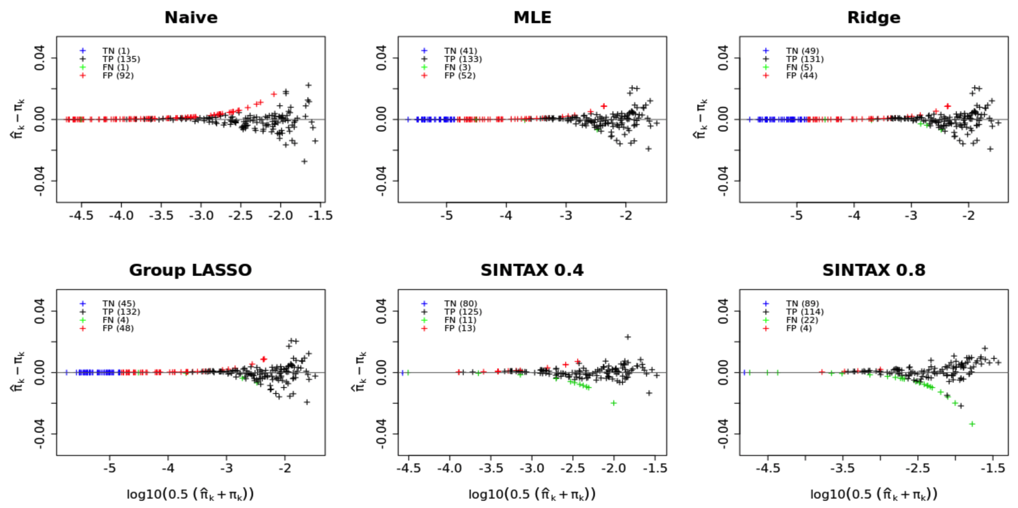
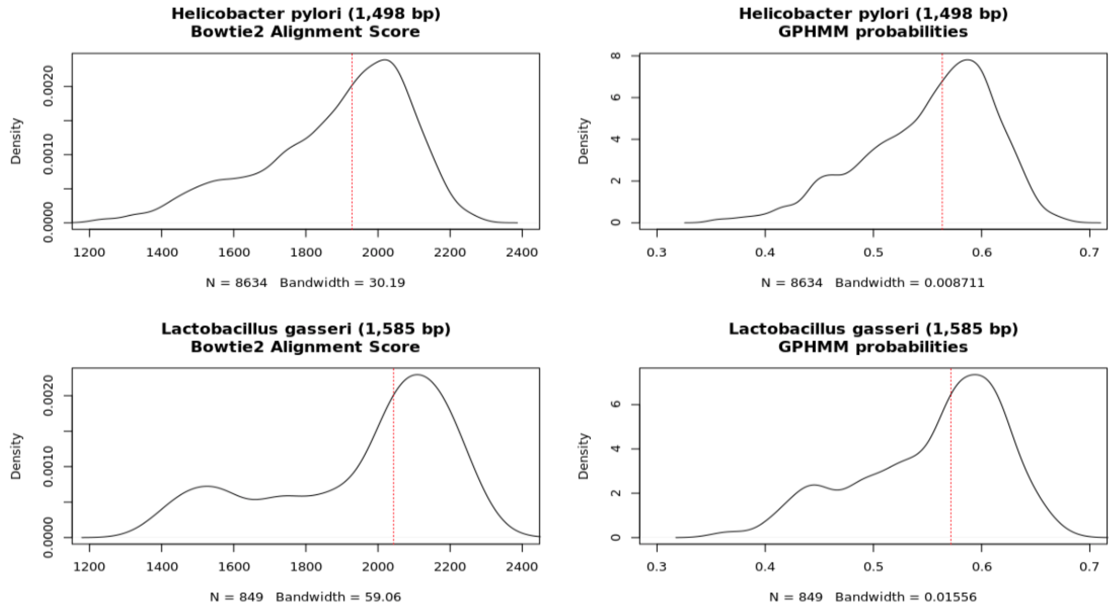
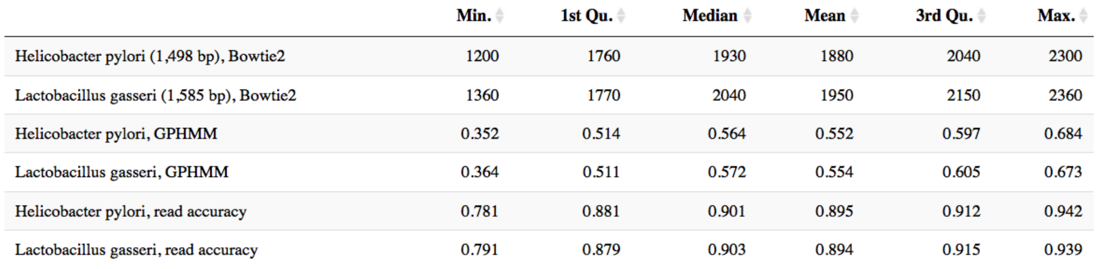


Figure 4.30: *Mean-difference plot of estimated vs. true bacterial frequencies for high Shannon diversity.* For simulated dataset *Vag2* with variability parameter $V = 10$ and Poisson sampling distribution, scatterplots of the differences between estimated bacterial frequencies $\hat{\pi}_k$ and true bacterial frequencies π_k versus the logarithm of the mean of the two values, for the naive, MLE, ridge, group LASSO, and SINTAX estimators. Red points show false positives (i.e., $\hat{\pi}_k > \epsilon$ and $\pi_k = 0$), green points false negatives (i.e., $\hat{\pi}_k < \epsilon$ and $\pi_k > 0$), and black points true positives (i.e., $\hat{\pi}_k > \epsilon$ and $\pi_k > 0$) with the number of false positives (FP), false negatives (FN), and true positives (TP) in parenthesis on the upper-left corner (Figure 4.23). Note that the x-axes are on different scales.





	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Helicobacter pylori (1,498 bp), Bowtie2	1200	1760	1930	1880	2040	2300
Lactobacillus gasseri (1,585 bp), Bowtie2	1360	1770	2040	1950	2150	2360
Helicobacter pylori, GPHMM	0.352	0.514	0.564	0.552	0.597	0.684
Lactobacillus gasseri, GPHMM	0.364	0.511	0.572	0.554	0.605	0.673
Helicobacter pylori, read accuracy	0.781	0.881	0.901	0.895	0.912	0.942
Lactobacillus gasseri, read accuracy	0.791	0.879	0.903	0.894	0.915	0.939

Figure 4.31: *Comparison of Bowtie2 alignment scores and GPHMM probabilities for Helicobacter pylori and Lactobacillus gasseri. Four upper density plots show the distributions of from left to right and top to bottom, Bowtie2 alignment scores for Helicobacter pylori, GPHMM probabilities for Helicobacter pylori, Bowtie2 alignment scores for Lactobacillus gasseri, and GPHMM probabilities for Lactobacillus gasseri. The table summarizes the statistical properties of the four distributions in the upper panel.*

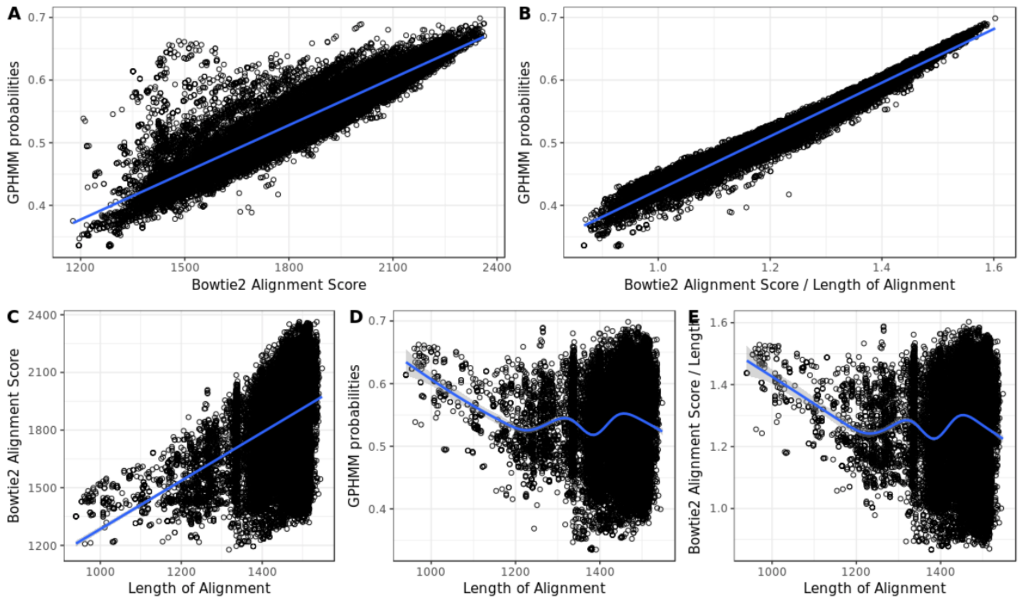


Figure 4.32: *Relationship between Bowtie2 alignment score, GPHMM probability and length of the alignment for all the reads sequenced from the mock community dataset. (A) Bowtie2 alignment scores against GPHMM probabilities, (B) Bowtie2 alignment scores divided by the length of the alignment against GPHMM probabilities, (c) Bowtie2 alignment scores against length of alignment, (D) GPHMM probabilities against length of alignment, and (E) Bowtie2 alignment scores divided by length of alignment against length of alignment.*

