On some statistical problems in single-cell transcriptome data analysis

Thesis submitted for the degree of Doctor of Philosophy in Statistics

by

Pronoy Kanti Mondal

under supervision of
Prof. Indranil Mukhopadhyay

Indian Statistical Institute
203 B T Road, Kolkata 700108, India
2021

Preface

This thesis is being submitted to fulfill the primary requirement for the degree of Doctor of Philosophy in Statistics at the Indian Statistical Institute.

The work has been carried out in the Human Genetics Unit, Biological Sciences Division, Indian Statistical Institute, Kolkata. Financial assistance was provided by Research Scholar Fellowship given by Indian Statistical Institute.

I would like to convey my sincere gratitude to my supervisor Prof. Indranil Mukhopadhyay. Without his constant support and guidance, this work would never have been complete. He was highly patient towards listening to my mistaken ideas and provided valuable suggestions to improve the quality of my work. He taught me not to compromise with the quality of research irrespective of obstacles I face, either academically or non-academically. I consider myself lucky to get the opportunity to work under him.

I would also like to convey heartfelt thanks to Professor Terry P. Speed, who provided me valuable suggestions during the development of some of the models described in this thesis.

Finally, I would like to thank my family and friends for their active cooperation and support, without which I could not pursue my research.

Much of this work was done during COVID 19 pandemic. While I am writing this, the world is not still fully operational. Students who are least susceptible to the disease are most affected by the pandemic due to lockdown. I would like to dedicate this work to students whose study has been affected by the pandemic, especially those who do not have access to online education.

Human Genetics Unit
Indian Statistical Institute
2021

Pronoy Kanti Mondal

Contents

Chapter 1: Introduction

1.1 To begin with

With the advent of next-generation sequencing (NGS) technology [126], huge amount of data are being generated regularly, using massively parallel sequencing. One of the applications of NGS technology is to profile the whole transcriptome data at single-cell level [33], commonly known as 'single-cell RNA sequencing' (scRNA-seq) data. Thousands to millions of cells can be profiled in a single experiment with the transcriptome consisting of tens of thousands of genes. This new technology is both time and cost-efficient from the technological point of view. However, although the data are expected to be much more informative than bulk RNA-seq data, they come with a few challenges like sparsity, heterogeneity, presence of noise from different sources etc. To draw proper inference from these data, researchers need to get equipped with proper statistical and computational tools that everyone can use. This dissertation aims in developing new and powerful statistical methods to analyze scRNA-seq data with more precision and efficiency. We investigate four important statistical problems in single-cell transcriptome data analysis.

Single-cell profiling can be applied to a wide range of experiments to discover biological mechanisms underlying different phenomena ranging from development to disease progression, neurology, immunology, digestive system, reproduction, organoid development, to name a few [114]. In a bulk RNA-seq experiment, a population of cells from a tissue is sequenced together, and as a result, cell to cell variability is lost. Capturing tissue heterogeneity is important in many applications necessitating the use of single-cell profiling. Moreover, information at the cellular level may help us in better pathogenesis of disease and precision medicine. On the other hand, bulk sequencing averages out cellular level expression values over a population of cells. This may introduce the erroneous effect of combining multiple subgroups known as Simpson's paradox. It is also possible that we miss subgroup-specific effects by combining cell subgroups, especially characteristics of rare subgroups.

Single-cell transcriptome profiling is becoming more and more popular these days because of the high resolution of the experiment leading to more information content in the data.

We have investigated four major problems for analyzing single-cell transcriptome data. Existing methods do not address the typical characteristics of scRNA-seq data leading to loss of information and compromise in accuracy. We have developed statistical methods taking into account the special features of the data and evaluate the performance of our proposed methods both theoretically and computationally. We have done rigorous data analysis using both simulated and real data. Our work also provides guidance to the end users regarding the appropriate applicability of the methods with respect to the arising situations and data characteristics.

A substantial amount of sparsity poses considerable challenges in modeling the distribution of scRNA-seq data. Zero expression values present in the data are called dropouts. Because of dropouts, the dataset becomes nonlinear in structure. The proportion of zeros may vary from gene to gene as well as from cell to cell. These zero values can arise from three sources: biological zeros, technical zeros, and sampling zeros. When the true underlying expression value is zero, and the corresponding data point shows zero value, it can be called 'biological zero'. Zeros may also arise from technical errors because tiny amount of RNA is present in a single cell, and cDNA may be lost during amplification. Zeros arising from this source can be attributed to 'technical zeros'. Even if the cDNA is present in the final sample, it may not be detected due to random detection leading to 'sampling zeros'. Differentiating these three sources of zeros is vital because biologists are interested only in zeros arising from biological sources. This large occurrence of zero values makes this data defiant to standard data analysis methods. For example, principal component analysis (PCA) is not appropriate for this data due to nonlinearity. Nonlinear methods like gaussian process latent variable modeling (GPLVM) [68] or t-stochastic neighbor embedding (tSNE) [76] produces a more meaningful result in dimensionality reduction for single-cell RNA-seq data. An option to tackle with these zeros is to impute the zero values based on non-zero observations. However, existing studies show that considering the zero values instead of imputation produces better results [48].

Another important characteristic that distinguishes single-cell expression data is the existence of outliers. Due to the high amount of amplification from very little starting material, some genes may be over-amplified, resulting in outliers. In addition to that, high variability in gene expression is also a characteristic that needs to be taken care of.

1.2 Modeling scRNA-seq data

Understanding the typical characteristics of scRNA-seq data is challenging because there are many sources of variations affecting the expression level of a gene in a given cell. The gene-specific effect, as well as cell-specific effect, can influence cell-specific gene expression. Both types of effects can even be responsible for dropouts. Gene-specific effects behind dropout can be classified as biological zeros, whereas cell-specific effects behind dropouts can be called technical zeros. Sampling zeros can occur when a gene is not detected because it is present at a shallow level in a sample. Another important characteristic of single-cell gene expression is that some gene expressions show a bimodal pattern, whereas others are unimodal. Distinguishing these two types of genes may be important to analyze the data better. We use characterization described by Robertson and Fryer [97] to distinguish these two types of genes. While developing our model for fitting a probability model to scRNA-seq data, we have taken into account the presence of outliers, high variability, and also heteroskedasticity in expression level.

We model the dropout event in single-cell gene expression data with a probit model with additive effects from genes and cells. We assume that gene-specific effects are fixed, whereas cell-specific effects are random. To estimate the parameters of this model, we use iteratively re-weighted least squares (IRLS) method along with EM algorithm. To estimate the factor behind sampling zeros, we use genes with similar zero proportions instead of single genes. Given a gene is expressed, we assume that the log-expression value consists of additive effects from cells and genes.

1.3 Testing for differential expression

Once the distribution of gene expression is modelled appropriately, we move on to perform differential expression analysis between two conditions. Differential expression analysis on single-cell data faces many unique challenges, e.g., heteroskedasticity between conditions, zero-inflated distribution, existence of both bimodal and unimodal genes, etc. Existing methods like DESingle [81], SC2P [131], MAST [39], etc., attempt to fit zero-inflated negative binomial, a mixture of Poisson and lognormal and zero-inflated lognormal distributions, respectively, for differential expression analysis. A common pitfall is that they do not try to distinguish between technical zeros and biological zeros because differences in only biological zeros should be compared in differential expression testing. Also, other methods do not clearly illustrate how their assumed distribution compares with the observed data. We propose two statistical tests for performing differential expression between groups, namely

RIBBON I and RIBBON II. RIBBON I is based on the ideology of testing for equality of both mean and variance for normal distribution based on a likelihood ratio test. RIBBON II is based on the ideology of testing equality of mixing proportions in mixture normal distribution between two groups. We have also derived the asymptotic distributions of our proposed test statistics. Under some weak conditions, we have proved that RIBBON I and RIBBON II asymptotically follow χ^2_3 and χ^2_2 distributions, respectively. Our proposed methods seem to outperform other existing methods on simulated data and benchmarking real datasets. To be honest with other methods, we compare them using simulations protocols as devised by others like MFA [20] and scDD [60] along with our own protocol. Cell cycle data from Buettner et al. [16] were used for benchmarking.

1.4 Pseudotime reconstruction

Single cells collected at one time point contain information of cells that are at different stages of expression with respect to time. Placing each cell on a hypothetical time trajectory by deconvoluting the information collected at one time point is a challenging task. However, uncovering such hypothetical timeline, called 'pseudotime trajectory' is a major key for many downstream analyses like transcriptional bursting detection, identifying important genes at different stages of cell regulation etc. Hence, the problem is to construct an ordering of cells to represent the underlying biological procedure properly. All single-cell transcriptome data cannot be clearly classified into discrete subgroups. Based on cell capture procedure, sometimes there is a continuous path of cells between two distinct cell types. The construction of pseudotime trajectory is appropriate for this kind of single-cell dataset. Another related problem is to construct branching based on the transcriptome data. Existing methods like Monocle [94], Slingshot [110], DPT [45], scVelo [12], Waterfall [103], TSCAN [55] etc. apply curve-fitting on reduced dimensional data. This includes reducing very high-dimensional data into two or three dimensions. However, the dimensionality reduction step may cause loss of information, leading to erroneous inference from the data. Based on our simulations, we have observed that these methods may often fail drastically to perform when the patterns of change in gene expression vary between genes.

We propose PseudoGA, a cell pseudotime reconstruction method based on genetic algorithm. We view the pseudotime reconstruction problem as finding the best permutation based on a cost function. We define the cost function as the sum of BIC values for all genes after fitting a smooth curve on the given permutation. The smooth curve is fitted based on the rank of genes, and hence the method is fully nonparametric. The search space being

too large to find an exact solution, we apply a genetic algorithm to find a near-optimal solution. We also develop an algorithm for performing branching with different clusters by using a method similar to a minimal spanning tree. To tackle datasets with a large number of cells, we first construct an ordering based on a subset of cells and extend the ordering to all cells based on nearest neighbor matching. To draw a more accurate inference, we fit the principal curve on a set of orderings obtained from different subsets sampled from the full dataset. Our algorithm shows good accuracy and robustness on benchmarking real data as well as simulated data. For real data comparison, we use five different datasets with known pseudotime information. Our method is scalable with respect to time and memory as well and amenable to parallel computing.

1.5 Batch effect correction

The next problem we consider is removing batch effects from two scRNA-seq data and identification of common cell types. This helps in integrating multiple single-cell transcriptome data. Among existing methods, SMNN [132] uses an anchoring-based approach where mutual neighbors are identified across datasets and the datasets are thereafter batch corrected using linear models. A drawback of this approach is that the batch effect may not be linear across datasets. Moreover, these methods inherently assume that biological variability and variability due to batch effect are independent. However, this assumption may not always hold in reality. Biological differences and technical effects are often interspersed. A different approach in Seurat [15] uses dynamic time warping following canonical correlation analysis. When the batch effect present between datasets is comparatively larger than the difference between clusters, this approach may not work well.

To address this problem, we identify common cell types across datasets first and then perform batch effects correction. First, we perform clustering of individual datasets and hence derive cell types for each of the datasets. Next, we apply an algorithm called Batch Corrected Gaussian Process Latent Variable modeling (BC-GPLVM) to take care of nonlinear batch effects correction across datasets. We find common cell types across datasets based on the clustering membership obtained through the reduced dimensional data. Based on the common cell types, we perform batch effects correction across datasets. This helps in identifying nonlinear batch effects as well as cluster-specific batch effects, i.e., the interaction between clusters and batches. Our proposed algorithm shows promising accuracy on both simulated and real datasets. To compare the accuracy of batch effects correction produced by different methods, we use transfer entropy [9] as a metric for the goodness of

mixing. Our method is also time and memory-efficient.

We have addressed a few problems that are at the core of scRNA-seq data analysis. Our proposed methods are novel, powerful and robust. Performances of these methods are very promising and hence can be used for further downstream analysis. We believe that our newly developed methods and algorithms will help statisticians as well as biologists in drawing appropriate inferences from scRNA-seq data.

1.6 Publication from the thesis

1. From Chapter 4: Pronoy Kanti Mondal, Udit Surya Saha, Indranil Mukhopadhyay (2021). PseudoGA: cell pseudotime reconstruction based on genetic algorithm. *Nucleic Acids Research* 2021 Jul 9; gkab457. doi: 10.1093/nar/gkab457. Online ahead of print.

Chapter 2: Modeling scRNA-seq expression data

2.1 Introduction

Rapid advances in technology have ushered a new era of getting an in-depth view of gene expression profiles from millions of cells [113, 52, 102, 91, 88]. Proper characterization of the distribution of gene expression at the single-cell level is necessary to address analytical problems and other downstream analyses relevant to these data. No existing statistical model is known to outperform others universally, across all platforms and species, bringing up the necessity to revisit the problem and developing a concrete statistical framework for analyzing such data. Gene expression is influenced by several biological and environmental factors that account for its overall variability. It is known that transcriptional bursts, cellular heterogeneity, stochastic variability, fluctuations due to gene co-expression networks, etc., have a significant effect on gene regulation leading to high variance of such data [73, 118, 128, 117, 34]. However, these variations are generally masked in bulk expression data that provides average value over a population of cells. The averaging, in effect, possibly introduces erroneously combining subgroups and leads to loss of information, a phenomenon known as 'Simpson's paradox'. While single-cell gene expression data hold promise to decipher the regulating factors behind biological mechanisms, its variability and dynamics of gene expression profile at the cellular level, pose considerable challenges in its analysis [96, 64].

RNA-seq is a technology widely used over the last several years to measure the amount of RNA present in a sample at a given time. Methods of analyzing bulk RNA-seq data are not intended to be applied to analyze single-cell RNA-seq data [53, 108]. The limitation is because those methods do not take care of cell-to-cell heterogeneity and other features of gene expression distribution that are specific to single-cell transcriptome data. Profiling

low amounts of mRNA at single-cell level requires amplification by more than a million-fold to be detected, which leads to nonlinear distortions of transcript abundance level. Additionally, each cell may lie in different biochemical states [118], and its effect on individual gene expression profiles need to be taken care of properly to analyze any scRNA-seq data. Along with this inherent stochasticity at individual cells, extreme sparsity is another typical feature usually absent in bulk RNA-seq data [11].

The most essential and typical feature of scRNA-seq is the 'dropout' event indicating a situation where a gene might be expressed in one cell while it fails to be detected in another one [95]. This behavior may occur due to two reasons. A gene may not be expressed due to intrinsic biological factors or expressed at a level too low to be detected. We call the 'zero' value due to this reason as 'biological zero'. Transcriptional bursting [40, 62] is known to cause an on and off switch in gene expression level in individual cells, which might be another determining factor of biological zeroes. On the other hand, there might be several technical reasons leading to failure in the amplification of mRNA in one cell. For example, transcripts from a particular gene may be missed or under-amplified during reverse transcription and cDNA amplification [129, 39]. This fact also gives rise to a 'zero' value in the data, and we designate this as 'technical zero'. The dropout probability in a cell also depends on the absolute expression level and the library quality of that cell, among many other factors.

Usually, existing approaches use hurdle model [39, 80], zero-inflated model [58, 81], generalized linear model [127], generalized additive model [117] or mixture model [60, 36, 131]. Most of these methods assume a common cause of zero expression at individual cells. However, only biological zeros are of interest to researchers in typical analyses as technical zeros are usually considered as noise. So, these two well-known types of sources of zeros need to be characterized differently to analyze the data better. We would be able to analyze scRNA-seq data more efficiently if we can separate technical artefacts that restrain its precision. None of the methods addresses how to differentiate between these two causes of zero expression clearly. The literature also lacks studies that evaluate the goodness of fit for different distribution assumptions on single-cell gene expression data.

We propose a novel statistical model for the distribution of gene expression in single-cell RNA-seq data using a two-part model [37]: one accounting for zero expressions and the other accounting for the positive part. The technical and biological factors behind the zero expressions are separated using an additive model with a probit link. Log transformed positive expression values are fitted with either a mixture of two normal distributions or unimodal normal distribution depending on the characteristic of individual genes. We also consider cell-specific stochasticity that may arise due to the transcription state, quality

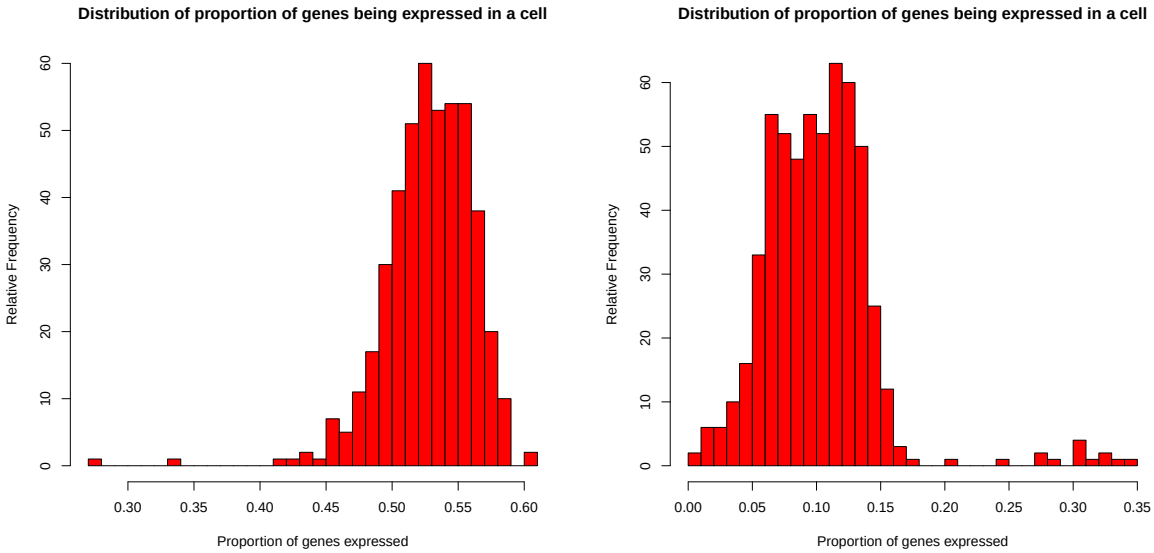
of library of that cell, and other biological reasons like different cell types etc. It may affect the expression values of genes and also the dropout events. Our proposed model takes into account several factors that control the gene expression in cells. We estimate the parameters of the model using EM algorithm.

2.2 Characteristics of single-cell RNA-seq data

To develop an appropriate model for single-cell gene expression profiles, one should consider few aspects that are typical for scRNA-seq data. A characteristic that distinguishes single-cell expression data from other gene expression data is the high abundance of zeros. The occurrence of zeros can be attributed to several factors: biological zeros, sampling zeros, and technical zeros [105]. A gene may not be expressed in some cells due to an underlying biological factor resulting in biological zeros. Due to the stochastic nature of count datasets, some genes may not show positive expression because the number of transcripts present in the cell is deficient, and RNA-seq protocol cannot detect the transcript due to sampling inefficiency. Another type of zero can arise purely due to technical reasons. Because of poor library preparation, capture inefficiency, or other technical reasons, some genes may be missed in a particular cell.

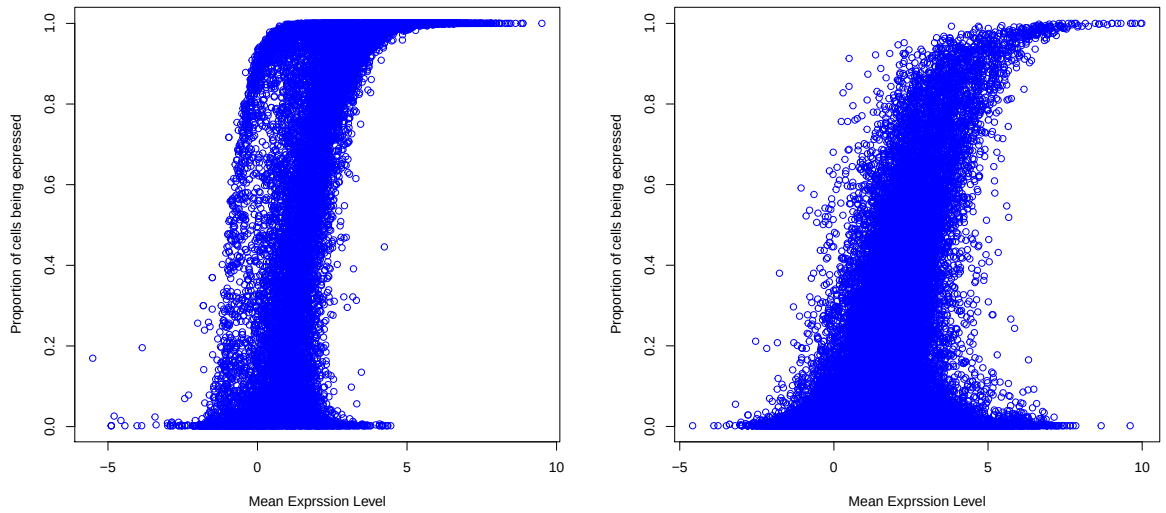
Based on two datasets (GEO accession number: GSE64016, GSE75688), a study of histograms reveals that only a tiny fraction of genes within the transcriptome is detected in a typical cell (Figure 2.1). A large proportion of genes that are not expressed assumes the value '0' in the dataset. The proportion of genes expressed in a particular cell is highly variable, and this can be attributed to variability due to technical reasons. A fact that is often ignored in existing studies is the relationship between the proportion of cells being expressed for a gene with the mean log expression level. The scatter plots (Figure 2.2) of mean gene expression level with the proportion of cells showing expression on the same datasets reveal that, the proportion of cells in which a gene is expressed is an increasing but non-linear function of mean expression level. As mean expression decreases, the proportion of cells containing expressed genes also decreases, giving rise to too many zeros due to amplification failure. Variability explained by this relationship can be attributed to variability due to dropouts from sampling. Our task is to discover true biological variability after removing the effects of technical variability and variability due to sampling.

Another essential characteristic that distinguishes scRNA-seq data from bulk RNA-seq data is the fact that single-cell expression distribution may be both bimodal or unimodal



(A) **(B)**

Figure 2.1: Histogram of the proportion of genes expressed in a cell shows that the proportion of expressed genes in a cell has very high variability and a cell-specific factor for dropouts should be considered in scRNA-seq datasets with GEO acession number: (A) GSE64016 (B) GSE75688.



(A) **(B)**

Figure 2.2: The proportion of cells where a gene is expressed is an increasing function of mean log expression level. This relationship can be exploited to estimate the factor behind sampling zeros in scRNA-seq datasets with GEO acession number: (A) GSE64016 (B) GSE75688.

[60, 82]. Due to transcriptional bursting, a gene might go through on and off switching that leads to the oscillation in transcription levels. For this reason and also due to some other technical and biological factors, a gene may not be expressed or expressed at a very low level. So the distribution of gene expression generally has two modes: one mode at zero or at a point very close to zero and another mode slightly away from zero. Moreover, the presence of large outliers makes the modeling more challenging. These features are typical of single-cell RNA-seq data. However, there may be some genes that do not show bimodality in the distribution of expression profile. We apply the criterion proposed by Holzmann et al. [47] to check the presence of bimodality. We fit a two-component Gaussian mixture model on log expression level. If $F(x;\theta)$ is the distribution function of X, the log expression level of a gene, according to Robertson and Fryer [97],

$$F(x) \text{ is } \begin{cases} \text{unimodal if} & 0 < \mu \leq \mu_0 \\ \text{bimodal if} & \mu > \mu_0, p \in (p_1, p_2) \end{cases}$$

where $\mu = \frac{|\mu_2 - \mu_1|}{\sigma_1}$ and $\mu_0 = \left\{\frac{2(\sigma^4-\sigma^2+1)^{\frac{3}{2}} - 2\sigma^6 - 3\sigma^4 - 3\sigma^2 + 2}{\sigma^2}\right\}^{\frac{1}{2}}$. Note that here, for $i = 1, 2$, $p_i^{-1} = 1 + \frac{\sigma^3 y_i}{\mu - y_i} \exp\{-\frac{1}{2}y_i^2 + \frac{1}{2}\left(\frac{y_i - \mu}{\sigma}\right)^2\}$, where y_1 and y_2 are roots of the equation $(\sigma^2 - 1)y^3 - \mu(\sigma^2 - 2)y^2 - \mu^2 y + \mu\sigma^2 = 0$ with $0 < y_1 < y_2 < \mu$ and $\sigma = \frac{\sigma_2}{\sigma_1}$ where (μ_1, μ_2) and (σ_1^2, σ_2^2) are the mean and variance parameters respectively for the two component distribution and p is the mixing proportion.

If we further assume $\sigma_1 = \sigma_2$, the condition simplifies to the fact that the distribution is unimodal if and only if $d \leq 1$ or $|\log(1-p) - \log(p)| \geq 2\log(d - \sqrt{d^2 - 1}) + 2d\sqrt{d^2 - 1}$ where $d = \frac{|\mu_1 - \mu_2|}{2\sqrt{\sigma_1 \sigma_2}}$ [47]. Based on the criterion assuming equal variance, we tested for bimodality (Figure 2.3) in two independent datasets (GSE accession numbers: GSE64016 and GSE75688). We observe that 47% and 68% genes are bimodal in GSE64016 and GSE75688 datasets respectively. So it is necessary to consider both unimodal as well bimodal distribution while modeling single-cell expression data.

Also, single-cell data show high variability, which should be taken care of in the model correctly. In addition to that, the model should also absorb outlier expression levels present. This restriction can be taken care of, to some extent, by fitting Gaussian distribution to log expression levels because log transformation is well known to decrease the skewness of any data. Another critical issue with single-cell data is that not all cells are of the same quality. Even after filtering out low-quality cells, all cells may not exhibit the same characteristic. Very often, there are rare cell types and cell subpopulations. One way to resolve this is by considering cell-specific random effects on expression profiles for every gene. We took

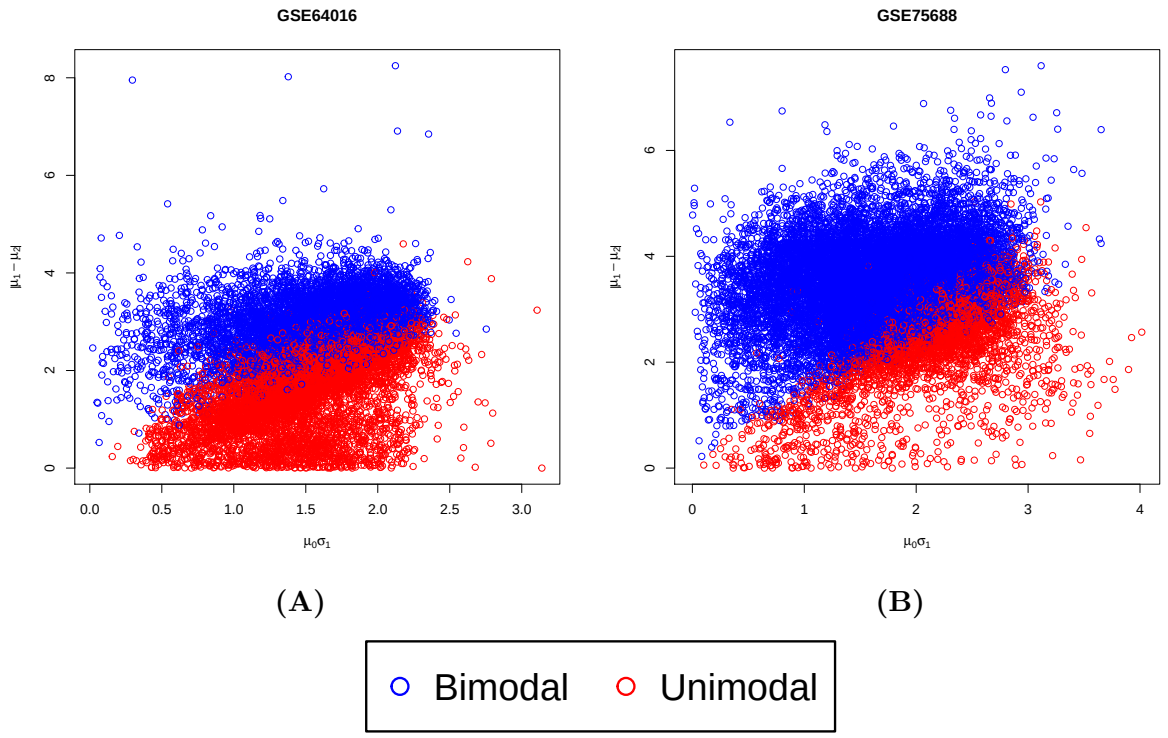


Figure 2.3: In both datasets, some genes are unimodal and others are bimodal. So, consideration of bimodal distribution is necessary.

all these characteristics into account while building a model named 'RIBBON', for gene expression profiles.

2.3 RIBBON

Some methods, developed to analyze single-cell RNA-seq data, use read counts to fit appropriate distribution on the data. However, though gene-level expression data are produced in read counts, transcript-level expressions are usually of continuous data type. In the case of UMI data or other types of expression data, expression levels of genes might be non-integer valued. Moreover, discrete read counts can easily be normalized into continuous data [1] by converting into FPKM (Fragments per kilobase per million) [83] or CPM (Counts per million) [31] level. So a model for gene expression with continuous distribution has wider applicability than a model for read counts.

Individual cell effect influences the expression level. Dropouts in a typical cell are in abundance. We propose a mixed model approach for overall expression levels assuming that the effects of genes are fixed whereas the effects of cells are random. This approach takes care of cell-specific effect on expression level due to the quality of library and effect of subgroups. We also assume random cell effects for dropout events as different cells might possess different rates for technical dropouts. We presume that the logarithm of the expression level of each gene is either unimodal or bimodal with different means and possibly unequal variances. The log transformation and random cell effect for every cell can take care of large outlier expression values. We assume that the probability of dropout in each cell follows a probit model with additive effects from cells and genes.

It is well-known that technical zeros and biological zeros are confounded in single-cell transcriptome data. No existing method addresses separating these two sources of zeros bringing about unwanted variation present in the data. In the previous section, we have described how RIBBON takes care of three types of sources of zeros. In our model, we aim to distinguish these three sources of zeros. Some works [39, 58, 81, 36, 131] modeled single-cell gene expression data with a two-component model, assuming that zero expression can only arise from dropout events.

To formalize our proposed model, first, assume that the distribution is either unimodal or bimodal normal. Let y_{ij} be the logarithm expression level of the j-th gene in cell i and E_{ij} be the event that gene j is expressed in cell i. We consider another variable z_{ij} that indicates the observed presence of gene expression value for j-th gene in i-th cell. Using the criterion described before, we model gene-specific expressions, either unimodal

or bimodal. Conditional on the fact that the gene expression distribution is bimodal, we introduce another indicator variable D_{ij}, a latent variable. $D_{ij} = 0$ means expression level of gene j in cell i comes from mode 1, otherwise from mode 2. To incorporate two different types of zeros, we assume that the probability that j-th gene is expressed in cell i is $\Phi(c_j)$ where c_j is the gene-specific fixed effect for dropout and $\Phi(x)$ is the distribution function of a standard normal variable. c_j is a gene-specific parameter accounting for zeros and hence it can be identified as the biological factor behind zeros. Once a gene is expressed in a cell, the probability that there is no technical error in that cell for the given gene is $\Phi(c + \mu_j + \alpha_{1i})$. Here, α_{1i} is the cell-specific random effect, μ_j is the overall mean in case of unimodal distribution, and it is the lower mode in bimodal distribution. α_{1i} is a cell-specific parameter and may depend on experimental conditions and hence can be called the technical factor behind zeros. The parameter c influences several genes equally across all cells and so the factor behind sampling zeros can be captured through this parameter.

With all these notations defined above, we first present our model in a diagrammatic way (Figure 2.4), followed by the mathematical presentation.

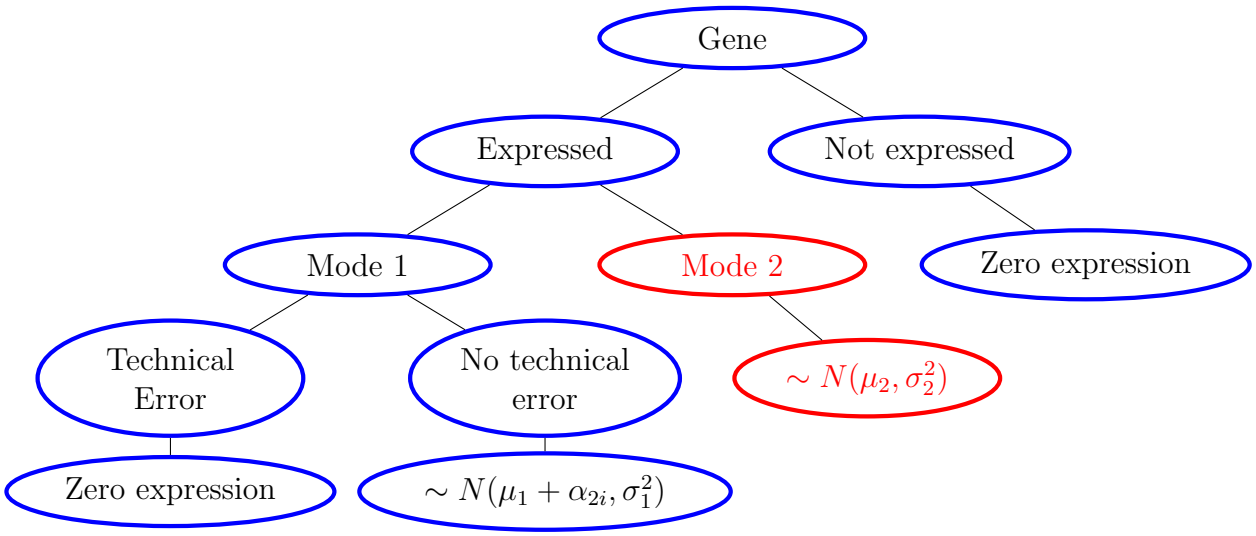


Figure 2.4: RIBBON model under bimodality assumption

$$Pr[E_{ij} = 1] = \Phi(c_j), P[D_{ij} = 1] = \pi_j, (z_{ij}|E_{ij} = 0) \equiv 0,$$

$$
\begin{aligned}
&Pr(z_{ij}=1|D_{ij},E_{ij}=1)=\Phi(c+\alpha_{1i}+\mu_{1j})I(D_{ij}=0)+I(D_{ij}=1),\\
&(y_{ij}|z_{ij}=0)\equiv 0,\\
&(y_{ij}|z_{ij}=1,D_{ij}=0)\sim N(\alpha_{2i}+\mu_{1j},\sigma^2_{1j}),\\
&(y_{ij}|z_{ij}=1,D_{ij}=1)\sim N(\mu_{2j},\sigma^2_{2j}),\\
&\alpha_{1i}\sim N(0,\tau_1^2),\ \ \alpha_{2i}\sim N(0,\tau_2^2)
\end{aligned}
\tag{2.1}
$$

We assume that the true expression levels of all cells for a given gene belong to one of the two modes. A cell belonging to any of the two modes is silenced due to underlying biological factor with a fixed probability leading to $E_{ij}=0$. Technical zeros, however, can occur only from the lower mode. So we assume D_{ij} and E_{ij} are independent. Now, we have,

$$
\begin{aligned}
&P[z_{ij}=1|D_{ij}=0]=\tfrac{P[z_{ij}=1,D_{ij}=0]}{P[D_{ij}=0]}=\tfrac{P[z_{ij}=1,D_{ij}=0,E_{ij}=1]+P[z_{ij}=1,D_{ij}=0,E_{ij}=0]}{P[D_{ij}=0]}\\
&=\tfrac{P[z_{ij}=1,D_{ij}=0,E_{ij}=1]}{P[Dij=0]}=\tfrac{P[E_{ij}=1]P[D_{ij}=0|E_{ij}=1]P[z_{ij}=1|D_{ij}=0,E_{ij}=1]}{P[D_{ij}=0]}\\
&=\tfrac{\Phi(c_j)(1-\pi_j)\Phi(c+\alpha_{1i}+\mu_j)}{(1-\pi_j)}=\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)\\
&P[z_{ij}=1|D_{ij}=1]=\tfrac{P[z_{ij}=1,D_{ij}=1]}{P[D_{ij}=1]}=\tfrac{P[z_{ij}=1,D_{ij}=1,E_{ij}=1]+P[z_{ij}=1,D_{ij}=1,E_{ij}=0]}{P[D_{ij}=1]}\\
&=\tfrac{P[z_{ij}=1,D_{ij}=1,E_{ij}=1]}{P[D_{ij}=1]}=\tfrac{P[E_{ij}=1]P[D_{ij}=1|E_{ij}=1]P[z_{ij}=1|D_{ij}=1,E_{ij}=1]}{P[D_{ij}=1]}\\
&=\tfrac{\Phi(c_j)\pi_j.1}{\pi_j}=\Phi(c_j)
\end{aligned}
$$

Under independence of E_{ij} and D_{ij}, note that $(y_{ij}|D_{ij},z_{ij},E_{ij})\equiv(y_{ij}|D_{ij},z_{ij})$. So eliminating E_{ij} the model (2.1) can be written as:

$$
\begin{aligned}
&P[D_{ij}=1]=\pi_j, P[z_{ij}=1|D_{ij}=1]=\Phi(c_j),\\
&P[z_{ij}=1|D_{ij}=0]=\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}),\\
&(y_{ij}|z_{ij}=0)\equiv 0,\\
&(y_{ij}|z_{ij}=1,D_{ij}=0)\sim N(\alpha_{2i}+\mu_{1j},\sigma^2_{1j}),\\
&(y_{ij}|z_{ij}=1,D_{ij}=1)\sim N(\mu_{2j},\sigma^2_{2j}),\\
&\alpha_{1i}\sim N(0,\tau_1^2),\ \ \alpha_{2i}\sim N(0,\tau_2^2)
\end{aligned}
\tag{2.2}
$$

The likelihood function for gene j is:

$$
\begin{aligned}
&L(\theta_j|\{D_{ij}\}_{i=1}^n,\{z_{ij}\}_{i=1}^n,\{y_{ij}\}_{i=1}^n)\\
&=\prod_{i=1}^n\Big\{\pi_j^{D_{ij}}(1-\pi_j)^{(1-D_{ij})}[(\Phi(c_j))^{z_{ij}}(1-\Phi(c_j))^{(1-z_{ij})}(\phi(y_{ij};\mu_{2j},\sigma_{2j}))^{z_{ij}}]^{D_{ij}}\\
&\times\big[(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))^{z_{ij}}(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))^{(1-z_{ij})}
\end{aligned}
$$

$$\times \left(\phi(y_{ij}, \mu_{1j} + \alpha_{2i}, \sigma_{1j})\right)^{z_{ij}}\Big]^{(1-D_{ij})} \phi(\alpha_{1i}; 0, \tau_1)\phi(\alpha_{2i}; 0, \tau_2)\Bigg\}$$

where $\theta_j = (\pi_j, c_j, \mu_{1j}, \mu_{2j}, \sigma^2_{1j}, \sigma^2_{2j}, c, \tau_1^2, \tau_2^2)'$ and $\phi(x; \mu, \sigma)$ is the p.d.f. of a $N(\mu, \sigma^2)$, and $\Phi(x)$ is the distribution function of a standard normal variable.

For genes showing unimodal trait, we assume ($D_{ij} \equiv 0$) in this model for all observations i.e. we drop the larger mode (red part of the diagram in Figure 2.4). A diagrammatic presentation of our proposed model when the distribution is unimodal is given in Figure 2.5 followed by a mathematical representation.

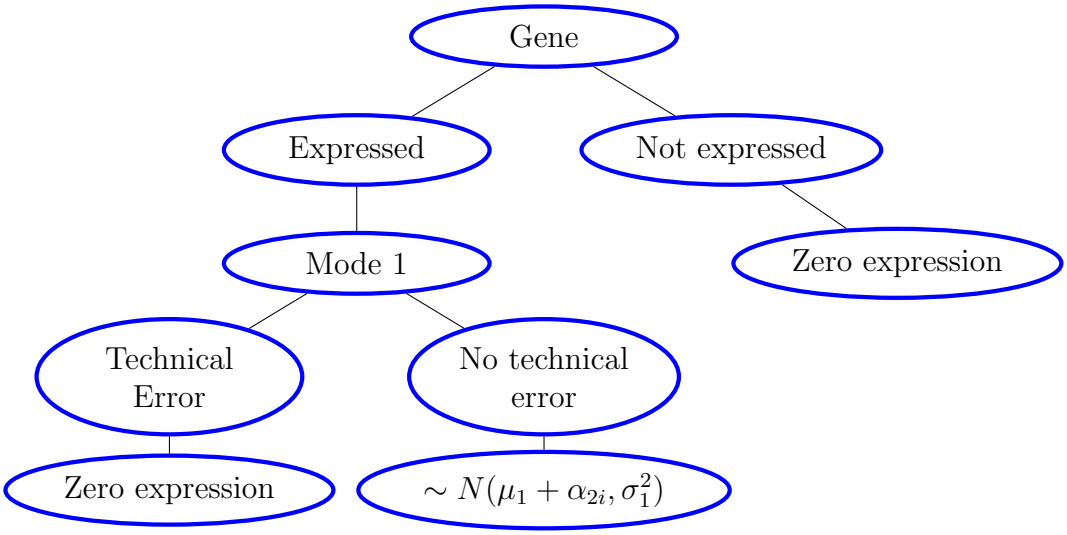


Figure 2.5: RIBBON model under unimodality assumption

$$\begin{aligned}
&Pr[E_{ij} = 1] = \Phi(c_j), (z_{ij}|E_{ij} = 0) \equiv 0,\\
&Pr(z_{ij} = 1|E_{ij} = 1) = \Phi(c + \alpha_{1i} + \mu_j),\\
&(y_{ij}|z_{ij} = 0) \equiv 0,\\
&(y_{ij}|z_{ij} = 1) \sim N(\alpha_{2i} + \mu_j, \sigma_j^2),\\
&\alpha_{1i} \sim N(0, \tau_1^2), \;\; \alpha_{2i} \sim N(0, \tau_2^2)
\end{aligned} \tag{2.3}$$

Taking the unique characteristics of single-cell RNA-seq data into account, we have proposed a model to fit the distribution of gene expression. Our model can be extended

or modified given some particular aspects that might better explain the distribution of expression of some specific genes. Note that, the essence of the problem is the presence of various types of genes in thousands of cells, all measured by single-cell technology. Naturally, all genes cannot exhibit a similar pattern which a single model with equal efficiency can explain. However, our proposed model is a very general one to encompass different situations while modeling scRNA-seq data.

Now, we describe the estimation procedure for the parameters of the model for unimodal distribution as given by (2.3) followed by the same for bimodal distribution (2.2).

2.4 Estimation of parameters for the unimodal model

We use EM algorithm to find the MLE of the parameters in all models. For unimodal model, using (2.3), the likelihood function for gene j based on n cells is:

$$\begin{aligned}
&L(\mu_j, \sigma_j^2, c_j, \tau_1^2, \tau_2^2, c | \{z_{ij}\}_{i=1}^n, \{y_{ij}\}_{i=1}^n, \{\alpha_{1i}\}_{i=1}^n, \{\alpha_{2i}\}_{i=1}^n) \\
&= \prod_{i=1}^n (\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))^{z_{ij}} (1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))^{(1-z_{ij})} \\
&\times \prod_{i=1}^n (\phi(y_{ij}; \mu_j+\alpha_{2i}, \sigma_j))^{z_{ij}} \phi(\alpha_{1i}; 0, \tau_1)\phi(\alpha_{2i}; 0, \tau_2)
\end{aligned}$$

Hence, the log-likelihood is for the j-th gene is given by

$$\begin{aligned}
&l(\mu_j, \sigma_j^2, \tau_1^2, \tau_2^2, c_j, c | \{y_{ij}\}_{i=1}^n, \{z_{ij}\}_{i=1}^n, \{\alpha_{1i}\}_{i=1}^n, \{\alpha_{2i}\}_{i=1}^n) \\
=&\text{Constant} + \sum_{i=1}^n \Big[z_{ij} \log(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)) \\
&+ (1-z_{ij}) \log(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)) \\
&+ z_{ij}(-\frac{1}{2}\log(\sigma_j^2) - \frac{(y_{ij}-\alpha_{2i}-\mu_j)^2}{2\sigma_j^2}) \Big] - \sum_{i=1}^n \big[\frac{1}{2}\log(\tau_1^2) + \frac{\alpha_{1i}^2}{2\tau_1^2} + \frac{1}{2}\log(\tau_2^2) + \frac{\alpha_{2i}^2}{2\tau_2^2} \big]
\end{aligned}$$

Consequently the joint log-likelihood for n_G genes is given by:

$$= \text{Constant} + \sum_{j=1}^{n_G} \sum_{i=1}^n [z_{ij} \log(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))$$

$$+ (1 - z_{ij}) \log(1 - \Phi(c_j)\Phi(c + \alpha_{1i} + \mu_j)) + z_{ij}(-\frac{1}{2}\log(\sigma_j^2) - \frac{(y_{ij} - \alpha_{2i} - \mu_j)^2}{2\sigma_j^2})]$$

$$- \sum_{i=1}^{n}[\frac{1}{2}\log(\tau_1^2) + \frac{\alpha_{1i}^2}{2\tau_1^2} + \frac{1}{2}\log(\tau_2^2) + \frac{\alpha_{2i}^2}{2\tau_2^2}]$$

2.4.1 Estimation of parameters in GLM with probit link having random effects

First, we discuss briefly how we estimate parameters in a generalized linear model with mixed effects using probit link. Then we prove one lemma that is used for estimating the parameters of the unimodal model for scRNA expression data.

Consider the generalized linear model $y_i \sim Ber(\Phi(x_i^t\beta + z_i^t u)), i = 1, 2, \ldots, n$ where β and u both are fixed effects. The log-likelihood is given by:

$l(\beta, u|\{y_i\}_{i=1}^n) = \sum_{i=1}^{n}[y_i \log(\Phi(x_i^t\beta + z_i^t u)) + (1 - y_i)\log(1 - \Phi(x_i^t\beta + z_i^t u))]$

Now, $\frac{\partial l}{\partial \beta} = \sum_{i=1}^{n} x_i \frac{(y_i - \Phi(x_i^t\beta + z_i^t u))\phi(x_i^t\beta + z_i^t u)}{\Phi(x_i^t\beta + z_i^t u)(1 - \Phi(x_i^t\beta + z_i^t u))}$ and $\frac{\partial l}{\partial u} = \sum_{i=1}^{n} z_i \frac{(y_i - \Phi(x_i^t\beta + z_i^t u))\phi(x_i^t\beta + z_i^t u)}{\Phi(x_i^t\beta + z_i^t u)(1 - \Phi(x_i^t\beta + z_i^t u))}$.

Also, $E[-\frac{\partial^2 l}{\partial \beta^2}] = \sum_{i=1}^{n} x_i \frac{\phi^2(x_i^t\beta + z_i^t u)}{\Phi(x_i^t\beta + z_i^t u)(1 - \Phi(x_i^t\beta + z_i^t u))} x_i^t$ and $E[-\frac{\partial^2 l}{\partial u^2}] = \sum_{i=1}^{n} z_i \frac{\phi^2(x_i^t\beta + z_i^t u)}{\Phi(x_i^t\beta + z_i^t u)(1 - \Phi(x_i^t\beta + z_i^t u))} z_i^t$.

We estimate the parameters using iteratively re-weighted least squares (IRLS), where the parameters at the k-th step are estimated by the following equations:

$$\begin{bmatrix} X^t W^{(k-1)} X & X^t W^{(k-1)} Z \\ Z^t W^{(k-1)} X & Z^t W^{(k-1)} Z \end{bmatrix} \begin{bmatrix} \beta^{(k)} \\ u^{(k)} \end{bmatrix} = \begin{bmatrix} X^t W^{(k-1)} y^{*(k)} \\ Z^t W^{(k-1)} y^{*(k)} \end{bmatrix}$$

where $W^{(k-1)}$ is a diagonal matrix with i-th diagonal element $\frac{\phi^2(x_i^t\beta^{(k-1)} + z_i^t u^{(k-1)})}{\Phi(x_i^t\beta^{(k-1)} + z_i^t u^{(k-1)})(1 - \Phi(x_i^t\beta^{(k-1)} + z_i^t u^{(k-1)}))}$ and $y^{*(k)} = x\beta^{(k-1)} + Zu^{(k-1)} + {W^{(k-1)}}^{-1} v^{(k-1)}$ where the i-th element of the vector $v^{(k-1)}$ is given by $v_i^{(k-1)} = \frac{(y_i - \Phi(x_i^t\beta^{(k-1)} + z_i^t u^{(k-1)}))\phi(x_i^t\beta^{(k-1)} + z_i^t u^{(k-1)})}{\Phi(x_i^t\beta^{(k-1)} + z_i^t u^{(k-1)})(1 - \Phi(x_i^t\beta^{(k-1)} + z_i^t u^{(k-1)}))}$. This equation can also be thought of coming from a quasi-likelihood where $y^* \sim N(X\beta + Zu, (W^{(k-1)})^{-1})$ and $\beta^{(k)}$ and $u^{(k)}$ are MLEs of β and u respectively. Now, if we further assume u to be a random parameter following $N(0, D)$, the equations for estimation are:

$$\begin{bmatrix} X^t W^{(k-1)} X & X^t W^{(k-1)} Z \\ Z^t W^{(k-1)} X & Z^t W^{(k-1)} Z + D^{-1} \end{bmatrix} \begin{bmatrix} \beta^{(k)} \\ u^{(k)} \end{bmatrix} = \begin{bmatrix} X^t W^{(k-1)} y^{*(k)} \\ Z^t W^{(k-1)} y^{*(k)} \end{bmatrix}$$

This equation can be used for estimation for mixed effect parameters in generalized linear model. When D is unknown, one way of solving this equation is through the following steps of EM algorithm:

E-step:

$E[\hat{u}^{(k)}|y] = (Z^tW^{(k-1)}Z + (D^{(k-1)})^{-1})^{-1}(Z^tW^{(k-1)}y^{*(k)} - Z^tW^{(k-1)}X\beta^{k-1})$

M-step:

$\hat{\beta}^{(k)} = (X^tW^{(k-1)}X)^{-1}(X^tW^{(k-1)}y^{*(k)} - X^tW^{(k-1)}ZE[\hat{u}^{(k)}|y])$

$\hat{D}^{(k)} = E[\hat{u}^{(k)}\hat{u}^{(k)}|y] = (E[\hat{u}^{(k)}|y])(E[\hat{u}^{(k)}|y])^t + (Z^tW^{(k-1)}Z + (D^{(k-1)})^{-1})^{-1}$

Lemma 1. *If* $X \sim N(\mu, \sigma^2)$, $E[\Phi(a+X)] = \Phi(\frac{a+\mu}{\sqrt{1+\sigma^2}})$, *and* $E[\phi(a+X)] = \frac{1}{\sqrt{1+\sigma^2}}\phi(\frac{a+\mu}{\sqrt{1+\sigma^2}})$

Proof. Consider a standard normal variable Y independent of X, so that
$E[\Phi(a+X)] = E_X[E_{Y|X}[I(Y \leq a+X)|X]] = E_{X,Y}[I(Y \leq a+X)] = P[Y \leq a+X]$
$= P[Y - X \leq a] = P[Z \leq a]$ where $Z \sim N(-\mu, (1+\sigma^2))$
So, $E[\Phi(a+X)] = P[(\frac{Z+\mu}{\sqrt{1+\sigma^2}}) \leq (\frac{a+\mu}{\sqrt{1+\sigma^2}})] = \Phi(\frac{a+\mu}{\sqrt{1+\sigma^2}})$

Differentiating both sides of the equality w.r.t a, we have, $E[\phi(a+X)] = \frac{1}{\sqrt{1+\sigma^2}}\phi(\frac{a+\mu}{\sqrt{1+\sigma^2}})$ □

2.4.2 Estimation of parameters of RIBBON

The parameters to be estimated are $\{c_j\}_{j=1}^{n_G}, \{\mu_j\}_{j=1}^{n_G}, \{\sigma_j^2\}_{i=1}^{n_G}, \{\alpha_{1i}\}_{i=1}^{n}, \{\alpha_{2i}\}_{i=1}^{n}, \tau_1^2, \tau_2^2, c$, where n is the number of cells and n_G is the number of genes. From a single gene, it is not possible to estimate c, because c and c_j becomes confounded. Genes are clustered into subsets with similar zero occurrences and gene specific parameters within each subset are estimated simultaneously. The joint distribution of $\{\{y_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$, $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$, $\{\alpha_{1i}\}_{i=1}^{n}$ and $\{\alpha_{2i}\}_{i=1}^{n}$ can be represented as:

$$\begin{aligned} &f(\{\{y_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}, \{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}, \{\alpha_{1i}\}_{i=1}^{n}, \{\alpha_{2i}\}_{i=1}^{n}) \\ =&f(\{\{y_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}, \{\alpha_{2i}\}_{i=1}^{n}|\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}, \{\alpha_{1i}\}_{i=1}^{n})f(\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}, \{\alpha_{2i}\}_{i=1}^{n}) \end{aligned}$$

The parameters are estimated in two steps. In the first step $\{\mu_j\}_{j=1}^{n_G}$, $\{\sigma_j^2\}_{j=1}^{n_G}$, τ_2^2 and $\{\alpha_{2i}\}_{i=1}^{n}$ are estimated by considering conditional distribution on $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$,$\{\alpha_{1i}\}_{i=1}^{n}$. In the second step, the remaining parameters are estimated by considering marginal distribution of $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$,$\{\alpha_{1i}\}_{i=1}^{n}$.

Step I:

It can easily be seen that, conditioned on $z_{ij} > 0$, $y_{ij} \sim N(\mu_j + \alpha_{2i}, \sigma_j^2)$. The conditional log-likelihood given $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$ and $\{\alpha_{1i}\}_{i=1}^{n}$ can be represented as:

$$l_1(\{\mu_j\}_{j=1}^{n_G}, \{\sigma_j^2\}_{j=1}^{n_G}|\{\{y_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}, \{\alpha_{2i}\}_{i=1}^{n})$$

$$=\text{Constant} + \sum_{j=1}^{n_G}\sum_{i=1}^{n}[z_{ij}(-\frac{1}{2}\log(\sigma_j^2) - \frac{(y_{ij}-\alpha_{2i}-\mu_j)^2}{2\sigma_j^2})] - \sum_{i=1}^{n}[\frac{1}{2}\log(\tau_2^2) + \frac{\alpha_{2i}^2}{2\tau_2^2}]$$

Optimization of the log-likelihood function is obtained by equating the partial derivative of the log-likelihood function with respect to the parameters to zero.

$$\begin{aligned}
\frac{\partial l}{\partial \mu_j} = 0 &\implies \sum_{i=1}^{n}\frac{(y_{ij}-\alpha_{2i}-\mu_j)z_{ij}}{\sigma_j^2} = 0\\
\frac{\partial l}{\partial \alpha_{2i}} = 0 &\implies \sum_{j=1}^{n_G}\frac{(y_{ij}-\alpha_{2i}-\mu_j)z_{ij}}{\sigma_j^2} - \frac{\alpha_{2i}}{\tau_2^2} = 0\\
\frac{\partial l}{\partial \sigma_j^2} = 0 &\implies \sum_{i=1}^{n}[-\frac{z_{ij}}{2\sigma_j^2} + \frac{z_{ij}(y_{ij}-\alpha_{2i}-\mu_j)^2}{2(\sigma_j^2)^2}] = 0\\
\frac{\partial l}{\partial \tau_2^2} = 0 &\implies \sum_{i=1}^{n}[-\frac{1}{2\tau_2^2} + \frac{\alpha_{2i}^2}{2(\tau_2^2)^2}] = 0
\end{aligned}$$

The parameters are estimated using EM algorithm with the following steps:

E-step

We first derive the conditional distribution of α_{2i}. The joint log-likelihood of $\{y_{ij}\}_{j=1}^{n_G}$ and α_{2i} is given by

$$\begin{aligned}
&l(\alpha_{2i}, \{y_{ij}\}_{j=1}^{n_G}|\{z_{ij}\}_{j=1}^{n_G}, \{\mu_j\}_{j=1}^{n_G}, \{\sigma_j^2\}_{j=1}^{n_G}, \tau_2^2)\\
=&-\frac{\sum_{j=1}^{n_G} z_{ij}}{2}\log(2\pi) - \frac{1}{2}\sum_{j=1}^{n_G} z_{ij}\log(\sigma_j^2) - \sum_{j=1}^{n_G}\frac{(y_{ij}-\mu_j-\alpha_{2i})^2 z_{ij}}{2\sigma_j^2}\\
&-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\tau_2^2) - \frac{\alpha_{2i}^2}{2\tau_2^2}
\end{aligned}$$

Now, the conditional log-likelihood of α_{2i} given $\{y_{ij}\}_{j=1}^{n_G}$ can be expressed as:

$$\begin{aligned}
&l(\alpha_{2i}|\{y_{ij}\}_{j=1}^{n_G}, \{z_{ij}\}_{j=1}^{n_G}, \{\mu_j\}_{j=1}^{n_G}, \{\sigma_j^2\}_{j=1}^{n_G}, \tau_2^2)\\
=&constant - \Big(\sum_{j=1}^{n_G}\frac{z_{ij}}{2\sigma_j^2} + \frac{1}{2\tau_2^2}\Big)\alpha_{2i}^2 + \sum_{j=1}^{n_G}\frac{(y_{ij}-\mu_j)z_{ij}}{\sigma_j^2}\alpha_{2i}
\end{aligned}$$

$$=constant - \frac{1}{2}\Big(\sum_{j=1}^{n_G}\frac{z_{ij}}{\sigma_j^2}+\frac{1}{\tau_2^2}\Big)\Bigg(\alpha_{2i}-\frac{\sum_{j=1}^{n_G}\frac{(y_{ij}-\mu_j)z_{ij}}{\sigma_j^2}}{(\sum_{j=1}^{n_G}\frac{z_{ij}}{\sigma_j^2}+\frac{1}{\tau_2^2})}\Bigg)^2$$

So, the conditional distribution of α_{2i} is normal with conditional mean $\frac{s_{1,\alpha_{1i}}}{s_{2,\alpha_{2i}}}$ and conditional variance $\frac{1}{s_{2,\alpha_{2i}}}$ where $s_{1,\alpha_{2i}}=\sum_{j=1}^{n_G}\frac{(y_{ij}-\mu_j)z_{ij}}{\sigma_j^2}$ and $s_{2,\alpha_{2i}}=\sum_{j=1}^{n_G}\frac{z_{ij}}{\sigma_j^2}+\frac{1}{\tau_2^2}$.

At the k-th iteration, we update $\alpha_{2i}^{(k)}$ as:
$\alpha_{2i}^{(k)}=\frac{s_{1,\alpha_{2i}}^{(k-1)}}{s_{2,\alpha_{2i}}^{(k-1)}}$ with $s_{1,\alpha_{2i}}^{(k-1)}=\sum_{j=1}^{n_G}\frac{(y_{ij}-\mu_j^{(k-1)})z_{ij}}{\sigma_j^{2(k-1)}}$ and $s_{2,\alpha_{2i}}^{(k-1)}=(\sum_{j=1}^{n_G}\frac{z_{ij}}{\sigma_j^{2(k-1)}}+\frac{1}{\tau_2^{2(k-1)}})$.

M-step

$\mu_j^{(k)}=\frac{\sum_{i=1}^{n}\frac{(y_{ij}-\alpha_{2i}^{(k)})z_{ij}}{\sigma_j^{2(k-1)}}}{\sum_{i=1}^{n}\frac{z_{ij}}{\sigma_j^{2(k-1)}}}$, $\sigma_j^{2(k)}=\frac{\sum_{i=1}^{n}(y_{ij}-\alpha_{2i}^{(k)}-\mu_j^{(k)})^2z_{ij}+\sum_{i=1}^{n}\frac{1}{s_{2,\alpha_{2i}}^{(k-1)}}z_{ij}}{\sum_{i=1}^{n}z_{ij}}$,

$\tau_2^{2(k)}=\frac{1}{n}\sum_{i=1}^{n}({\alpha_{2i}^{(k)}}^2+\frac{1}{s_{2,\alpha_{2i}}^{(k-1)}})$. [since, $Var[\alpha_{2i}^{(k)}|\{z_{ij}\}_{j=1}^{n_G},\{y_{ij}\}_{j=1}^{n_G}]=\frac{1}{s_{2,\alpha_{2i}}^{(k-1)}}$]

Step II:

Having estimated $\{\mu_j\}_{j=1}^{n_G}$, $\{\alpha_{2i}\}_{i=1}^{n}$, $\{\sigma_j^2\}_{j=1}^{n_G}$ and τ_2^2, we now estimate $\{c_j\}_{j=1}^{n_G}$, c, $\{\alpha_{1i}\}_{i=1}^{n}$ and τ_1^2 using the likelihood of $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$ and $\{\alpha_{1i}\}_{i=1}^{n}$ only. The log-likelihood for $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$ and $\{\alpha_{1i}\}_{i=1}^{n}$ is:

$$\begin{aligned}&l_2(\{c_j\}_{j=1}^{n_G},c,\tau_1^2|\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G},\{\alpha_{1i}\}_{i=1}^{n},\{\mu_j\}_{j=1}^{n_G})\\ =&\sum_{j=1}^{n_G}\sum_{i=1}^{n}[z_{ij}\log(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))+(1-z_{ij})\log(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))]\\ &-\sum_{i=1}^{n}[\frac{1}{2}\log(\tau_1^2)+\frac{\alpha_{1i}^2}{2\tau_1^2}]\end{aligned}$$

Optimization of the log-likelihood function is obtained by equating the partial derivative of the log-likelihood function with respect to the parameters to zero.

$$\frac{\partial l}{\partial c_j}=0 \implies$$
$$\sum_{i=1}^{n}[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)}-\frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]\Phi(c+\alpha_{1i}+\mu_j)\phi(c_j)=0$$

$$\frac{\partial l}{\partial c} = 0 \implies$$
$$\sum_{j=1}^{n_G}\sum_{i=1}^{n}[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]\Phi(c_j)\phi(c+\alpha_{1i}+\mu_j) = 0$$
$$\frac{\partial l}{\partial \alpha_{1i}} = 0 \implies$$
$$\sum_{j=1}^{n_G}[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]\Phi(c_j)\phi(c+\alpha_{1i}+\mu_j) - \frac{\alpha_{1i}}{\tau_1^2} = 0$$
$$\frac{\partial l}{\partial \tau_1^2} = 0 \implies$$
$$\sum_{i=1}^{n}[-\frac{1}{2\tau_1^2} + \frac{\alpha_{1i}^2}{2(\tau_1^2)^2}] = 0$$

The parameters are estimated with EM algorithm coupled with Iteratively Re-weighted Least Squares (IRLS). $\{\alpha_{1i}\}_{i=1}^n$, $\{\alpha_{2i}\}_{i=1}^n$, and $\{c_j\}_{j=1}^{n_G}$ are assumed to be latent variables. The expectation step and the maximization steps are described below:

E-step:

We consider finding conditional expectation of α_{1i}. Define a vector $u_l = y_{ij}$ where $l = (i-1)n_G + j$, $1 \le i \le n$, $1 \le j \le n_G$. Define X and Z as design matrices of $\{\mu_j\}_{j=1}^{n_G}$ and $\{\alpha_{1i}\}_{i=1}^n$ respectively, so that, $X_{li} = 1$ and $Z_{lj} = 1$ if $l = (i-1)n_G + j$, $1 \le i \le n$, $1 \le j \le n_G$ and $X_{li} = 0, Z_{lj} = 0$ otherwise.

In section 2.4.1, for $l = (i-1)n_G + j$, by treating the term $\Phi(c_j^{(k-1)})$ as constant, the diagonal matrix $W^{(k-1)}$ can be expressed as:
$W_{ll}^{(k-1)} = \frac{(\phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)};0,1))^2\Phi(c_j^{(k-1)})}{(\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))}$.
So, $Z^tW^{(k-1)}Z = \sum_{j=1}^{n_G} \frac{(\phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)};0,1))^2\Phi(c_j^{(k-1)})}{(\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))}$.
$((D^{(k-1)})^{-1})_{ii} = \frac{1}{\tau_2^{2(k-1)}}$ and $(Z^tW^{(k-1)}y^{*(k)} - Z^tW^{(k-1)}X\beta^{k-1}) = (Z^tW^{(k-1)}Z)\alpha_1 + Z^tr$
where $r_l = \frac{(z_{ij}-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))\phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)};0,1)}{\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)})(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))}$, $l = (i-1)n_G + j$.
The update for α_1 is
$\alpha_1^{(k)} = (Z^tW^{(k-1)}Z + (D^{(k-1)})^{-1})^{-1}(Z^tW^{(k-1)}y^{*(k)} - Z^tW^{(k-1)}X\beta^{k-1})$, which can also be rewritten as:

$\alpha_{1i}^{(k)} = \alpha_{1i}^{(k-1)} + \frac{s_{1,\alpha_{1i}}^{(k-1)}}{s_{2,\alpha_{1i}}^{(k-1)}}$ where

$s_{1,\alpha_{1i}}^{(k-1)} = \sum_{j=1}^{n_G} \frac{(z_{ij} - \Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))\phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)};0,1)}{\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)})(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))} - \frac{\alpha_{1i}^{(k-1)}}{\tau_1^{2(k-1)}}$ and

$s_{2,\alpha_{1i}}^{(k-1)} = \sum_{j=1}^{n_G} \frac{(\phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)};0,1))^2\Phi(c_j^{(k-1)})}{(\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))} + \frac{1}{\tau_1^{2(k-1)}}$.

M-step: $c^{(k)}$ is updated according to Fisher-scoring step from the above partial differential equation:

$c^{(k)} = c^{(k-1)} + \frac{s_{1,c}^{(k-1)}}{s_{2,c}^{(k-1)}}$ with

$s_{1,c}^{(k-1)} = \sum_{i=1}^{n}\sum_{j=1}^{n_G} \frac{(z_{ij} - \Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))\phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)};0,1)}{\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)})(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))}$ and

$s_{2,c}^{(k-1)} = \sum_{i=1}^{n}\sum_{j=1}^{n_G} \frac{\Phi(c_j^{(k-1)})(\phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)};0,1))^2}{(\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_j+\alpha_{1i}^{(k-1)}))}$.

Maximum likelihood estimator of binomial model can be obtained by equating estimated frequency to observed frequency and using Lemma 1 as

$\Phi(c_j^{(k)})E[\Phi(c^{(k)}+\mu_j+\alpha_{1i}^{(k)})] = \frac{1}{n}\sum_{i=1}^{n} z_{ij}$, or $\Phi(c_j^{(k)})\Phi(\frac{c+\mu_j+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1)}}) = \frac{1}{n}\sum_{i=1}^{n} z_{ij}$

So, MLE of $\Phi(c_j)$ given other parameters is :

$\Phi(c_j^{(k)}) = min(1, \frac{\frac{1}{n}\sum_{i=1}^{n} z_{ij}}{\Phi(\frac{c+\mu_j+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1)}})}) \implies c_j^{(k)} = \Phi^{-1}(min(1, \frac{\frac{1}{n}\sum_{i=1}^{n} z_{ij}}{\Phi(\frac{c+\mu_j+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)})}{\sqrt{1+var(\alpha_1)}})})$

and $\tau_1^{2(k)} = \frac{1}{n}\sum_{i=1}^{n}({\alpha_{1i}^{(k)}}^2 + \frac{1}{s_{2,\alpha_{1i}}^{(k-1)}})$, [since $Var[\alpha_{1i}^{(k)}|\{z_{ij}\}_{j=1}^{n_G}, \{y_{ij}\}_{j=1}^{n_G}] = \frac{1}{s_{2,\alpha_{1i}}^{(k-1)}}$]

2.4.3 Asymptotic distribution of estimates

We derive the asymptotic distribution of the parameters when the number of cells becomes large. It is a well-known fact that if $\hat{\theta}_n$ is the mle of θ based on n independent and identically distributed observations following some common distribution F_θ and θ_0 is the true value of θ, $\sqrt{n}(\hat{\theta}_n - \theta_0)$ asymptotically follows $N(0, nI(\theta_0)^{-1})$, with $I(\theta_0) = \lim_{n\to\infty} I_n(\theta_0)$ where

$$I_n(\theta_0) = E_{\theta_0}[-\tfrac{d^2}{d\theta^2} l(\theta; X_1, ..., X_n)]$$

Now, we calculate the second derivatives with respect to parameters:

$$\frac{\partial^2 l}{\partial \mu_j^2} = -\frac{\sum_{i=1}^{n} z_{ij}}{\sigma_j^2}$$
$$\frac{\partial^2 l}{\partial (\sigma_j^2)^2} = \sum_{i=1}^{n}[\frac{z_{ij}}{2(\sigma_j^2)^2} - \frac{z_{ij}(y_{ij} - \alpha_{2i} - \mu_j)^2}{(\sigma_j^2)^3}]$$
$$\frac{\partial^2 l}{\partial (\tau_2^2)^2} = \sum_{i=1}^{n}[-\frac{1}{2(\tau_2^2)^2} + \frac{\alpha_{2i}^2}{(\tau_2^2)^3}]$$
$$\frac{\partial^2 l}{\partial \mu_j \partial \sigma_j^2} = -\frac{\sum_{i=1}^{n} z_{ij}(y_{ij} - \alpha_{2i} - \mu_j)}{(\sigma_j^2)^2}$$
$$\frac{\partial^2 l}{\partial \mu_j \partial \tau_2^2} = 0, \frac{\partial^2 l}{\partial \sigma_j^2 \partial \tau_2^2} = 0$$

So that, $E[-\frac{\partial^2 l}{\partial \mu_j^2}] = \frac{\sum_{i=1}^{n} E[z_{ij}]}{\sigma_j^2}$, $E[-\frac{\partial^2 l}{\partial (\sigma_j^2)^2}] = \frac{\sum_{i=1}^{n} E[z_{ij}]}{2(\sigma_j^2)^2}$, $E[-\frac{\partial^2 l}{\partial (\tau_2^2)^2}] = \frac{n}{2(\tau_2^2)^2}$, $E[-\frac{\partial^2 l}{\partial \mu_j \partial \sigma_j^2}] = 0$, $E[-\frac{\partial^2 l}{\partial \mu_j \partial \tau_2^2}] = 0$, $E[-\frac{\partial^2 l}{\partial \sigma_j^2 \partial \tau_2^2}] = 0$.