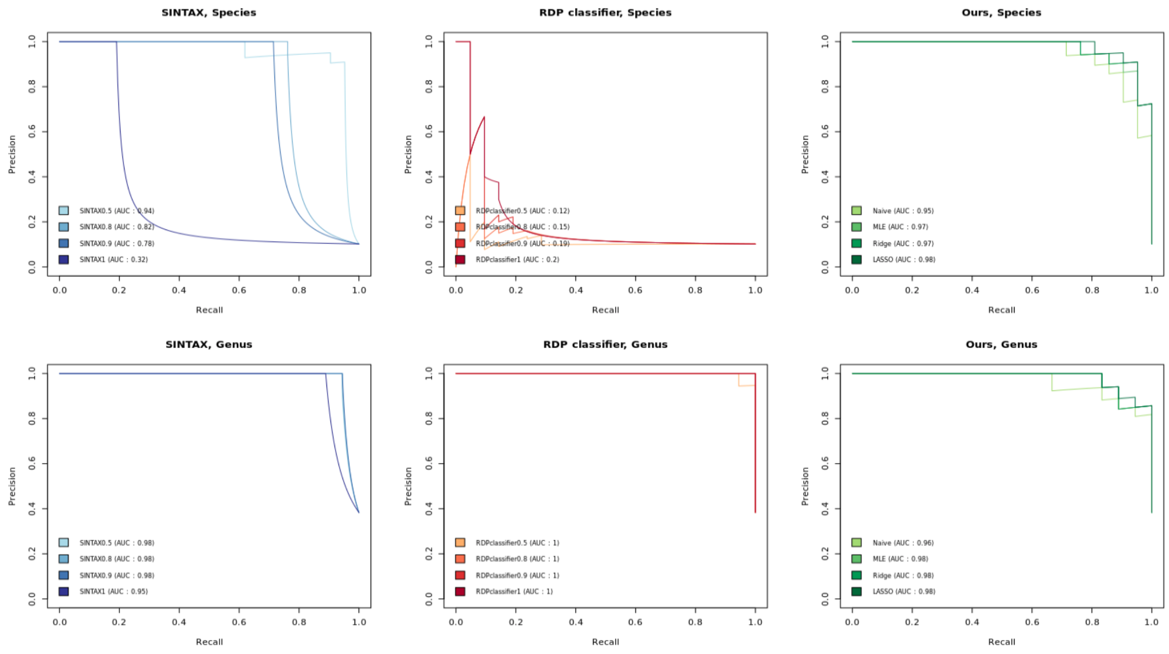


Figure 4.33: *Precision-Recall curve for SINTAX, RDP classifier, and our methods. SINTAX and RDP classifier are shown for bootstrap cutoff of* 0.5, 0.8, 0.9, *and* 1. *Plots in the first row show species level classification and plots in the second raw genus level classification.*

Chapter 5

Conclusion

This thesis provides novel statistical and computational methods for the analysis of single-cell transcriptome sequencing (scRNA-seq) and metagenomics.

In Chapter 2, I presented a general and flexible zero-inflated negative binomial-based wanted variation extraction (ZINB-WaVE) method, which extract low-dimensional signal from scRNA-seq read counts, accounting for zero inflation (dropouts), over-dispersion, and the discrete nature of the data. In that work, my collaborators and I showed with simulated and real data, that the model and its associated estimation procedure are able to give a more stable and accurate low-dimensional representation of the data than principal component analysis (PCA) and zero-inflated factor analysis (ZIFA) without the need for a preliminary (usually tedious) normalization step.

In Chapter 3, an application of the ZINB-WaVE model was presented that identifies excess zero counts and generates gene and cell-specific weights to unlock bulk RNA-seq differential expression (DE) pipelines for zero-inflated data. The method is shown to outperform competing methods on simulated bulk and single-cell RNA-seq datasets. We also demonstrated a gain in DE performance in two publicly available real datasets. The user-friendly open source implementation of ZINB-WaVE on Bioconductor will ease the technical aspects of scRNA-seq data analysis and help with the discovery of novel applications of the model.

Finally, in Chapter 4, I proposed a method to estimate bacterial abundances in human metagenomes using full-length 16S sequencing reads. I showed that although long sequencing reads still suffer from a high sequencing error rate, it is possible to model the sequencing error profile of a sequencing machine and take advantage of sequencing reads that span the entire length of the 16S gene to provide a better resolution that shorter reads usually used in the field. The publicly available CRAN package gphmm developed for this project will contribute and help other researchers to improve our understanding of microbial communities that surround us and impact our health.

In the future, I envision adding new features and continuing the development of the publicly available software packages zinbwave and gphmm, and create new user-friendly tools to analyze genomic data. I strongly believe that one day genomics could form a key part of our everyday health-care allowing us to optimize our health on a whole different level, from improving our diet to diagnosing diseases at a very early stage. In that perspective, I hope to continue working on state-of-the-art genomic technologies and get involved in the ethical, legal, and social questions raised by genomic research and medicine.

Bibliography

[1] *Genomics: definition.* URL: https://en.wikipedia.org/wiki/Genomics.

[2] *Hippocrate citation.* URL: https://www.brainyquote.com/authors/hippocrates.

[3] Nikhil Chaudhary et al. "16S Classifier: A Tool for Fast and Accurate Taxonomic Classification of 16S rRNA Hypervariable Regions in Metagenomic Datasets". In: *PLOS ONE* 10.2 (Feb. 2015). Ed. by Andrew R. Dalby, e0116106. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0116106. URL: http://dx.plos.org/10.1371/journal.pone.0116106.

[4] Ian S. Hagemann. "Overview of Technical Aspects and Chemistries of Next-Generation Sequencing". In: *Clinical Genomics*. Elsevier, 2015, pp. 3–19. DOI: 10.1016/B978-0-12-404748-8.00001-0. URL: http://linkinghub.elsevier.com/retrieve/pii/B9780124047488000010.

[5] Laura M. Zahn. "A Fantastic Voyage in Genomics". In: *Science* 358.6359 (Oct. 2017), pp. 56–57. ISSN: 0036-8075. DOI: 10.1126/science.358.6359.56. URL: http://www.sciencemag.org/lookup/doi/10.1126/science.358.6359.56.

[6] Aviv Regev et al. "Science Forum: The Human Cell Atlas." In: *eLife* 6 (Dec. 2017). ISSN: 2050-084X. DOI: 10.7554/eLife.27041. URL: http://www.ncbi.nlm.nih.gov/pubmed/29206104.

[7] Orit Rozenblatt-Rosen et al. "The Human Cell Atlas: from vision to reality." In: *Nature* 550.7677 (Oct. 2017), pp. 451–453. ISSN: 1476-4687. DOI: 10.1038/550451a. URL: http://www.ncbi.nlm.nih.gov/pubmed/29072289.

[8] Ron Sender, Shai Fuchs, and Ron Milo. "Revised Estimates for the Number of Human and Bacteria Cells in the Body". In: *PLOS Biology* 14.8 (Aug. 2016), e1002533. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.1002533. URL: http://dx.plos.org/10.1371/journal.pbio.1002533.

[9] Aleksandra A Kolodziejczyk et al. "The technology and biology of single-cell RNA sequencing". In: *Molecular Cell* 58.4 (2015), pp. 610–620.

[10] E Z Macosko et al. "Highly Parallel Genome-wide Expression Profiling of Individual Cells Using Nanoliter Droplets". In: *Cell* 161.5 (2015), pp. 1202–1214. ISSN: 1097-4172. DOI: 10.1016/j.cell.2015.05.002. URL: http://www.ncbi.nlm.nih.gov/pubmed/26000488.

[11] Bosiljka Tasic et al. “Adult mouse cortical cell taxonomy revealed by single cell transcriptomics”. In: *Nature Neuroscience* 19 (2016), pp. 335–346. ISSN: 1097-6256. DOI: `10.1038/nn.4216`. URL: `http://dx.doi.org/10.1038/nn.4216`.

[12] A. Zeisel et al. “Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq”. In: *Science* 347.6226 (Mar. 2015), pp. 1138–1142. ISSN: 0036-8075. DOI: `10.1126/science.aaa1934`. URL: `http://www.sciencemag.org/cgi/doi/10.1126/science.aaa1934`.

[13] Qiaolin Deng et al. “Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells.” In: *Science* 343.6167 (2014), pp. 193–6. ISSN: 1095-9203. DOI: `10.1126/science.1245316`.

[14] Anoop P. Patel et al. “Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma”. In: *Science* 344.6190 (2014), pp. 1396–1401. ISSN: 0036-8075, 1095-9203. DOI: `10.1126/science.1254257`.

[15] Rhonda Bacher and Christina Kendziorski. “Design and computational analysis of single-cell RNA-sequencing experiments”. In: *Genome Biology* 17.63 (2016).

[16] Peter V Kharchenko, Lev Silberstein, and David T Scadden. “Bayesian approach to single-cell differential expression analysis.” In: *Nature Methods* 11.7 (2014), pp. 740–2. ISSN: 1548-7105. DOI: `10.1038/nmeth.2967`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/24836921`.

[17] Saiful Islam et al. “Quantitative single-cell RNA-seq with unique molecular identifiers”. In: *Nature Methods* 11.1 (2014), pp. 163–166. ISSN: 1548-7105. DOI: `10.1038/nmeth.2772`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/24363023`.

[18] Po-Yuan Tung et al. “Batch effects and the effective design of single-cell gene expression studies”. In: *Scientific Reports* 7 (2017), p. 39921. URL: `http://dx.doi.org/10.1038/srep39921%20http://10.0.4.14/srep39921%20http://www.nature.com/articles/srep39921#supplementary-information`.

[19] Catalina A. Vallejos et al. “Normalizing single-cell RNA sequencing data: challenges and opportunities”. In: *Nature Methods* 14 (2017), pp. 565–571.

[20] Georgi K Marinov et al. “From single-cell to cell-pool transcriptomes: stochasticity in gene expression and RNA splicing.” In: *Genome research* 24.3 (2014), pp. 496–510. ISSN: 1549-5469. DOI: `10.1101/gr.161034.113`. URL: `http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3941114&tool=pmcentrez&rendertype=abstract`.

[21] Alex A Pollen et al. “Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex”. In: *Nature Biotechnology* 32.10 (2014), pp. 1053–8. ISSN: 1087-0156. DOI: `10.1038/nbt.2967`. URL: `http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=4191988&tool=pmcentrez&rendertype=abstract`.

[22] Florian Buettner et al. “Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells”. In: *Nature Biotechnology* 33.2 (2015), pp. 155–160.