Now, $E[z_{ij}] = E[E[z_{ij}|\alpha_{1i}]] = E[\Phi(c_j)\Phi(c + \alpha_{1i} + \mu_j)] = \Phi(c_j)\Phi(\frac{(c+\mu_j)}{\sqrt{1+\tau_1^2}})$

So, denoting $\theta_{1,j} = (\mu_j, \sigma_j^2, \tau_2^2)$, $\sqrt{n}(\hat{\theta}_{1,j}^{(n)} - \theta_{1,j})$ asymptotically follows normal distribution with mean 0 and variance $diag(\frac{n\sigma_j^2}{\sum_{i=1}^{n} E[z_{ij}]}, \frac{2n\sigma_j^4}{\sum_{i=1}^{n} E[z_{ij}]}, \tau_2^4) = diag(\frac{\sigma_j^2}{\Phi(c_j)\Phi(\frac{(c+\mu_j)}{\sqrt{1+\tau_1^2}})}, \frac{2\sigma_j^4}{\Phi(c_j)\Phi(\frac{(c+\mu_j)}{\sqrt{1+\tau_1^2}})}, 2\tau_2^2)$.

Because our parameters are estimated in two separate steps, μ_js are assumed to be fixed in estimation of (c_j, c, τ_1^2). Similarly,

$$\frac{\partial^2 l}{\partial c^2} = \sum_{j=1}^{n_G}\sum_{i=1}^{n}[-\frac{z_{ij}\Phi(c_j)\phi(c + \alpha_{1i} + \mu_j)}{(\Phi(c_j)\Phi(c + \alpha_{1i} + \mu_j))^2} - \frac{(1 - z_{ij})\Phi(c_j)\phi(c + \alpha_{1i} + \mu_j)}{(1 - \Phi(c_j)\Phi(c + \alpha_{1i} + \mu_j))^2}]\phi(c + \alpha_{1i} + \mu_j)\Phi(c_j)$$
$$+ \sum_{j=1}^{n_G}\sum_{i=1}^{n}[\frac{z_{ij}}{\Phi(c_j)\Phi(c + \alpha_{1i} + \mu_j)} - \frac{(1 - z_{ij})}{(1 - \Phi(c_j)\Phi(c + \alpha_{1i} + \mu_j))}]\phi'(c + \alpha_{1i} + \mu_j)\Phi(c_j)$$

$$\frac{\partial^2 l}{\partial(\tau_2^2)^2} = \sum_{i=1}^{n}[\frac{1}{2(\tau_1^2)^2} - \frac{\alpha_{1i}^2}{(\tau_1^2)^3}]$$

$$\frac{\partial^2 l}{\partial(c_j)^2} = \sum_{i=1}^{n}[-\frac{z_{ij}}{(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))^2} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))^2}](\Phi(c+\alpha_{1i}+\mu_j))^2(\phi(c_j))^2$$
$$+ [\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]\Phi(c+\alpha_{1i}+\mu_j)\phi'(c_j)$$

$$\frac{\partial^2 l}{\partial c \partial c_j} = \sum_{i=1}^{n}[-\frac{z_{ij}}{(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))^2} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))^2}]$$
$$\times \Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)\phi(c_j)\phi(c+\alpha_{1i}+\mu_j)$$
$$+ \sum_{i=1}^{n}[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]\phi(c+\alpha_{1i}+\mu_j)\phi(c_j)$$

So, $\frac{1}{n}E[-\frac{\partial^2 l}{\partial c^2}] = \frac{1}{n}E[E[-\frac{\partial^2 l}{\partial c^2}|\alpha_{1i}]] = \sum_{j=1}^{n_G} E[\frac{1}{n}\sum_{i=1}^{n}[\frac{(\Phi(c_j)\phi(c+\alpha_{1i}+\mu_j))^2}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]]$.

Denote, by $\psi_2(a,p,\sigma^2) = E[\frac{(\phi(X+a))^2}{\Phi(X+a)(1-p\Phi(X+a))}]$ where $X \sim N(0,\sigma^2)$ for $-\infty < a < \infty$, and $0 < p < 1$. Note that, $\psi_2(a,p,\sigma^2)$ is finite for all a, p, σ^2, because $\frac{(\phi(x+a))^2}{\Phi(x+a)(1-p\Phi(x+a))}$ is a bounded function of x for a given a, p, and σ^2. Taking $a = (\mu + c_j)$, $X = \alpha_{1i}$, $p = \Phi(c_j)$, and $\sigma^2 = \tau^2$, we have, $E[\frac{(\Phi(c_j)\phi(c+\alpha_{1i}+\mu_j))^2}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}] = \Phi(c_j)\psi_2((c+\mu_j), \Phi(c_j), \tau^2)$

So, $\frac{1}{n}\sum_{i=1}^{n}[\frac{(\Phi(c_j)\phi(c+\alpha_{1i}+\mu_j))^2}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}] \xrightarrow{P} \Phi(c_j)\psi_2((c+\mu_j), \Phi(c_j), \tau_1^2)$.

Now, $\frac{1}{n}\sum_{i=1}^{n}[\frac{(\Phi(c_j)\phi(c+\alpha_{1i}+\mu_j))^2}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]$ being a bounded quantity, by Bounded Convergence Theorem, $E[\frac{1}{n}\sum_{i=1}^{n}[\frac{(\Phi(c_j)\phi(c+\alpha_{1i}+\mu_j))^2}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j)(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]] \to \Phi(c_j)\psi_2((c+\mu_j), \Phi(c_j), \tau_1^2)$ as well. So, the limit $\frac{1}{n}E[-\frac{\partial^2 l}{\partial c^2}]$ has the value: $\sum_{j=1}^{n_G} \Phi(c_j)\psi_2((c+\mu_j), \Phi(c_j), \tau_1^2) = \zeta_{11}$ (say).

Also, $\frac{1}{n}E[-\frac{\partial^2 l}{\partial(c_j)^2}] = \frac{1}{n}E[E[-\frac{\partial^2 l}{\partial(c_j)^2}|\alpha_{1i}]] = E[\frac{1}{n}\sum_{i=1}^{n}[\frac{\Phi(c+\alpha_{1i}+\mu_j)(\phi(c_j))^2}{(\Phi(c_j))^2(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]$

Now, since α_{1i}s are i.i.d.,

$$\frac{1}{n}\sum_{i=1}^{n}[\frac{\Phi(c+\alpha_{1i}+\mu_j)(\phi(c_j))^2}{(\Phi(c_j))^2(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))} \xrightarrow{P} \frac{E[\Phi(c+\alpha_{1i}+\mu_j)](\phi(c_j))^2}{(\Phi(c_j))^2(1-\Phi(c_j)E[\Phi(c+\alpha_{1i}+\mu_j)])}$$

$$= \frac{\Phi(\frac{c+\mu_j}{\sqrt{1+\tau_1^2}})(\phi(c_j))^2}{(\Phi(c_j))^2(1-\Phi(c_j)\Phi(\frac{c+\mu_j}{\sqrt{1+\tau_1^2}}))}.$$

Also, note that, $\frac{\Phi(c+\alpha_{1i}+\mu_j)}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))} \leq \frac{1}{(1-\Phi(c_j))}$.
Hence, by Bounded Convergence Theorem,
$E[\frac{1}{n}\sum_{i=1}^{n}[\frac{\Phi(c+\alpha_{1i}+\mu_j)(\phi(c_j))^2}{(\Phi(c_j))^2(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}] \to \frac{\Phi(\frac{c+\mu_j}{\sqrt{1+\tau_1^2}})(\phi(c_j))^2}{(\Phi(c_j))^2(1-\Phi(c_j)\Phi(\frac{c+\mu_j}{\sqrt{1+\tau_1^2}}))} = \zeta_{12}$ (say).
Now, $\frac{1}{n}E[-\frac{\partial^2 l}{\partial c \partial c_j}] = E[\frac{1}{n}\sum_{i=1}^{n}\frac{\phi(c_j)\phi(c+\alpha_{1i}+\mu_j)}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}]$.

Denote by $\psi_1(a,p,\sigma^2) = E[\frac{\phi(a+X)}{1-p\Phi(a+X)}]$, where $X \sim N(0,\sigma^2)$, $-\infty < a < \infty$, and $0 < p < 1$. Taking $a = (c+\mu_j), p = \Phi(c_j), X = \alpha_{1i}$, and $\sigma^2 = \tau^2$, we have, $E[\frac{\phi(c_j)\phi(c+\alpha_{1i}+\mu_j)}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}] = \phi(c_j)\psi_1((c+\mu_j), \Phi(c_j), \tau_1^2)$.
So, $\frac{1}{n}\sum_{i=1}^{n}\frac{\phi(c_j)\phi(c+\alpha_{1i}+\mu_j)}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))} \xrightarrow{P} \phi(c_j)\psi_1((c+\mu_j), \Phi(c_j), \tau_1^2)$. Since $f(x) = \frac{\phi(a+x)}{1-p\Phi(a+x)}$ is a bounded function of x, taking $p = \Phi(c_j)$ and by Bounded Convergence Theorem,

$$E[\frac{1}{n}\sum_{i=1}^{n}\frac{\phi(c_j)\phi(c+\alpha_{1i}+\mu_j)}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_j))}] \to \phi(c_j)\psi_1((c+\mu_j), \Phi(c_j), \tau_1^2) = \zeta_{22} \text{ (say) as } n \to \infty,$$

Moreover, $\frac{1}{n}E[-\frac{\partial^2 l}{\partial c \partial c_j}]$ also has the same limit.

Denoting $\theta_{2,j} = (c, c_j, \tau_1^2)$, it follows that $\sqrt{n}(\hat{\theta}_{2,j}^{(n)} - \theta_{2,j})$ asymptotically follows $N(0,\Sigma)$ where:

$$\Sigma = \begin{pmatrix} \zeta_{11} & \zeta_{12} & 0 \\ \zeta_{21} & \zeta_{22} & 0 \\ 0 & 0 & 2\tau_1^4 \end{pmatrix}$$

Putting the estimates of the parameters evaluated above, we can estimate ψ_1 and ψ_2 that eventually provide the estimates of the variances of the parameters

2.5 Estimation of parameters for bimodal model

As discussed in Section 2.3, we propose two models for modeling the gene expression data depending on the number of modes as obtained through a testing procedure (Section

2.2). Here we describe the method of estimating the parameters for our proposed bimodal distribution. Using the model (2.2), the likelihood function for gene j based on n cells is:

$$
\begin{aligned}
&L(\theta_j|\{z_{ij}\}_{i=1}^n,\{y_{ij}\}_{i=1}^n\{\alpha_{1i}\}_{i=1}^n,\{\alpha_{2i}\}_{i=1}^n,\{D_{ij}\}_{i=1}^n)\\
=&\prod_{i=1}^n\Big\{\pi_j^{D_{ij}}(1-\pi_j)^{(1-D_{ij})}[(\Phi(c_j))^{z_{ij}}(1-\Phi(c_j))^{(1-z_{ij})}(\phi(y_{ij};\mu_{2j},\sigma_{2j}))^{z_{ij}}]^{D_{ij}}\\
&\times\big[(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))^{z_{ij}}(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))^{(1-z_{ij})}\\
&\times(\phi(y_{ij};\mu_{1j}+\alpha_{2i},\sigma_{1j}))^{z_{ij}}\big]^{(1-D_{ij})}\phi(\alpha_{1i};0,\tau_1)\phi(\alpha_{2i};0,\tau_2)\Big\}
\end{aligned}
$$

where $\theta_j=(\pi_j,\mu_{1j},\mu_{2j},\sigma_{1j}^2,\sigma_{2j}^2,c_j,c,\tau_1^2,\tau_2^2)'$ and $\phi(x;\mu,\sigma)$ is the p.d.f. of a $N(\mu,\sigma^2)$, and $\Phi(x)$ is the distribution function of a standard normal variable.

Hence the log-likelihood is for the j-th gene is given by,

$$
\begin{aligned}
&l(\pi_j,\mu_{1j},\mu_{2j},\sigma_{1j}^2,\sigma_{2j}^2,\tau_1^2,\tau_2^2,c_j,c|\{y_{ij}\}_{i=1}^n,\{z_{ij}\}_{i=1}^n,\{E_{ij}\}_{i=1}^n,\{D_{ij}\}_{i=1}^n,\{\alpha_{1i}\}_{i=1}^n,\{\alpha_{2i}\}_{i=1}^n)\\
&=\text{Constant}+\sum_{i=1}^n\Big\{D_{ij}\log(\pi_j)+(1-D_{ij})\log(1-\pi_j)\\
&+D_{ij}\Big[z_{ij}\log(\Phi(c_j))+(1-z_{ij})\log(1-\Phi(c_j))+z_{ij}\big[-\frac{1}{2}\log(2\pi)-\frac{1}{2}\log(\sigma_{2j}^2)-\frac{(y_{ij}-\mu_{2j})^2}{2\sigma_{2j}^2}\big]\Big]\\
&+(1-D_{ij})\Big[z_{ij}\log(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))+(1-z_{ij})\log(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))\\
&+z_{ij}\big[-\frac{1}{2}\log(2\pi)-\frac{1}{2}\log(\sigma_{1j}^2)-\frac{(y_{ij}-\mu_{1j}-\alpha_{2i})^2}{2\sigma_{1j}^2}\big]\Big]\Big\}\\
&+\sum_{i=1}^n\big[-\frac{1}{2}\log(2\pi)-\frac{1}{2}\log(\tau_1^2)-\frac{\alpha_{1i}^2}{2\tau_1^2}-\frac{1}{2}\log(2\pi)-\frac{1}{2}\log(\tau_2^2)-\frac{\alpha_{2i}^2}{2\tau_2^2}\big]
\end{aligned}
$$

Consequently the log-likelihood function of n_G genes is given by

$$
\begin{aligned}
&l(\{\pi_j\}_{j=1}^{n_G},\{\mu_{1j}\}_{j=1}^{n_G},\{\mu_{2j}\}_{j=1}^{n_G},\{\sigma_{1j}^2\}_{j=1}^{n_G},\{\sigma_{2j}^2\}_{j=1}^{n_G},\tau_1^2,\tau_2^2,\{c_j\}_{j=1}^{n_G},c|\\
&\{\{y_{ij}\}_{i=1}^n\}_{j=1}^{n_G},\{\{z_{ij}\}_{i=1}^n\}_{j=1}^{n_G},\{\{D_{ij}\}_{i=1}^n\}_{j=1}^{n_G},\{\alpha_{1i}\}_{i=1}^n,\{\alpha_{2i}\}_{i=1}^n)\\
&=\text{Constant}+\sum_{i=1}^n\sum_{j=1}^{n_G}[D_{ij}\log(\pi_j)+(1-D_{ij})\log(1-\pi_j)+D_{ij}[z_{ij}\log(\Phi(c_j))+(1-z_{ij})\log(1-\\
&\Phi(c_j))+z_{ij}[-\tfrac{1}{2}\log(2\pi)-\tfrac{1}{2}\log(\sigma_{2j}^2)-\tfrac{(y_{ij}-\mu_{2j})^2}{2\sigma_{2j}^2}]]+(1-D_{ij})[z_{ij}\log(\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))+\\
&(1-z_{ij})\log(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))+z_{ij}[-\tfrac{1}{2}\log(2\pi)-\tfrac{1}{2}\log(\sigma_{1j}^2)-\tfrac{(y_{ij}-\mu_{1j}-\alpha_{2i})^2}{2\sigma_{1j}^2}]]]+
\end{aligned}
$$

$\sum_{i=1}^{n}[-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\tau_1^2) - \frac{\alpha_{1i}^2}{2\tau_1^2} - \frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\tau_2^2) - \frac{\alpha_{2i}^2}{2\tau_2^2}]$

2.5.1 Estimation of parameters

Similar to the unimodal case, genes are clustered into subsets with similar zero occurrences and gene specific parameters within each subset are estimated simultaneously. Similar to the unimodal case, we first maximize the likelihood for $\{\{y_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$, $\{\alpha_{2i}\}_{i=1}^{n}$ conditioned on $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$ and $\{\alpha_{1i}\}_{i=1}^{n}$ to maximize over $\{\mu_{1j}\}_{j=1}^{n_G}$, $\{\mu_{2j}\}_{j=1}^{n_G}$, $\{\sigma_{1j}^2\}_{j=1}^{n_G}$, $\{\sigma_{2j}^2\}_{j=1}^{n_G}$, $\{\alpha_{2i}\}_{i=1}^{n}$ and τ_2^2, where n is the number of cells and n_G is the number of genes. After that, we maximize the marginal likelihood for $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$, $\{\alpha_{1i}\}_{i=1}^{n}$ to maximize over remaining parameters.

Note that, $(y_{ij}|z_{ij} > 0)$ follows mixture normal with same mean vector $((\mu_{1j}+\alpha_{2i}), \mu_{2j})$, same variance vector $(\sigma_{1j}^2, \sigma_{2j}^2)$ but different mixture proportion π'_j. First, a mixture normal distribution is fitted on positive expression values for each gene. We also introduce the latent variable D'_{ij} which is the indicator variable for whether the i-th cell for j-th gene belongs to mode 2 conditioned on the fact that $z_{ij} > 0$. The purpose of this is to estimate the parameters in two steps.

Step I

In the first step, $\{\mu_{1j}\}_{j=1}^{n_G}, \{\mu_{2j}\}_{j=1}^{n_G}, \{\sigma_{1j}^2\}_{j=1}^{n_G}.\{\sigma_{2j}^2\}_{j=1}^{n_G}, \{\alpha_{2i}\}_{i=1}^{n}$ are estimated by conditioning on $\{\{z_{ij}\}_{i=1}^{n}\}_{j=1}^{n_G}$. This can be done by the following procedure:

1. Start with initial values of $\mu_{1j}^{(0)}, \mu_{2j}^{(0)}, \sigma_{1j}^{(0)}, \sigma_{2j}^{(0)}, \pi_j'^{(0)}$.

2. Estimate mixing probability for each cell corresponding to every gene:
   $D'^{(n)}[i,j] = \frac{\pi_j'^{(n-1)}\phi(y_{ij};\mu_{2j}^{(n-1)},\sigma_{2j}^{(n-1)})}{\pi_j'^{(n-1)}\phi(y_{ij};\mu_{2j}^{(n-1)},\sigma_{2j}^{(n-1)})+(1-\pi_j'^{(n-1)})\phi(y_{ij};\mu_{1j}^{(n-1)},\sigma_{1j}^{(n-1)})}$.

3. The parameters of the each of the two component normal distributions can be estimated in the following manner:
   $\pi_j'^{(n)} = \frac{\sum_{i=1}^{n} z_{ij}D'^{(n)}_{ij}}{\sum_{i=1}^{n} z_{ij}}$, $\mu_{1j}^{(n)} = \frac{\sum_{i=1}^{n} z_{ij}(1-D'^{(n)}_{ij})y_{ij}}{\sum_{i=1}^{n} z_{ij}(1-D'^{(n)}_{ij})}$, $\mu_{2j}^{(n)} = \frac{\sum_{i=1}^{n} z_{ij}(D'^{(n)}_{ij})y_{ij}}{\sum_{i=1}^{n} z_{ij}(D'^{(n)}_{ij})}$
   $\sigma_{1j}^{2\,(n)} = \frac{\sum_{i=1}^{n} z_{ij}(1-D'^{(n)}_{ij})(y_{ij}-\mu_{1j}^{(n)})^2}{\sum_{i=1}^{n} z_{ij}(1-D'^{(n)}_{ij})}$, $\sigma_{2j}^{2\,(n)} = \frac{\sum_{i=1}^{n} z_{ij}(D'^{(n)}_{ij})(y_{ij}-\mu_{2j}^{(n)})^2}{\sum_{i=1}^{n} z_{ij}(D'^{(n)}_{ij})}$.

4. Step 2 and 3 are repeated until the parameters converge.

Thus μ_{2j}, σ^2_{2j} get determined in this step and $(\mu_{1j}, \alpha_{2i}, \sigma^2_{1j}, \tau^2_2, \pi_j)$ are then estimated based on the following equations:

$$\begin{aligned}
&\frac{\partial l}{\partial \mu_{1j}} = 0 \implies \sum_{i=1}^{n}(1-D'_{ij})\frac{(y_{ij}-\alpha_{2i}-\mu_{1j})z_{ij}}{\sigma^2_{1j}} = 0\\
&\frac{\partial l}{\partial \alpha_{2i}} = 0 \implies \sum_{j=1}^{n_G}(1-D'_{ij})\frac{(y_{ij}-\alpha_{2i}-\mu_{1j})z_{ij}}{\sigma^2_{1j}} - \sum_{j=1}^{n_G}\frac{\alpha_{2i}z_{ij}}{\tau^2_2} = 0\\
&\frac{\partial l}{\partial \sigma^2_{1j}} = 0 \implies \sum_{i=1}^{n}(1-D'_{ij})z_{ij}[-\frac{1}{2\sigma^2_{1j}} + \frac{(y_{ij}-\alpha_{2i}-\mu_{1j})^2}{2(\sigma^2_{1j})^2}] = 0\\
&\frac{\partial l}{\partial \tau^2_2} = 0 \implies \sum_{i=1}^{n}[-\frac{1}{2\tau^2_2} + \frac{\alpha^2_{2i}}{2(\tau^2_2)^2}] = 0\\
&\frac{\partial l}{\partial \pi_j} = 0 \implies \sum_{i=1}^{n}[\frac{D'_{ij}}{\pi_j} - \frac{(1-D'_{ij})}{(1-\pi_j)}] = 0
\end{aligned}$$

Parameters are solved using EM-algorithm with the following steps:

E-step:

We first derive the conditional distribution of α_{2i}. The joint log-likelihood of $\{y_{ij}\}_{j=1}^{n_G}$ and α_{2i} is given by

$$\begin{aligned}
&l(\alpha_{2i}, \{y_{ij}\}_{j=1}^{n_G}|\{z_{ij}\}_{j=1}^{n_G}, \{\mu_{1j}\}_{j=1}^{n_G}, \{\sigma^2_{1j}\}_{j=1}^{n_G}, \tau^2_2)\\
=&-\frac{\sum_{j=1}^{n_G} z_{ij}(1-D'_{ij})}{2}\log(2\pi) - \frac{1}{2}\sum_{j=1}^{n_G} z_{ij}(1-D'_{ij})\log(\sigma^2_{1j})\\
&-\sum_{j=1}^{n_G}\frac{(1-D'_{ij})(y_{ij}-\mu_{1j}-\alpha_{2i})^2 z_{ij}}{2\sigma^2_{1j}} - \frac{1}{2}\log(2\pi) - \frac{1}{2}\log(2\tau^2_2) - \frac{\alpha^2_{2i}}{2\tau^2_2}
\end{aligned}$$

So, the conditional log-likelihood of α_{2i} given $\{y_{ij}\}_{j=1}^{n_G}$ can be expressed as:

$$\begin{aligned}
&l(\alpha_{2i}|\{y_{ij}\}_{j=1}^{n_G}, \{z_{ij}\}_{j=1}^{n_G}, \{\mu_{1j}\}_{j=1}^{n_G}, \{\sigma^2_{1j}\}_{j=1}^{n_G}, \tau^2_2)\\
=&constant - (\sum_{j=1}^{n_G}\frac{z_{ij}(1-D'_{ij})}{2\sigma^2_{1j}} + \frac{1}{2\tau^2_2})\alpha^2_{2i} + \sum_{j=1}^{n_G}\frac{(1-D'_{ij})(y_{ij}-\mu_{1j})z_{ij}}{\sigma^2_{1j}}\alpha_{2i}
\end{aligned}$$

Hence, the conditional distribution of α_{2i} is normal with conditional mean $\frac{s_{1,\alpha_{1i}}}{s_{2,\alpha_{2i}}}$ and conditional variance $\frac{1}{s_{2,\alpha_{2i}}}$ where

$s_{1,\alpha_{2i}} = \sum_{j=1}^{n_G} \frac{(1-D'_{ij})(y_{ij}-\mu_{1j})z_{ij}}{\sigma_{1j}^2}$ and $s_{2,\alpha_{2i}} = \sum_{j=1}^{n_G} \frac{(1-D'_{ij})z_{ij}}{\sigma_{1j}^2} + \frac{1}{\tau_2^2}$.

So, at the k-th iteration, we update $\alpha_{2i}^{(k)}$ as $\alpha_{2i}^{(k)} = \frac{s_{1,\alpha_{2i}}^{(k-1)}}{s_{2,\alpha_{2i}}^{(k-1)}}$ where

$s_{1,\alpha_{2i}}^{(k-1)} = \sum_{j=1}^{n_G} \frac{(1-D'_{ij})(y_{ij}-\mu_j)z_{ij}}{\sigma_{1j}^{2\ (k-1)}}$and $s_{2,\alpha_{2i}}^{(k-1)} = (\sum_{j=1}^{n_G} \frac{(1-D'_{ij})z_{ij}}{\sigma_{1j}^{2\ (k-1)}} + \frac{1}{\tau_2^{2(k-1)}})$.

M-step:

$\mu_{1j}{}^{(k)} = \frac{\sum_{i=1}^{n}(1-D'_{ij})(y_{ij}-\alpha_{2i}^{(k)})z_{ij}}{\sum_{i=1}^{n} z_{ij}}$

Since $Var[\alpha_{2i}^{(k)}|\{z_{ij}\}_{j=1}^{n_G}, \{y_{ij}\}_{j=1}^{n_G}] = \frac{1}{s_{2,\alpha_{2i}}^{(k-1)}}$, we have,

$\tau_2^{2(k)} = \frac{1}{n}\sum_{i=1}^{n}(\alpha_{2i}^{(k)^2} + \frac{1}{s_{2,\alpha_{2i}}^{(k-1)}})$, and $\sigma_{1j}^{2\ (k)} = \frac{\sum_{i=1}^{n}(1-D'_{ij})(y_{ij}-\alpha_{2i}^{(k)}-\mu_{1j}^{(k)})^2 z_{ij} + \sum_{i=1}^{n}\frac{(1-D'_{ij})}{s_{2,\alpha_{2i}}^{(k-1)}} z_{ij}}{\sum_{i=1}^{n}(1-D'_{ij})z_{ij}}$.

Step II

In this step, we estimate c_j, c, α_{1i} and τ_1^2. Differentiating the likelihood with respect to these parameters, the following equations are obtained.

$$\frac{\partial l}{\partial c_j} = 0 \implies$$
$$\sum_{i=1}^{n}(1-D_{ij})[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))}]\Phi(c+\alpha_{1i}+\mu_{1j})\phi(c_j) = 0$$
$$\frac{\partial l}{\partial c} = 0 \implies$$
$$\sum_{j=1}^{n_G}\sum_{i=1}^{n}(1-D_{ij})[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))}]\Phi(c_j)\phi(c+\alpha_{1i}+\mu_{1j}) = 0$$
$$\frac{\partial l}{\partial \alpha_{1i}} = 0 \implies$$
$$\sum_{j=1}^{n_G}(1-D_{ij})[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))}]\Phi(c_j)\phi(c+\alpha_{1i}+\mu_{1j})$$

$$-\sum_{j=1}^{n_G}\frac{\alpha_{1i}}{\tau_1^2}=0$$

$$\frac{\partial l}{\partial \tau_1^2}=0 \implies \sum_{i=1}^{n}[-\frac{1}{2\tau_1^2}+\frac{\alpha_{1i}^2}{2(\tau_1^2)^2}]=0.$$

We estimate the parameters using EM algorithm coupled with IRLS. The iteration steps are same as in unimodal case with an additional $(1-D_{ij})$ factor. D'_{ij} values obtained in the previous step are used as initial values of D_{ij}. The expectation and the maximization steps are described below:

E-step:

To update at the k-th step for α_{1i}, all other terms remain the same as in the unimodal case with an additional $(1-D_{ij})$ factor. Hence, the update at k-th step is,

$E[D_{ij}^{(k)}|z_{ij}=0]=P(D_{ij}^{(k)}=1|z_{ij}=0)=\frac{P(z_{ij}=0|D_{ij}^{(k)}=1)P(D_{ij}^{(k)}=1)}{P(z_{ij}=0)}$

$=\frac{P(z_{ij}=0|D_{ij}^{(k)}=1)P(D_{ij}^{(k)}=1)}{P(z_{ij}=0|D_{ij}^{(k)}=1)P(D_{ij}^{(k)}=1)+P(z_{ij}=0|D_{ij}^{(k)}=0)P(D_{ij}^{(k)}=0)}$

$=\frac{\pi_j^{(k-1)}(1-\Phi(c_j^{(k-1)}))}{\pi_j(1-\Phi(c_j^{(k-1)}))+(1-\pi_j^{(k-1)})(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\alpha_{1i}^{(k-1)}+\mu_{1j}))}$

Similarly, $E[D_{ij}^{(k)}|z_{ij}=1]=\frac{\pi_j^{(k-1)}\Phi(c_j^{(k-1)})}{\pi_j^{(k-1)}\Phi(c_j^{(k-1)})+(1-\pi_j^{(k-1)})\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\alpha_{1i}^{(k-1)}+\mu_{1j})}$

$\alpha_{1i}^{(k)}=\alpha_{1i}^{(k-1)}+\frac{s_{1,\alpha_{1i}}^{(k-1)}}{s_{2,\alpha_{1i}}^{(k-1)}}$ with

$s_{1,\alpha_{1i}}^{(k-1)}=\sum_{j=1}^{n_G}\frac{(1-E[D_{ij}^{(k)}|z_{ij}])(z_{ij}-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k-1)}))}{\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k-1)})(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k-1)}))}\phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k-1)},0,1)-\frac{\alpha_{1i}^{(k-1)}}{\tau_1^{2(k-1)}}$ and

$s_{2,\alpha_{1i}}^{(k-1)}=\sum_{j=1}^{n_G}\frac{(1-E[D_{ij}^{(k)}|z_{ij}])\Phi(c_j^{(k-1)})(\phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k-1)},0,1))^2}{(\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k-1)}))(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k-1)}))}+\frac{1}{\tau_1^{2(k-1)}}$

2.5.2 M-step

The update for c at the k-th step should be,

$c^{(k)}=c^{(k-1)}+\frac{s_{1,c}^{(k-1)}}{s_{2,c}^{(k-1)}}$ with

$s_{1,c}^{(k-1)} = \sum_{i=1}^{n}\sum_{j=1}^{n_G} \frac{(1-E[D_{ij}^{(k)}|z_{ij}])(z_{ij}-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k)}))}{\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k)})(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k)}))}\phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k)})$ and

$s_{2,c}^{(k-1)} = \sum_{i=1}^{n}\sum_{j=1}^{n_G} \frac{(1-E[D_{ij}^{(k)}|z_{ij}])\Phi(c_j^{(k-1)})(\phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k)}))^2}{(\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k)}))(1-\Phi(c_j^{(k-1)})\Phi(c^{(k-1)}+\mu_{1j}^{(k-1)}+\alpha_{1i}^{(k)}))}$

Note that, we have estimated π_j' in Step I. $\pi_j^{(k)}$ is estimated using the relationship between π_j' and other parameters.

$\pi_j'^{(k)} = P(D_{ij}^{(k)} = 1|z_{ij} > 0) = \frac{P(D_{ij}^{(k)}=1, z_{ij}>0)}{P(z_{ij}>0)}$

$= \frac{\pi_j^{(k)}\Phi(c_j^{(k-1)})}{\pi_j^{(k)}\Phi(c_j^{(k-1)})+(1-\pi_j^{(k)})\Phi(c_j^{(k-1)})\Phi(\frac{c^{(k)}+\mu_{1j}+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1^{(k)})}})} = \frac{\pi_j^{(k)}}{\pi_j^{(k)}+(1-\pi_j^{(k)})\Phi(\frac{c^{(k)}+\mu_{1j}+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1^{(k)})}})},$

so that $\pi_j^{(k)} = \frac{\pi_j'^{(k)}\Phi(\frac{c^{(k)}+\mu_{1j}+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1^{(k)})}})}{(1-\pi_j'^{(k)})+\pi_j'^{(k)}\Phi(\frac{c^{(k)}+\mu_{1j}+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1^{(k)})}})}$

Maximum likelihood estimator of binomial model is obtained as,

$$\Phi(c_j^{(k)})((1-\pi_j^{(k)})E[\Phi(c^{(k)}+\mu_{1j}+\alpha_{1i}^{(k)})]+\pi_j^{(k)}) = \frac{1}{n}\sum_{i=1}^{n} z_{ij}$$

$$\text{i.e.}\Phi(c_j^{(k)})((1-\pi_j^{(k)})\Phi(\frac{c+\mu_{1j}+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1^{(k)})}})+\pi_j^{(k)}) = \frac{1}{n}\sum_{i=1}^{n} z_{ij}.$$

So, MLE of c_j and τ_1 are obtained as:

$$\Phi(c_j^{(k)}) = min(1, \frac{\frac{1}{n}\sum_{i=1}^{n} z_{ij}}{((1-\pi_j^{(k)})\Phi(\frac{c^{(k)}+\mu_{1j}+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1^{(k)})}})+\pi_j^{(k)})})$$

$$\text{or, } c_j^{(k)} = \Phi^{-1}(min(1, \frac{\frac{1}{n}\sum_{i=1}^{n} z_{ij}}{((1-\pi_j^{(k)})\Phi(\frac{c^{(k)}+\mu_{1j}+\frac{1}{n}\sum_{i=1}^{n}\alpha_{1i}^{(k)}}{\sqrt{1+var(\alpha_1^{(k)})}})+\pi_j^{(k)})}))$$

$$\tau_1^{2(k)} = \frac{1}{n}\sum_{i=1}^{n}({\alpha_{1i}^{(k)}}^2 + \frac{1}{s_{2,\alpha_{1i}}^{(k-1)}}), \qquad [Var[\alpha_{1i}^{(k)}|\{z_{ij}\}_{j=1}^{n_G}, \{y_{ij}\}_{j=1}^{n_G}] = \frac{1}{s_{2,\alpha_{1i}}^{(k-1)}}]$$

2.5.3 Asymptotic distribution of estimates

Similar to unimodal setup, the estimates here are maximum likelihood estimators and hence asymptotically follow the normal distribution. So, we first calculate the second derivatives with respect to the parameters:

$$\frac{\partial^2 l}{\partial \mu_{1j}^2} = \sum_{j=1}^{n_G}(1-D'_{ij})\frac{z_{ij}}{\sigma_{1j}^2}, \quad \frac{\partial^2 l}{\partial (\sigma_{1j}^2)^2} = \sum_{i=1}^{n}(1-D'_{ij})z_{ij}[\frac{1}{2(\sigma_{1j}^2)^2} - \frac{(y_{ij}-\alpha_{2i}-\mu_{1j})^2}{(\sigma_{1j}^2)^3}],$$
$$\frac{\partial^2 l}{\partial \mu_{2j}^2} = \sum_{j=1}^{n_G} D'_{ij}\frac{z_{ij}}{\sigma_{2j}^2}, \quad \frac{\partial^2 l}{\partial (\sigma_{2j}^2)^2} = \sum_{i=1}^{n} D'_{ij}z_{ij}[\frac{1}{2(\sigma_{2j}^2)^2} - \frac{(y_{ij}-\alpha_{2i}-\mu_{2j})^2}{(\sigma_{2j}^2)^3}],$$
$$\frac{\partial^2 l}{\partial \pi_j^2} = \sum_{i=1}^{n}[-\frac{D'_{ij}}{\pi_j^2} - \frac{(1-D'_{ij})}{(1-\pi_j)^2}], \quad \frac{\partial^2 l}{\partial \tau_2^2} = \sum_{i=1}^{n}[\frac{1}{2(\tau_2^2)^2} - \frac{\alpha_{2i}^2}{(\tau_2^2)^3}],$$
$$\frac{\partial^2 l}{\partial \mu_{1j}\partial \mu_{2j}} = \frac{\partial^2 l}{\partial \mu_{1j}\partial \sigma_{2j}^2} = \frac{\partial^2 l}{\partial \mu_{1j}\partial \pi_j} = \frac{\partial^2 l}{\partial \mu_{1j}\partial \tau_2^2} = \frac{\partial^2 l}{\partial \mu_{2j}\partial \sigma_{1j}^2} = \frac{\partial^2 l}{\partial \mu_{2j}\partial \pi_j} = \frac{\partial^2 l}{\partial \mu_{2j}\partial \tau_2^2} = \frac{\partial^2 l}{\partial \sigma_{1j}^2\partial \sigma_{2j}^2}$$
$$= \frac{\partial^2 l}{\partial \sigma_{1j}^2\partial \pi_j} = \frac{\partial^2 l}{\partial \sigma_{1j}^2\partial \tau_2^2} = \frac{\partial^2 l}{\partial \sigma_{2j}^2\partial \pi_j} = \frac{\partial^2 l}{\partial \sigma_{2j}^2\partial \tau_2^2} = \frac{\partial^2 l}{\partial \pi_j\partial \tau_2^2} = 0,$$
$$\frac{\partial^2 l}{\partial \mu_{1j}\partial \sigma_{1j}^2} = -\sum_{i=1}^{n}(1-D'_{ij})\frac{(y_{ij}-\alpha_{2i}-\mu_{1j})z_{ij}}{(\sigma_{1j}^2)^2}, \quad \frac{\partial^2 l}{\partial \mu_{2j}\partial \sigma_{2j}^2} = -\sum_{i=1}^{n} D'_{ij}\frac{(y_{ij}-\mu_{2j})z_{ij}}{(\sigma_{2j}^2)^2}$$

It follows that $E[-\frac{\partial^2 l}{\partial \mu_{1j}\partial \mu_{2j}}] = E[-\frac{\partial^2 l}{\partial \mu_{1j}\partial \sigma_{2j}^2}] = E[-\frac{\partial^2 l}{\partial \mu_{1j}\partial \pi_j}] = E[-\frac{\partial^2 l}{\partial \mu_{1j}\partial \tau_2^2}] = E[-\frac{\partial^2 l}{\partial \mu_{2j}\partial \sigma_{1j}^2}] = E[-\frac{\partial^2 l}{\partial \mu_{2j}\partial \pi_j}] = E[-\frac{\partial^2 l}{\partial \mu_{2j}\partial \tau_2^2}] = E[-\frac{\partial^2 l}{\partial \sigma_{1j}^2\partial \sigma_{2j}^2}] = E[-\frac{\partial^2 l}{\partial \sigma_{1j}^2\partial \pi_j}] = E[-\frac{\partial^2 l}{\partial \sigma_{1j}^2\partial \tau_2^2}] = E[-\frac{\partial^2 l}{\partial \sigma_{2j}^2\partial \pi_j}] = E[-\frac{\partial^2 l}{\partial \sigma_{2j}^2\partial \tau_2^2}] = E[-\frac{\partial^2 l}{\partial \pi_j\partial \tau_2^2}] = E[-\frac{\partial^2 l}{\partial \mu_{1j}\partial \sigma_{1j}^2}] = E[-\frac{\partial^2 l}{\partial \mu_{2j}\partial \sigma_{2j}^2}] = 0$

Now, note that, $E[(1-D_{ij})z_{ij}] = E[E[(1-D_{ij})z_{ij}|\alpha_{1i}]] = E[(1-\pi_j)\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})] = (1-\pi_j)\Phi(c_j)\Phi(\frac{c+\mu_{1j}}{\sqrt{1+\tau_1^2}})$, and $E[(D_{ij}z_{ij}] = E[E[D_{ij}z_{ij}|\alpha_{1i}]] = \pi_j\Phi(c_j)$.

So, denoting $\theta_{1,j} = (\mu_{1j}, \mu_{2j}, \sigma_{1j}^2, \sigma_{2j}^2, \pi_j, \tau_2^2)$, $\sqrt{n}(\hat{\theta}_{1,j}^{(n)} - \theta_{1,j})$ asymptotically follows multivariate normal distribution with mean 0 and diagonal variance covariance matrix Σ (say).

Hence $\sqrt{n}(\hat{\theta}_{1,j}^{(n)} - \theta_{1,j})$ asymptotically follows $N(0, \Sigma)$ where $\Sigma = diag(\frac{\sigma_{1j}^2}{(1-\pi_j)\Phi(c_j)\Phi(\frac{c+\mu_{1j}}{\sqrt{1+\tau_1^2}})}$, $\frac{\sigma_{2j}^2}{\pi_j\Phi(c_j)}, \frac{2\sigma_{1j}^4}{(1-\pi_j)\Phi(c_j)\Phi(\frac{c+\mu_{1j}}{\sqrt{1+\tau_1^2}})}, \frac{2\sigma_{2j}^4}{\pi_j\Phi(c_j)}, \pi_j(1-\pi_j), 2\tau_2^4)$

The parameters $\theta_{2,j} = (c_j, c, \tau_1^2)$ are estimated in the second step keeping other parameters fixed. Similar to the unimodal case, here again, we calculate the second-order derivatives.

$$\frac{\partial^2 l}{\partial c^2} = \sum_{j=1}^{n_G}\sum_{i=1}^{n}(1-D_{ij})[-\frac{z_{ij}}{\Phi(c_j)(\Phi(c+\alpha_{1i}+\mu_{1j}))^2} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))^2}]$$
$$\times (\Phi(c_j))^2(\phi(c+\alpha_{1i}+\mu_{1j}))^2$$
$$+\sum_{j=1}^{n_G}\sum_{i=1}^{n}(1-D_{ij})[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))}]\Phi(c_j)\phi'(c+\alpha_{1i}+\mu_{1j})$$
$$\frac{\partial^2 l}{\partial c_j^2} = \sum_{i=1}^{n}(1-D_{ij})[-\frac{z_{ij}}{(\Phi(c_j))^2\Phi(c+\alpha_{1i}+\mu_{1j})} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))^2}]$$
$$\times (\Phi(c+\alpha_{1i}+\mu_{1j}))^2(\phi(c_j))^2$$
$$+\sum_{i=1}^{n}(1-D_{ij})[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))}]\Phi(c+\alpha_{1i}+\mu_{1j})\phi'(c_j)$$
$$\frac{\partial^2 l}{\partial c \partial c_j} = \sum_{i=1}^{n}(1-D_{ij})[-\frac{z_{ij}}{\Phi(c_j)(\Phi(c+\alpha_{1i}+\mu_{1j}))^2} - \frac{(1-z_{ij})}{(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))^2}]$$
$$\times \Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})\phi(c_j)\phi(c+\alpha_{1i}+\mu_{1j})$$
$$+\sum_{i=1}^{n}(1-Dij)[\frac{z_{ij}}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})} - \frac{(1-z_{ij})}{1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})}]\phi(c_j)\phi(c+\alpha_{1i}+\mu_{1j})$$

Similar to the unimodal setup, we have, as $n \to \infty$,

$$\frac{1}{n}E[-\frac{\partial^2 l}{\partial c^2}] = \frac{1}{n}E[E[-\frac{\partial^2 l}{\partial c^2}|\alpha_{1i}]]$$

$$= \sum_{j=1}^{n_G}(1-\pi_j)E[\frac{1}{n}\sum_{i=1}^{n}[\frac{(\Phi(c_j)\phi(c+\alpha_{1i}+\mu_{1j}))^2}{\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j})(1-\Phi(c_j)\Phi(c+\alpha_{1i}+\mu_{1j}))}]]$$

$$\to \sum_{j=1}^{n_G}(1-\pi_j)\Phi(c_j)\psi_2((c+\mu_{1j}),\Phi(c_j),\tau_1^2) = \zeta_{11} \text{ (ψ_2 is defined earlier).}$$

$$\frac{1}{n}E[-\frac{\partial^2 l}{\partial c \partial c_j}] \to (1-\pi_j)\frac{\Phi(\frac{c+\mu_{1j}}{\sqrt{1+\tau_1^2}})(\phi(c_j))^2}{(\Phi(c_j))^2(1-\Phi(c_j)\Phi(\frac{c+\mu_{1j}}{\sqrt{1+\tau_1^2}}))} = \zeta_{12} \text{ (say),}$$

$$\frac{1}{n}E[-\frac{\partial^2 l}{\partial (c_j)^2}] \to (1-\pi_j)\phi(c_j)\psi_1((c+\mu_{1j}),\Phi(c_j),\tau_1^2) = \zeta_{22} \text{ (ψ_1 is defined earlier).}$$

Denoting (c, c_j, τ_2^2) by $\theta_{2,j}$, it follows that $\sqrt{n}(\hat{\theta}_{2,j}^{(n)} - \theta_{2,j})$ asymptotically follows $N(0, \Sigma)$, where

$$\Sigma = \begin{pmatrix} \zeta_{11} & \zeta_{12} & 0 \\ \zeta_{12} & \zeta_{22} & 0 \\ 0 & 0 & 2\tau_1^4 \end{pmatrix}$$

Putting the estimates of the parameters evaluated above, we can estimate ψ_1 and ψ_2 that eventually provide the estimates of the variances of the parameters.

2.6 Simulation Protocol and Goodness of fit with Real Data

The underlying model assumption of RIBBON readily leads to a protocol for continuous single-cell gene expression data simulation. Given any data, cell-specific and gene-specific parameters can be estimated and subsequently used for data simulation. We study the accuracy of different single-cell gene expression models based on benchmarking real datasets. We have used six single-cell expression data (see Section 2.8) to assess the performance of RIBBON. We fit the real datasets using existing models for single-cell RNA-seq data, and compare their goodness of fit. DESingle, SC2P, MAST, and scDD are used as candidates for comparison because of their applicability and availability of their codes.

First, we estimate the gene-specific and cell-specific parameters by fitting the individual models to the data to assess the goodness of fit by different methods. Using these estimated parameters and the assumed statistical model for each of the methods, we generate one pseudo-dataset for each method with the same number of cells and the same number of

genes as in the original dataset. The data generation scheme assures that the structure of original dataset is maintained in the pseudo-data. Empirical CDFs from expression values of every gene are estimated from the original dataset and the simulated dataset generated by each method. The empirical CDF function for data $\{X_1, X_2, \ldots, X_n\}$ is defined as $\hat{F}_n(x) = \frac{1}{n}\sum_{i=1}^{n} I\{X_i \leq x\}$. We calculate two-sample Kolmogorov-Smirnov (KS) statistics for every gene with empirical CDFs from the original data and the simulated data. For two independent datasets $\{X_1, X_2, \ldots, X_{n_1}\}$ and $\{Y_1, Y_2, \ldots, Y_{n_2}\}$, the KS statistic between these two samples is defined as $\sqrt{\frac{n_1 n_2}{(n_1+n_2)}} sup_x |\hat{F}_{1,n_1}(x) - \hat{F}_{2,n_2}(x)|$ where $\hat{F}_{1,n_1}$ and $\hat{F}_{2,n_2}$ are empirical CDFs obtained from original data and simulated data respectively. If a model fits well to a gene expression profile, the empirical CDF $\hat{F}_{2,n_2}$ is expected to be a good approximation of the empirical CDF from the original data $\hat{F}_{1,n_1}$ and hence lower value of KS statistic is indicative of better fit. However, since the distribution of single-cell expression is discontinuous, the distribution of KS statistic does not have a distribution-free property. It is well known that the KS statistic for discrete distribution is stochastically smaller than that for a continuous distribution. The exact cutoff depends on the discrete distribution under consideration [107, 43, 79]. Instead of measuring KS statistics based on zero-inflated continuous data, we find the goodness of fit statistic based on two parts: KS statistic based on continuous part and absolute difference in the proportion of zeros. The boxplots with KS statistics for individual genes from all six datasets are shown in Figure 2.6. The boxplots for absolute difference in zero proportions from original and simulated data are shown in Figure 2.7.

In all of the datasets under consideration, RIBBON outperforms other methods in the goodness of fit for modeling nonzero expressions. The improvement of RIBBON over MAST can be attributed to modeling the fraction of genes as bimodal distribution. RIBBON and MAST behave similarly in estimating zero proportions from the data.

2.7 Discussion

We propose a model to analyze single-cell RNA-seq data. One can extend the distributional characterization to other analyses with single-cell RNA-seq data like differential expression analysis, gene set enrichment analysis, cell clustering, lineage mapping, spatial mapping, cell cycle modeling, etc. We distinguish technical zeros from biological zeros so that an independent user can include that proportion of biological zeros in the analysis. Our model is bimodal, capturing the on and off nature of single-cell gene expression levels.

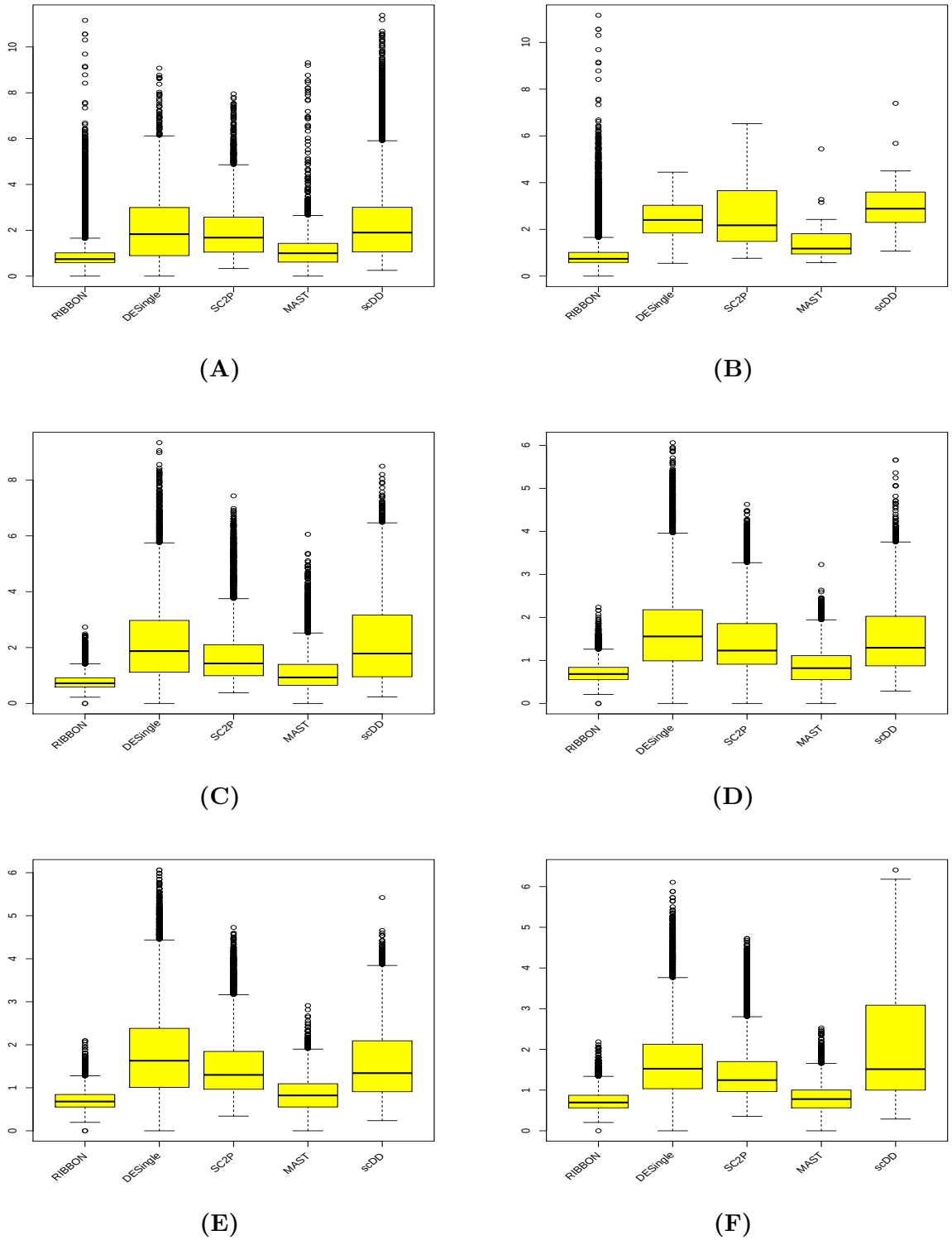


Figure 2.6: (A),(B),(C),(D),(E) and (F) represent represent boxplots of KS statistic for nonzero expressions only from different methods for six real datasets. RIBBON outperforms all other existing methods in accuracy. Difference between RIBBON and MAST is smallest.

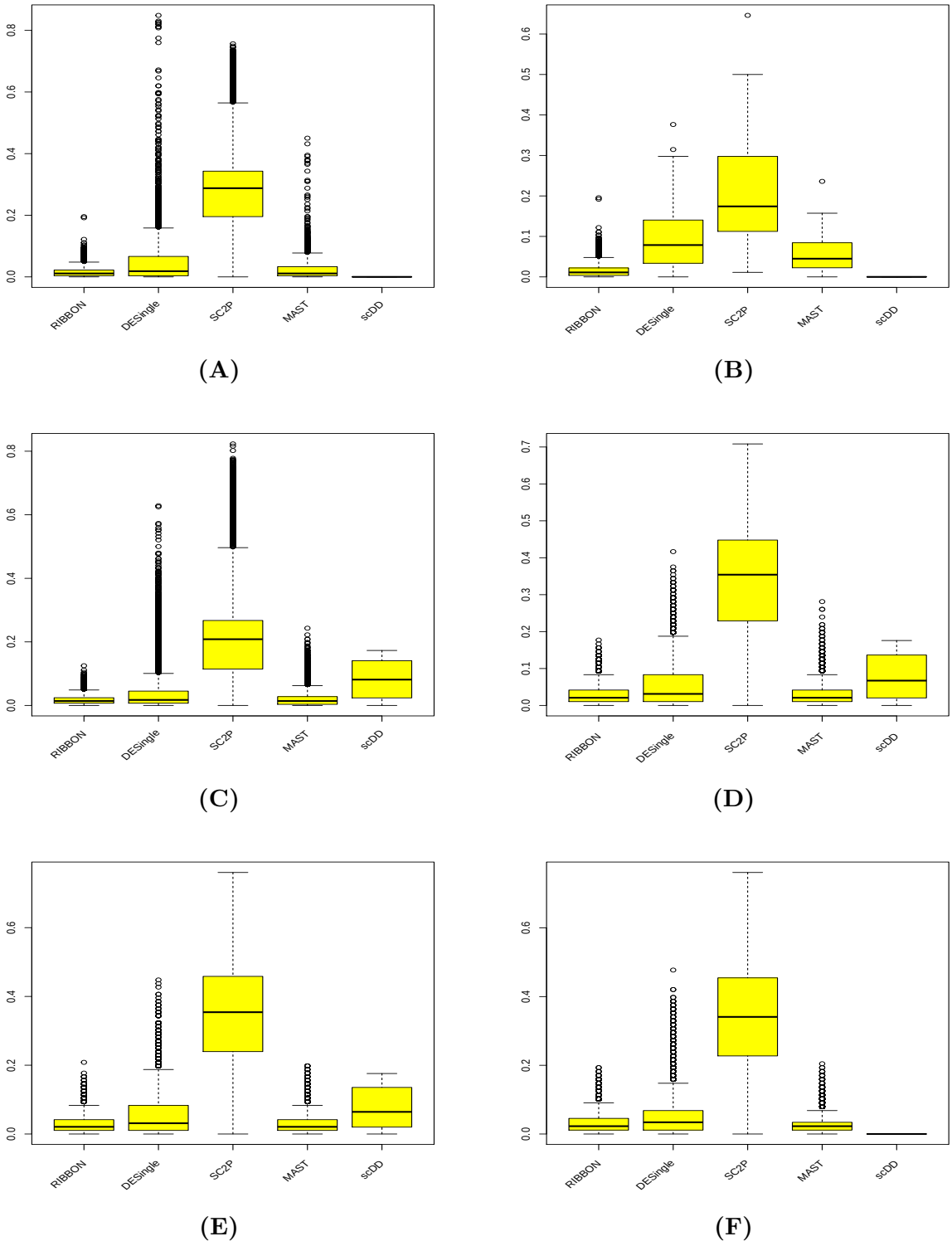


Figure 2.7: (A),(B),(C),(D),(E) and (F) represent absolute difference in proportion of zeros between original data and simulated data for different methods for six real datasets. RIBBON outperforms all other existing methods in accuracy. Difference between RIBBON and MAST is smallest.

It can capture cell subgroup effects, cell-specific effects, and gene effects since we assume random cell-specific effects. One can estimate the parameters easily with the help of an EM-type algorithm which can also be formulated as Iteratively Reweighted Least Squares algorithm. Extensive simulation shows that the parameters can be estimated reliably using this algorithm. Our estimation process is scalable for time and memory. The method offers promising accuracy in terms of Kolmogorov-Smirnov statistic on real data. Hence it can readily be used for all types of analysis with single-cell RNA-seq data.

2.8 Brief description of real datasets

(I) HSMM data (GEO Accession id: GSE52529): Single-cell RNA-sequencing was performed on Human Skeletal Muscle Myoblasts [117] to exploit variation in gene expression. The study aimed to reveal regulatory circuitry governing cell differentiation and other biological processes. Single-cell mRNA sequencing was performed on 271 cells using TruSeq protocol.
(II) Lung data (GEO Accession id: GSE52583): Treutlein et al. [121] generated this dataset to construct pseudotime based on mouse lung RNA-seq data. The authors performed whole transcriptome analysis on distal mouse lung epithelial cells from various developmental stages of mouse embryos and adult mice. We considered a subset of genes from data used by Qiu et al. in our analysis.
(III) HSCs and CML stem cells (GEO Accession id: GSE81730): To distinguish CML stem cell markers from hematopoietic stem cell markers, single-cell RNA-seq was performed on these two types of cells isolated from the same patient [75]. Whole transcriptome data are available on 288 CML stem cells.
(IV, V & VI) G1, S and G2M cells (ArrayExpress Accession id: E-MTAB-2805): Buettner et al. [16] aimed to study gene expression patterns at the single-cell level across the different cell cycle stages in mESC. A Single-cell RNA-Seq experiment was performed on mouse mESC cells that were flow cytometry sorted into G1, S, and G2M phases of the cell cycle. These three types of cells constituted three different datasets in our analysis.

2.9 Code and software availability

Reproducible codes for all figures, data, and software for RIBBON are available at: http://github.com/indranillab/ribbon .

Chapter 3: Testing differential scRNA-seq expression data

3.1 Introduction

Once we fit a distribution to scRNA data, we can test for differential expression that identifies the difference in cell-specific effects in two different populations. It is clear that the appropriate testing for differential expression requires new development that is beyond the realm of bulk RNA data analysis. Our model RIBBON already captures the different typical characteristics of scRNA-seq data. Now we proceed to develop a statistical method to test whether the differential expressions are same in two different groups. These two groups may be normal and tumor cells [28], cells belonging to two different stages of a disease [100], etc. Existing approaches [39, 131, 81] do not differentiate technical zeros from biological zeros. The literature is not rich with tests to perform testing based on mixing proportion under bimodality assumption.

However, we observe that it is difficult to apply RIBBON directly for this testing purpose. To capture differences in subgroup-specific cell effects, we marginalise the cell-specific effect on the overall expression level. Since the differential expression is performed on a gene-by-gene basis, it is really impossible to distinguish sampling zeros from biological zeros. At the same time, it is important to separate technical zeros from overall zero occurrences. So we modify our original model a little for differential expression testing.

3.2 Developing testing procedure

Let y_{ij} be the logarithm expression level of the j-th gene in cell i and z_{ij} be an indicator variable denoting the event that gene j is detected in cell i. To incorporate two different types of zeros, we assume that the probability that j-th gene is detected in cell i is given

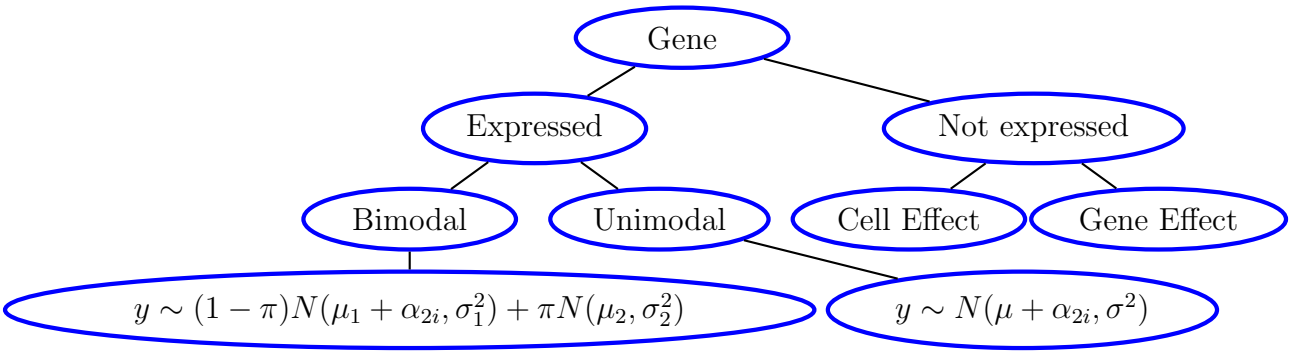


Figure 3.1: Model used for testing differential expression

by $\Phi(c_j + \alpha_i)$. c_j is the gene-specific fixed effect for dropout, and α_i is the cell-specific fixed effect. Note that c_j takes care of both biological zero and sampling zero we described earlier and α_i is the factor behind technical zero. We assume that the distribution is either unimodal or bimodal normal. Using the criterion described before, we model gene-specific expressions, either unimodal or bimodal. Conditional on the fact that the gene expression distribution is bimodal, we introduce another indicator variable D_{ij}, a latent variable. $D_{ij} = 0$ means expression level of gene j in cell i comes from mode 1, otherwise from mode 2.

With all these notations define above, we first present our model for testing differential expression in a diagrammatic way (Figure 3.1), followed by the mathematical presentation.

When the distribution of gene expression is bimodal, the data generation model for gene j can be mathematically described as:

$$
\begin{aligned}
&Pr[z_{ij} = 1] = \Phi(c_j + \alpha_i), \\
&P[D_{ij} = 1] = \pi_j, \\
&(y_{ij}|z_{ij} = 0) \equiv 0, \\
&(y_{ij}|z_{ij} = 1, D_{ij} = 0) \sim N(\alpha_{2i} + \mu_{1j}, \sigma_{1j}^2) \\
&(y_{ij}|z_{ij} = 1, D_{ij} = 1) \sim N(\mu_{2j}, \sigma_{2j}^2) \\
&\alpha_{2i} \sim N(0, \tau_2^2)
\end{aligned} \tag{3.1}
$$

For a gene showing bimodal characteristic, the likelihood function is given by:

$$L(\theta_j|\{D_{ij}\}_{i=1}^n,\{z_{ij}\}_{i=1}^n,\{y_{ij}\}_{i=1}^n)=\prod_{i=1}^n\Big\{\pi_j^{D_{ij}}(1-\pi_j)^{(1-D_{ij})}\phi(\alpha_{2i};0,\tau_2)$$
$$\times(\Phi(c_j+\alpha_i))^{z_{ij}}(1-\Phi(c_j+\alpha_i))^{(1-z_{ij})}\big((\phi(y_{ij};\alpha_{2i}+\mu_{1j},\sigma_{1j}))^{(1-D_{ij})}(\phi(y_{ij};\mu_{2j},\sigma_{2j}))^{D_{ij}}\big)^{z_{ij}}\Big\}$$
$$=L_\pi\times L_0\times L_2 \tag{3.2}$$

where $L_\pi=\prod_{i=1}^n\pi_j^{D_{ij}}(1-\pi_j)^{(1-D_{ij})}$, $L_0=\prod_{i=1}^n(\Phi(c_j+\alpha_i))^{z_{ij}}(1-\Phi(c_j+\alpha_i))^{(1-z_{ij})}$ and $L_2=\prod_{i=1}^n\big((\phi(y_{ij};\alpha_{2i}+\mu_{1j},\sigma_{1j}))^{(1-D_{ij})}(\phi(y_{ij};\mu_{2j},\sigma_{2j}))^{D_{ij}}\big)^{z_{ij}}\phi(\alpha_{2i};0,\tau_2)$ where the set of parameters is denoted by $\theta_j=(\pi_j,c_j,\mu_{1j},\mu_{2j},\sigma_{1j}^2,\sigma_{2j}^2,\tau_2^2)'$.

When the distribution of a gene expression is unimodal, the generation model can be mathematically represented as:

$$\begin{aligned}&Pr[z_{ij}=1]=\Phi(c_j+\alpha_i),\\&(y_{ij}|z_{ij}=0)\equiv 0,\\&(y_{ij}|z_{ij}=1)\sim N(\alpha_{2i}+\mu_j,\sigma_j^2)\\&\alpha_{2i}\sim N(0,\tau_2^2),\end{aligned} \tag{3.3}$$

Using the notation $\theta_j=(c_j,\mu_j,\sigma_j^2,\tau_2^2)$, the likelihood function for gene j is:

$$\begin{aligned}&L(\theta_j|\{z_{ij}\}_{i=1}^n,\{y_{ij}\}_{i=1}^n)\\&=\prod_{i=1}^n(\Phi(c_j+\alpha_i))^{z_{ij}}(1-\Phi(c_j+\alpha_i))^{(1-z_{ij})}(\phi(y_{ij};\alpha_{2i}+\mu_j,\sigma_j))^{z_{ij}}\\&=L_0\times L_1\end{aligned} \tag{3.4}$$

where $L_0=\prod_{i=1}^n(\Phi(c_j+\alpha_i))^{z_{ij}}(1-\Phi(c_j+\alpha_i))^{(1-z_{ij})}$ and $L_1=\prod_{i=1}^n(\phi(y_{ij};\alpha_{2i}+\mu_j,\sigma_j))^{z_{ij}}$.

We have stated that the probability of a dropout corresponding to cell i and gene j is $\Phi(\alpha_i+c_j)$ where α_i is the cell specific effect and c_j is the gene specific effect. For a given gene j, let $c_j^{(1)}$ and $c_j^{(2)}$ be the gene specific parameters for two groups. Similarly, denote by $z_{ij}^{(k)}$, the z_{ij} values belonging to k-th group; $k=1,2$.

The models proposed by (3.1) and (3.3) taking into account the possible bimodal and unimodal nature of expression values can be used for testing differential gene expression profiles between two groups. It has the flexibility to test different hypotheses depending on the problem and the interest of the researcher. Here we describe two tests, one for unimodal and another for bimodal expression distribution. It is clear that to see any gene-specific effect in two groups is of utmost interest. This test should depend on c_js. Note that L_0 in the log-likelihoods of both unimodal and bimodal models, contains the same expression relating to c_j whereas L_1 and L_2 do not depend on c_j for any group. So this test needs to be done for both types of distributions. Moreover, for unimodal distribution, we are mainly interested to see any shift in mean and variance among the two groups. But in a bimodal distribution, it is very important to see whether the proportions coming from different modes are same in two groups.

To test whether the gene expression profiles are same in two groups when the distributions are unimodal, our hypotheses of interest would be:

$$H_0 : (c_j^{(1)}, \mu_j^{(1)}, \sigma_j^{(1)}) = (c_j^{(2)}, \mu_j^{(2)}, \sigma_j^{(2)}) \text{ against } H_1 : (c_j^{(1)}, \mu_j^{(1)}, \sigma_j^{(1)}) \neq (c_j^{(2)}, \mu_j^{(2)}, \sigma_j^{(2)})$$

Now from (3.4), it is clear that information of c_j comes from only L_0 whereas L_1 provides information for other parameters of the hypotheses. So we construct the test for $H_0 : c_j^{(1)} = c_j^{(2)}$ using L_0 only and that for $H_0 : (\mu_j^{(1)}, \sigma_j^{(1)}) = (\mu_j^{(2)}, \sigma_j^{(2)})$ using L_1 only.

First we describe the test for $H_0 : c_j^{(1)} = c_j^{(2)}$, vs $H_1 : c_j^{(1)} \neq c_j^{(2)}$. Let n_1 and n_2 be the number of observations for Group 1 and Group 2 respectively for gene j, $j = 1, \ldots, N$. Use the notation, $\boldsymbol{z}_j^{(k)} = (z_{1j}^{(k)}, \ldots, z_{n_k j}^{(k)})$ for $k = 1, 2$. Now define $l_k(c_j^{(k)} | \boldsymbol{z}_j^{(k)})$ for $k = 1, 2$, as the marginal log-likelihood of $z_{ij}^{(k)}$'s conditioned on estimated values of α_i's, i.e.,

$$l_k(c_j^{(k)} | \boldsymbol{z}_j) = \sum_{i=1}^{n_k} [z_{ij}(\log(\Phi(c_j^{(k)} + \hat{\alpha}_i^{(k)}))) + (1 - z_{ij})(\log(1 - \Phi(c_j^{(k)} + \hat{\alpha}_i^{(k)})))]$$

Similarly, define by $l_0()$, the likelihood of the j-th gene for the combined data. The $-2\log$ of likelihood ratio test statistic based on the Bernoulli model is given by

$$T_0 = 2\big(l_1(\hat{c}_j^{(1)} | \boldsymbol{z}_j^{(1)}) + l_2(\hat{c}_j^{(2)} | \boldsymbol{z}_j^{(2)}) - l_0(\hat{c}_j | \boldsymbol{z}_j^{(1)}, \boldsymbol{z}_j^{(2)})\big) \tag{3.5}$$

The first test RIBBON I is based on unimodality assumption. In differential expression analysis, it is important to capture difference in cell specific effects across groups. So we use distribution of y_{ij}'s after marginalizing over α_{2i}'s. We denote by $y_{ij}^{(k)}$, the log expression value in cell i for the j-th gene belonging to group k. Note that $y_{1j}^{(1)}$, $y_{2j}^{(1)}$,..., $y_{n_1 j}^{(1)}$ are

i.i.d. observations from $N(\mu_j^{(1)}, \sigma_j^{2(1)})$ and $y_{1j}^{(2)}$, $y_{2j}^{(2)}$,..., $y_{n_2 j}^{(2)}$ are i.i.d. observations from $N(\mu_j^{(2)}, \sigma_j^{2(2)})$. Denoting $\boldsymbol{y}_j^{(k)} = (y_{1j}^{(k)}, \ldots, y_{n_1 j}^{(k)})$ for $k = 1, 2$, the log-likelihood function for gene j from group k is:

$$l_k(\mu_j^{(k)}, \sigma_j^{2(k)} | \boldsymbol{y}_j^{(k)}, \boldsymbol{z}_{1j}^{(k)}) = \sum_{i=1}^{n_k} z_{ij}^{(k)} \log(\phi(y_{ij}^{(k)}; \mu_j^{(k)}, \sigma_j^{2(k)})); \qquad k = 1, 2 \tag{3.6}$$

Under H_0, when $\mu_j^{(1)} = \mu_j^{(2)} = \mu_j, \sigma_j^{(1)} = \sigma_j^{(2)} = \sigma_j^2$, the log-likelihood function for gene j is,

$$l_0(\mu_j, \sigma_j^2 | \boldsymbol{y}_j^{(1)}, \boldsymbol{z}_{1j}^{(1)}, \boldsymbol{y}_j^{(2)}, \boldsymbol{z}_{1j}^{(2)}) = \sum_{k=1}^{2} \sum_{i=1}^{n_k} z_{ij}^{(k)} \log(\phi(y_{ij}^{(k)}; \mu_j, \sigma_j^2)) \tag{3.7}$$

Similarly defining l_0 as the log-likelihood of combined data under H_0, we have the $-2\log$-likelihood ratio test statistic for testing $H_0 : (\mu_j^{(1)}, \sigma_j^{2(1)}) = (\mu_j^{(2)}, \sigma_j^{2(2)})$ vs $H_1 : (\mu_j^{(1)}, \sigma_j^{2(1)}) \neq (\mu_j^{(2)}, \sigma_j^{2(2)})$ as

$$T_1 = 2\big(l_1(\hat{\mu}_j^{(1)}, \hat{\sigma}_j^{2(1)} | \boldsymbol{y}_j^{(1)}, \boldsymbol{z}_j^{(1)}) + l_2(\hat{\mu}_j^{(2)}, \hat{\sigma}_j^{2(2)} | \boldsymbol{y}_j^{(2)}, \boldsymbol{z}_j^{(2)}) - l_0(\hat{\mu}_j, \hat{\sigma}_j^2 | \boldsymbol{y}_j^{(1)}, \boldsymbol{z}_j^{(1)}, \boldsymbol{y}_j^{(2)}, \boldsymbol{z}_j^{(2)})\big) \tag{3.8}$$

To test we need to find the asymptotic distribution of RIBBON I which is given by the test statistic $T_0 + T_1$. Since the MLEs of different parameters on both T_0 and T_1 cannot be obtained in a compact form, we use EM algorithm to estimate them and consequently develop an asymptotic test procedure based on the statistic $T_0 + T_1$ for testing in case of unimodal distribution.