[23] Cole Trapnell et al. “The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells.” In: *Nature Biotechnology* 32.4 (2014), pp. 381–6. ISSN: 1546-1696. DOI: 10.1038/nbt.2859. URL: http://dx.doi.org/10.1038/nbt.2859%5Cnhttp://www.nature.com/doifinder/10.1038/nbt.2859.

[24] Zhicheng Ji and Hongkai Ji. “TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis”. In: *Nucleic Acids Research* 44.13 (2016), e117. ISSN: 13624962. DOI: 10.1093/nar/gkw430.

[25] Jaehoon Shin et al. “Single-Cell RNA-Seq with Waterfall Reveals Molecular Cascades underlying Adult Neurogenesis”. In: *Cell Stem Cell* 17.3 (2015), pp. 360–372. ISSN: 18759777. DOI: 10.1016/j.stem.2015.07.013.

[26] Kieran Campbell, Chris P Ponting, and Caleb Webber. “Laplacian eigenmaps and principal curves for high resolution pseudotemporal ordering of single-cell RNA-seq profiles”. In: *bioRxiv* (2015), p. 027219. DOI: 10.1101/027219. URL: http://biorxiv.org/content/early/2015/09/18/027219.abstract.

[27] Rahul Satija et al. “Spatial reconstruction of single-cell gene expression data”. In: *Nature Biotechnology* 33.5 (2015), pp. 495–502. ISSN: 1087-0156. DOI: 10.1038/nbt.3192. URL: http://dx.doi.org/10.1038/nbt.3192.

[28] Alex K Shalek et al. “Single-cell RNA-seq reveals dynamic paracrine control of cellular variation.” In: *Nature* 510.7505 (2014), pp. 363–9. ISSN: 1476-4687. DOI: 10.1038/nature13437. URL: http://www.nature.com.docelec.univ-lyon1.fr/nature/journal/v510/n7505/full/nature13437.html#f1.

[29] Jellert T. Gaublomme et al. “Single-Cell Genomics Unveils Critical Regulators of Th17 Cell Pathogenicity”. In: *Cell* 163.6 (2015), pp. 1400–1412. ISSN: 10974172. DOI: 10.1016/j.cell.2015.11.009.

[30] Stephanie C Hicks, Mingxiang Teng, and Rafael A Irizarry. “On the widespread and critical impact of systematic bias and batch effects in single-cell RNA-Seq data”. In: *bioRxiv* (2015), p. 025528. DOI: 10.1101/025528. URL: http://biorxiv.org/content/early/2015/08/25/025528.abstract.

[31] Greg Finak et al. “MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data”. In: *Genome Biology* 16.1 (2015), p. 1.

[32] Mikhail Belkin and Partha Niyogi. “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation”. In: *Neural Computation* 15.6 (2003), pp. 1373–1396. ISSN: 0899-7667. DOI: 10.1162/089976603321780317. URL: http://www.mitpressjournals.org/doi/abs/10.1162/089976603321780317.

[33] Laurens Van Der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. ISSN: 1532-4435. DOI: 10.1007/s10479-011-0841-3.

[34] Emma Pierson and Christopher Yau. "Dimensionality reduction for zero-inflated single cell gene expression analysis". In: *Genome Biology* 16.241 (2015). ISSN: 1474-760X. DOI: 10.1101/019141. URL: http://biorxiv.org/content/early/2015/06/14/019141.abstract.

[35] Johann a Gagnon-Bartsch and Terence P Speed. "Using control genes to correct for unwanted variation in microarray data." In: *Biostatistics* 13.3 (2012), pp. 539–52. ISSN: 1468-4357. DOI: 10.1093/biostatistics/kxr034. URL: http://www.ncbi.nlm.nih.gov/pubmed/22101192.

[36] Davide Risso et al. "Normalization of RNA-seq data using factor analysis of control genes or samples". In: *Nat Biotech* 32.9 (2014), pp. 896–902. ISSN: 1087-0156. DOI: 10.1038/nbt.2931. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=4404308&tool=pmcentrez&rendertype=abstract.

[37] Nathan Srebro, Jason D M Rennie, and Tommi S Jaakkola. "Maximum-Margin Matrix Factorization". In: *Advances in Neural Information Processing Systems 17* Section 4 (2005), pp. 1329–1336. ISSN: 10495258. DOI: 10.1.1.59.118.

[38] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. *Spectral Regularization Algorithms for Learning Large Incomplete Matrices.* 2010. DOI: 10.1016/j.surg.2006.10.010.Use. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3087301&tool=pmcentrez&rendertype=abstract.

[39] Aleksandra A. Kolodziejczyk et al. "Single Cell RNA-Sequencing of Pluripotent States Unlocks Modular Transcriptional Variation". In: *Cell Stem Cell* 17.4 (2015), pp. 471–485. ISSN: 18759777. DOI: 10.1016/j.stem.2015.09.011.

[40] Aaron T L Lun and John C Marioni. "Overcoming confounding plate effects in differential expression analyses of single-cell RNA-seq data". In: *bioRxiv* (2016). DOI: 10.1101/073973. URL: http://biorxiv.org/content/early/2016/09/08/073973.

[41] Russell B Fletcher et al. "Deconstructing Olfactory Stem Cell Trajectories at Single-Cell Resolution". In: *Cell Stem Cell* 20.6 (2017), pp. 817–830.

[42] Fanny Perraudeau et al. "Bioconductor workflow for single-cell RNA sequencing: Normalization, dimensionality reduction, clustering, and lineage inference". In: *F1000Research* 6 (2017).

[43] Michael Cole and Davide Risso. *scone: Single Cell Overview of Normalized Expression data.* R package version 1.1.2. 2017. URL: https://bioconductor.org/packages/scone.

[44] Grace XY Zheng et al. "Massively parallel digital transcriptional profiling of single cells". In: *Nature Communications* 8 (2017), p. 14049.

[45] Bo Li and Colin N Dewey. "RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome". In: *BMC Bioinformatics* 12.1 (2011), p. 323. ISSN: 1471-2105. DOI: `10.1186/1471-2105-12-323`. URL: `http://dx.doi.org/10.1186/1471-2105-12-323`.

[46] C Trapnell et al. "Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation". In: *Nature Biotechnology* 28.5 (2010), pp. 511–515.

[47] J H Bullard et al. "Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments". In: *BMC Bioinformatics* 11.1 (2010), p. 94.

[48] M D Robinson and A Oshlack. "A scaling normalization method for differential expression analysis of RNA-seq data". In: *Genome Biology* 11.3 (2010), R25.

[49] Elizabeth Purdom and Davide Risso. *clusterExperiment: Compare Clusterings for Single-Cell Sequencing*. R package version 1.3.3. 2017. URL: `https://bioconductor.org/packages/clusterExperiment`.

[50] George C Tseng and Wing H Wong. "Tight Clustering: A Resampling-Based Approach for Identifying Stable and Tight Patterns in Data". In: *Biometrics* 61.1 (2005), pp. 10–16.

[51] Rahul Satija, Andrew Butler, and Paul Hoffman. *Seurat: Tools for Single Cell Genomics*. R package version 2.0.1. 2017. URL: `https://CRAN.R-project.org/package=Seurat`.

[52] Ludo Waltman and Nees Jan van Eck. "A smart local moving algorithm for large-scale modularity-based community detection". In: *The European Physical Journal B* 86.11 (2013), p. 471.

[53] M D Robinson, D J McCarthy, and G K Smyth. "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data". In: *Bioinformatics* 26.1 (2010), p. 139.

[54] Andrew McDavid, Greg Finak, and Masanao Yajima. *MAST: Model-based Analysis of Single Cell Transcriptomics*. R package version 1.3.2. 2017. URL: `https://github.com/RGLab/MAST/`.

[55] Bernard Desgraupes. *clusterCrit: Clustering Indices*. R package version 1.2.7. 2016. URL: `https://CRAN.R-project.org/package=clusterCrit`.

[56] Davis J Mccarthy et al. "Scater: pre-processing, quality control, normalization and visualization of single-cell RNA-seq data in R". In: *Bioinformatics* 33 (2017), pp. 1179–1186. ISSN: 1367-4803. DOI: `10.1093/bioinformatics/btw777`.

[57] Kelly Street et al. "Slingshot: Cell lineage and pseudotime inference for single-cell transcriptomics". In: *bioRxiv* (2017), p. 128843.

[58] Alexandra-Chloé Villani et al. "Single-cell RNA-seq reveals new types of human blood dendritic cells, monocytes, and progenitors". In: *Science* 356.6335 (2017), eaah4573.

[59] W Evan Johnson, Cheng Li, and Ariel Rabinovic. “Adjusting batch effects in microarray expression data using empirical Bayes methods”. In: *Biostatistics* 8.1 (2007), pp. 118–127.

[60] Jeffrey T Leek et al. “The sva package for removing batch effects and other unwanted variation in high-throughput experiments”. In: *Bioinformatics* 28.6 (2012), pp. 882–883.

[61] F William Townes et al. “Varying-Censoring Aware Matrix Factorization for Single Cell RNA-Sequencing”. In: *bioRxiv* (2017), p. 166736.

[62] Davis J McCarthy et al. “scater: pre-processing, quality control, normalisation and visualisation of single-cell RNA-seq data in R”. In: *bioRxiv* (2016), p. 69633.

[63] Jeffrey T Leek and John D Storey. “Capturing heterogeneity in gene expression studies by surrogate variable analysis”. In: *PLoS Genet* 3.9 (2007), e161.

[64] Zhixiang Lin et al. “Simultaneous dimension reduction and adjustment for confounding variation”. In: *Proceedings of the National Academy of Sciences* 113.51 (2016), pp. 14662–14667. DOI: `10.1073/pnas.1617317113`. URL: `http://www.pnas.org/content/113/51/14662.abstract`.

[65] Michael I Love, John B Hogenesch, and Rafael A Irizarry. “Modeling of RNA-seq fragment sequence bias reduces systematic errors in transcript abundance estimation”. In: *Nature Biotechnology* 34.12 (2016), p. 1287.

[66] Aviv Regev et al. “The Human Cell Atlas”. In: *bioRxiv* (2017), p. 121202.

[67] Bo Wang et al. “Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning”. In: *Nat Methods* 14 (2017), pp. 414–416. ISSN: 1548-7105. URL: `http://dx.doi.org/10.1038/nmeth.4207%20http://10.0.4.14/nmeth.4207%20http://www.nature.com/nmeth/journal/vaop/ncurrent/abs/nmeth.4207.html#supplementary-information`.

[68] Koen Van den Berge et al. “zingeR: unlocking RNA-seq tools for zero-inflation and single cell applications”. In: *bioRxiv* (2017), p. 157982.

[69] Michael I Love, Wolfgang Huber, and Simon Anders. “Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2”. In: *Genome Biology* 15.12 (Dec. 2014), p. 550. ISSN: 1465-6906. DOI: `10.1186/s13059-014-0550-8`. URL: `http://genomebiology.com/2014/15/12/550`.

[70] Mark D Robinson, Davis J McCarthy, and Gordon K Smyth. “edgeR: a Bioconductor package for differential expression analysis of digital gene expression data.” In: *Bioinformatics (Oxford, England)* 26.1 (Jan. 2010), pp. 139–40. ISSN: 1367-4811. DOI: `10.1093/bioinformatics/btp616`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/19910308%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2796818`.

[71] Charity W Law et al. "voom: Precision weights unlock linear model analysis tools for RNA-seq read counts." In: *Genome biology* 15.2 (Jan. 2014), R29. ISSN: 1465-6914. DOI: 10.1186/gb-2014-15-2-r29. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=4053721%7B%5C&%7Dtool=pmcentrez%7B%5C&%7Drendertype=abstract.

[72] Zhong Wang, Mark Gerstein, and Michael Snyder. "RNA-Seq: a revolutionary tool for transcriptomics". In: *Nature Reviews Genetics* 10.1 (Jan. 2009), pp. 57–63. ISSN: 1471-0056. DOI: 10.1038/nrg2484. URL: http://www.nature.com/doifinder/10.1038/nrg2484.

[73] Sara Goodwin, John D. McPherson, and W. Richard McCombie. "Coming of age: ten years of next-generation sequencing technologies". In: *Nature Reviews Genetics* 17.6 (May 2016), pp. 333–351. ISSN: 1471-0056. DOI: 10.1038/nrg.2016.49. URL: http://www.nature.com/doifinder/10.1038/nrg.2016.49.

[74] Tapio Lönnberg et al. "Single-cell RNA-seq and computational analysis using temporal mixture modelling resolves Th1/Tfh fate bifurcation in malaria." In: *Science immunology* 2.9 (Mar. 2017). DOI: 10.1126/sciimmunol.aal2192. URL: http://www.ncbi.nlm.nih.gov/pubmed/28345074%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5365145.

[75] Florian Buettner et al. "Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells". In: *Nature Biotechnology* 33.2 (Jan. 2015), pp. 155–160. ISSN: 1087-0156. DOI: 10.1038/nbt.3102. URL: http://www.ncbi.nlm.nih.gov/pubmed/25599176%20http://www.nature.com/doifinder/10.1038/nbt.3102.

[76] Anoop P. Patel et al. "Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma". In: *Science* 344.6190 (2014). URL: http://science.sciencemag.org/content/344/6190/1396.

[77] Aleksandra A Kolodziejczyk et al. "Single Cell RNA-Sequencing of Pluripotent States Unlocks Modular Transcriptional Variation." In: *Cell stem cell* 17.4 (Oct. 2015), pp. 471–85. ISSN: 1875-9777. DOI: 10.1016/j.stem.2015.09.011. URL: http://www.ncbi.nlm.nih.gov/pubmed/26431182%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4595712.

[78] Li Li et al. "Single-Cell RNA-Seq Analysis Maps Development of Human Germline Cells and Gonadal Niche Interactions". In: *Cell Stem Cell* 20.6 (June 2017), 858–873.e4. ISSN: 19345909. DOI: 10.1016/j.stem.2017.03.007. URL: http://www.ncbi.nlm.nih.gov/pubmed/28457750%20http://linkinghub.elsevier.com/retrieve/pii/S1934590917300784.

[79] Dmitry Usoskin et al. "Unbiased classification of sensory neuron types by large-scale single-cell RNA sequencing". In: *Nature Neuroscience* 18.1 (Nov. 2014), pp. 145–153. ISSN: 1097-6256. DOI: 10.1038/nn.3881. URL: http://www.nature.com/doifinder/10.1038/nn.3881.

[80] Aleksandra A. Kolodziejczyk et al. "The Technology and Biology of Single-Cell RNA Sequencing". In: *Molecular Cell* 58.4 (May 2015), pp. 610–620. ISSN: 10972765. DOI: 10.1016/j.molcel.2015.04.005. URL: http://linkinghub.elsevier.com/retrieve/pii/S1097276515002610.

[81] T. Nakamura et al. "SC3-seq: a method for highly parallel and quantitative measurement of single-cell gene expression". In: *Nucleic Acids Research* 43.9 (May 2015), e60–e60. ISSN: 0305-1048. DOI: 10.1093/nar/gkv134. URL: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkv134.

[82] Angela R Wu et al. "Quantitative assessment of single-cell RNA-sequencing methods". In: *Nature Methods* 11.1 (Oct. 2013), pp. 41–46. ISSN: 1548-7091. DOI: 10.1038/nmeth.2694. URL: http://www.ncbi.nlm.nih.gov/pubmed/24141493%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4022966%20http://www.nature.com/doifinder/10.1038/nmeth.2694.

[83] Saiful Islam et al. "Quantitative single-cell RNA-seq with unique molecular identifiers". In: *Nature Methods* 11.2 (Dec. 2013), pp. 163–166. ISSN: 1548-7091. DOI: 10.1038/nmeth.2772. URL: http://www.ncbi.nlm.nih.gov/pubmed/24363023%20http://www.nature.com/doifinder/10.1038/nmeth.2772.

[84] Saiful Islam et al. "Characterization of the single-cell transcriptional landscape by highly multiplex RNA-seq." In: *Genome research* 21.7 (July 2011), pp. 1160–7. ISSN: 1549-5469. DOI: 10.1101/gr.110882.110. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3129258%7B%5C&%7Dtool=pmcentrez%7B%5C&%7Drendertype=abstract.

[85] Simone Picelli et al. "Full-length RNA-seq from single cells using Smart-seq2". In: *Nature Protocols* 9.1 (Jan. 2014), pp. 171–181. ISSN: 1754-2189. DOI: 10.1038/nprot.2014.006. URL: http://www.ncbi.nlm.nih.gov/pubmed/24385147%20http://www.nature.com/doifinder/10.1038/nprot.2014.006.

[86] Tamar Hashimshony et al. "CEL-Seq2: sensitive highly-multiplexed single-cell RNA-Seq." In: *Genome biology* 17 (Apr. 2016), p. 77. ISSN: 1474-760X. DOI: 10.1186/s13059-016-0938-8. URL: http://www.ncbi.nlm.nih.gov/pubmed/27121950%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4848782.

[87] Greg Finak et al. "MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data". In: *Genome Biology* 16.1 (Dec. 2015), p. 278. ISSN: 1474-760X. DOI: 10.1186/s13059-015-0844-5. URL: http://genomebiology.com/2015/16/1/278.

[88] Arjun Raj and Alexander van Oudenaarden. "Nature, nurture, or chance: stochastic gene expression and its consequences." In: *Cell* 135.2 (Oct. 2008), pp. 216–26. ISSN: 1097-4172. DOI: 10.1016/j.cell.2008.09.050. URL: http://www.ncbi.nlm.nih.gov/pubmed/18957198%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3118044.

[89] Arjun Raj et al. "Stochastic mRNA Synthesis in Mammalian Cells". In: *PLoS Biology* 4.10 (Sept. 2006). Ed. by Ueli Schibler, e309. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.0040309. URL: http://www.ncbi.nlm.nih.gov/pubmed/17048983%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1563489%20http://dx.plos.org/10.1371/journal.pbio.0040309.

[90] Emma Pierson and Christopher Yau. "ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis". In: *Genome Biology* 16.1 (Dec. 2015), p. 241. ISSN: 1474-760X. DOI: 10.1186/s13059-015-0805-z. URL: http://genomebiology.com/2015/16/1/241.

[91] Davide Risso et al. "ZINB-WaVE : A general and flexible method for signal extraction from single-cell RNA-seq data". In: *bioRxiv* (2017). DOI: 10.1101/125112. URL: http://www.biorxiv.org/content/early/2017/04/06/125112.

[92] Manu Setty et al. "Wishbone identifies bifurcating developmental trajectories from single-cell data." In: *Nature biotechnology* 34.6 (June 2016), pp. 637–45. ISSN: 1546-1696. DOI: 10.1038/nbt.3569. URL: http://www.ncbi.nlm.nih.gov/pubmed/27136076%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4900897.

[93] Xiaojie Qiu et al. "Reversed graph embedding resolves complex single-cell trajectories". In: *Nature Methods* (Aug. 2017). DOI: 10.1038/nmeth.4402. URL: https://www.nature.com/nmeth/journal/vaop/ncurrent/full/nmeth.4402.html.

[94] Kelly Street et al. "Slingshot: Cell lineage and pseudotime inference for single-cell transcriptomics". In: *bioRxiv* (2017). DOI: 10.1101/128843.

[95] Aaron T.L. Lun, Davis J. McCarthy, and John C. Marioni. "A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor". In: *F1000Research* 5 (Oct. 2016), p. 2122. ISSN: 2046-1402. DOI: 10.12688/f1000research.9501.2. URL: https://f1000research.com/articles/5-2122/v2.

[96] Peter V Kharchenko, Lev Silberstein, and David T Scadden. "Bayesian approach to single-cell differential expression analysis." In: *Nature methods* 11.7 (July 2014), pp. 740–2. ISSN: 1548-7105. DOI: 10.1038/nmeth.2967. URL: http://www.ncbi.nlm.nih.gov/pubmed/24836921%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4112276.