Theorem 1. *Under* $H_0 : c_j^{(1)} = c_j^{(2)}, (\mu_j^{(1)}, \sigma_j^{2(1)}) = (\mu_j^{(2)}, \sigma_j^{2(2)})$ *we have,*

$$T_0 + T_1 \xrightarrow{d} \chi_3^2 \text{ as } n_1, n_2 \to \infty$$

Note that the statistic in the theorem is the sum of two statistics T_0 and T_1. We first study the asymptotic distributions of these two statistics in the next two lemmas and merge them to prove Theorem 1.

Lemma 1.

$$T_0 \xrightarrow{d} \chi_1^2 \text{ as } n_1, n_2 \to \infty \text{ under } H_0$$

To prove Lemma 1, we need to make the following assumption:

Assumption 1. $\frac{1}{n}\sum_{i=1}^{n}\frac{\phi^2(c_j+\alpha_i)}{\Phi(c_j+\alpha_i)(1-\Phi(\alpha_i+c_j))}$ *converges to a constant, say* κ*, as* $n\to\infty$

Note that $f(x)=\frac{\phi^2(x)}{\Phi(x)(1-\Phi(x))}$ is a bounded function with $|f(x)|\leq f(0)$. If the sequence does not converge, it will oscillate. So, this is a very general assumption, that holds in any single-cell RNA-seq data. For a given cell type probability of zero expression remains same across cells and hence this assumption is unlikely to fail to hold. Note that the points at which the sequence oscillates are very close to each other, all lying in $(0,1)$ interval. So, even if this assumption fails, we can either take Cesàro mean or the mean of oscillatory points if the number of such points is finite, as an approximate value of κ.

Proof. To prove Lemma 1, first we have to estimate the parameters $\{\alpha_i\}_{i=1}^n$, $\{c_j\}_{j=1}^N$ where $n=n_1+n_2$. Based on n cells and N genes and using L_0 function in (3.4), we have the likelihood function involving z_{ij} as,

$$L(\{c_j\}_{j=1}^N,\{\alpha_i\}_{i=1}^n|\{\{z_{ij}\}_{i=1}^n\}_{j=1}^N)=\prod_{j=1}^{N}\prod_{i=1}^{n}(\Phi(c_j+\alpha_i))^{z_{ij}}(1-\Phi(c_j+\alpha_i))^{(1-z_{ij})}$$

Hence, the log likelihood is given by,

$$\begin{aligned}&l(\{c_j\}_{j=1}^N,\{\alpha_i\}_{i=1}^n|\{\{z_{ij}\}_{i=1}^n\}_{j=1}^N)\\&=\sum_{j=1}^{N}\sum_{i=1}^{n}[z_{ij}\log(\Phi(c_j+\alpha_i))+(1-z_{ij})\log(1-\Phi(c_j+\alpha_i))]\end{aligned}$$

We maximize the likelihood with respect to $\{\alpha_i\}_{i=1}^n$ and $\{c_j\}_{j=1}^N$ using fisher scoring method. The partial derivatives are,

$$\frac{\partial l}{\partial c_j}=\sum_{i=1}^{n}\frac{(z_{ij}-\Phi(c_j+\alpha_i))\phi(c_j+\alpha_i)}{\Phi(c_j+\alpha_i)(1-\Phi(\alpha_i+c_j))},\ \frac{\partial l}{\partial \alpha_i}=\sum_{j=1}^{N}\frac{(z_{ij}-\Phi(c_j+\alpha_i))\phi(c_j+\alpha_i)}{\Phi(c_j+\alpha_i)(1-\Phi(\alpha_i+c_j))}$$

and the expected second derivatives are,

$$E\Big[-\frac{\partial^2 l}{\partial c_j^2}\Big]=\sum_{i=1}^{n}\frac{\phi^2(c_j+\alpha_i)}{\Phi(c_j+\alpha_i)(1-\Phi(\alpha_i+c_j))}$$

$$E\Big[-\frac{\partial^2 l}{\partial \alpha_i^2}\Big] = \sum_{j=1}^{N} \frac{\phi^2(c_j+\alpha_i)}{\Phi(c_j+\alpha_i)(1-\Phi(\alpha_i+c_j))}$$
$$E\Big[-\frac{\partial^2 l}{\partial \alpha_i \partial c_j}\Big] = \frac{\phi^2(c_j+\alpha_i)}{\Phi(c_j+\alpha_i)(1-\Phi(\alpha_i+c_j))}$$

Now, the parameter of our interest is c_j because the test of hypothesis is based on testing $c_j^{(1)} = c_j^{(2)}$. So, keeping α_is fixed, we maximize the likelihood function to estimate c_j, $c_j^{(1)}$ and $c_j^{(2)}$.

To estimate initial α_is we ignore the non-diagonal terms in information matrix in the Fisher-Scoring method, and estimate the parameters using iteration with estimates at the k-th step as:

$c_j^{(k)} = c_j^{(k-1)} + \frac{s_{1,c_j}^{(k-1)}}{s_{2,c_j}^{(k-1)}}$ where

$s_{1,c_j}^{(k-1)} = \sum_{i=1}^{n} \frac{(z_{ij}-\Phi(c_j^{(k-1)}+\alpha_i^{(k-1)}))\phi(c_j^{(k-1)}+\alpha_i^{(k-1)})}{\Phi(c_j^{(k-1)}+\alpha_i^{(k-1)})(1-\Phi(\alpha_i^{(k-1)}+c_j^{(k-1)}))}$ and $s_{2,c_j}^{(k-1)} = \sum_{i=1}^{n} \frac{\phi^2(c_j^{(k-1)}+\alpha_i^{(k-1)})}{\Phi(c_j^{(k-1)}+\alpha_i^{(k-1)})(1-\Phi(\alpha_i^{(k-1)}+c_j^{(k-1)}))}$,

$\alpha_i^{(k)} = \alpha_i^{(k-1)} + \frac{s_{1,\alpha_i}^{(k-1)}}{s_{2,\alpha_i}^{(k-1)}}$ where

$s_{1,\alpha_i}^{(k-1)} = \sum_{j=1}^{N} \frac{(z_{ij}-\Phi(c_j^{(k-1)}+\alpha_i^{(k-1)}))\phi(c_j^{(k-1)}+\alpha_i^{(k-1)})}{\Phi(c_j^{(k-1)}+\alpha_i^{(k-1)})(1-\Phi(\alpha_i^{(k-1)}+c_j^{(k-1)}))}$ and $s_{2,\alpha_i}^{(k-1)} = \sum_{j=1}^{N} \frac{\phi^2(c_j^{(k-1)}+\alpha_i^{(k-1)})}{\Phi(c_j^{(k-1)}+\alpha_i^{(k-1)})(1-\Phi(\alpha_i^{(k-1)}+c_j^{(k-1)}))}$.

Please note that the parameters would converge to the actual MLE, because the log-likelihood function is a strictly concave function of the parameters and, as a result, MLE is unique.

Also, to make the model identifiable and to capture the gene-specific effects on groups, we set, $\sum_i \alpha_i^{(1)} = \sum_i \alpha_i^{(2)} = 0$ where $\alpha_i^{(k)}$ are the values of α_is restricted to k-th group. So, after each iteration, we set $\alpha_i^{(1)} = \alpha_i^{(1)} - \bar{\alpha}^{(1)}$ and $\alpha_i^{(2)} = \alpha_i^{(2)} - \bar{\alpha}^{(2)}$ where $\bar{\alpha}^{(k)} = \frac{1}{n_k}\sum_{i=1}^{n_k} \alpha_i^{(k)}$

c_j's estimated in this step are taken to be estimated c_j's under null distribution. To estimate $\{c_j^{(1)}\}_{j=1}^N$ and $\{c_j^{(2)}\}_{j=1}^N$, we keep the α_i's same and estimate the parameters under alternative distribution from the two groups separately following the same method.

Let n be the number of cells and N be the number of genes and c_j and α_i are actual parameter values, $i = 1, \ldots, n; j = 1, \ldots, N$. Then by Section 2.2 of [38], $\hat{\alpha}_i$s and $\hat{c}_j$s estimated in the initial step asymptotically follow normal distribution with $\sqrt{n}(\hat{c}_j - c_j) = O(1)$ for each $j = 1, \ldots, N$ and $(\hat{\alpha}_i - \alpha_i) = O(1)$ for each $i = 1, \ldots, n$ as $n \to \infty$.

Let n_1 and n_2 be the number of cells for j-th gene in two groups respectively. Then using Section 2.2 of [38], as $n_k \to \infty$, $\sqrt{n_k}(\hat{c}_j^{(k)} - c_j) = O(1)$ for $k = 1, 2$. Now we will find

the asymptotic distribution of $\hat{c}_j^{(1)}$ and $\hat{c}_j^{(2)}$. Denote the joint log-likelihood function for all observations from group 1 by l_1 and note that $l_1'(\hat{c}_j^{(1)}) = 0$ since $\hat{c}_j^{(1)}$ is the MLE of $c_j^{(1)}$. Thus we have,

$l_1'(\hat{c}_j^{(1)}) = l_1'(c_j) + (\hat{c}_j^{(1)} - c_j)l_1''(c_j) + \cdots$

So,

$$\sqrt{n_1}(\hat{c}_j^{(1)} - c_j) \approx -\sqrt{n_1}\frac{l'(c_j)}{l''(c_j)} = \frac{\frac{l_1'(c_j)}{\sqrt{n_1}}}{-\frac{l_1''(c_j)}{n_1}} \tag{3.9}$$

Note that, the denominator in (3.9) is $-\frac{l_1''(c_j)}{n_1} = \frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\phi^2(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))}$. As, $n \to \infty$, $(\hat{\alpha}_i^{(1)} - \alpha_i^{(1)}) = O_p(1)$. Using Assumption 1, $\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\phi^2(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))} \xrightarrow{P} \kappa$. Similarly, $\frac{1}{n_2}\sum_{i=1}^{n_2}\frac{\phi^2(c_j+\hat{\alpha}_i^{(2)})}{\Phi(c_j+\hat{\alpha}_i^{(2)})(1-\Phi(\hat{\alpha}_i^{(2)}+c_j))} \xrightarrow{P} \kappa$ and $\frac{1}{n}\sum_{i=1}^{n}\frac{\phi^2(c_j+\hat{\alpha}_i)}{\Phi(c_j+\hat{\alpha}_i)(1-\Phi(\hat{\alpha}_i+c_j))} \xrightarrow{P} \kappa$

Now, the numerator is $\frac{l_1'(c_j)}{\sqrt{n_1}} = \frac{1}{\sqrt{n_1}}\sum_{i=1}^{n_1}\frac{(z_{ij}-\Phi(c_j+\hat{\alpha}_i^{(1)}))\phi(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))}$.

Take, $X_i = \frac{(z_{ij}-\Phi(c_j+\hat{\alpha}_i^{(1)}))\phi(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))}$, Then $E[X_i] = \frac{(\Phi(c_j+\alpha_i^{(1)})-\Phi(c_j+\hat{\alpha}_i^{(1)}))\phi(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))}$, and $E[X_i^2] = \frac{\phi^2(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))}$.

Now, $\frac{l_1'(c_j)}{\sqrt{n_1}} = \frac{1}{\sqrt{n_1}}\sum_{i=1}^{n_1}X_i$, so that $Var[\frac{l_1'(c_j)}{\sqrt{n_1}}] \xrightarrow{P} \frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\phi^2(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))}$.

Now, from equation (3.9) , it follows that, $\sqrt{n_1}(\hat{c}_j^{(1)}-c_j)$ asymptotically follows $N(0, \frac{n_1}{s^1_{2,c_j}})$ where $s^1_{2,c_j} = \sum_{i=1}^{n_1}\frac{\phi^2(c_j+\hat{\alpha}_i^{(1)})}{\Phi(c_j+\hat{\alpha}_i^{(1)})(1-\Phi(\hat{\alpha}_i^{(1)}+c_j))}$ [We have shown that $\sqrt{n_1}(\hat{c}_j^{(1)} - c_j)$ asymptotically follows normal distribution before.]

Similarly, $\sqrt{n_2}(\hat{c}_j^{(2)}-c_j)$ asymptotically follows $N(0, \frac{n_2}{s^2_{2,c_j^{(2)}}})$ where $s^2_{2,c_j^{(2)}} = \sum_{i=1}^{n_2}\frac{\phi^2(c_j^{(2)}+\hat{\alpha}_i^{(2)})}{\Phi(c_j^{(2)}+\hat{\alpha}_i^{(2)})(1-\Phi(\hat{\alpha}_i^{(2)}+c_j^{(2)}))}$

Let us denote by l_0, the joint log-likelihood of the combined data, l_1 denote the log-likelihood from group 1 and let l_2 denote the log-likelihood from group 2. Note that, $l_0 = l_1 + l_2$. Now we derive the distribution of T_0. First note that,

$$T_0 = 2[l_1(\hat{c}_j^{(1)}) + l_2(\hat{c}_j^{(2)}) - l_0(\hat{c}_j)] = 2[l_1(\hat{c}_j^{(1)}) + l_2(\hat{c}_j^{(2)}) - l_1(\hat{c}_j) - l_2(\hat{c}_j)]$$

Now,

$$l_1(\hat{c}_j) = l_1(\hat{c}_j^{(1)}) + (\hat{c}_j - \hat{c}_j^{(1)})l_1'(\hat{c}_j^{(1)}) + \frac{1}{2}(\hat{c}_j - \hat{c}_j^{(1)})^2 l_1''(\hat{c}_j^{(1)}) + o_p(1) \tag{3.10}$$

Similarly,

$$l_2(\hat{c}_j) = l_2(\hat{c}_j^{(2)}) + (\hat{c}_j - \hat{c}_j^{(2)})l_2'(\hat{c}_j^{(2)}) + \frac{1}{2}(\hat{c}_j - \hat{c}_j^{(2)})^2 l_2''(\hat{c}_j^{(2)}) + o_p(1) \tag{3.11}$$

Note that, under null hypothesis, $-\frac{l_1''(\hat{c}_j^{(1)})}{n_1} \to \kappa$ and $-\frac{l_2''(\hat{c}_j^{(2)})}{n_2} \to \kappa$ for some $\kappa > 0$. [This happens due to the fact that, α_i's are asymptotically normal.]
So, from equation (3.10) and (3.11),

$$T_0 \xrightarrow{d} 2[-\frac{1}{2}(\hat{c}_j - \hat{c}_j^{(1)})^2 l_1''(\hat{c}_j^{(1)}) - \frac{1}{2}(\hat{c}_j - \hat{c}_j^{(2)})^2 l_2''(\hat{c}_j^{(2)})] \xrightarrow{d} \kappa[n_1(\hat{c}_j^{(1)} - \hat{c}_j)^2 + n_2(\hat{c}_j^{(2)} - \hat{c}_j)^2]$$

From equation (3.9), we have $(\hat{c}_j - c_j) = -\frac{l'(c_j)}{l''(c_j)}$ and similarly $(\hat{c}_j^{(1)} - c_j) = -\frac{l_1'(c_j)}{l_1''(c_j)}$ and $(\hat{c}_j^{(2)} - c_j) = -\frac{l_2'(c_j)}{l_2''(c_j)}$.
In addition to this, the fact that $\lim_{n_1\to\infty} -\frac{l_1''(c_j)}{n_1} = \lim_{n_2\to\infty} -\frac{l_2''(c_j)}{n_2} = \lim_{n\to\infty} -\frac{l_0''(c_j)}{n} = \kappa$ together imply that $n_1(\hat{c}_j^{(1)} - c_j) + n_2(\hat{c}_j^{(2)} - c_j) \approx n(\hat{c}_j - c_j)$ or $\hat{c}_j \approx \frac{n_1\hat{c}_j^{(1)} + n_1\hat{c}_j^{(1)}}{n_1+n_2}$
So, $T_0 \xrightarrow{d} \kappa\frac{n_1 n_2}{n_1+n_2}(\hat{c}_j^{(1)} - \hat{c}_j^{(2)})^2$ or, $T_0 \xrightarrow{d} \chi_1^2$ under null hypothesis. □

Now we need to find the asymptotic distribution of T_1. For notational simplicity, we drop the z_{ij} term, i.e., we assume that z_{ij}'s are all 1. If some z_{ij}s take the value 0, we consider the terms with z_{ij} equal to 1 only and apply the asymptotics. Now the effective sample size becomes the total number of observations with z_{ij} equal to 1. We fix a gene j and denote the expression value corresponding to i-th cell in the k-th group to be $Y_i^{(k)}$ which is same as the $y_{ij}^{(k)}$ according to our original notation. Similarly, we drop the subscript j from all the parameters. We shall show that the likelihood ratio test statistic for testing equality of means and variances based on normal likelihood follows χ_2^2 distribution as given in the following Lemma.

Lemma 2. *Under* $H_0 : (\mu_1, \sigma_1^2) = (\mu_2, \sigma_2^2)$,

$$T_1 \xrightarrow{d} \chi_2^2 \text{ as } n_1, n_2 \to \infty.$$

Proof. Let l denote the log likelihood function of normal distribution for a single observation. Let $\hat{\mu}$ and $\hat{\sigma}^2$ be the MLEs of μ and σ^2 respectively, under null distribution. Similarly, let $(\hat{\mu}_1, \hat{\sigma}_1^2)$ and $(\hat{\mu}_2, \hat{\sigma}_2^2)$ be the MLEs of (μ, σ^2) for the two groups under alternative distribution. The -2 log-likelihood ratio statistic can be written as:

$$T_1 = 2\Big[\sum_{i=1}^{n_1} l(\hat{\mu}_1, \hat{\sigma}_1^2|Y_i^{(1)}) + \sum_{i=1}^{n_2} l(\hat{\mu}_2, \hat{\sigma}_2^2|Y_i^{(2)}) - \sum_{i=1}^{n_1} l(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) - \sum_{i=1}^{n_2} l(\hat{\mu}, \hat{\sigma}^2|Y_i^{(2)})\Big]$$

Now, by Taylor's series expansion,

$$\begin{aligned} l(\hat{\mu}_1, \hat{\sigma}_1^2|Y_i^{(1)}) &= l(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + (\hat{\mu}_1 - \hat{\mu})\dot{l}_\mu(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) \\ &+ (\hat{\sigma}_1^2 - \hat{\sigma}^2)\dot{l}_{\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + \frac{1}{2}(\hat{\mu}_1 - \hat{\mu})^2\ddot{l}_{\mu,\mu}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) \\ &+ \frac{1}{2}(\hat{\sigma}_1^2 - \hat{\sigma}^2)^2\ddot{l}_{\sigma^2,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + (\hat{\mu}_1 - \hat{\mu})(\hat{\sigma}_1^2 - \hat{\sigma}^2)\ddot{l}_{\mu,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + \cdots \end{aligned}$$

where $\dot{l}_\mu = \dfrac{\partial l}{\partial \mu}, \ddot{l}_\mu = \dfrac{\partial^2 l}{\partial \mu^2}, \dot{l}_{\sigma^2} = \dfrac{\partial l}{\partial \sigma^2}, \ddot{l}_{\sigma^2,\sigma^2} = \dfrac{\partial^2 l}{\partial (\sigma^2)^2}, \ddot{l}_{\mu,\sigma^2} = \dfrac{\partial^2 l}{\partial \mu \partial \sigma^2} = \dfrac{\partial^2 l}{\partial \sigma^2 \partial \mu}.$

Now, because $\hat{\mu}$ and $\hat{\sigma}^2$ are MLEs under the null distribution,

$$\sum_{i=1}^{n_1} \dot{l}_\mu(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + \sum_{i=1}^{n_2} \dot{l}_\mu(\hat{\mu}, \hat{\sigma}^2|Y_i^{(2)}) = \sum_{i=1}^{n_1} \dot{l}_{\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + \sum_{i=1}^{n_2} \dot{l}_{\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(2)}) = 0.$$

It can be easily shown that

$$\sum_{i=1}^{n_1} (\hat{\mu}_1 - \hat{\mu})(\hat{\sigma}_1^2 - \hat{\sigma}^2)\ddot{l}_{\mu,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + \sum_{i=1}^{n_2} (\hat{\mu}_1 - \hat{\mu})(\hat{\sigma}_1^2 - \hat{\sigma}^2)\ddot{l}_{\mu,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(2)}) \approx 0$$

(because $(n_1+n_2)(\hat{\mu}_1-\hat{\mu})(\hat{\sigma}_1^2-\hat{\sigma}^2) = (\sqrt{(n_1+n_2)}(\hat{\mu}_1-\hat{\mu}))(\sqrt{(n_1+n_2)}(\hat{\sigma}_1^2-\hat{\sigma}^2))$ asymptotically follow W_1W_2 where W_1 and W_2 are two independent mean 0 normal variables. Also, $\frac{1}{(n_1+n_2)}[\sum_{i=1}^{n_1} \ddot{l}_{\mu,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + \sum_{i=1}^{n_2} \ddot{l}_{\mu,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(2)})] \xrightarrow{P} 0$ in probability because $E[\ddot{l}_{\mu,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y)] = 0$ for $Y \sim N(\mu, \sigma^2)$.)

Hence, we have,

$$\begin{aligned} T_1 \approx 2 \sum_{i=1}^{n_1} [&\frac{1}{2}(\hat{\mu}_1 - \hat{\mu})^2\ddot{l}_{\mu,\mu}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)}) + \frac{1}{2}(\hat{\sigma}_1^2 - \hat{\sigma}^2)^2\ddot{l}_{\sigma^2,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(1)})] \\ + \sum_{i=1}^{n_2} [&\frac{1}{2}(\hat{\mu}_2 - \hat{\mu})^2\ddot{l}_{\mu,\mu}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(2)}) + \frac{1}{2}(\hat{\sigma}_2^2 - \hat{\sigma}^2)^2\ddot{l}_{\sigma^2,\sigma^2}(\hat{\mu}, \hat{\sigma}^2|Y_i^{(2)})] \end{aligned}$$

Under null distribution, let us assume $Y_i^{(k)} \sim N(\mu, \sigma^2)$. So,

$$T_1 \approx \frac{n_1(\hat{\mu}_1 - \hat{\mu})^2}{\hat{\sigma}^2} + \frac{n_2(\hat{\mu}_2 - \hat{\mu})^2}{\hat{\sigma}^2} + \frac{n_1(\hat{\sigma}_1^2 - \hat{\sigma}^2)^2}{2\hat{\sigma}^4} + \frac{n_2(\hat{\sigma}_2^2 - \hat{\sigma}^2)^2}{2\hat{\sigma}^4}$$

$$= \frac{n_1 n_2}{(n_1+n_2)} \frac{(\hat{\mu}_1 - \hat{\mu}_2)^2}{\hat{\sigma}^2} + \frac{n_1 n_2}{(n_1+n_2)} \frac{(\hat{\sigma}_1^2 - \hat{\sigma}_2^2)^2}{2\hat{\sigma}^4}$$

Now, $\sqrt{\frac{n_1 n_2}{(n_1+n_2)}} \begin{pmatrix} \hat{\mu}_1 - \hat{\mu}_2 \\ \hat{\sigma}_1^2 - \hat{\sigma}_2^2 \end{pmatrix}$ asymptotically follows $N(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \mu_2 & 0 \\ 0 & \mu_4 - \mu_2^2 \end{pmatrix})$, under H_0. So, T_1 asymptotically follows sum of two weighted χ_1^2 variable where the weights are eigenvalues of $\Sigma^{\frac{1}{2}} A \Sigma^{\frac{1}{2}}$ where $\Sigma = \begin{pmatrix} \hat{\mu}_2 & 0 \\ 0 & \hat{\mu}_4 - \hat{\mu}_2^2 \end{pmatrix}$ and $A = \begin{pmatrix} \frac{1}{\hat{\mu}_2} & 0 \\ 0 & \frac{1}{2\hat{\mu}_2^2} \end{pmatrix}$.
If we further assume that $\mu_4 = 3\mu_2^2$, which holds under normality assumption on $Y_i^{(k)}$, T_1 asymptotically follows χ_2^2 distribution.

□

Remark 1. *If we drop the assumption of normality and we only assume that $Y_i^{(k)}$s have finite fourth moment, T_1 asymptotically follows weighted sum of two χ_1^2 variables.*

Now we are in a position to prove Theorem 2.

Proof of Theorem 1.

Proof. Let Z_i denote the indicator that a given gene is detected in cell i. let Y_i denote the actual expression value of that gene in cell i. Note that T_0 is a function of the r.v. Z_i's only. On the other hand, T_1 is a function of Y_i's conditioned on $Z_i = 1$. Since, asymptotic distribution of T_1 is independent of Z_i's, T_0 and T_1 are independent.

Hence, by Lemma 1 and Lemma 2, under $H_0 : c_j^{(1)} = c_j^{(2)}, (\mu_j^{(1)}, \sigma_j^{(1)}) = (\mu_j^{(2)}, \sigma_j^{(2)})$,

$$T_0 + T_1 \xrightarrow{d} \chi_3^2 \text{ as } n_1, n_2 \to \infty.$$

□

Remark 2. *If we drop the assumption of normality and only assume that, $Y_i^{(k)}$s have finite fourth moments, by Remark of **Lemma 2**, RIBBON I asymptotically follows weighted sum of three χ_1^2 variables.*

The second test RIBBON II tests for equality of mixing proportions in bimodality set up along with equality of gene specific effects in two groups. So our hypotheses of interest in this case would be:

$$H_0 : c_j^{(1)} = c_j^{(2)}, \pi_j^{(1)} = \pi_j^{(2)} \text{ against } H_1 : c_j^{(1)} \neq c_j^{(2)}, \pi_j^{(1)} \neq \pi_j^{(2)}$$

Similar to RIBBON I, we use the distribution of y_{ij}'s after marginalizing over α_{2i}'s. Note that under H_0, for both groups, gene expression follow the same bimodal distribution and hence the mixing proportions would be approximately same. On the other hand, if H_0 is not true, means and variances may not change much but the mixing proportions in two groups could be substantially different giving two different distributions. Moreover, change in mean and expression is expected to be captured by RIBBON I. it is clear that all observations come from the family of distributions: $\pi_j N(\mu_{1j}, \sigma^2_{1j}) + (1-\pi_j)N(\mu_{2j}, \sigma^2_{2j})$. So,

$$y^{(1)}_{1j}, y^{(1)}_{2j}, \ldots, y^{(1)}_{n_1 j} \overset{i.i.d.}{\sim} \pi^{(1)}_j N(\mu_{1j}, \sigma^2_{1j}) + (1-\pi^{(1)}_j)N(\mu_{2j}, \sigma^2_{2j})$$
$$\text{and } y^{(2)}_{1j}, y^{(2)}_{2j}, \ldots, y^{(2)}_{n_2 j} \overset{i.i.d.}{\sim} \pi^{(2)}_j N(\mu_{1j}, \sigma^2_{1j}) + (1-\pi^{(2)}_j)N(\mu_{2j}, \sigma^2_{2j})$$

Since L_0 and L_2 do not involve common parameters and we have already developed a testing procedure for testing gene specific effect, now it remains to test $H_0 : \pi^{(1)}_j = \pi^{(2)}_j$ vs $H_1 : \pi^{(1)}_j \neq \pi^{(2)}_j$. This testing can be rephrased as testing for equality of mixing proportions between two normal components. Define the conditional log-likelihood for k-th group as,

$$\begin{aligned} l_k(\pi^{(k)}_j | \boldsymbol{y}^{(k)}_j, \boldsymbol{z}^{(k)}_j) = \sum_{i=1}^{n_k} z^{(k)}_{ij} [\log(\pi^{(k)}_j) E[D_{ij} | y_{ij}, \hat{\pi}_j, \hat{\mu}_1, \hat{\mu}_2, \hat{\sigma}^2_1, \hat{\sigma}^2_2] \\ + \sum_{i=1}^{n_k} \log(1-\pi^{(k)}_j)(1 - E[D_{ij} | y_{ij}, \hat{\pi}_j, \hat{\mu}_1, \hat{\mu}_2, \hat{\sigma}^2_1, \hat{\sigma}^2_2])] \end{aligned}$$

Similarly denoting the joint log-likelihood for the combined data under H_0 by l_0, we have $-2\log$ of likelihood ratio statistic as,

$$T_2 = 2\big(l_1(\hat{\pi}^{(1)}_j | \boldsymbol{y}^{(1)}_j, \boldsymbol{z}^{(1)}_j) + l_2(\hat{\pi}^{(2)}_j | \boldsymbol{y}^{(2)}_j, \boldsymbol{z}^{(2)}_j) - l_0(\hat{\pi}_j | \boldsymbol{y}^{(1)}_j, \boldsymbol{z}^{(1)}_j, \boldsymbol{y}^{(2)}_j, \boldsymbol{z}^{(2)}_j)\big) \tag{3.12}$$

Theorem 2. *Under* $H_0 : c^{(1)}_j = c^{(2)}_j, \pi^{(1)}_j = \pi^{(2)}_j$ *we have,*

$$T_0 + \frac{T_2}{\beta} \xrightarrow{d} \chi^2_2 \text{ as } n_1, n_2 \to \infty \text{ for some constant } \beta (0 < \beta < 1)$$

Remark: The asymptotic distribution remains the same even if $\mu_1 = \mu_2$ and $\sigma^2_1 = \sigma^2_2$, i.e. two components are identical. So, the cutoff from the same distribution can be applied for testing differential expression on all genes.

Here again, for notational simplicity, we ignore the z_{ij} term, i.e., we assume that z_{ij}'s are all 1. If some z_{ij}s are 0, we collect the terms with z_{ij} equal to 1 only and apply the same asymptotics. Now the effective sample size becomes total number of observations with z_{ij} equal to 1. We fix a gene j and denote the expression value corresponding to i-th cell in the k-th group to be $Y_i^{(k)}$ which is same as $y_{ij}^{(k)}$ according to our original notation. Similarly, we drop the subscript j from all the parameters. Consider a set of observations from two groups: $Y_1^{(1)}, Y_2^{(1)}, \ldots, Y_{n_1}^{(1)}$ from group 1 and $Y_1^{(2)}, Y_2^{(2)}, \ldots, Y_{n_2}^{(2)}$ from group 2. Let the $-2\log$-likelihood ratio test statistic be T_2 for testing $H_0 : \pi_1 = \pi_2$ vs $H_1 : \pi_1 \neq \pi_2$ under the assumption that, $Y_i^{(k)} \sim \pi_k N(\mu_1, \sigma_1^2) + (1 - \pi_k)N(\mu_2, \sigma_2^2)$, $k = 1, 2$. In Lemma 4, we shall derive the asymptotic distribution of T_2 under H_0.

Lemma 3. *Define* $U_{i,n}^{(k)} = \frac{\hat{\pi}_n \phi(Y_i^{(k)}; \hat{\mu}_{1,n}, \hat{\sigma}_{1,n})}{\hat{\pi}_n \phi(Y_i^{(k)}; \hat{\mu}_{1,n}, \hat{\sigma}_{1,n}) + (1-\hat{\pi}_n)\phi(Y_i^{(k)}; \hat{\mu}_{2,n}, \hat{\sigma}_{2,n})}$, $k = 1, 2$, *based on* n *observations. Under null hypothesis, if* $\pi_1 = \pi_2 = \pi$, $E[U_{i,n}^{(k)}] \to \pi$ *as* $n \to \infty$.

Proof. If $(\hat{\mu}_{1,n}, \hat{\mu}_{2,n}, \hat{\sigma}_{1,n}, \hat{\sigma}_{2,n}, \hat{\pi}_n)$ are MLEs of $(\mu_1, \mu_2, \sigma_1, \sigma_2, \pi)$, by asymptotic properties of MLEs, $(\hat{\mu}_{1,n}, \hat{\mu}_{2,n}, \hat{\sigma}_{1,n}, \hat{\sigma}_{2,n}, \hat{\pi}_n) \xrightarrow{P} (\mu_1, \mu_2, \sigma_1, \sigma_2, \pi)$.
So, $U_{i,n}^{(k)} \xrightarrow{P} W_i^{(k)}$ where $W_i^{(k)} = \frac{\pi\phi(Y_i^{(k)}; \mu_1, \sigma_1)}{\pi\phi(Y_i^{(k)}; \mu_1, \sigma_1) + (1-\pi)\phi(Y_i^{(k)}; \mu_2, \sigma_2)}$.
If D_i is the indicator denoting that the i-th observation comes from mode 1,

$$E[D_i|Y_i] = \frac{\pi\phi(Y_i^{(k)}; \mu_1, \sigma_1)}{\pi\phi(Y_i^{(k)}; \mu_1, \sigma_1) + (1-\pi)\phi(Y_i^{(k)}; \mu_2, \sigma_2)} = W_i^{(k)}.$$

So, $E[W_i^{(k)}] = E[D_i] = \pi$. Now, $|U_{i,n}^{(k)}| \leq 1$ a.s. and hence $\{U_{i,n}^{(k)}\}_{n=1}^{\infty}$ are uniformly integrable because $E[|U_{i,n}^{(k)}| I(|U_{i,n}^{(k)}| > 2)] = 0$ for all n. By Dominated Convergence Theorem (DCT), $E[|U_{i,n}^{(k)} - W_i^{(k)}|] \to 0$ and hence $E[U_{i,n}^{(k)}] \to E[W_i^{(k)}] = \pi$ as $n \to \infty$. □

This indicates that the unconditional mean of $U_{i,n}^{(k)}$ is π asymptotically under the null hypothesis. Hence, a test for equality of proportions based on $U_{i,n}^{(k)}$ is meaningful.

Lemma 4. *T_2 asymptotically follows a constant multiple of χ^2_1 distribution under H_0 : $\pi^{(1)}_j = \pi^{(2)}_j$.*

Proof. Note that, $\hat{\pi}_{1,n} = \frac{\sum_{i=1}^{n_1} U^{(1)}_{i,n}}{n_1}$, $\hat{\pi}_{2,n} = \frac{\sum_{i=1}^{n_2} U^{(2)}_{i,n}}{n_2}$ and $\hat{\pi}_n = \frac{\sum_{i=1}^{n_1} U^{(1)}_{i,n} + \sum_{i=1}^{n_1} U^{(2)}_{i,n}}{(n_1+n_2)}$.

$$
\begin{aligned}
T_2 &= 2[\sum_{i=1}^{n_1}[U^{(1)}_{i,n}\log(\hat{\pi}_{1,n}) + (1-U^{(1)}_{i,n})\log(1-\hat{\pi}_{1,n})] \\
&+ \sum_{i=1}^{n_2}[U^{(2)}_{i,n}\log(\hat{\pi}_{2,n}) + (1-U^{(2)}_{i,n})\log(1-\hat{\pi}_{2,n})] - \sum_{i=1}^{n_1}[U^{(1)}_{i,n}\log(\hat{\pi}_n) + (1-U^{(1)}_{i,n})\log(1-\hat{\pi}_n)] \\
&- \sum_{i=1}^{n_2}[U^{(2)}_{i,n}\log(\hat{\pi}_n) + (1-U^{(2)}_{i,n})\log(1-\hat{\pi}_n)]] \\
&= -2\sum_{i=1}^{n_1}[U^{(1)}_{i,n}\log(\frac{\hat{\pi}}{\hat{\pi}_{1,n}}) + (1-U^{(1)}_{i,n})\log(\frac{1-\hat{\pi}_n}{1-\hat{\pi}_{1,n}})] - 2\sum_{i=1}^{n_2}[U^{(2)}_{i,n}\log(\frac{\hat{\pi}_n}{\hat{\pi}_{2,n}}) \\
&+ (1-U^{(2)}_{i,n})\log(\frac{1-\hat{\pi}_n}{1-\hat{\pi}_{2,n}})] \\
&= -2\sum_{i=1}^{n_1}[U^{(1)}_{i,n}\log(1+\frac{\hat{\pi}_n-\hat{\pi}_{1,n}}{\hat{\pi}_{1,n}}) + (1-U^{(1)}_{i,n})\log(1+\frac{\hat{\pi}_{1,n}-\hat{\pi}_n}{1-\hat{\pi}_{1,n}})] \\
&- 2\sum_{i=1}^{n_2}[U^{(2)}_{i,n}\log(1+\frac{\hat{\pi}_n-\hat{\pi}_{2,n}}{\hat{\pi}_{2,n}}) + (1-U^{(2)}_{i,n})\log(1+\frac{\hat{\pi}_{2,n}-\hat{\pi}_n}{1-\hat{\pi}_{2,n}})] \\
&= -2\sum_{i=1}^{n_1}[U^{(1)}_{i,n}(\frac{\hat{\pi}_n-\hat{\pi}_{1,n}}{\hat{\pi}_{1,n}} - \frac{(\hat{\pi}_n-\hat{\pi}_{1,n})^2}{2\hat{\pi}^2_{1,n}}) + (1-U^{(1)}_{i,n})(\frac{\hat{\pi}_{1,n}-\hat{\pi}_n}{1-\hat{\pi}_{1,n}} - \frac{(\hat{\pi}_{1,n}-\hat{\pi}_n)^2}{2(1-\hat{\pi}_{1,n})^2})] \\
&- 2\sum_{i=1}^{n_2}[U^{(2)}_{i,n}(\frac{\hat{\pi}_n-\hat{\pi}_{2,n}}{\hat{\pi}_{2,n}} - \frac{(\hat{\pi}_n-\hat{\pi}_{2,n})^2}{2\hat{\pi}^2_{2,n}}) + (1-U^{(2)}_{i,n})(\frac{\hat{\pi}_{2,n}-\hat{\pi}_n}{1-\hat{\pi}_{2,n}} - \frac{(\hat{\pi}_{2,n}-\hat{\pi}_n)^2}{2(1-\hat{\pi}_{2,n})^2})] + o_p(1) \\
&= \frac{n_1(\hat{\pi}_n-\hat{\pi}_{1,n})^2}{\hat{\pi}_{1,n}(1-\hat{\pi}_{1,n})} + \frac{n_2(\hat{\pi}_n-\hat{\pi}_{2,n})^2}{\hat{\pi}_{2,n}(1-\hat{\pi}_{2,n})} + o_p(1)
\end{aligned}
$$

$$
\begin{aligned}
\text{Now, } U^{(k)}_{i,n} &= W^{(k)}_i + (\frac{\partial W^{(k)}_i}{\partial \pi})(\hat{\pi}_n - \pi) + (\frac{\partial W^{(k)}_i}{\partial \mu_1})(\hat{\mu}_{1,n} - \mu_1) \\
&+ (\frac{\partial W^{(k)}_i}{\partial \mu_2})(\hat{\mu}_{2,n} - \mu_2) + (\frac{\partial W^{(k)}_i}{\partial \sigma^2_1})(\hat{\sigma}^2_{1,n} - \sigma^2_1) + (\frac{\partial W^{(k)}_i}{\partial \sigma^2_2})(\hat{\sigma}^2_{2,n} - \sigma^2_2) + O_p(\frac{1}{n})
\end{aligned}
$$

$$\text{Here, } \frac{\partial W_i^{(k)}}{\partial \pi} = \frac{\phi(Y_i^{(k)};\mu_1,\sigma_1)\phi(Y_i^{(k)},\mu_2,\sigma_2)}{(\pi\phi(Y_i^{(k)};\mu_1,\sigma_1)+(1-\pi)\phi(Y_i^{(k)},\mu_2,\sigma_2))^2}$$

$$\frac{\partial W_i^{(k)}}{\partial \mu_1} = \frac{\pi(1-\pi)\phi(Y_i^{(k)};\mu_2,\sigma_2)\phi(Y_i^{(k)},\mu_1,\sigma_1)}{(\pi\phi(Y_i^{(k)};\mu_1,\sigma_1)+(1-\pi)\phi(Y_i^{(k)});\mu_2,\sigma_2)^2}\frac{(\mu_1-Y_i^{(k)})}{\sigma_1^2}$$

$$\frac{\partial W_i^{(k)}}{\partial \mu_2} = \frac{\pi(1-\pi)\phi(Y_i^{(k)};\mu_2,\sigma_2)\phi(Y_i^{(k)},\mu_1,\sigma_1)}{(\pi\phi(Y_i^{(k)};\mu_1,\sigma_1)+(1-\pi)\phi(Y_i^{(k)});\mu_2,\sigma_2)^2}\frac{(\mu_2-Y_i^{(k)})}{\sigma_2^2}$$

$$\frac{\partial W_i^{(k)}}{\partial \sigma_1^2} = \frac{\pi(1-\pi)\phi(Y_i^{(k)};\mu_2,\sigma_2)\phi(Y_i^{(k)},\mu_1,\sigma_1)}{(\pi\phi(Y_i^{(k)};\mu_1,\sigma_1)+(1-\pi)\phi(Y_i^{(k)});\mu_2,\sigma_2)^2}\frac{(\mu_1-Y_i^{(k)})^2}{2(\sigma_1^2)^2}$$

$$\frac{\partial W_i^{(k)}}{\partial \sigma_2^2} = \frac{\pi(1-\pi)\phi(Y_i^{(k)};\mu_2,\sigma_2)\phi(Y_i^{(k)},\mu_1,\sigma_1)}{(\pi\phi(Y_i^{(k)};\mu_1,\sigma_1)+(1-\pi)\phi(Y_i^{(k)});\mu_2,\sigma_2)^2}\frac{(\mu_2-Y_i^{(k)})^2}{2(\sigma_2^2)^2}$$

We also have,

$$\hat{\pi}_1 = \frac{\sum_{i=1}^{n_1} U_{i,n}^{(1)}}{n_1} = \frac{\sum_{i=1}^{n_1} W_i^{(1)}}{n_1} + \Big(\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \pi}\Big)(\hat{\pi}_n-\pi)$$
$$+\Big(\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \mu_1}\Big)(\hat{\mu}_{1,n}-\mu_1)+\Big(\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \mu_2}\Big)(\hat{\mu}_{2,n}-\mu_2)$$
$$++\Big(\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \sigma_1^2}\Big)(\hat{\sigma}_{1,n}^2-\sigma_1^2)+\Big(\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \sigma_2^2}\Big)(\hat{\sigma}_{2,n}^2-\sigma_2^2)+O_p(\frac{1}{n})$$

Since, $X_i^{(k)}$s are i.i.d., by WLLN, we have, $\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \pi} \xrightarrow{P} c_\pi$,
$\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \mu_1} \xrightarrow{P} c_{\mu_1}$, $\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \mu_2} \xrightarrow{P} c_{\mu_2}$, $\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \sigma_1^2} \xrightarrow{P} c_{\sigma_1^2}$, $\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial W_i^{(1)}}{\partial \sigma_2^2} \xrightarrow{P} c_{\sigma_2^2}$.

Therefore,

$$\sqrt{n_1}(\hat{\pi}_{1,n}-\pi_1) = \sqrt{n_1}(\frac{1}{n_1}\sum_{i=1}^{n_1}W_i^{(1)}-\pi)+c_\pi\sqrt{n_1}(\hat{\pi}_n-\pi)+c_{\mu_1}\sqrt{n_1}(\hat{\mu}_{1,n}-\mu_1)$$
$$+c_{\mu_2}\sqrt{n_1}(\hat{\mu}_{2,n}-\mu_2)+c_{\sigma_1^2}\sqrt{n_1}(\hat{\sigma}_{1,n}^2-\sigma_1^2)+c_{\sigma_2^2}\sqrt{n_1}(\hat{\sigma}_{2,n}^2-\sigma_2^2)+o_p(1)$$

Similarly,

$$\sqrt{n_2}(\hat{\pi}_{2,n}-\pi_2) = \sqrt{n_2}(\frac{1}{n_2}\sum_{i=1}^{n_2}W_i^{(2)}-\pi)+c_\pi\sqrt{n_2}(\hat{\pi}_n-\pi)+c_{\mu_1}\sqrt{n_2}(\hat{\mu}_{1,n}-\mu_1)$$

$$+ c_{\mu_2}\sqrt{n_2}(\hat{\mu}_{2,n} - \mu_2) + c_{\sigma_1^2}\sqrt{n_2}(\hat{\sigma}^2_{1,n} - \sigma_1^2) + c_{\sigma_2^2}\sqrt{n_2}(\hat{\sigma}^2_{2,n} - \sigma_2^2) + o_p(1).$$

Also, $\sqrt{\frac{n_1 n_2}{n_1+n_2}}(\hat{\pi}_{1,n} - \hat{\pi}_{2,n}) = \sqrt{\frac{n_1 n_2}{n_1+n_2}}(\frac{1}{n_1}\sum_{i=1}^{n_1} W_i^{(1)} - \frac{1}{n_2}\sum_{i=1}^{n_2} W_i^{(2)}) + o_p(1)$

Hence, we have, $\hat{\pi}_{1,n} \xrightarrow{P} \pi$, $\hat{\pi}_{2,n} \xrightarrow{P} \pi$ and so $\sqrt{\frac{n_1 n_2}{n_1+n_2}}(\hat{\pi}_{1,n} - \hat{\pi}_{2,n}) = N(0, \sigma_W^2)$ under H_0, where $\sigma_W^2 = Var[W_1^{(k)}]$

Therefore, $T_2 = \frac{n_1(\hat{\pi}-\hat{\pi}_1)^2}{\pi(1-\pi)} + \frac{n_2(\hat{\pi}-\hat{\pi}_2)^2}{\pi(1-\pi)} + o_p(1) = \frac{n_1 n_2}{(n_1+n_2)}\frac{(\hat{\pi}_1-\hat{\pi}_2)^2}{\pi(1-\pi)} + o_p(1)$ and hence T_2 asymptotically follows $\frac{\sigma_W^2}{\pi(1-\pi)}\chi_1^2$.

Now, if $\hat{\sigma}_W^2$ is a consistent estimator of σ_W^2 and $\hat{\pi} \xrightarrow{P} \pi$ we have by Slutsky's theorem,

$$\frac{T_2}{\hat{\beta}} \xrightarrow{d} \chi_1^2 \text{ as } n_1, n_2 \to \infty \text{ where } \hat{\beta} = \frac{\hat{\sigma}_W^2}{\hat{\pi}(1-\hat{\pi})}.$$

□

Remark 3. *Note that $\sigma_W^2 < \pi(1-\pi)$. To prove this consider a random variable $V_i^{(k)}$ such that $V_i^{(k)}|W_i^{(k)} \sim Ber(W_i^{(k)})$.*

Now $Var[V_i^{(k)}] > Var[E[V_i^{(k)}|W_i^{(k)}]] = Var[W_i^{(k)}]$ and so $Var[W_i^{(k)}] < \pi(1-\pi)$.

Remark 4. *We estimate π by $\hat{\pi}_n$ and σ_W^2 by $\frac{1}{n_1+n_2-1}\sum_{k=1}^{2}\sum_{i=1}^{n_k}(U_{i,n}^{(k)} - \hat{\pi}_n)^2$.*

Now we can prove Theorem 2 using Lemma 2 and Lemma 4 as follows.

Proof of Theorem 2.

Proof. Similar to Theorem 1, T_0 is a function of the random variable Z_i's only and T_2 is a function of Y_i's conditioned on $Z_i = 1$, $i = 1, \ldots, n$. Since, asymptotic distribution of T_2 is independent of Z_i's, T_0 and T_2 are independent.

Now using Lemma 2 and Lemma 4, the proof of Theorem 2 follows immediately. □

3.3 Simulation study for testing differential expression

To compare the performance of different methods, we investigate the power and ROC curve using simulated data. We perform the simulations with three different model assumptions.

One type of simulation is performed using the underlying model used by RIBBON, and benchmarking is also conducted based on data generated using MFA and scDD simulation protocol. SC2P [131], DESingle [81], and MAST [39] are taken as candidate tools for comparison for their better accuracy over other methods and availability of their software. Since SC2P provides two different p-values: one for the differential proportion of zeros and the other for differential mean expression level; we have included both in the comparison. Similarly, we have considered two types of p-values for MAST: p-value for log fold change and p-value based on chi-square statistic. In all the scenarios under consideration, RIBBON shows more consistency and robust accuracy than other methods.

3.3.1 Simulation using model of RIBBON

We perform simulations with RIBBON for two scenarios: one with 100 cells in each group and the other with 1000 cells in each group, for 5000 genes for a single individual. We generate expression values for half of the genes from bimodal distribution whereas for the remaining half, we use unimodal distribution. For bimodal distribution, we generate the parameters as given below:

$$\mu_{1j} = \min(Y_{1j}, Y_{2j}), \mu_{2j} = \max(Y_{1j}, Y_{2j}) \text{ where } Y_{1j}, Y_{2j} \overset{i.i.d.}{\sim} N(0, 0.3),$$
$$\sigma^2_{1j} = \min(Z_{1j}, Z_{2j}), \sigma^2_{2j} = \max(Z_{1j}, Z_{2j}) \text{ where } Z_{1j}, Z_{2j} \overset{i.i.d.}{\sim} Gamma(1, \frac{1}{3}),$$
$$c_j \sim N(0, 1), \pi_j \sim Beta(0.5, 0.5).$$

For unimodal distribution, we use the following scheme:

$$\mu_j \sim N(0, 0.3), \sigma^2_j \sim Gamma(1, \frac{1}{3}), c_j \sim N(0, 1).$$

Under alternative hypothesis, we generate two sets of parameters independently. Keeping the false positive rate (FPR) fixed at 1% level, we compare the performance of the methods based on the true positive rate (TPR). These along with ROC curves for two types of simulations are shown in Figures 3.2 and 3.3. RIBBON I and RIBBON II show the highest power at 1% FPR in bimodal simulations and outperform other methods in terms of ROC curves, when the distribution of gene expression is bimodal.

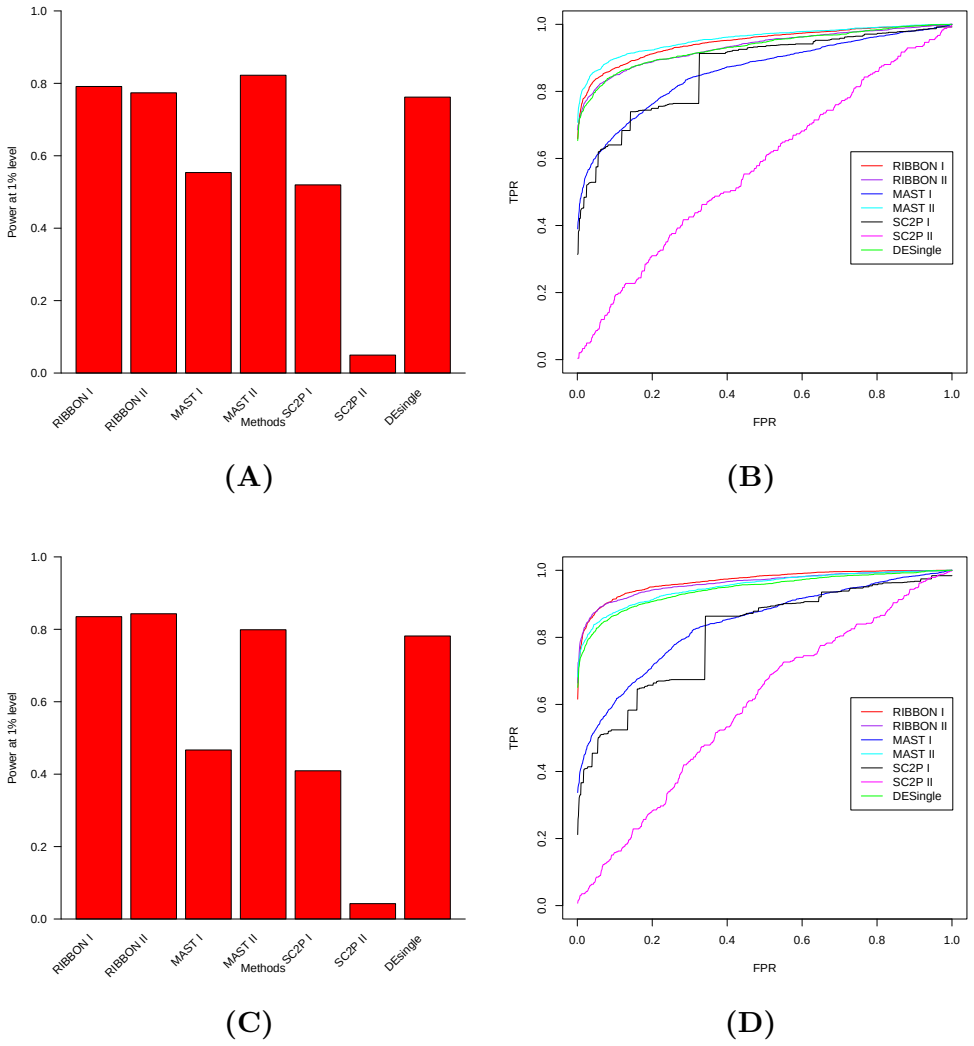


Figure 3.2: (A): Power at 1% FPR, (B) ROC curve with data simulated from RIBBON when the simulated gene expressions are unimodal, (C) Power at 1% FPR, (D) ROC curve with data simulated from RIBBON when the gene expressions are bimodal. The number of cells in each simulation is 100. RIBBON I and RIBBON II outperform all other methods when the distribution is bimodal.

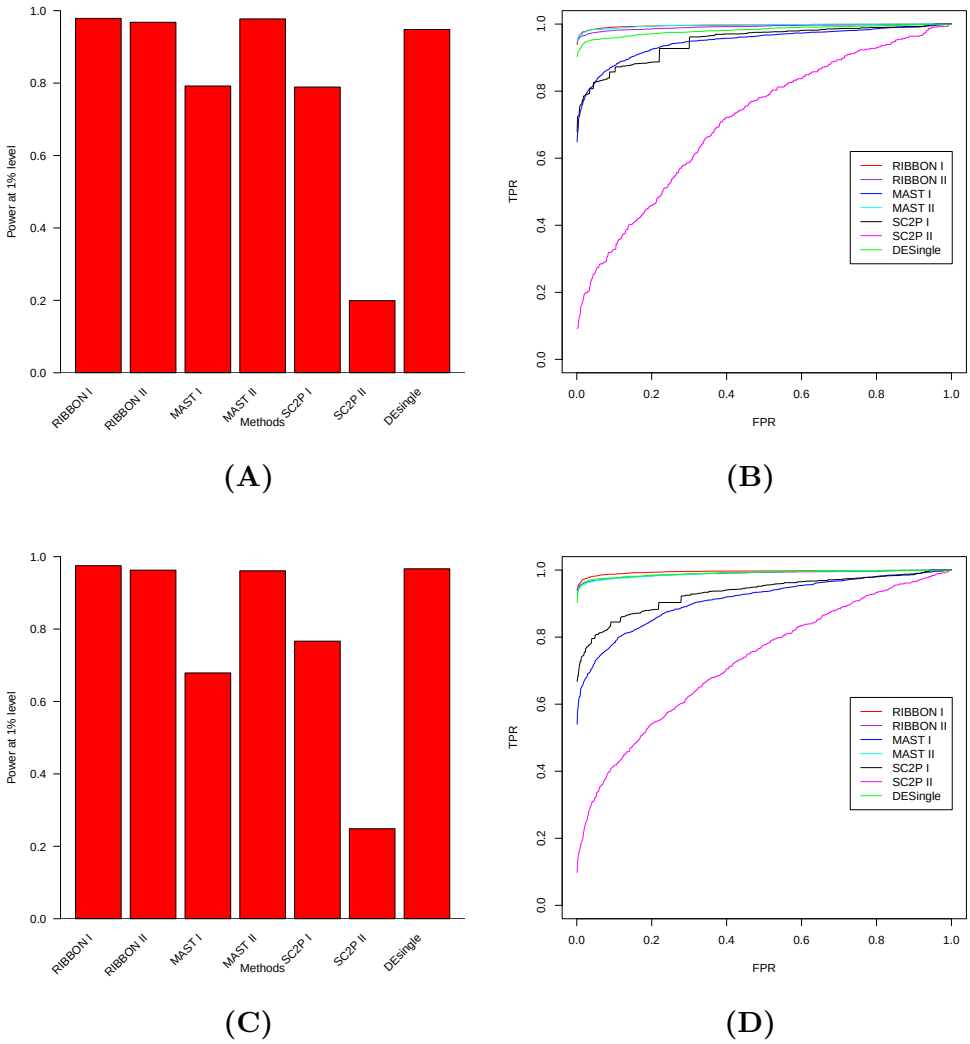


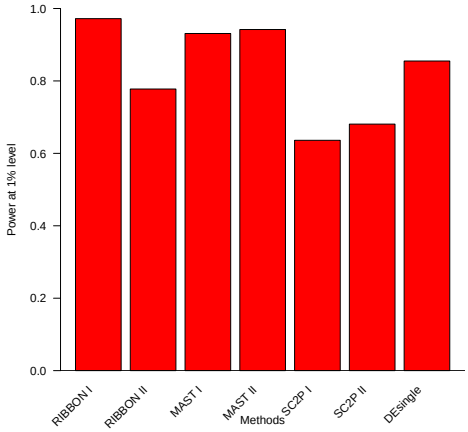
Figure 3.3: (A): Power at 1% FPR, (B) ROC curve with data simulated from RIBBON when the simulated gene expressions are unimodal, (C) Power at 1% FPR, (D) ROC curve with data simulated from RIBBON when the gene expressions are bimodal. The number of cells in each simulation is 1000. Because of the large number of observations, ROC curves of many tests are close to 1, though for bimodal simulation, RIBBON I and RIBBON II are slightly better than other methods.

3.3.2 Simulation using Splatter

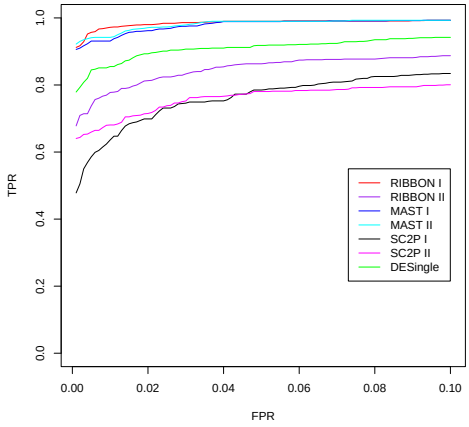
We also follow simulation methods other than that of our own. Using Bioconductor package Splatter [133], we simulate data with two different simulation models: scDD and MFA. In scDD simulation [60], we have considered both EE and EP models to generate data from the null distribution. Under the EE model, expression values are generated from a single-mode with identical distribution for each gene under two conditions. We generate expression values for each gene from a bimodal distribution with similar distribution from both conditions under the EP model. We consider DE and DP models to generate data from the alternative distribution. Under DE model, all genes are unimodal with possibly different means and variances across two conditions. Under the DP model, there are two modes in each condition with equal component means; however, the mixture from the two modes varies. We perform these two types of simulations with 100 cells in each group, and simulate 1000 genes under each of the three models. In the simulation with the MFA model, cells from different branches are assumed to belong to two groups. Five thousand genes with differential expression in two branches and the same number of genes with identical distribution in two branches, are selected for comparison. Barplots of powers in these two types of simulations and ROC curves are shown in Figure 3.4. RIBBON I and RIBBON II seem to outperform all other methods in these three types of scenarios. RIBBON I exceeds RIBBON II in performance in DE simulation, whereas RIBBON II seems to have an edge over RIBBON I at 1% FPR in DP simulation.

3.4 Real Data Analysis

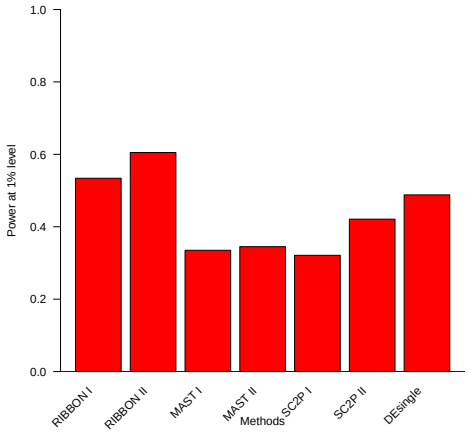
We have used single-cell data from Buettner et al. [16] to validate the performance of our method on real data. The study was aimed to observe the effect of cell cycle on gene expression levels. Single-cell RNA-seq was performed on cells with G1, G2M, and S stages of mouse cells. A single-cell experiment was performed on mouse mESC cells that were flow cytometry sorted into G1, S, and G2M phases of the cell cycle. These three types of cells constituted three different datasets in our analysis. Three pairwise comparisons for differential expression are performed using this dataset. Names of genes that are responsible for the cell cycle are downloaded from the NCBI database. We calculate the false discovery rate and true positive rate based on top differentiated genes found by these seven methods and the list of genes responsible for cell cycle obtained from the database. The ROC curves for the top FDR level up to 0.1 are shown in Figure 3.5. Figure 3.6 shows complete ROC curves for these three comparisons.


Power at 1% level
1.0
0.8
0.6
0.4
0.2
0.0
RIBBON I
RIBBON II
MAST I
MAST II
SC2P I
SC2P II
DESingle
Methods

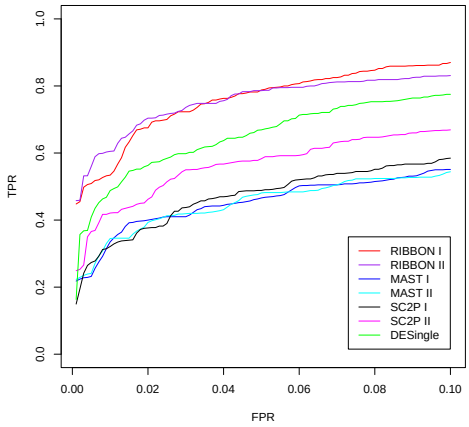
(A)


TPR
FPR
0.00
0.02
0.04
0.06
0.08
0.10
0.0
0.2
0.4
0.6
0.8
1.0
RIBBON I
RIBBON II
MAST I
MAST II
SC2P I
SC2P II
DESingle

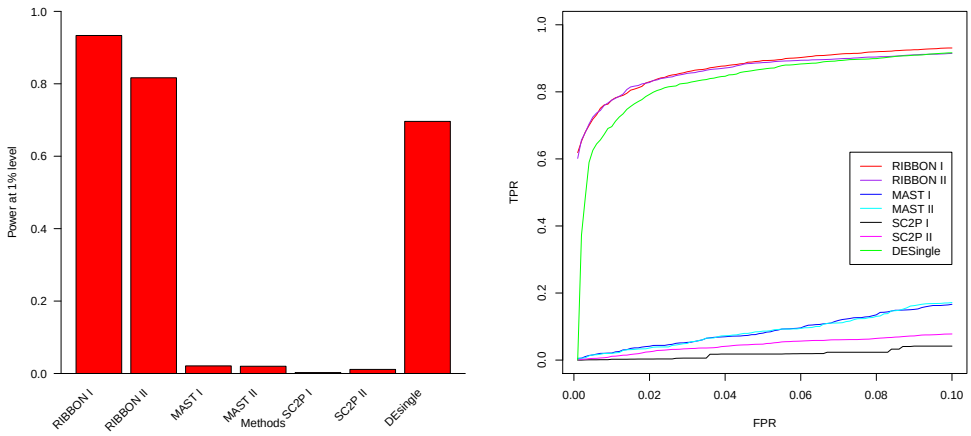
(B)


Power at 1% level
1.0
0.8
0.6
0.4
0.2
0.0
RIBBON I
RIBBON II
MAST I
MAST II
SC2P I
SC2P II
DESingle
Methods

(C)

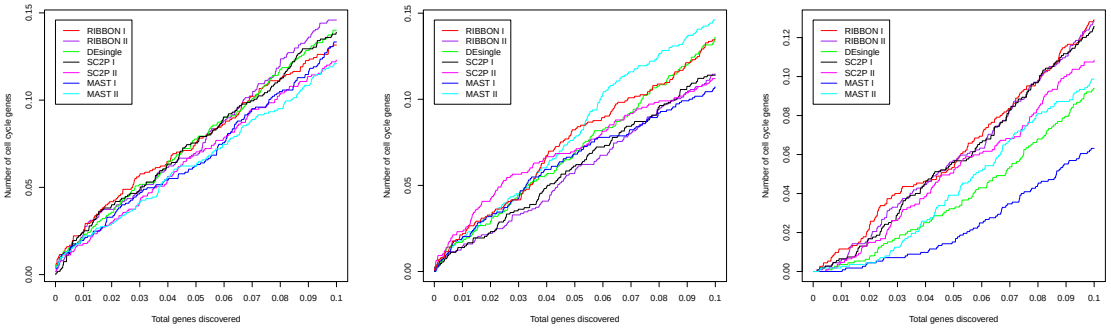

TPR
FPR
0.00
0.02
0.04
0.06
0.08
0.10
0.0
0.2
0.4
0.6
0.8
1.0
RIBBON I
RIBBON II
MAST I
MAST II
SC2P I
SC2P II
DESingle

(D)



(A) **(B)**

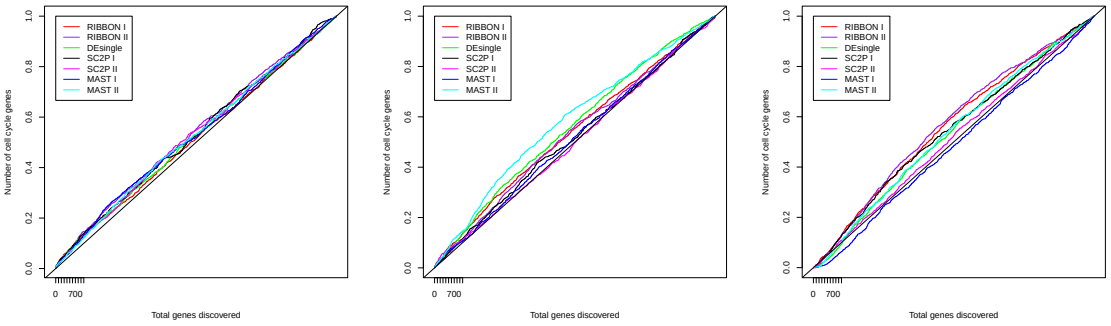
Figure 3.4: (A) Power at 1% FPR with data simulated from sCDD model EE, (B) ROC curve with data simulated from scDD model DE, (C) Power at 1% FPR with data simulated from scDD model DP, (D) ROC curve with data simulated from scDD model DP, (E) Power at 1% FPR with data simulated from MFA, (F) ROC curve with data simulated from MFA. In DE simulation, RIBBON I is the best; in DP simulation, RIBBON II is the best, whereas in MFA simulation, both RIBBON I and RIBBON II perform best.



(A) Comaprison between G1 and G2M cells **(B)** Comaprison between G2M and S cells **(C)** Comparison between S and G1 cells

Figure 3.5

RIBBON II outperforms other methods in comparison I and III whereas RIBBON I outperforms others in comparison III.



(A) Comaprison between subgroups of G1 cells **(B)** Comaprison between subgroups of G2M cells **(C)** Comparison between two subgroups of S cells

Figure 3.6

3.5 Multiple testing problem

Under our model's assumption genes are correlated and hence there is a dependency structure among p-values obtained from different genes. Since no specific dependency structure between genes is known, we recommend using Benjamini-Yekutielli procedure [10]. Let $P_{(1)} < P_{(2)} < P_{(3)} < \cdots < P_N$ be the ordered p-values for N genes and let $H_{(1)}, H_{(2)}, \ldots, H_{(N)}$ be corresponding null hypotheses respectively. Then FDR control at level α works as follows:

- For a given α, find the largest k such that $P_{(k)} < \frac{k}{N.c(N)}\alpha$ (i.e. $k = \underset{j\in\{1,2,..,N\}}{\operatorname{argmax}} \left(j.I_{P_{(j)}\leq \frac{j}{m}\alpha}\right)$) where $c(N)$ is the harmonic number i.e $c(N) = \sum_{j=1}^{N} \frac{1}{j}$. Note that, $c(N)$ can be approximated by:
  $c(N) = \sum_{j=1}^{N} \frac{1}{j} \approx \ln(N) + \gamma + \frac{1}{2N}$ where γ is the Euler-Mascheroni constant ($\gamma = 0.57721$).

- Reject the null hypotheses for all $H_{(i)}$ for $i = 1, 2, ..., k$.

So, the adjusted p-value is given by $\frac{N.c(N)}{k}P_{(k)}$, for $k = 1, 2, \ldots, N$.

3.6 Discussion

We extend our model RIBBON to find two tests for differential expression, RIBBON I and RIBBON II. RIBBON I is aimed in detecting the change in overall mean and variance, whereas RIBBON II can detect the change in mixing proportion for mixture normal distribution. Both of these tests separate the biological factor behind zeros from the technical factor behind zeros. We have found the asymptotic distributions of these two statistics under weaker assumption. It is ideal to apply RIBBON I to unimodal genes and RIBBON II to bimodal genes. Both RIBBON I and RIBBON II show promising accuracy in simulated data as well as in benchmarking real data. This also ensures the robustness of these two tests for testing differential expression in general. Our tests are based on comparison of likelihood ratio statistic between two hypotheses. As a result, though all results in this work are based on differential expression between two groups, they can easily be extended to perform differential expression among more than two groups.

3.7 Code and software availability

Reproducible codes for all figures, data, and software for RIBBON are available at: http://github.com/indranillab/ribbon .

Chapter 4: PseudoGA: Cell pseudotime reconstruction based on genetic algorithm

4.1 Introduction

Cellular level gene expression profile can reveal the heterogeneity within a tissue and provides valuable information about ongoing biological processes inside a cell [32, 84, 50]. In bulk RNA-seq data, averaging over large number of cells may hide the true biological signal coming from a heterogeneous mixture of cells. This phenomenon, commonly known as Simpson's paradox, may give misleading conclusions. Biological processes like tissue development, cellular differentiation, tumor development, cell cycle etc. go through transcriptomic stages in cell specific manner. To understand the mechanism of the ongoing process it is essential to study the transcriptomic signature that triggers and controls these programmed changes [35]. There is an underlying order [74, 18, 13] behind these transcriptomic stages that remains unexplored mainly due to the collection of cells at a single time point and inability to track the function over time. Clearly, not all cells are at the same stage during a biological process leading to cell to cell variability in gene expression profile. So capturing cells at a particular time would display different stages of cells that should be ordered according to a time scale, known as 'pseudotime'.