[97] Maria K. Jaakkola et al. "Comparison of methods to detect differentially expressed genes between single-cell populations". In: *Briefings in Bioinformatics* (July 2016), bbw057. ISSN: 1467-5463. DOI: 10.1093/bib/bbw057. URL: http://www.ncbi.nlm.nih.gov/pubmed/27373736%20http://bib.oxfordjournals.org/lookup/doi/10.1093/bib/bbw057.

[98] Charlotte Soneson and Mark D. Robinson. "Bias, Robustness And Scalability In Differential Expression Analysis Of Single-Cell RNA-Seq Data". In: *bioRxiv* (2017). URL: http://biorxiv.org/content/early/2017/05/28/143289.

[99] Davis J McCarthy, Yunshun Chen, and Gordon K Smyth. "Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation." In: *Nucleic acids research* 40.10 (May 2012), pp. 4288–97. ISSN: 1362-4962. DOI: 10.1093/nar/gks042. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3378882%7B%5C&%7Dtool=pmcentrez%7B%5C&%7Drendertype=abstract.

[100] Qiaolin Deng et al. "Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells." In: *Science (New York, N.Y.)* 343.6167 (Jan. 2014), pp. 193–6. ISSN: 1095-9203. DOI: 10.1126/science.1245316. URL: http://www.ncbi.nlm.nih.gov/pubmed/24408435.

[101] Daniel Bottomly et al. "Evaluating gene expression in C57BL/6J and DBA/2J mouse striatum using RNA-Seq and microarrays." In: *PloS one* 6.3 (Jan. 2011), e17820. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0017820. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3063777%7B%5C&%7Dtool=pmcentrez%7B%5C&%7Drendertype=abstract.

[102] Xiaobei Zhou, Helen Lindsay, and Mark D Robinson. "Robustly detecting differential expression in RNA sequencing data using observation weights." In: *Nucleic acids research* 42.11 (June 2014), e91. ISSN: 1362-4962. DOI: 10.1093/nar/gku310. URL: http://www.ncbi.nlm.nih.gov/pubmed/24753412%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4066750.

[103] Joseph N Paulson et al. "Differential abundance analysis for microbial marker-gene surveys." In: *Nature methods* 10.12 (Sept. 2013), pp. 1200–2. ISSN: 1548-7105. DOI: 10.1038/nmeth.2658. arXiv: NIHMS150003. URL: http://www.ncbi.nlm.nih.gov/pubmed/24076764%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4010126%20http://www.nature.com/doifinder/10.1038/nmeth.2658%20http://dx.doi.org/10.1038/nmeth.2658.

[104] Davis J McCarthy, Yunshun Chen, and Gordon K Smyth. "Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation." In: *Nucleic acids research* 40.10 (May 2012), pp. 4288–97. ISSN: 1362-4962. DOI: 10.1093/nar/gks042. URL: http://www.ncbi.nlm.nih.gov/pubmed/22287627%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3378882.

[105] Paul J. McMurdie and Susan Holmes. "phyloseq: An R Package for Reproducible Interactive Analysis and Graphics of Microbiome Census Data". In: *PLoS ONE* 8.4 (Apr. 2013). Ed. by Michael Watson, e61217. DOI: `10.1371/journal.pone.0061217`. URL: `http://dx.plos.org/10.1371/journal.pone.0061217`.

[106] Richard Bourgon, Robert Gentleman, and Wolfgang Huber. "Independent filtering increases detection power for high-throughput experiments." In: *Proceedings of the National Academy of Sciences of the United States of America* 107.21 (May 2010), pp. 9546–51. ISSN: 1091-6490. DOI: `10.1073/pnas.0914005107`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/20460310%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2906865`.

[107] Yaov Benjamini and Yosef Hochberg. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 57.1 (1995), pp. 289–300. URL: `https://www.jstor.org/stable/2346101?seq=1%7B%5C#%7Dpage%7B%5C_%7Dscan%7B%5C_%7Dtab%7B%5C_%7Dcontents`.

[108] Charlotte Soneson and Mark D Robinson. "iCOBRA: open, reproducible, standardized and live method benchmarking". In: *Nature Methods* 13.4 (Mar. 2016), pp. 283–283. ISSN: 1548-7091. DOI: `10.1038/nmeth.3805`. URL: `http://www.nature.com/doifinder/10.1038/nmeth.3805`.

[109] Debarka Sengupta et al. "Fast, scalable and accurate differential expression analysis for single cells". In: *bioRxiv* (2016).

[110] Mark A van de Wiel et al. "ShrinkBayes: a versatile R-package for analysis of count-based sequencing data in complex study designs". In: *BMC Bioinformatics* 15.1 (2014), p. 116. ISSN: 1471-2105. DOI: `10.1186/1471-2105-15-116`. URL: `http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-15-116`.

[111] Dirk F. Moore. "Asymptotic Properties of Moment Estimators for Overdispersed Counts and Proportions". In: *Biometrika* 73.3 (Dec. 1986), p. 583. ISSN: 00063444. DOI: `10.2307/2336522`. URL: `http://www.jstor.org/stable/2336522?origin=crossref`.

[112] P. (Peter) McCullagh and John A. Nelder. *Generalized linear models*. Second. New York: Chapman and Hall, 1989, p. 511. ISBN: 9780412317606. URL: `https://www.crcpress.com/Generalized-Linear-Models-Second-Edition/McCullagh-Nelder/p/book/9780412317606`.

[113] Simon N. Wood. "Thin plate regression splines". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65.1 (Feb. 2003), pp. 95–114. ISSN: 1369-7412. DOI: `10.1111/1467-9868.00374`. URL: `http://doi.wiley.com/10.1111/1467-9868.00374`.

[114] Cole Trapnell et al. "Differential analysis of gene regulation at transcript resolution with RNA-seq." In: *Nature biotechnology* 31.1 (Jan. 2013), pp. 46–53. ISSN: 1546-1696. DOI: `10.1038/nbt.2450`. URL: `http://dx.doi.org/10.1038/nbt.2450`.

[115] Alexey Sergushichev. "An algorithm for fast preranked gene set enrichment analysis using cumulative statistic calculation". In: *bioRxiv* (June 2016), p. 060012. DOI: `10.1101/060012`. URL: `https://www.biorxiv.org/content/early/2016/06/20/060012`.

[116] Dvir Aran, Zicheng Hu, and Atul J Butte. "xCell: Digitally portraying the tissue cellular heterogeneity landscape". In: *doi.org* (June 2017), p. 114165. DOI: `10.1101/114165`. URL: `https://www.biorxiv.org/content/early/2017/06/15/114165`.

[117] A. Subramanian et al. "Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles". In: *Proceedings of the National Academy of Sciences* 102.43 (Oct. 2005), pp. 15545–15550. ISSN: 0027-8424. DOI: `10.1073/pnas.0506580102`. URL: `http://www.pnas.org/cgi/doi/10.1073/pnas.0506580102`.

[118] Andrew Butler and Rahul Satija. "Integrated analysis of single cell transcriptomic data across conditions, technologies, and species". In: *doi.org* (July 2017), p. 164889. DOI: `10.1101/164889`. URL: `https://www.biorxiv.org/content/early/2017/07/18/164889`.

[119] Cameron Colin A. and Pravik K. Trivedi. "No Title". In: *Regression analysis of Count Data*. Second. Cambrdridge: Cambridge University Press, 2013. Chap. 4.3.

[120] Charlotte Soneson and Mark D Robinson. "Towards unified quality verification of synthetic count data with countsimQC". In: *Bioinformatics* (Oct. 2017). ISSN: 1367-4803. DOI: `10.1093/bioinformatics/btx631`. URL: `http://academic.oup.com/bioinformatics/article/doi/10.1093/bioinformatics/btx631/4345646/Towards-unified-quality-verification-of-synthetic`.

[121] Stephanie C Hicks, Mingxiang Teng, and Rafael A Irizarry. "On the widespread and critical impact of systematic bias and batch effects in single-cell RNA-Seq data". In: *bioRxiv* (2015). URL: `http://biorxiv.org/content/early/2015/12/27/025528`.

[122] Koen Van den Berge et al. "zingeR: unlocking RNA-seq tools for zero-inflation and single cell applications". In: *bioRxiv* (June 2017), p. 157982. DOI: `10.1101/157982`. URL: `https://www.biorxiv.org/content/early/2017/06/30/157982`.

[123] Dominic Grün, Lennart Kester, and Alexander van Oudenaarden. "Validation of noise models for single-cell transcriptomics". In: *Nature Methods* 11.6 (Apr. 2014), pp. 637–640. ISSN: 1548-7091. DOI: `10.1038/nmeth.2930`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/24747814%20http://www.nature.com/doifinder/10.1038/nmeth.2930`.

[124] Christoph Ziegenhain et al. "Comparative Analysis of Single-Cell RNA Sequencing Methods". In: *Molecular Cell* 65.4 (Feb. 2017), pp. 631–643. ISSN: 10972765. DOI: 10.1016/j.molcel.2017.01.023. URL: http://linkinghub.elsevier.com/retrieve/pii/S1097276517300497.

[125] Bhupinder Pal et al. "Construction of developmental lineage relationships in the mouse mammary gland by single-cell RNA profiling". In: *Nature Communications* 8.1 (Dec. 2017), p. 1627. ISSN: 2041-1723. DOI: 10.1038/s41467-017-01560-x. URL: http://www.nature.com/articles/s41467-017-01560-x.

[126] Keisuke Fujita, Mitsuhiro Iwaki, and Toshio Yanagida. "Transcriptional bursting is intrinsically caused by interplay between RNA polymerases on DNA". In: *Nature Communications* 7 (Dec. 2016), p. 13788. ISSN: 2041-1723. DOI: 10.1038/ncomms13788. URL: http://www.nature.com/doifinder/10.1038/ncomms13788.

[127] Lizhen Xu et al. "Assessment and Selection of Competing Models for Zero-Inflated Microbiome Data." In: *PloS one* 10.7 (2015), e0129606. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0129606. URL: http://www.ncbi.nlm.nih.gov/pubmed/26148172%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4493133.

[128] Joseph K Pickrell et al. "Understanding mechanisms underlying human gene expression variation with RNA sequencing." In: *Nature* 464.7289 (Apr. 2010), pp. 768–72. ISSN: 1476-4687. DOI: 10.1038/nature08872. URL: http://www.ncbi.nlm.nih.gov/pubmed/20220758%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3089435.

[129] S. Greenblum, P. J. Turnbaugh, and E. Borenstein. "Metagenomic systems biology of the human gut microbiome reveals topological shifts associated with obesity and inflammatory bowel disease". In: *Proceedings of the National Academy of Sciences* 109.2 (Jan. 2012), pp. 594–599. ISSN: 0027-8424. DOI: 10.1073/pnas.1116053109. URL: http://www.pnas.org/cgi/doi/10.1073/pnas.1116053109.

[130] B. J. Baker et al. "Lineages of Acidophilic Archaea Revealed by Community Genomic Analysis". In: *Science* 314.5807 (Dec. 2006), pp. 1933–1935. ISSN: 0036-8075. DOI: 10.1126/science.1132690. URL: http://www.sciencemag.org/cgi/doi/10.1126/science.1132690.

[131] Emiley A. Eloe-Fadrosh et al. "Metagenomics uncovers gaps in amplicon-based detection of microbial diversity". In: *Nature Microbiology* 1.4 (Feb. 2016), p. 15032. ISSN: 2058-5276. DOI: 10.1038/nmicrobiol.2015.32. URL: http://www.nature.com/articles/nmicrobiol201532.

[132] *http://www.alimetrics.net/en/index.php/dna-sequence-analysis.*

[133] J. M. Janda and S. L. Abbott. “16S rRNA Gene Sequencing for Bacterial Identification in the Diagnostic Laboratory: Pluses, Perils, and Pitfalls”. In: *Journal of Clinical Microbiology* 45.9 (Sept. 2007), pp. 2761–2764. ISSN: 0095-1137. DOI: 10.1128/JCM.01228-07. URL: http://jcm.asm.org/cgi/doi/10.1128/JCM.01228-07.

[134] J. R. Cole et al. “The Ribosomal Database Project: improved alignments and new tools for rRNA analysis”. In: *Nucleic Acids Research* 37.Database (Jan. 2009), pp. D141–D145. ISSN: 0305-1048. DOI: 10.1093/nar/gkn879. URL: http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gkn879.

[135] C. Quast et al. “The SILVA ribosomal RNA gene database project: improved data processing and web-based tools”. In: *Nucleic Acids Research* 41.D1 (Jan. 2013), pp. D590–D596. ISSN: 0305-1048. DOI: 10.1093/nar/gks1219. URL: http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gks1219.

[136] T. Z. DeSantis et al. “Greengenes, a Chimera-Checked 16S rRNA Gene Database and Workbench Compatible with ARB”. In: *Applied and Environmental Microbiology* 72.7 (July 2006), pp. 5069–5072. ISSN: 0099-2240. DOI: 10.1128/AEM.03006-05. URL: http://aem.asm.org/cgi/doi/10.1128/AEM.03006-05.

[137] A F Yassin et al. “Nocardia paucivorans sp. nov.” In: *International journal of systematic and evolutionary microbiology* 50 Pt 2 (Mar. 2000), pp. 803–9. ISSN: 1466-5026. DOI: 10.1099/00207713-50-2-803. URL: http://www.ncbi.nlm.nih.gov/pubmed/10758891.

[138] G E Fox, J D Wisotzkey, and P Jurtshuk. “How close is close: 16S rRNA sequence identity may not be sufficient to guarantee species identity.” In: *International journal of systematic bacteriology* 42.1 (Jan. 1992), pp. 166–70. ISSN: 0020-7713. DOI: 10.1099/00207713-42-1-166. URL: http://www.ncbi.nlm.nih.gov/pubmed/1371061.

[139] S. F. Stoddard et al. “rrnDB: improved tools for interpreting rRNA gene abundance in bacteria and archaea and a new foundation for future development”. In: *Nucleic Acids Research* 43.D1 (Jan. 2015), pp. D593–D598. ISSN: 0305-1048. DOI: 10.1093/nar/gku1201. URL: http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gku1201.

[140] Tom Coenye and Peter Vandamme. “Intragenomic heterogeneity between multiple 16S ribosomal RNA operons in sequenced bacterial genomes”. In: *FEMS Microbiology Letters* 228.1 (Nov. 2003), pp. 45–49. ISSN: 03781097. DOI: 10.1016/S0378-1097(03)00717-1. URL: http://femsle.oxfordjournals.org/cgi/doi/10.1016/S0378-1097(03)00717-1.

[141] Kei Kitahara and Kentaro Miyazaki. “Revisiting bacterial phylogeny”. In: *Mobile Genetic Elements* 3.1 (Jan. 2013), e24210. ISSN: 2159-256X. DOI: 10.4161/mge.24210. URL: http://www.tandfonline.com/doi/abs/10.4161/mge.24210.

[142] *https://en.wikipedia.org/wiki/Taxonomic_rank.*

[143] J. F. Petrosino et al. “Metagenomic Pyrosequencing and Microbial Identification”. In: *Clinical Chemistry* 55.5 (May 2009), pp. 856–866. ISSN: 0009-9147. DOI: 10.1373/clinchem.2008.107565. URL: http://www.clinchem.org/cgi/doi/10.1373/clinchem.2008.107565.

[144] Aarti Desai et al. “Identification of Optimum Sequencing Depth Especially for De Novo Genome Assembly of Small Genomes Using Next Generation Sequencing Data”. In: *PLoS ONE* 8.4 (Apr. 2013). Ed. by Shu-Dong Zhang, e60204. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0060204. URL: http://dx.plos.org/10.1371/journal.pone.0060204.

[145] Curtis Huttenhower et al. “Structure, function and diversity of the healthy human microbiome”. In: *Nature* 486.7402 (June 2012), pp. 207–214. ISSN: 0028-0836. DOI: 10.1038/nature11234. URL: http://www.nature.com/doifinder/10.1038/nature11234.

[146] Barbara A. Methé et al. “A framework for human microbiome research”. In: *Nature* 486.7402 (June 2012), pp. 215–221. ISSN: 0028-0836. DOI: 10.1038/nature11209. URL: http://www.nature.com/doifinder/10.1038/nature11209.

[147] M. Kim et al. “Towards a taxonomic coherence between average nucleotide identity and 16S rRNA gene sequence similarity for species demarcation of prokaryotes”. In: *INTERNATIONAL JOURNAL OF SYSTEMATIC AND EVOLUTIONARY MICROBIOLOGY* 64.Pt 2 (Feb. 2014), pp. 346–351. ISSN: 1466-5026. DOI: 10.1099/ijs.0.059774-0. URL: http://ijs.microbiologyresearch.org/content/journal/ijsem/10.1099/ijs.0.059774-0.

[148] Erin B Fichot and R Sean Norman. “Microbial phylogenetic profiling with the Pacific Biosciences sequencing platform”. In: *Microbiome* 1.1 (2013), p. 10. ISSN: 2049-2618. DOI: 10.1186/2049-2618-1-10. URL: http://www.microbiomejournal.com/content/1/1/10.

[149] K. J. Travers et al. “A flexible and efficient template format for circular consensus sequencing and SNP detection”. In: *Nucleic Acids Research* 38.15 (Aug. 2010), e159–e159. ISSN: 0305-1048. DOI: 10.1093/nar/gkq543. URL: http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gkq543.

[150] Sergey Koren et al. “Hybrid error correction and de novo assembly of single-molecule sequencing reads”. In: *Nature Biotechnology* 30.7 (July 2012), pp. 693–700. ISSN: 1087-0156. DOI: 10.1038/nbt.2280. URL: http://www.nature.com/doifinder/10.1038/nbt.2280.

[151] Robert C Edgar. “UPARSE: highly accurate OTU sequences from microbial amplicon reads.” In: *Nature methods* 10.10 (Oct. 2013), pp. 996–8. ISSN: 1548-7105. DOI: 10.1038/nmeth.2604. URL: http://www.ncbi.nlm.nih.gov/pubmed/23955772.

[152] L. Fu et al. “CD-HIT: accelerated for clustering the next-generation sequencing data”. In: *Bioinformatics* 28.23 (Dec. 2012), pp. 3150–3152. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bts565. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/bts565.

[153] R. C. Edgar. “Search and clustering orders of magnitude faster than BLAST”. In: *Bioinformatics* 26.19 (Oct. 2010), pp. 2460–2461. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btq461. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btq461.

[154] Mohammadreza Ghodsi, Bo Liu, and Mihai Pop. “DNACLUST: accurate and efficient clustering of phylogenetic marker genes”. In: *BMC Bioinformatics* 12.1 (2011), p. 271. ISSN: 1471-2105. DOI: 10.1186/1471-2105-12-271. URL: http://www.biomedcentral.com/1471-2105/12/271.

[155] Michael J Rosen et al. “Denoising PCR-amplified metagenome data”. In: *BMC Bioinformatics* 13.1 (2012), p. 283. ISSN: 1471-2105. DOI: 10.1186/1471-2105-13-283. URL: http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-13-283.

[156] Benjamin J Callahan, Paul J McMurdie, and Susan P Holmes. “Exact sequence variants should replace operational taxonomic units in marker-gene data analysis”. In: *The ISME Journal* 11.12 (Dec. 2017), pp. 2639–2643. ISSN: 1751-7362. DOI: 10.1038/ismej.2017.119. URL: http://www.nature.com/doifinder/10.1038/ismej.2017.119.

[157] Benjamin J Callahan et al. “DADA2: High-resolution sample inference from Illumina amplicon data”. In: *Nature Methods* 13.7 (May 2016), pp. 581–583. ISSN: 1548-7091. DOI: 10.1038/nmeth.3869. URL: http://www.nature.com/doifinder/10.1038/nmeth.3869.

[158] *https://qiime2.org*.

[159] Robert C Edgar. “UNOISE2: improved error-correction for Illumina 16S and ITS amplicon sequencing”. In: *bioRxiv* (2016), p. 081257. DOI: 10.1101/081257. URL: http://biorxiv.org/content/early/2016/10/15/081257.abstract%5Cnhttp://biorxiv.org/lookup/doi/10.1101/081257.