Genes responsible for circadian rhythm, metabolism, cell death process etc. are regulated in a synchronized manner in different cells. Function of a cell may be affected by stages in development process, cell state transition, spatial effect, interaction with environment, cell-cell interaction and other internal ongoing processes. Effects of these simultaneous processes add to the heterogeneity in expression levels of thousands of genes at cellular level [67, 51]. Thus arranging cells according to a pseudotime trajectory with respect to its

transcriptional stages may provide more insight into the mechanism of how transcriptomic changes govern biological procedures at molecular level [50, 22, 6]. This information might have important applications in therapeutics and system biology [74, 112, 103]. The pseudotime need not be the physical time in a biological process; it could be a hypothetical time scale or pseudotime, depending on the developmental stage, position in cellular hierarchy, cell cycle stage and other biological processes.

Available methods in the literature mainly focus on dimensionality reduction followed by mapping of cells to a trajectory. The dimensionality reduction is performed by principal component analysis (PCA) [122], independent component analysis (ICA) [109], t-stochastic neighbor embedding (t-SNE) [76], diffusion map (DM) [26] or DDRTree [94]. Pseudotime inference is based on reduced dimensional data instead of full data. After dimensionality reduction, few methods build minimal spanning tree [117, 55], principal curve [110] or reverse graph embedding [94] to learn a principal tree from the data and creates a pseudotime path. Instead of following tree construction approach, diffusion pseudotime [45] ranks cells based on eigenvectors of the matrix whose elements follow Gaussian distributions with respect to euclidean distance between two cells and kNN graph is created using the diffusion map. scVelo [12] follows a different approach by inferring pseudotime based on the amount of pre-mRNAs and mature mRNAs present in a cell.

Existing pseudotime construction algorithms are mainly based on construction of minimal spanning tree, kNN graph or principal curve fitted on first two reduced dimensions. The accuracy of a method depends on the dimensionality reduction method being used in the first step and the amount of information that is lost during converting original data to lower dimensions. To check whether different types of dimensionality reduction algorithm can indeed construct the true pseudotime properly and retain most of the information that is in the original data, we simulate three dimensional data, under three scenarios (Figure 4.1). In each case, first two components are time dependent variables and all variables are scaled by standard deviation.

We apply different algorithms for each scenario (Figure 4.2). In scenario 1, the first two variables are perfectly linear with pseudotime and the third variable is noise. First PCA and ICA components show linear trend with pseudotime. However, high variance for the second component adds more noise in its estimation while other dimensionality reduction methods do not show a clear picture of the pseudotime variable. In scenario 2, when there is a cascade like change in expression level of one variable, the pseudotime structure gets disrupted, though all methods show good characteristic of clustering. In scenario 3, both the variables are sinusoidal with phase difference. All dimensionality reduction methods fail to provide a clear picture of the temporal structure of the data. In all these three

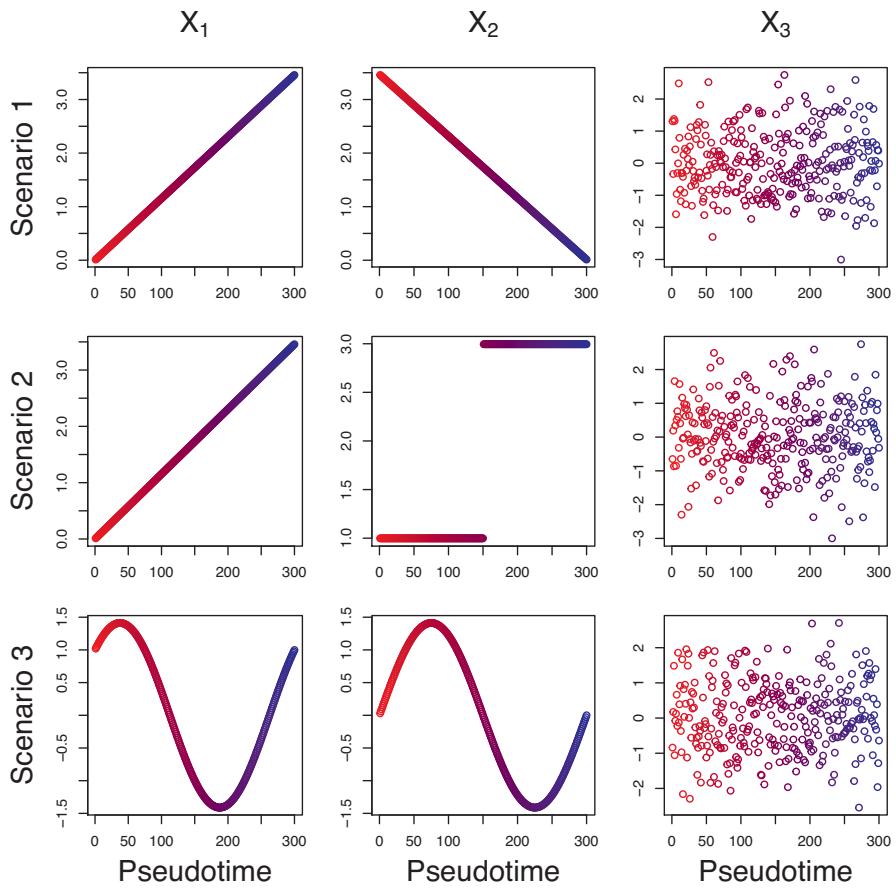


Figure 4.1: Three simulated scenarios each containing three variables (X_1, X_2, X_3) having same variance with different types of trends. Scenario 1: X_1 increasing, X_2 decreasing and X_3 random noise; Scenario 2: X_1 increasing, X_2 piecewise constant and X_3 random noise; Scenario 3: X_1 and X_2 sinusoidal with phase difference, X_3 random noise.

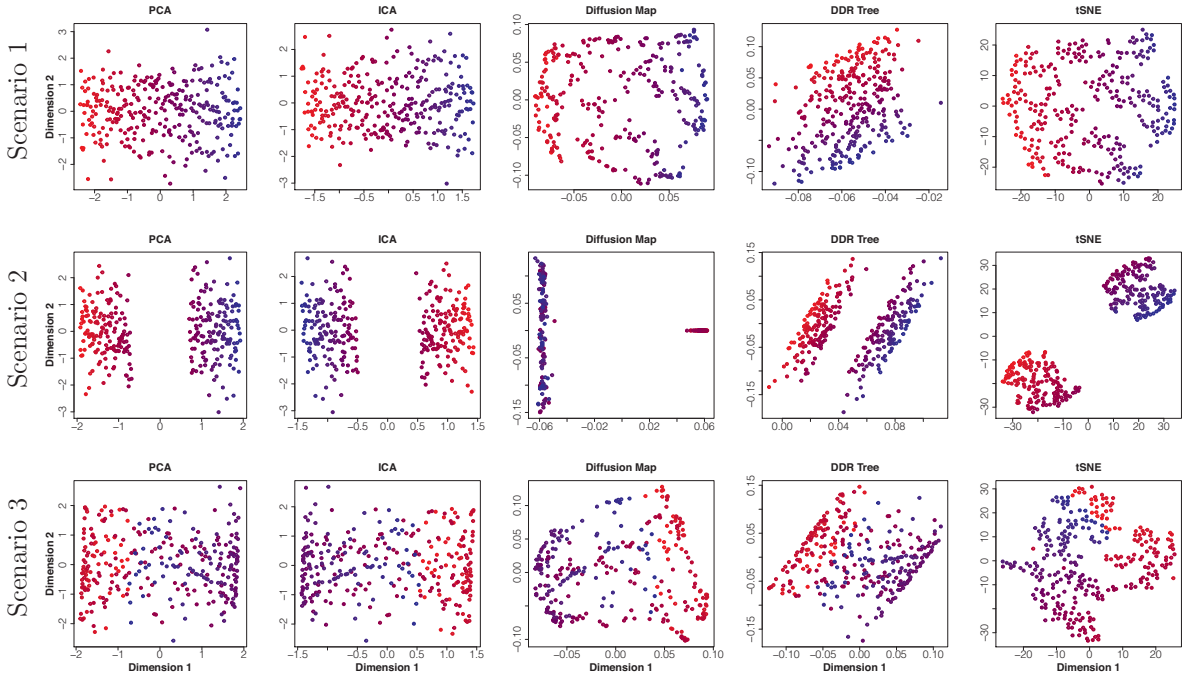


Figure 4.2: First two dimensions of outputs produced by PCA, ICA, diffusion map , DDRTree and t-SNE when applied to three scenarios as in Figure 4.1. Trajectory building algorithms based on minimal spanning tree, kNN graph or principal curve based on reduced dimensional data may not always retrieve the accurate behavior of actual pseudotime as the geometric patterns of the low dimensional space often do not truly reflect change in pseudotime.

simulations, scatter plots of first two dimensions after applying dimensionality reduction techniques do not necessarily show visibly clear pattern of change in cell states along pseudotime. Certain trajectory reconstruction methods may fail to estimate approximate pseudotime values from some of these low-dimensional representations.

Our simulation shows that dimensionality reduction techniques may not always capture the full information about the pseudotime trajectory especially when few genes behave typically, like piece-wise linear etc. This simulation makes it clear that any method that is directly based on the actual gene expression values would have a higher chance to use more information and might provide more efficient and robust pseudotime ordering. We propose a novel method for pseudotime ordering of cells that is directly based on actual gene expression levels. Our method 'PseudoGA' uses genetic algorithm to come up with a best possible trajectory of cells that explains expression patterns for individual genes. Another advantage of this method is that it can identify any lineage structure or branching while constructing pseudotime trajectory.

4.2 Material and Methods

For pseudotime estimation we apply genetic algorithm, which is appropriate for the current problem, to develop ordering of cells in the entire cell population. If the lineage structure or the branching between cell populations is of interest in addition to pseudotemporal ordering, cells are clustered into homogeneous subpopulations before applying the algorithm. The subpopulation structure can also be provided as input. Next, we apply the same algorithm to construct ordering of cells within same cell types. Finally, another subroutine concatenates the ordered paths from different clusters to form a tree like structure. Another highlight of our method is that our pipeline produces an undirected tree, connecting the paths from each cluster when no information on root cell is available. However, if the root cell or the cluster is identified or specified, our algorithm would provide an ordered tree. No transformation or dimensionality reduction is used in the pseudotime estimation step. We utilize full information from gene expression values within cells. However, if the lineage or branching structure is not of interest, the entire cell population is considered as one single subpopulation.

4.2.1 Pseudotime ordering of cells

Data generated from a single-cell whole transcriptome sequencing can be represented in a matrix $S = \big((s_{ij})\big)$ where s_{ij} is the gene expression corresponding to i-th cell and j-

th gene (Figure 4.3A). Since expressions of all genes do not depend on pseudotime, a preliminary gene filtering is recommended to improve the accuracy of estimation. Cells are clustered optimally in at least two clusters. For pseudotime estimation, we select the top genes, that are differentially expressed between clusters. We can perform this step without clustering cells using variety of approaches like selection of highly variable genes [124, 16], exploring relation between coefficient of variation and mean expression level [14, 24], dropout-based feature selection [5] etc. However, application of our method to the entire dataset also produces similar results. Based on the expression levels of many genes together in a collection of cells, our objective is to place each cell on a certain time point to create a pesudotime trajectory. Most often, the use of trajectory inference in the analysis of single-cell transcriptome data is reliant only on the ordering of cells and not on absolute values of the positions of cells on the trajectory. Quantitative positions on pseudotime trajectory may have no physical interpretation at all e.g. in cellular hierarchy data. Moreover, in reality, even if physical interpretation of values on pseudotime trajectory exists, a distance metric on cellular expression profile may not directly scale with the stretch between those two cells in the process under consideration. So, in this work, we consider discrete trajectories by finding the best permutation of cells such that the permutation explains gene expression level changes across transcriptome along the corresponding trajectory. The extent to which a pseudotime trajectory interprets specific changes in gene expression level can also be described in terms of a cost function. This cost or penalty is obtained by fitting a smooth curve with the expression values as a dependent variable and the pseudotime values as the explanatory variable. It may be noted that the problem of finding the best fitted pseudotime is similar to traveling salesman problem (TSP) [7]. Here also given a complete undirected graph with certain edge weights, the problem is to find the Hamiltonian path with the shortest weight. The pseudotime problem we are dealing with is slightly different because the cost associated with a pseudotime path need not be the sum of costs between two consecutive cells. However, like TSP, the search space of our problem is the set of all permutations and we apply genetic algorithm to find a near optimal solution for any function defined on this space. Given the fact that the search space is discrete and grows exponentially with the number of cells, some heuristics is inevitable to find a near optimal solution. Genetic algorithm is known to perform reasonably well in a wide spectrum of problems [54, 115] including the ones where the search space is the set of all permutations [3, 2, 87, 49, 27, 65, 93].

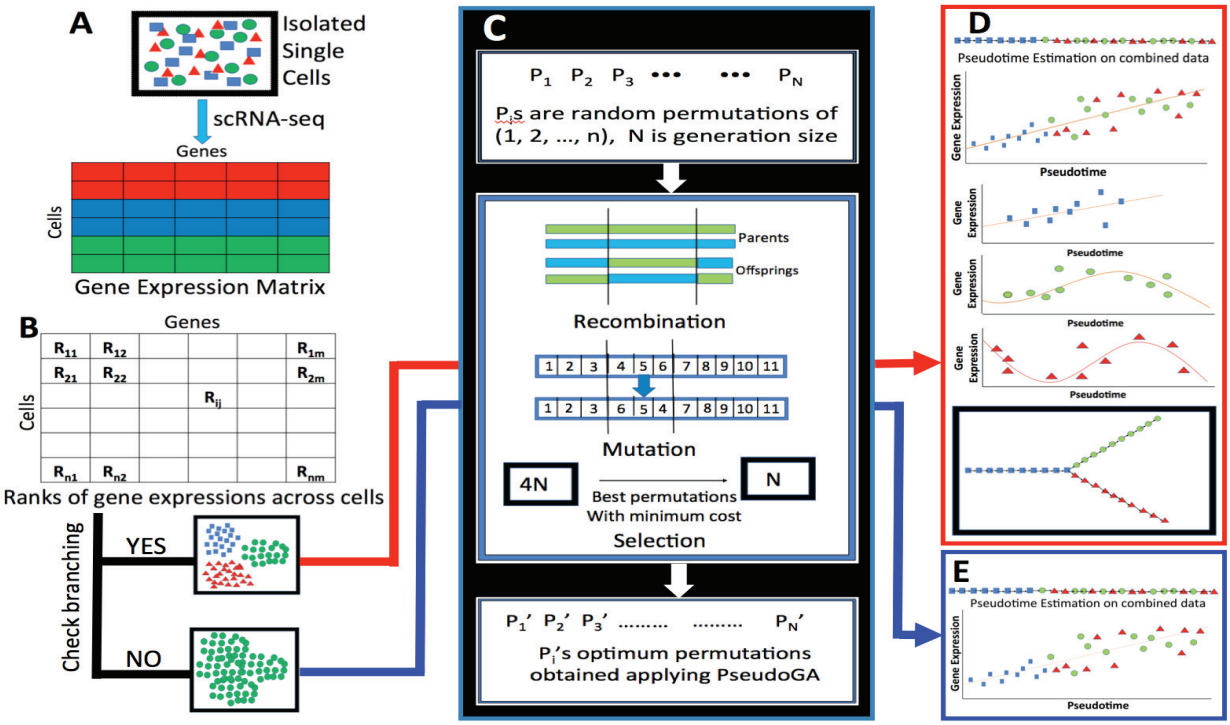


Figure 4.3: Outline of PseudoGA algorithm. (A) Single-cell transcription profiles are used as input data. (B) Expression matrix is transformed into ranks of individual genes across cells. To check branching, cells are clustered based on expression profiles into homogeneous groups of cells; otherwise keep the entire dataset. (C) PseudoGA algorithm is applied to each cluster or full dataset. The solution space is the set of all possible ordering of cells. A group of candidate solutions is considered as a population. Starting with an initial population, the population is made to go through recombination, mutation and selection to arrive at improved solutions. (D) Based on pseudotime of individual clusters, behaviors of gene expression profiles are examined. Paths from different clusters are combined to make joint inference for the entire data. (E) Creates one pseudotime trajectory based on full data.

Chromosome A	7 6 5 1 2 3 4 8
Chromosome B	6 1 4 2 5 7 3 8

Figure 4.4: Two different chromosomal representations of permutations.

4.2.2 Representation of ordering

Ordering of n objects can easily be represented by a permutation of natural numbers $1, 2, 3, \ldots, n$. Genetic algorithm [42] is a computational procedure that mimics biologically inspired operators such as mutation, crossover and selection to tackle the optimization problems (Figure 4.3C). It uses the idea of these biological phenomena in a computational or algorithmic paradigm and not in the actual biological sense. For example, a crossover in genetic algorithm generates a new list of permutations for evaluation in the next iteration, similar to a genomic crossover that generates a new set of markers on the chromosome. So, to apply genetic algorithm in an optimization problem, one first needs to find a suitable chromosomal representation (Figure 4.4) of a candidate solution, using which genetic operators like mutation, recombination, and selection can run on the space of all possible solutions.

In our work, we have used the permutation representation of ordering. We index the cells by $1, 2, \ldots, n$ where 1st, 2nd,..., n-th cells are chosen randomly. We represent a pseudotime ordering of cells indexed with $i_1, i_2, \ldots, i_n$ by the vector $(i_1, i_2, \ldots, i_n)$ which is indeed a permutation of $1, 2, \ldots, n$. Since the chromosomal representation is only for computational purpose and has no biological significance, recombination, mutation and selection operators when applied on a permutation give birth to a new one that needs to be checked for a better solution.

4.2.3 Cost function

Expression values of a gene over the pseudotime path may be a linear or nonlinear function of pseudotime. To make our proposed algorithm more general, we assume that the rank of the expression values over cells is a polynomial of pseudotime of degree at most 3 (Figure 4.3D, 4.3E). By using ranks instead of actual expression values (Figure 4.3B), we avoid the particular effect of any specific functional form of the gene expression, while retaining the general pattern. This non-parametric approach allows us to include a wide range of functional forms for gene regulation and also the outliers. There must be a tradeoff between

number of parameters and degree of the polynomial that is used to fit the model. Since it is well established that some genes may behave in a cyclic manner with pseudotime, we allow the polynomial degree up to 3 for fitting the data. Our careful extensive inspection and other studies [117, 55, 102, 121, 21, 85, 77] observe that gene expression regulation along pseudotime usually reveals expression patterns mainly of three types: (1) expression that increases or decreases with time, (2) expression that first increases and then decreases or vice versa, and (3) expression that first increases, then decreases, and then increases again with pseudotime, or vice versa. Our model can capture these three types of genes, assuming that ranks of gene expression values along pseudotime trajectory, can be either linear, quadratic or cubic function of the pseudotime. Mathematically, this model is:

$$R_{ij} = f_j(t_i) + \epsilon_{ij}$$

where R_{ij} is the rank of cell i in the expression levels of gene j, t_i is the pseudotime for cell i and ϵ_{ij} is the random error term. f_j is an unknown function according to which gene expression changes over pseudotime. In our set up, $t_i \in \{1, 2, \ldots, n\}$, for all i and $\{t_1, t_2, \ldots, t_n\}$ is a permutation of $\{1, 2, \ldots, n\}$. If we approximate f_j by a cubic polynomial, the regression equation becomes:

$$R_{ij} = \beta_{j0} + \beta_{j1}t_i + \beta_{j2}t_i^2 + \beta_{j3}t_i^3 + \epsilon_{ij}$$

Let $\hat{\beta}_{jk}$ be the least square estimate of β_{jk}, $k = 0, 1, 2, 3$, for a pseudotime ordering $(t_1, t_2, t_3, \ldots, t_n)$. Then, the cost associated with the ordering of j-th gene by cubic polynomial is given by Bayes Information Criterion: $\text{BIC}_{3j} = n\ln(\sigma_{j\epsilon}^2) + 3\ln(n)$ with
$\sigma_{j\epsilon}^2 = \frac{1}{n}\sum_{i=1}^{n}(R_{ij} - \hat{\beta}_{j0} - \hat{\beta}_{j1}t_i - \hat{\beta}_{j2}t_i^2 - \hat{\beta}_{j3}t_i^3)^2$

Similarly, we define BIC_{1j} and BIC_{2j} as BIC values associated respectively with fitting linear and quadratic polynomial on the rank of expression values with pseudotime as explanatory variable. Now the cost associated with j-th gene for the given pseudotime is $C_j = min(\text{BIC}_{1j}, \text{BIC}_{2j}, \text{BIC}_{3j})$. Overall cost associated with the pseudotime $\{t_1, t_2, \ldots, t_n\}$ for the whole transcriptome expression profile is $C = \sum_{j=1}^{n_G} C_j$ where n_G is the total number of genes. It is important to note that we treat the zero expressions in the data as numeric zeros and use them in ranking. If there are ties in expression values, we assign average rank to all observations with ties. The introduction of the cost function f_j adds more flexibility to our model. Any prior knowledge leading to more specific form of f_j can easily be incorporated in the model and the entire downstream protocol will follow accordingly. Naturally, this would result in more efficient estimation of pseudotime.

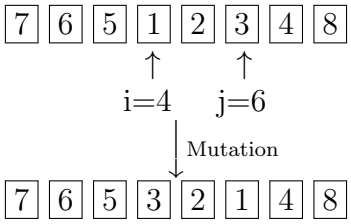


Figure 4.5: Mutation with the segment between position 4 to 6 being reversed.

4.2.4 Genetic algorithm for pseudotime construction

Let $\mathbb{X}$ be the space of all permutations of the set $\{1, 2, \ldots, n\}$. The cost function C is a function $C : \mathbb{X} \to \mathbb{R}$, where $\mathbb{R}$ denotes the real line. C contains penalty incurred due to non-optimality of an ordering. Hence, the optimal pseudotime ordering is obtained by minimising $C(x)$ with respect to x that moves in the space of all possible permutations *i.e.* $\mathbb{X}$. If x_{opt} is the optimal ordering, we have $x_{opt} = \arg\min_{x \in \mathbb{X}} C(x)$. Since the solution space is discrete, standard useful analytical tools like continuity or differentiability cannot be applied to find an optimal solution. Hence, we apply genetic algorithm to find x_{opt}. The algorithm uses the entire information from the dataset without any dimensionality reduction. Although it may not always find global optimality, it provides a reasonably good solution, at least, because no information is lost due to dimensionality reduction. Note that some other discrete optimisation algorithms may be used to address this problem. But we restrict to genetic algorithm and its modification tuned to this problem, mainly due to its wide applicability and better performance.

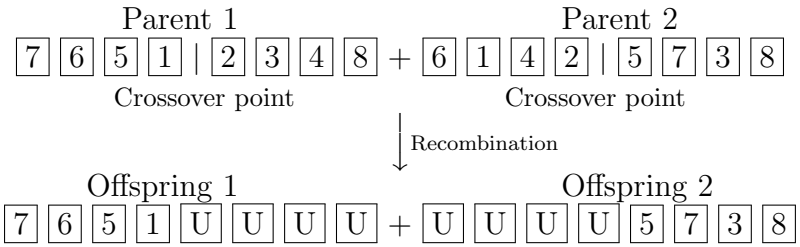
We consider three operators, mutation, recombination and selection in one single iteration. Mutation (Figure 4.5) creates a new vector y from a given permutation x by randomly choosing two positions i and j with $i \leq j$ such that $y_k = x_k(1 - I_{\{i \leq k \leq j\}}) + x_{i+j-k} I_{\{i \leq k \leq j\}}$ where $I_{\{.\}}$ denotes the indicator function. If x_i and x_j are the values of x at positions i and j respectively, after mutation the new values would be $y_i = x_j$ and $y_j = x_i$. This mutation operator is essentially an inversion [106] applied on a portion of the chromosome with two randomly chosen endpoints. Mutation adds N extra mutated individuals to an existing population of size N.

Several recombination or crossover operators on permutations have been suggested e.g. partially mapped crossover [41], order crossover [29], cycle crossover [87] etc. We propose a recombination operator (Figure 4.6) for this context that is similar to a partially mapped operator [41]. First, the set of individuals is divided into two subpopulations of equal size. Crossing over occurs between pairs with one from each group. Instead of taking only two cut points as in partially mapped crossover, we consider the number of cut points to be a Poisson random variable. If the random number generated is N, $(N + 1)$ fragments of equal length are formed from the parent string. Alternate fragments from one of the parents, say string I, are retained in one of the two newly formed offspring strings. To fill up the missing positions, the entries not retained in the newly formed string are recorded. A bipartite graph is constructed with the positional indices of those left behind entries of the two parent strings as vertices on two sides. All possible edges between vertices on the two sides are considered and the absolute differences between the ordinal position values are taken as edge weights. Based on the bipartite graph thus created, minimal bipartite matching is constructed. Entries in the positions considered in the other string, say string II, are put into the corresponding positions in string I based on the minimal bipartite matching graph. Using the same approach, another child is created by interchanging the role of string I and string II. Thus for an existing parent population of size N, same number of offsprings are added to it. It can be easily pointed out that the offsprings generated from this operation are not unique because the solution of minimal bipartite matching is not unique.

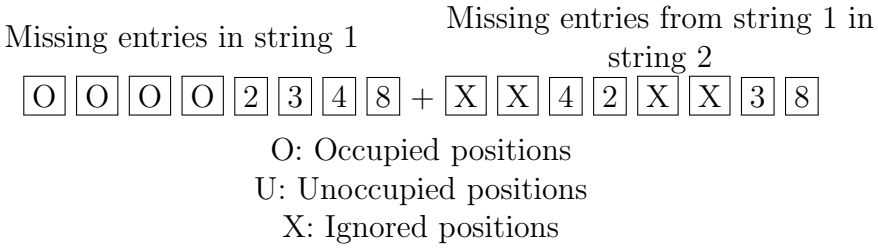
Mutation and recombination make a pool of $4N$ individuals from a pool of N. In the selection step (Figure 4.7), only the top one quarter *i.e.* top $\frac{N}{4}$ individuals with minimum cost, calculated on the basis of estimated cost function, are passed on to the next generation. This would keep the size of candidate solutions for x vector same in each iteration.

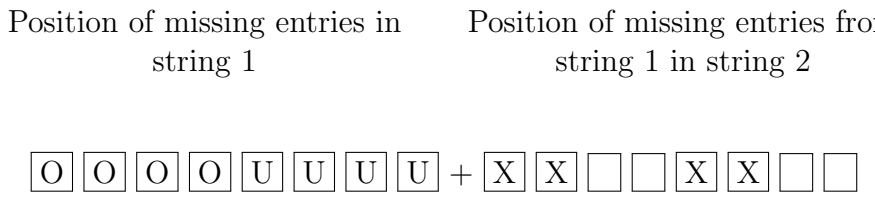
4.2.5 Construction of branching and lineage by joining different clusters

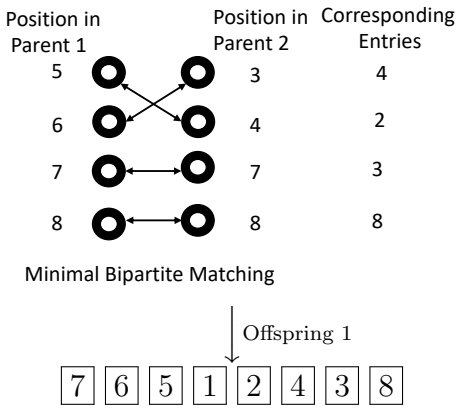
Till now we consider applying our method to the entire dataset. However, if we want to see any existence of branching, first we have to cluster the data. PseudoGA will be applied on each cluster considering it as the full data. Once the pseudotime orderings within the clusters are formed, we can construct lineages assuming a continuum between clusters (Figure 4.3D). This is important in many applications like construction of developmental trajectory, detection of bifurcation, building cellular lineage etc. Note that, the ordering within each cluster has two termination points. The distances between termination points


Parent 1
7 6 5 1 | 2 3 4 8 +
Crossover point
Parent 2
6 1 4 2 | 5 7 3 8
Crossover point
Recombination
Offspring 1
7 6 5 1 U U U U +
Offspring 2
U U U U 5 7 3 8

Offspring 1


Missing entries in string 1
Missing entries from string 1 in string 2
O O O O 2 3 4 8 + X X 4 2 X X 3 8
O: Occupied positions
U: Unoccupied positions
X: Ignored positions


Position of missing entries in string 1
Position of missing entries from string 1 in string 2
O O O O U U U U + X X □ □ X X □ □


Position in Parent 1
Position in Parent 2
Corresponding Entries
5 3 4
6 4 2
7 7 3
8 8 8
Minimal Bipartite Matching
Offspring 1
7 6 5 1 2 4 3 8

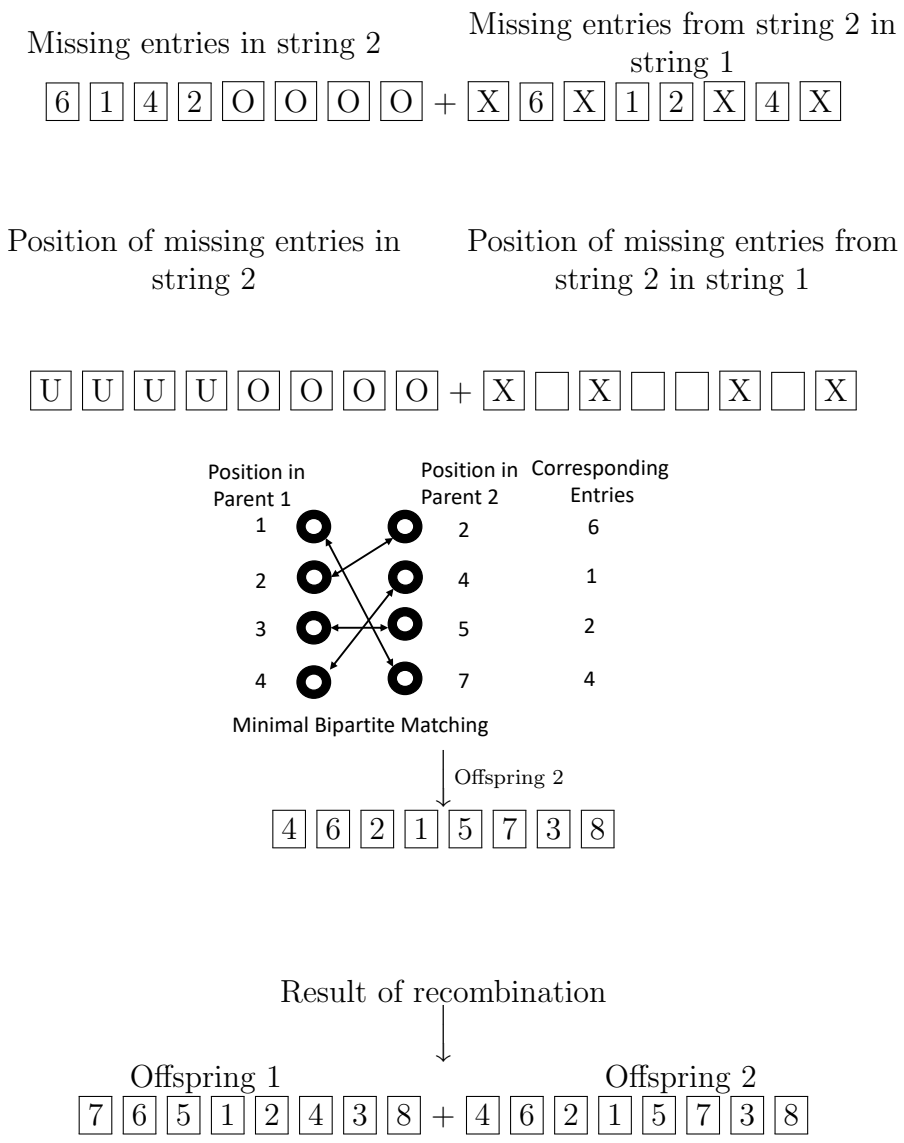


Figure 4.6: Recombination with breakpoint between position 4 and 5

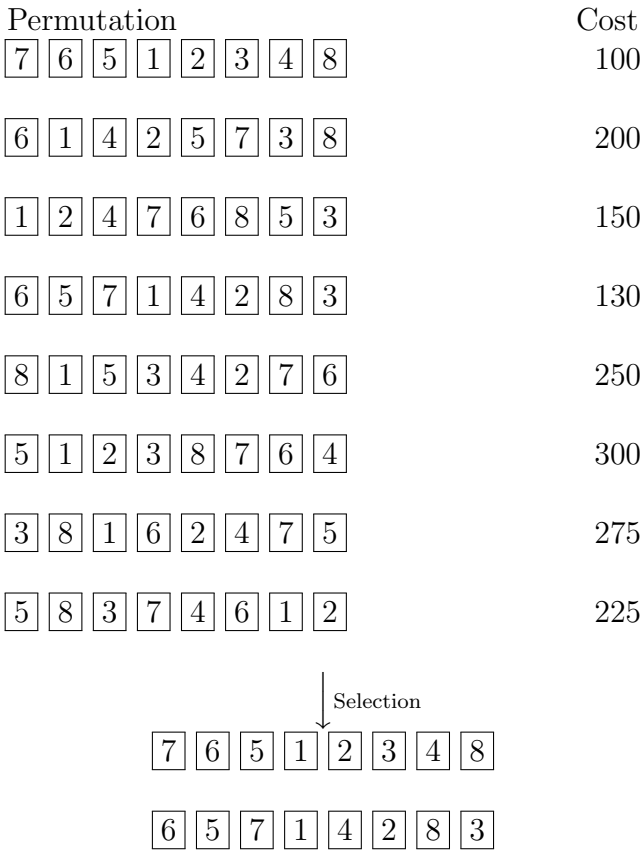


Figure 4.7: Selection with 8 permutations

across clusters are computed using extreme cells at either side of the path. The distance between two paths $\underset{\sim}{x} = (i_1, i_2, \ldots, i_m)$ and $\underset{\sim}{y} = (j_1, j_2, \ldots, j_n)$ is defined as $d(\underset{\sim}{x}, \underset{\sim}{y}) = min(C(x_1), C(x_2), C(x_3), C(x_4))$ where
$x_1 = (j_{(\lfloor \frac{n}{4} \rfloor)}, \ldots, j_2, j_1, i_1, i_2, \ldots, i_{(\lfloor \frac{m}{4} \rfloor)})$,
$x_2 = (j_{(\lfloor \frac{3n}{4} \rfloor+1)}, \ldots, j_{(n-1)}, j_n, i_1, i_2, \ldots, i_{(\lfloor \frac{m}{4} \rfloor)})$,
$x_3 = (i_{(\lfloor \frac{m}{4} \rfloor)}, \ldots, i_2, i_1, j_1, j_2, \ldots, j_{(\lfloor \frac{n}{4} \rfloor)})$, and
$x_4 = (i_{(\lfloor \frac{3m}{4} \rfloor+1)}, \ldots, i_{(m-1)}, i_m, j_1, j_2, \ldots, j_{(\lfloor \frac{n}{4} \rfloor)})$, where $C(x)$ is the cost function as defined before and the 'floor' function $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x.