[160] Amnon Amir et al. “Deblur Rapidly Resolves Single-Nucleotide Community Sequence Patterns”. In: *mSystems* 2.2 (Apr. 2017). Ed. by Jack A. Gilbert, pp. 00191–16. ISSN: 2379-5077. DOI: 10.1128/mSystems.00191-16. URL: http://msystems.asm.org/lookup/doi/10.1128/mSystems.00191-16.

[161] Qiong Wang et al. “Naive Bayesian classifier for rapid assignment of rRNA sequences into the new bacterial taxonomy.” In: *Applied and environmental microbiology* 73.16 (Aug. 2007), pp. 5261–7. ISSN: 0099-2240. DOI: 10.1128/AEM.00062-07. URL: http://www.ncbi.nlm.nih.gov/pubmed/17586664%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1950982.

[162] Robert C. Edgar. *UTAX*. URL: http://www.drive5.com/usearch/manual/utax_algo.html.

[163] J. G. Caporaso et al. "PyNAST: a flexible tool for aligning sequences to a template alignment". In: *Bioinformatics* 26.2 (Jan. 2010), pp. 266–267. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btp636. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btp636.

[164] Erica Plummer and Jimmy Twin. "A Comparison of Three Bioinformatics Pipelines for the Analysis of Preterm Gut Microbiota using 16S rRNA Gene Sequencing Data". In: *Journal of Proteomics & Bioinformatics* 8.12 (2015). ISSN: 0974276X. DOI: 10.4172/jpb.1000381. URL: http://www.omicsonline.org/open-access/a-comparison-of-three-bioinformatics-pipelines-for-the-analysis-ofpreterm-gut-microbiota-using-16s-rrna-gene-sequencing-data-jpb-1000381.php?aid=65142.

[165] J Gregory Caporaso et al. "QIIME allows analysis of high-throughput community sequencing data". In: *Nature Methods* 7.5 (May 2010), pp. 335–336. ISSN: 1548-7091. DOI: 10.1038/nmeth.f.303. URL: http://www.nature.com/doifinder/10.1038/nmeth.f.303.

[166] P. D. Schloss et al. "Introducing mothur: Open-Source, Platform-Independent, Community-Supported Software for Describing and Comparing Microbial Communities". In: *Applied and Environmental Microbiology* 75.23 (Dec. 2009), pp. 7537–7541. ISSN: 0099-2240. DOI: 10.1128/AEM.01541-09. URL: http://aem.asm.org/cgi/doi/10.1128/AEM.01541-09.

[167] F Meyer et al. "The metagenomics RAST server – a public resource for the automatic phylogenetic and functional analysis of metagenomes". In: *BMC Bioinformatics* 9.1 (2008), p. 386. ISSN: 1471-2105. DOI: 10.1186/1471-2105-9-386. URL: http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-9-386.

[168] Samuel S Minot, Niklas Krumm, and Nicholas B Greenfield. *One Codex: A Sensitive and Accurate Data Platform for Genomic Microbial Identification*. Tech. rep. Sept. 2015. DOI: 10.1101/027607. URL: http://biorxiv.org/lookup/doi/10.1101/027607.

[169] Robert Edgar. *SINTAX: a simple non-Bayesian taxonomy classifier for 16S and ITS sequences*. Tech. rep. Sept. 2016. DOI: 10.1101/074161. URL: http://biorxiv.org/lookup/doi/10.1101/074161.

[170] Ron Sender, Shai Fuchs, and Ron Milo. *Revised estimates for the number of human and bacteria cells in the body*. Tech. rep. Jan. 2016. DOI: 10.1101/036103. URL: http://biorxiv.org/lookup/doi/10.1101/036103.

[171] Anthony Rhoads and Kin Fai Au. “PacBio Sequencing and Its Applications”. In: *Genomics, Proteomics & Bioinformatics* 13.5 (Oct. 2015), pp. 278–289. ISSN: 16720229. DOI: `10.1016/j.gpb.2015.08.002`. URL: `http://linkinghub.elsevier.com/retrieve/pii/S1672022915001345`.

[172] Ben Langmead and Steven L Salzberg. “Fast gapped-read alignment with Bowtie 2”. In: *Nature Methods* 9.4 (Mar. 2012), pp. 357–359. ISSN: 1548-7091. DOI: `10.1038/nmeth.1923`. URL: `http://www.nature.com/doifinder/10.1038/nmeth.1923`.

[173] Y. Ono, K. Asai, and M. Hamada. “PBSIM: PacBio reads simulator–toward accurate genome assembly”. In: *Bioinformatics* 29.1 (Jan. 2013), pp. 119–121. ISSN: 1367-4803. DOI: `10.1093/bioinformatics/bts649`. URL: `http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/bts649`.

[174] Peter J. Turnbaugh et al. “A core gut microbiome in obese and lean twins”. In: *Nature* 457.7228 (Jan. 2009), pp. 480–484. ISSN: 0028-0836. DOI: `10.1038/nature07540`. URL: `http://www.nature.com/doifinder/10.1038/nature07540`.

[175] Dan Knights, Kara G Lassen, and Ramnik J Xavier. “Advances in inflammatory bowel disease pathogenesis: linking host genetics and the microbiome”. In: *Gut* 62.10 (Oct. 2013), pp. 1505–1510. ISSN: 0017-5749. DOI: `10.1136/gutjnl-2012-303954`. URL: `http://gut.bmj.com/lookup/doi/10.1136/gutjnl-2012-303954`.

[176] Junjie Qin et al. “A metagenome-wide association study of gut microbiota in type 2 diabetes”. In: *Nature* 490.7418 (Sept. 2012), pp. 55–60. ISSN: 0028-0836. DOI: `10.1038/nature11450`. URL: `http://www.nature.com/doifinder/10.1038/nature11450`.

[177] Nur Arslan. “Obesity, fatty liver disease and intestinal microbiota”. In: *World Journal of Gastroenterology* 20.44 (2014), p. 16452. ISSN: 1007-9327. DOI: `10.3748/wjg.v20.i44.16452`. URL: `http://www.wjgnet.com/1007-9327/full/v20/i44/16452.htm`.

[178] Human Microbiome Project Consortium. “Structure, function and diversity of the healthy human microbiome.” In: *Nature* 486.7402 (June 2012), pp. 207–14. ISSN: 1476-4687. DOI: `10.1038/nature11234`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/22699609%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3564958`.

[179] Human Microbiome Project Consortium. “A framework for human microbiome research.” In: *Nature* 486.7402 (June 2012), pp. 215–21. ISSN: 1476-4687. DOI: `10.1038/nature11209`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/22699610%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3377744`.

[180] Patrick D. Schloss, Dirk Gevers, and Sarah L. Westcott. “Reducing the Effects of PCR Amplification and Sequencing Artifacts on 16S rRNA-Based Studies”. In: *PLoS ONE* 6.12 (Dec. 2011). Ed. by Jack Anthony Gilbert, e27310. ISSN: 1932-6203. DOI: `10.1371/journal.pone.0027310`. URL: `http://dx.plos.org/10.1371/journal.pone.0027310`.

[181] S. F. Stoddard et al. "rrnDB: improved tools for interpreting rRNA gene abundance in bacteria and archaea and a new foundation for future development". In: *Nucleic Acids Research* 43.D1 (Jan. 2015), pp. D593–D598. ISSN: 0305-1048. DOI: `10.1093/nar/gku1201`. URL: `http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gku1201`.

[182] Schloss PD et al. "Sequencing 16S rRNA gene fragments using the PacBio SMRT DNA sequencing system". In: *PeerJ 4:e1869* (2016). DOI: `10.7717/peerj.1869`.

[183] Jun Hang et al. "16S rRNA gene pyrosequencing of reference and clinical samples and investigation of the temperature stability of microbiome profiles". In: *Microbiome* 2.1 (2014), p. 31. ISSN: 2049-2618. DOI: `10.1186/2049-2618-2-31`. URL: `http://www.microbiomejournal.com/content/2/1/31`.

[184] Burrows and Wheeler. "A Block-sorting Lossless Data Compression Algorithm". In: (1994).

[185] Paolo Ferragina and Giovanni Manzini. "Indexing compressed text". In: *Journal of the ACM* 52.4 (July 2005), pp. 552–581. ISSN: 00045411. DOI: `10.1145/1082036.1082039`. URL: `http://portal.acm.org/citation.cfm?doid=1082036.1082039`.

[186] Heng Li and Richard Durbin. "Fast and accurate short read alignment with Burrows-Wheeler transform." In: *Bioinformatics (Oxford, England)* 25.14 (July 2009), pp. 1754–60. ISSN: 1367-4811. DOI: `10.1093/bioinformatics/btp324`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/19451168%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2705234`.

[187] Heng Li and Richard Durbin. "Fast and accurate long-read alignment with Burrows-Wheeler transform." In: *Bioinformatics (Oxford, England)* 26.5 (Mar. 2010), pp. 589–95. ISSN: 1367-4811. DOI: `10.1093/bioinformatics/btp698`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/20080505%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2828108`.

[188] R. Li et al. "SOAP2: an improved ultrafast tool for short read alignment". In: *Bioinformatics* 25.15 (Aug. 2009), pp. 1966–1967. ISSN: 1367-4803. DOI: `10.1093/bioinformatics/btp336`. URL: `http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btp336`.

[189] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. "Optimization by simulated annealing". In: *SCIENCE* 220 (1983), pp. 671–680.

[190] Richard Durbin et al. *Biological sequence analysis*. Cambridge: Cambridge University Press, 1998. ISBN: 9780511790492. DOI: `10.1017/CBO9780511790492`. URL: `http://ebooks.cambridge.org/ref/id/CBO9780511790492`.

[191] Alan Agresti. *An Introduction to Categorical Data Analysis*. Hoboken, NJ, USA: John Wiley & Sons, Inc., Mar. 2007, p. 106. ISBN: 9780470114759. DOI: `10.1002/0470114754`. URL: `http://doi.wiley.com/10.1002/0470114754`.

[192] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Vol. 12. 4. Boston, MA: Springer US, Dec. 1989, pp. 100, 102, 114. ISBN: 978-0-412-31760-6. DOI: 10.1007/978-1-4899-3242-6. URL: http://projecteuclid.org/euclid.aos/1176346819%20http://link.springer.com/10.1007/978-1-4899-3242-6.

[193] Ulrich Bodenhofer et al. "msa: an R package for multiple sequence alignment". In: *Bioinformatics* (Aug. 2015), btv494. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btv494. URL: http://bioinformatics.oxfordjournals.org/lookup/doi/10.1093/bioinformatics/btv494.

[194] Alexios Ghalanos and Stefan Theussl. *Rsolnp: General Non-linear Optimization Using Augmented Lagrange Multiplier Method*. 2015.

[195] J. Friedman, T. Hastie, and R. Tibshirani. "A note on the group lasso and a sparse group lasso". In: (Jan. 2010). URL: http://arxiv.org/abs/1001.0736.

[196] Robert Tibshirani. "Regression Shrinkage and Selection Via the Lasso". In: *Journal of the Royal Statistical Society, Series B* 58 (1994), pp. 267–288.

[197] R. C. Edgar. "Search and clustering orders of magnitude faster than BLAST". In: *Bioinformatics* 26.19 (Oct. 2010), pp. 2460–2461. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btq461. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btq461.

[198] S. G. Acinas et al. "PCR-Induced Sequence Artifacts and Bias: Insights from Comparison of Two 16S rRNA Clone Libraries Constructed from the Same Sample". In: *Applied and Environmental Microbiology* 71.12 (Dec. 2005), pp. 8966–8969. ISSN: 0099-2240. DOI: 10.1128/AEM.71.12.8966-8969.2005. URL: http://aem.asm.org/cgi/doi/10.1128/AEM.71.12.8966-8969.2005.

[199] Daniel Aird et al. "Analyzing and minimizing PCR amplification bias in Illumina sequencing libraries". In: *Genome Biology* 12.2 (2011), R18. ISSN: 1465-6906. DOI: 10.1186/gb-2011-12-2-r18. URL: http://genomebiology.biomedcentral.com/articles/10.1186/gb-2011-12-2-r18.

[200] Quan Peng et al. "Reducing amplification artifacts in high multiplex amplicon sequencing by using molecular barcodes". In: *BMC Genomics* 16.1 (Dec. 2015), p. 589. ISSN: 1471-2164. DOI: 10.1186/s12864-015-1806-8. URL: http://www.biomedcentral.com/1471-2164/16/589.

[201] Tomáš Větrovský and Petr Baldrian. "The Variability of the 16S rRNA Gene in Bacterial Genomes and Its Consequences for Bacterial Community Analyses". In: *PLoS ONE* 8.2 (Feb. 2013). Ed. by Josh Neufeld, e57923. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0057923. URL: http://dx.plos.org/10.1371/journal.pone.0057923.

[202] Laura A. Hug et al. "A new view of the tree of life". In: *Nature Microbiology* 1.5 (Apr. 2016), p. 16048. ISSN: 2058-5276. DOI: 10.1038/nmicrobiol.2016.48. URL: http://www.nature.com/articles/nmicrobiol201648.

[203] Scott R. Santos and Howard Ochman. “Identification and phylogenetic sorting of bacterial lineages with universally conserved genes and proteins”. In: *Environmental Microbiology* 6.7 (July 2004), pp. 754–759. ISSN: 1462-2912. DOI: `10.1111/j.1462-2920.2004.00617.x`. URL: `http://doi.wiley.com/10.1111/j.1462-2920.2004.00617.x`.

[204] Madhumita Mahalanabis et al. “Cell lysis and DNA extraction of gram-positive and gram-negative bacteria from whole blood in a disposable microfluidic chip”. In: *Lab on a Chip* 9.19 (2009), p. 2811. ISSN: 1473-0197. DOI: `10.1039/b905065p`. URL: `http://xlink.rsc.org/?DOI=b905065p`.

[205] Erik Borgström et al. “Phasing of single DNA molecules by massively parallel barcoding”. In: *Nature Communications* 6 (June 2015), p. 7173. ISSN: 2041-1723. DOI: `10.1038/ncomms8173`. URL: `http://www.nature.com/doifinder/10.1038/ncomms8173`.

[206] Riccardo Rosselli et al. “Direct 16S rRNA-seq from bacterial communities: a PCR-independent approach to simultaneously assess microbial diversity and functional activity potential of each taxon”. In: *Scientific Reports* 6.1 (Oct. 2016), p. 32165. ISSN: 2045-2322. DOI: `10.1038/srep32165`. URL: `http://www.nature.com/articles/srep32165`.

[207] T. Z. DeSantis et al. “Greengenes, a Chimera-Checked 16S rRNA Gene Database and Workbench Compatible with ARB”. In: *Applied and Environmental Microbiology* 72.7 (July 2006), pp. 5069–5072. ISSN: 0099-2240. DOI: `10.1128/AEM.03006-05`. URL: `http://aem.asm.org/cgi/doi/10.1128/AEM.03006-05`.

[208] Juan Jovel et al. “Characterization of the Gut Microbiome Using 16S or Shotgun Metagenomics”. In: *Frontiers in Microbiology* 7 (Apr. 2016). ISSN: 1664-302X. DOI: `10.3389/fmicb.2016.00459`. URL: `http://journal.frontiersin.org/Article/10.3389/fmicb.2016.00459/abstract`.

[209] Morgan G I Langille et al. “Predictive functional profiling of microbial communities using 16S rRNA marker gene sequences”. In: *Nature Biotechnology* 31.9 (Aug. 2013), pp. 814–821. ISSN: 1087-0156. DOI: `10.1038/nbt.2676`. URL: `http://www.nature.com/doifinder/10.1038/nbt.2676`.

[210] Thomas J. Sharpton. “An introduction to the analysis of shotgun metagenomic data”. In: *Frontiers in Plant Science* 5 (June 2014). ISSN: 1664-462X. DOI: `10.3389/fpls.2014.00209`. URL: `http://journal.frontiersin.org/article/10.3389/fpls.2014.00209/abstract`.

[211] Christopher L. Hemme et al. “Comparative metagenomics reveals impact of contaminants on groundwater microbiomes”. In: *Frontiers in Microbiology* 6 (Oct. 2015). ISSN: 1664-302X. DOI: `10.3389/fmicb.2015.01205`. URL: `http://journal.frontiersin.org/Article/10.3389/fmicb.2015.01205/abstract`.

[212] Susannah J Salter et al. “Reagent and laboratory contamination can critically impact sequence-based microbiome analyses”. In: *BMC Biology* 12.1 (Dec. 2014), p. 87. ISSN: 1741-7007. DOI: 10.1186/s12915-014-0087-z. URL: http://bmcbiol.biomedcentral.com/articles/10.1186/s12915-014-0087-z.

[213] D. Li et al. “MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph”. In: *Bioinformatics* 31.10 (May 2015), pp. 1674–1676. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btv033. URL: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv033.

[214] E W Myers et al. “A whole-genome assembly of Drosophila.” In: *Science (New York, N.Y.)* 287.5461 (Mar. 2000), pp. 2196–204. ISSN: 0036-8075. URL: http://www.ncbi.nlm.nih.gov/pubmed/10731133.

[215] B. Haider et al. “Omega: an Overlap-graph de novo Assembler for Metagenomics”. In: *Bioinformatics* 30.19 (Oct. 2014), pp. 2717–2722. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btu395. URL: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu395.

[216] Anton Bankevich et al. “SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing”. In: *Journal of Computational Biology* 19.5 (May 2012), pp. 455–477. ISSN: 1066-5277. DOI: 10.1089/cmb.2012.0021. URL: http://online.liebertpub.com/doi/abs/10.1089/cmb.2012.0021.

[217] Derrick E Wood and Steven L Salzberg. “Kraken: ultrafast metagenomic sequence classification using exact alignments”. In: *Genome Biology* 15.3 (2014), R46. ISSN: 1465-6906. DOI: 10.1186/gb-2014-15-3-r46. URL: http://genomebiology.biomedcentral.com/articles/10.1186/gb-2014-15-3-r46.

[218] Lorian Schaeffer et al. “Pseudoalignment for metagenomic read assignment”. In: (Oct. 2015). URL: http://arxiv.org/abs/1510.07371.

[219] Yu-Wei Wu, Blake A. Simmons, and Steven W. Singer. “MaxBin 2.0: an automated binning algorithm to recover genomes from multiple metagenomic datasets”. In: *Bioinformatics* 32.4 (Feb. 2016), pp. 605–607. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btv638. URL: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv638.

[220] Dongwan D Kang et al. “MetaBAT, an efficient tool for accurately reconstructing single genomes from complex microbial communities.” In: *PeerJ* 3 (2015), e1165. ISSN: 2167-8359. DOI: 10.7717/peerj.1165. URL: http://www.ncbi.nlm.nih.gov/pubmed/26336640%20http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4556158.

[221] Karthik Anantharaman et al. "Thousands of microbial genomes shed light on interconnected biogeochemical processes in an aquifer system". In: *Nature Communications* 7 (Oct. 2016), p. 13219. ISSN: 2041-1723. DOI: `10.1038/ncomms13219`. URL: `http://www.nature.com/doifinder/10.1038/ncomms13219`.

[222] Christopher T. Brown et al. "Unusual biology across a group comprising more than 15% of domain Bacteria". In: *Nature* 523.7559 (June 2015), pp. 208–211. ISSN: 0028-0836. DOI: `10.1038/nature14486`. URL: `http://www.nature.com/doifinder/10.1038/nature14486`.

[223] J. A. Frank et al. "Improved metagenome assemblies and taxonomic binning using long-read circular consensus sequence data". In: *Scientific Reports* 6.1 (July 2016), p. 25373. ISSN: 2045-2322. DOI: `10.1038/srep25373`. URL: `http://www.nature.com/articles/srep25373`.