A common approach to construct lineage from disjoint clusters of homogeneous populations of cells is by constructing minimal spanning tree (MST) on cluster centres [103, 55, 110]. Here we adopt a similar approach. Following Kruskal's algorithm for minimal spanning tree, the termination points with minimum distances are joined until a tree structure is constructed, taking into consideration that no cycle is formed. If multiple clusters join with a single termination point in a path, a new branching point is added near that termination point. In this way, we construct an undirected graph with tree like structure with branching points. If the purpose is to find a directed graph of clusters, the user would provide either a root cell or a root cluster. Now, a directed network is constructed using the root cluster and the undirected graph. In the HSMM dataset [94, 119], three clusters have been observed while performing cluster-wise pseudotime estimation. The t-SNE plot applied on this data clearly shows a lineage structure (Figure 4.8). The network between clusters by assuming cluster 2 as the root cluster is shown in Figure 4.9. The network can be visualized with any low dimensional representation of the data. It is consistent with the overall structure of reduced dimensional representations produced by PCA, diffusion map and t-SNE (Figure 4.10). In all these three embeddings, there is a transition from cluster 2 to a bifurcation into cluster 1 and cluster 3. Our pseudotime ordering agrees to the ordering with all three low dimensional embeddings. PseudoGA branching trajectory is very similar to the lineage produced by Monocle 2 [94] on the same dataset.

4.2.6 Pseudotime estimation with large number of cells

Genetic Algorithm for finding optimal permutation scales poorly with number of cells. Some modification of our algorithm is required to construct pseudotime with large number of cells. First, we subsample a smaller number of cells and apply our proposed method.

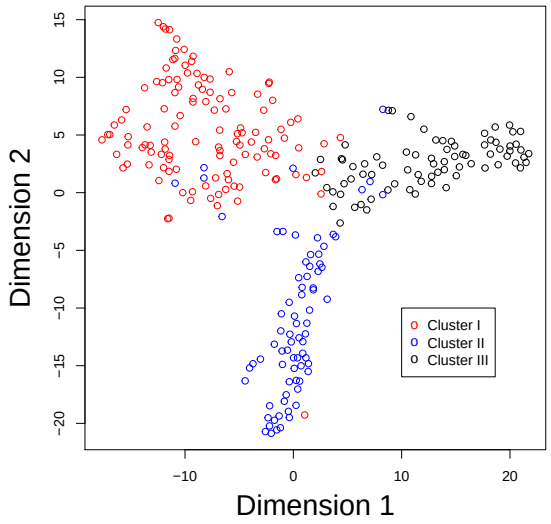


Figure 4.8: t-SNE plot of the HSMM data with memberships obtained from k-means clustering.

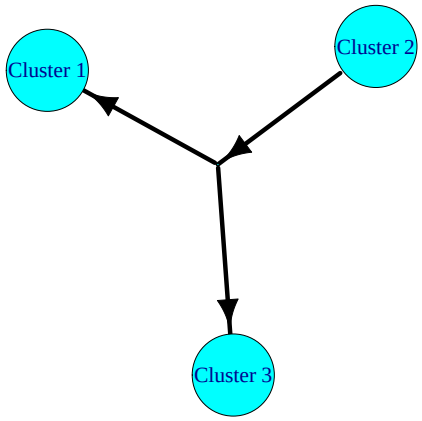
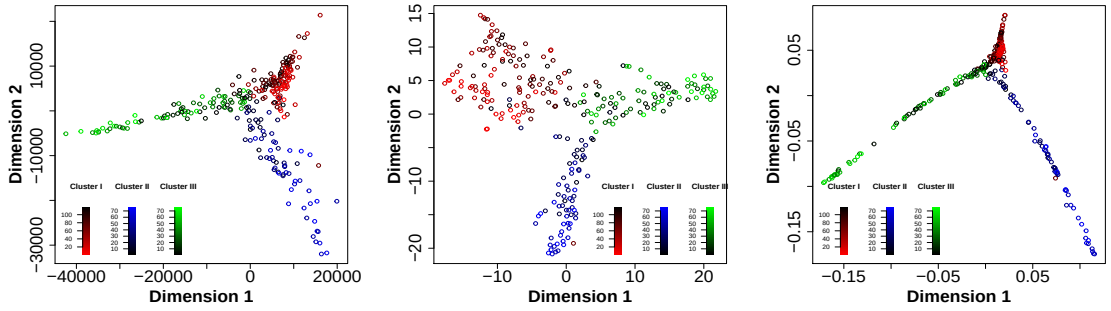


Figure 4.9: Network between clusters in HSMM data estimated by PseudoGA.



(A) **(B)** **(C)**

Figure 4.10: Visualization of PseudoGA ordering with **(A)** PCA, **(B)** t-SNE, and **(C)** Diffusion Map, using HSMM dataset. Clustering was performed using k-means clustering on reduced dimensional data obtained using t-SNE. We calculate 23 different indices and choose the optimum number of clusters by majority voting. This can be done using R package "NbClust" [23].

Algorithm 1: Algorithm for pseudotime ordering of cells and lineage construction

Input: Cell by gene matrix obtained from single-cell RNA-seq data. Choose an ϵ, a small preassigned positive quantity.

Output: Near optimum pseudotime ordering of cells.

Clustering: Perform clustering on cells to partition the cell population into homogeneous subpopulations. On each of the subpopulations, perform the next step. If we are not interested in branching, go directly to Pseudotime estimation with full dataset without any clustering.

Pseudotime estimation: Construct $\mathbb{Y}_0 = \{Y_1, \ldots, Y_n\}$: initial set of random permutations of cells.

while *Minimum cost function over the population converges* **do**

Step 1: Perform *Recombination* on $\mathbb{Y}_0$ to generate offsprings. Set of permutations becomes $\mathbb{Y}_1 = \{Y_1, \ldots, Y_n, Y_{1(o)}, \ldots, Y_{n(o)}\}$, where $\{Y_{1(o)}, \ldots, Y_{n(o)}\}$ are the offspring from $\{Y_1, \ldots, Y_n\}$ due to recombination. Here $\mathbb{C}(\mathbb{Y}_1) = 2n$, where $\mathbb{C}(A)$ is the cardinality (number of elements) of a set A.

Step 2: Perform *Mutation* on each element of $\mathbb{Y}_1$ to find a new augmented set of permutations $\mathbb{Y}_2 = \{\mathbb{Y}_1, \mathbb{Y}_1^{(m)}\}$ with $\mathbb{Y}_1^{(m)} = \{Y_1^{(m)}, \ldots, Y_n^{(m)}, Y_{1(o)}^{(m)}, \ldots, Y_{n(o)}^{(m)}\}$, where $Y_i^{(m)}$ and $Y_{i(o)}^{(m)}$ are new permutations due to mutation from Y_i and $Y_{i(o)}$ respectively for each $i = 1, \ldots, n$. Clearly $\mathbb{C}(\mathbb{Y}_2) = 4n$.

Step 3: Calculate cost for each permutation in $\mathbb{Y}_2$ and order them as $\mathcal{C}_{(1)}, \ldots, \mathcal{C}_{(4n)}$, where $\mathcal{C}_{(r)}$ is r-th ordered value of $\{\mathcal{C}_{(1)}, \ldots, \mathcal{C}_{(4n)}\}$. Selection is based on choosing minimum 25% i.e. n permutations corresponding to $\{\mathcal{C}_{(1)}, \ldots, \mathcal{C}_{(n)}\}$. Denote this new set of permutations as $\mathbb{Y}_0^{(1)}$ obtained after first iteration.

Step 4: Go back to Step 1 - 3 until $|\mathcal{C}_{(1)}^{\text{new}} - \mathcal{C}_{(1)}^{\text{old}}| < \epsilon$

end

Tree construction: Construct the branching or lineage by joining pseudotime orderings from different clusters using our proposed method.

Out of N total cells, pseudotime is estimated based on a subset of n ($n < N$) cells. Suppose $(t_1, t_2, \ldots, t_n)$ is the vector of estimated pseudotime for n cells. We define a score for every cell j, $S_j = \frac{1}{r} \sum_{k \in N_r(j)} t_k$ where $N_r(j)$ is the set of r nearest neighbors of cell j. The vector $S = (S_1, S_2, \ldots, S_N)$ or the ordering of S can be considered as the pseudotime for the original set of N cells.

To increase the efficiency, instead of inferring trajectory based on one subset, one can consider pooled inference from multiple subsamples as well. Based on B (say 30) subsamples each of size M (say 100) from the same dataset, we construct pesudotime trajectories separately. We find score ($S_{jb},\ b = 1, \ldots, B$) of the j-th cell corresponding to each b-th trajectory and construct a principal curve based on B dimensional scores of individual cells. The principal curve has been used for pseudotime reconstruction in different manners [110, 78, 19]. In our algorithm, ordering of cells on the principal curve is the final pseudotime trajectory for the entire dataset. Naturally, larger the number of subsamples, more will be the accuracy. However, we observe that 30 subsamples would show a significant improvement in correlation (0.99) with actual pseudotime (Figure 4.11). Inferring the final trajectory based on multiple estimates makes this approach robust to unwanted variation present in the data.

4.3 Results

We evaluate our method 'PseudoGA' and compare it to other methods using extensive simulations and five different real datasets including one that contains a large number of cells. In all datasets under consideration, we measure the accuracy using appropriate measures. Monocle [117], TSCAN [55], Slingshot [110], DPT [45], Waterfall [103] and scVelo [12] are used for comparison since they are all de novo pseudotime reconstruction techniques based on unique approaches and their open source codes are available. The benchmarking also indicates how different dimensionality reduction methods perform in constructing pseudotime trajectory. scVelo has been used for comparison on real data only because in synthetic gene expression datasets, expression values are directly simulated without mimicking exact RNA-seq experiment whereas scVelo requires raw reads for estimation.

4.3.1 Pseudotime determination using real data

We consider five real datasets for benchmarking. We evaluate reference trajectories for all these datasets based on the given information like time of collection, stage etc. To

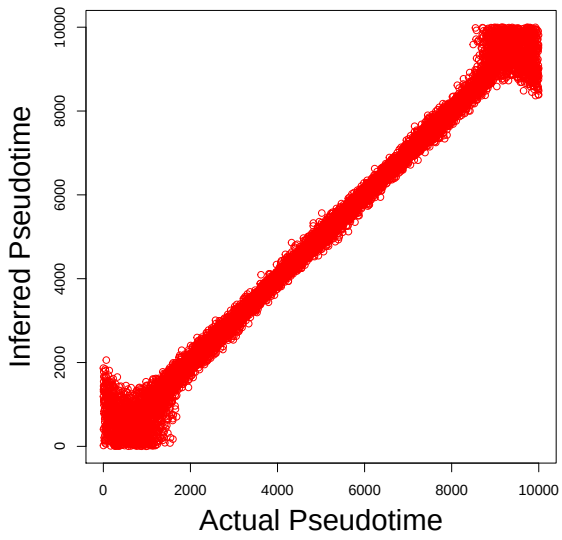


Figure 4.11: Pseudotime ordering by PseudoGA with multiple subsamples based on large data.

compare precisions of different estimates, we use absolute Spearman's rank correlation. Moreover, PseudoGA estimates can be visualized in low dimensional embeddings using any dimensionality reduction method including PCA. We also attempt to explore genes that are highly correlated with estimated pseudotime and whether they have any significance in the context of known actual pseudotime. We now briefly describe the real datasets and the interpretations of pseudotime for the concerned experiments.

Skeletal myoblasts are set to undergo a well-characterized sequence of morphological and transcriptional changes during differentiation. Primary human skeletal muscle myoblasts (HSMM) were expanded under high mitogen conditions and then differentiated by switching to low mitogen media (GSE52529) [117, 119]. RNA-seq libraries were sequenced from each of several hundred cells taken over a time-course of serum-induced differentiation. Around 49 to 77 cells were captured at each of the four time points (0, 24, 48 and 72 hours). The capture time here can be assumed to be the underlying pseudotime. First, we perform pseudotime analysis on the entire dataset. First principal component (PC) shows an increasing pattern with pseudotime estimated by PseudoGA (Figure 4.12B). Plot of PC II exhibits a parabolic pattern with respect to pseudotime (Figure 4.13).

If we are interested to see any lineage or branch structure with respect to pseudotime, we have to cluster the original data and apply our algorithm on each cluster. Clustering with t-SNE creates three clusters with one cluster consisting of observations from 0 hours only (Cluster II), two other clusters (Cluster I and III) with mixture of observations from 24 hours, 48 hours and 72 hours (Figure 4.15A). Cells from three different time points in cluster I are well separated whereas the cells from three populations in cluster III are mixed up. For visualisation, we plot PC I and PC II with respect to pseudotime as estimated by PseudoGA that show overall linear or quadratic trend in all these clusters (Figure 4.15B).

PseudoGA has the highest correlation among all methods under consideration when applied on the entire dataset (Figure 4.12A) although Slingshot performs slightly better when clusters are considered separately (Figure 4.14). However, PseudoGA seems more robust because its performance is consistently good in all scenarios. We find top 6 genes having highest correlation with pseudotime for the whole HSMM dataset as well as for three clusters separately (Tables 4.1, 4.2) and observe the change of expression along pseudotemporal path (Figures 4.16 and 4.17).

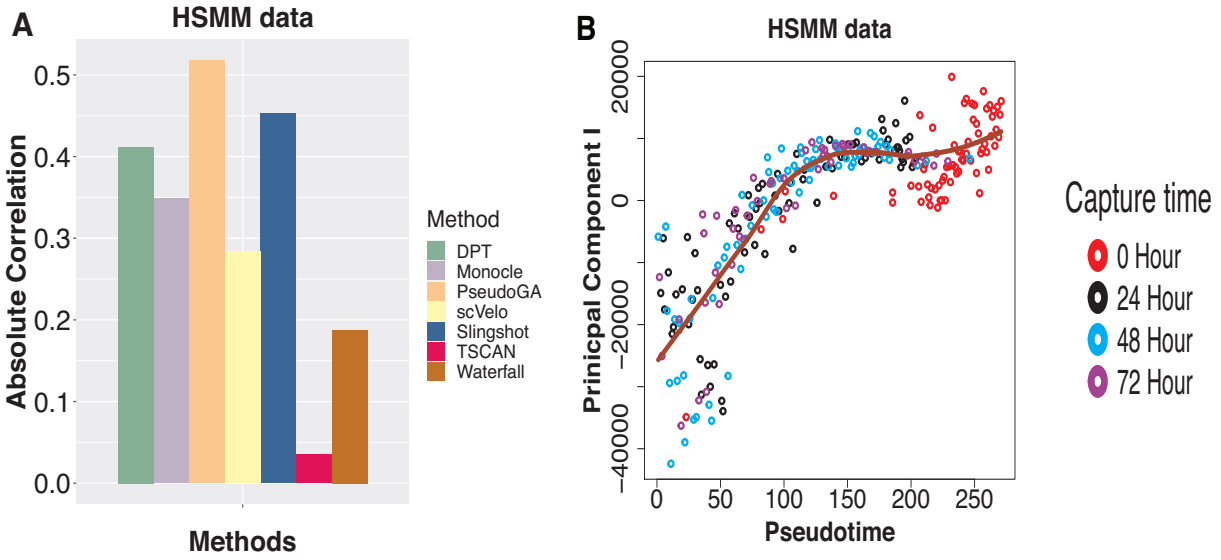


Figure 4.12: (A) Absolute Spearman's rank correlation between HSMM data capture time and pseudotime produced by different methods for the entire dataset. PseudoGA shows the highest correlation among all methods. (B) Plot of PC I with pseudotime estimated by PseudoGA shows monotonically increasing pattern.

Cluster id	Genes with highest correlation	Common function
Entire dataset	TNNT2, MT2A, TXN, ACTC1, NCAM1, MACF1	Muscle functioning

Table 4.1: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA ignoring clusters and their common function in the HSMM dataset.

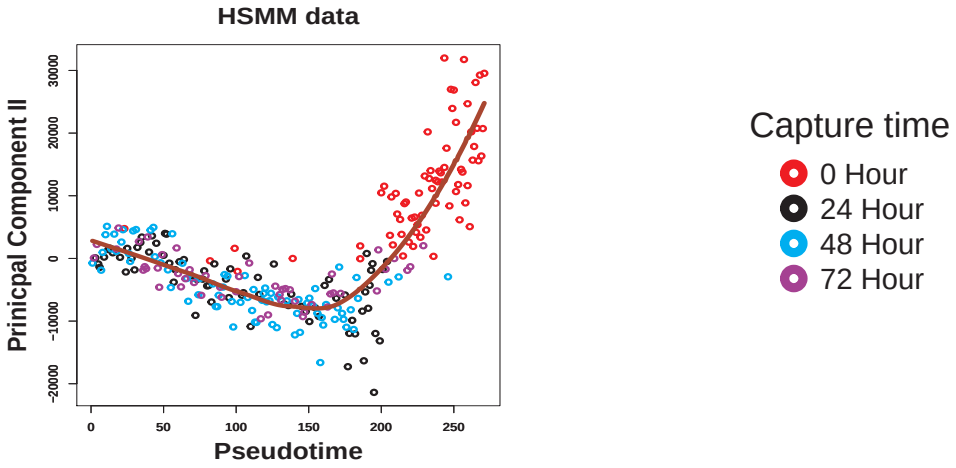


Figure 4.13: Plot of second principal component on the whole data with pseudotime estimated by PseudoGA. PC II shows quadratic pattern with pseudotime estimated by PseudoGA.

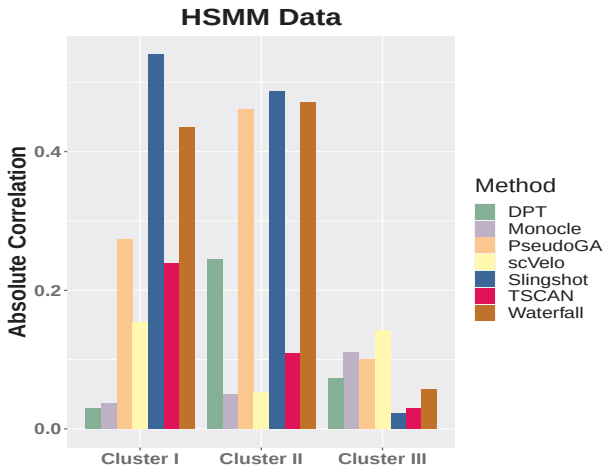


Figure 4.14: Absolute Spearman's rank correlation between HSMM data capture time and pseudotime produced by different methods. PseudoGA remains among top three methods across all three clusters.

Cluster id	Genes with highest correlation	Common function
Cluster I	TXN, LTBP1, HMGA1, EEF1A1, H19, FN1	Oxidative stress
Cluster II	MT2A, CD59, MT1E, MT1L, TAGLN2, TXN	Oxidative stress
Cluster III	HMGA1, COX6A2, MT1G, MT2A, TAGLN2, AC112721.2	Heme binding

Table 4.2: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA in each cluster and their common functions in HSMM dataset.

Our next dataset contains single-cell RNA-seq data from 1861 mouse dendritic cells stimulated with three pathogenic components. This dataset is used to examine the variation between individual cells exposed to the same stimulus and study the complex dynamic responses to the stimulus exhibited by multicellular populations (GSE48968) [102]. Cells were captured initially without any stimulus, and at one, two, four and six hours after applying the stimulus. Cell capture time in this case can be considered as the pseudotime. To compare accuracy of different methods, cells that were applied different types of stimuli, were sorted out. Three different types of stimuli, namely LPS, PAM and PIC were applied.

We apply pseudotime estimation algorithms on all these three types of data with different stimuli. Application of PseudoGA on the entire data for each stimulus shows that estimated pseudotimes are in overall congruence with the actual pseudotime (Figure 4.18A). First two principal components show strong functional relationship with pseudotime estimated by PseudoGA (Figure 4.18B, 4.19). Under all these three types of stimuli, PseudoGA shows the best performance among these seven methods (Figure 4.18A). Only in the data for mice treated with LPS, Monocle performs better than PseudoGA although for other two datasets its performance is not really good. On the other hand, PseudoGA consistently shows high correlation and clear pattern with pseudotime for all stimuli. Top 6

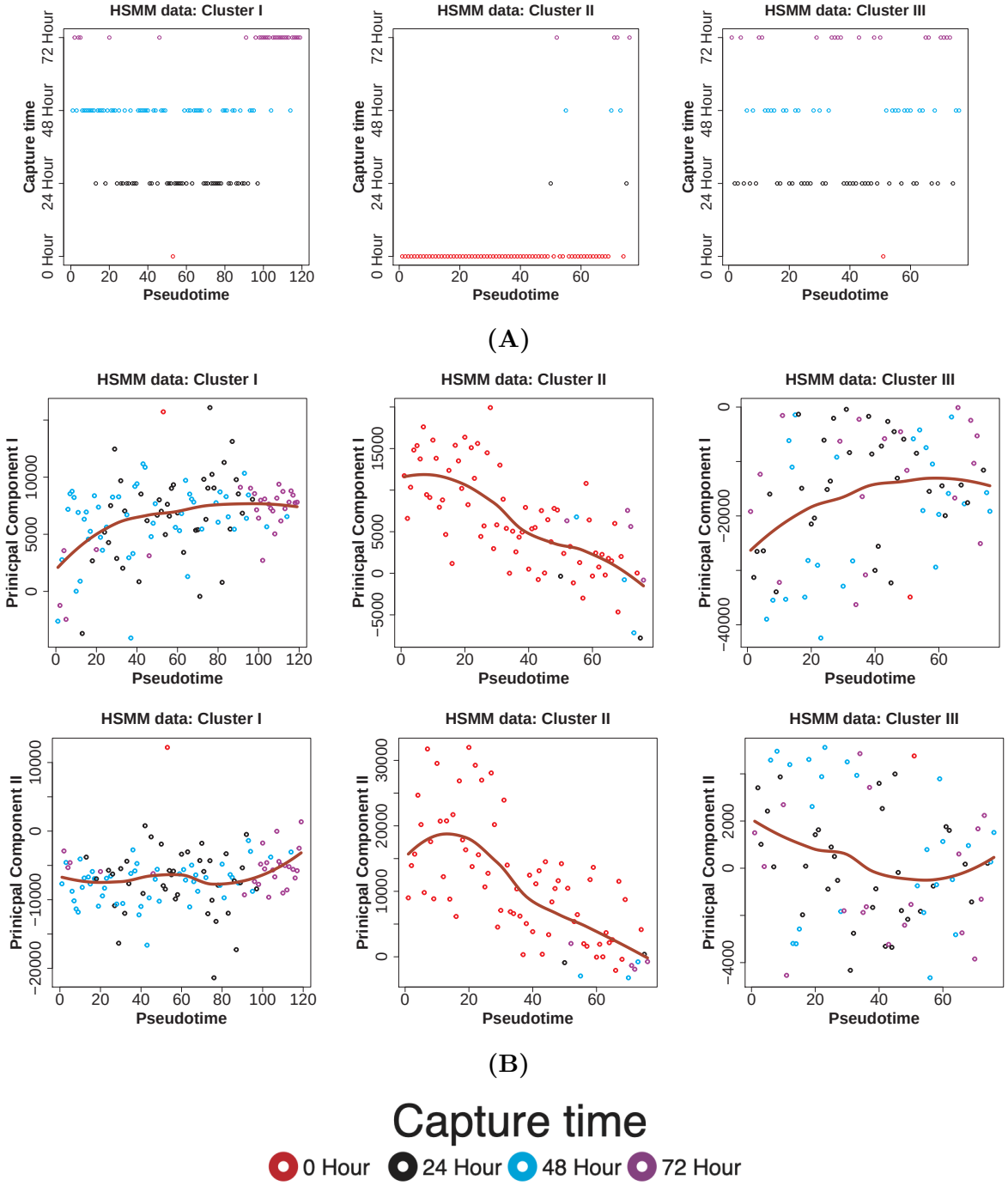


Figure 4.15: (A) Pseudotime ordering by PseudoGA on HSMM dataset on three different clusters (B) Functional relationship between first two principal components and pseudotime estimated by PseudoGA on three different clusters. In all cases, the relationship is either linear or quadratic.

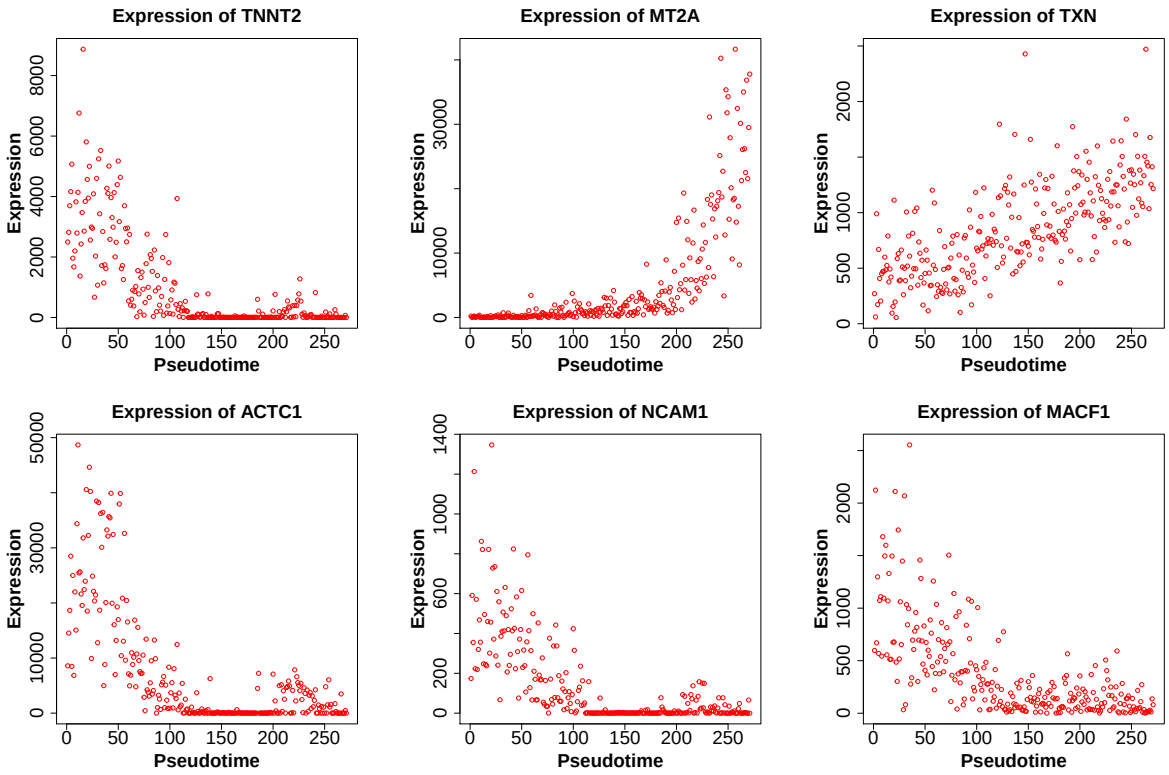
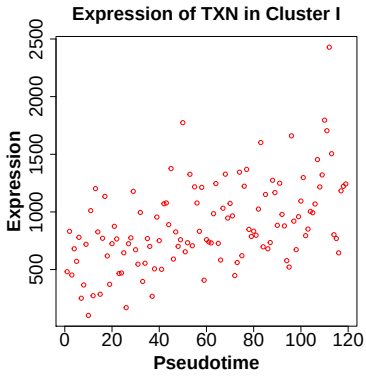
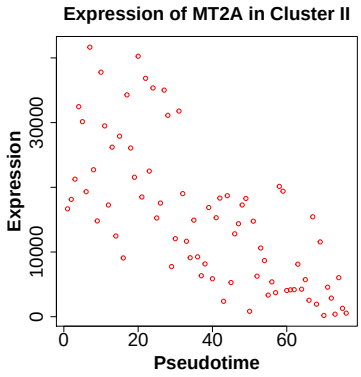
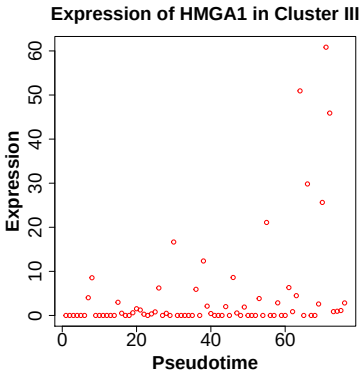
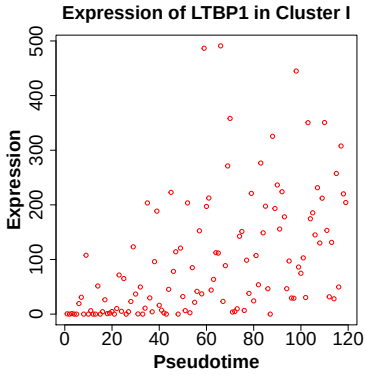


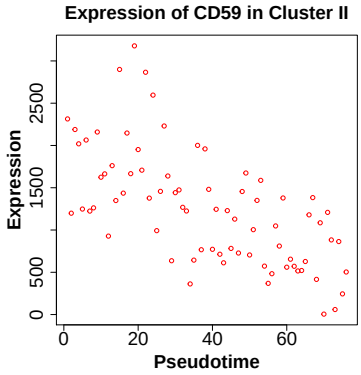
Figure 4.16: Change of expression pattern with pseudotime in HSMM dataset showing top 6 genes with highest correlation on the entire dataset.

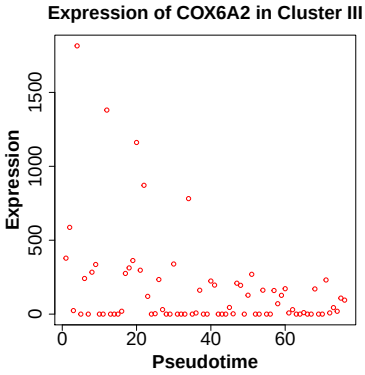

Expression of TXN in Cluster I
Expression
Pseudotime

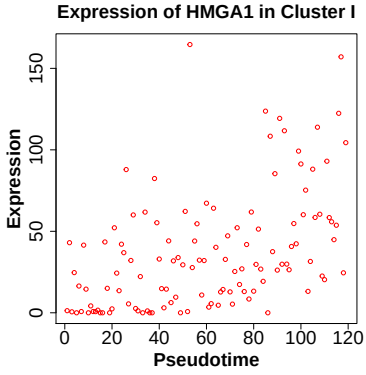

Expression of MT2A in Cluster II
Expression
Pseudotime

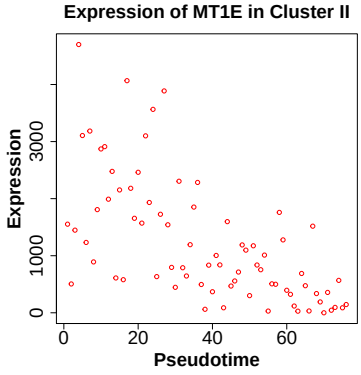

Expression of HMGA1 in Cluster III
Expression
Pseudotime

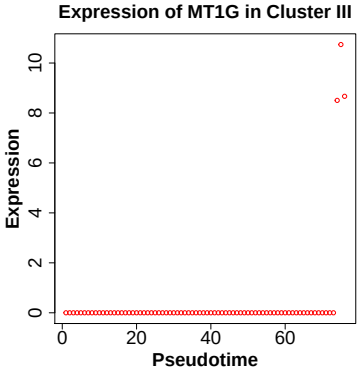

Expression of LTBP1 in Cluster I
Expression
Pseudotime


Expression of CD59 in Cluster II
Expression
Pseudotime


Expression of COX6A2 in Cluster III
Expression
Pseudotime


Expression of HMGA1 in Cluster I
Expression
Pseudotime


Expression of MT1E in Cluster II
Expression
Pseudotime


Expression of MT1G in Cluster III
Expression
Pseudotime

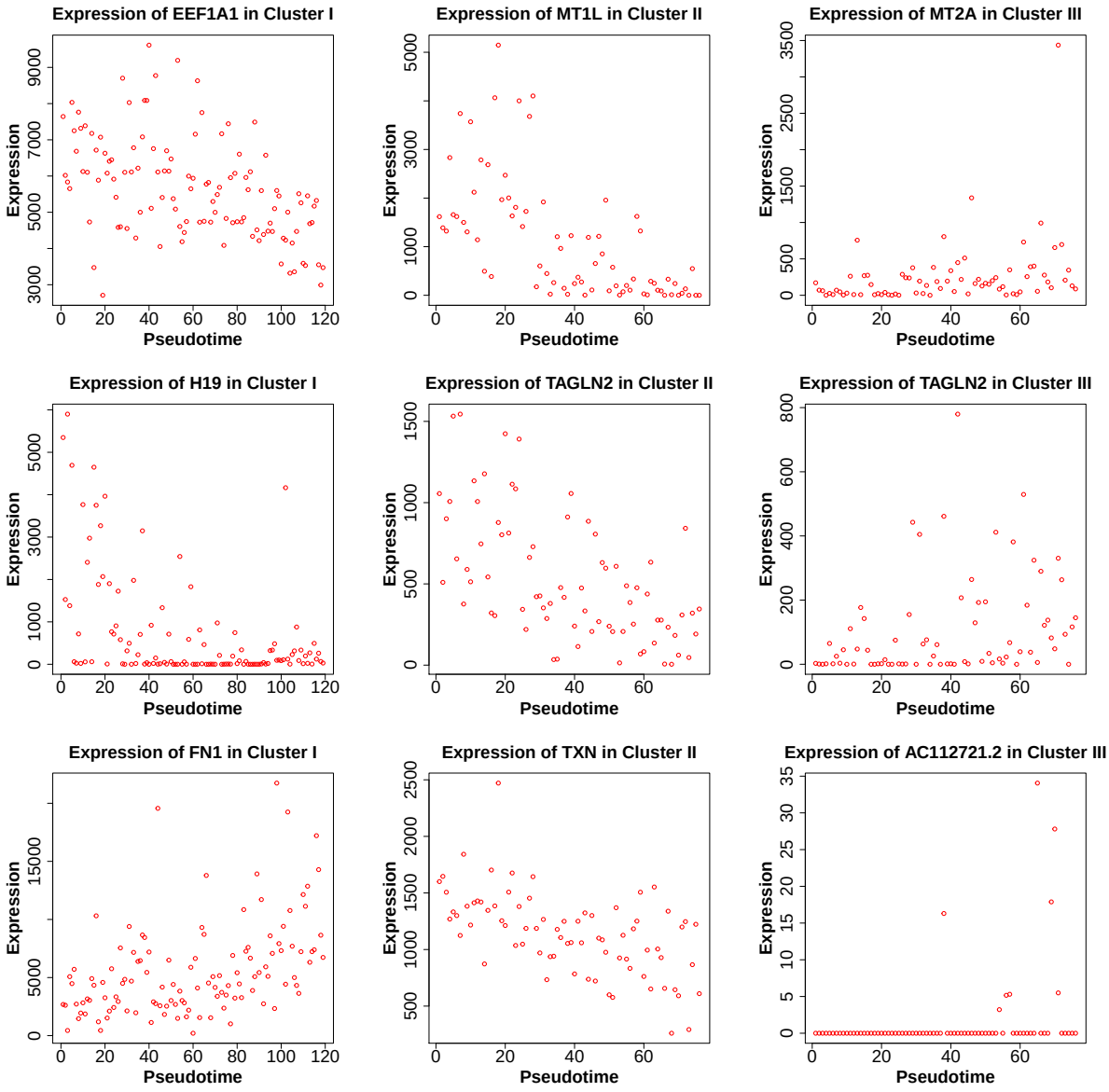


Figure 4.17: Change of expression pattern with pseudotime in HSMM dataset showing top 6 genes with highest correlation in each of the three clusters.

genes having highest correlation with pseudotime in different clusters and their common functions are shown in Tables 4.3-4.5. The change of expressions with pseudotime along pseudotemporal path produced by PseudoGA in these three scenarios are shown in Figures 4.20, 4.21, 4.22.

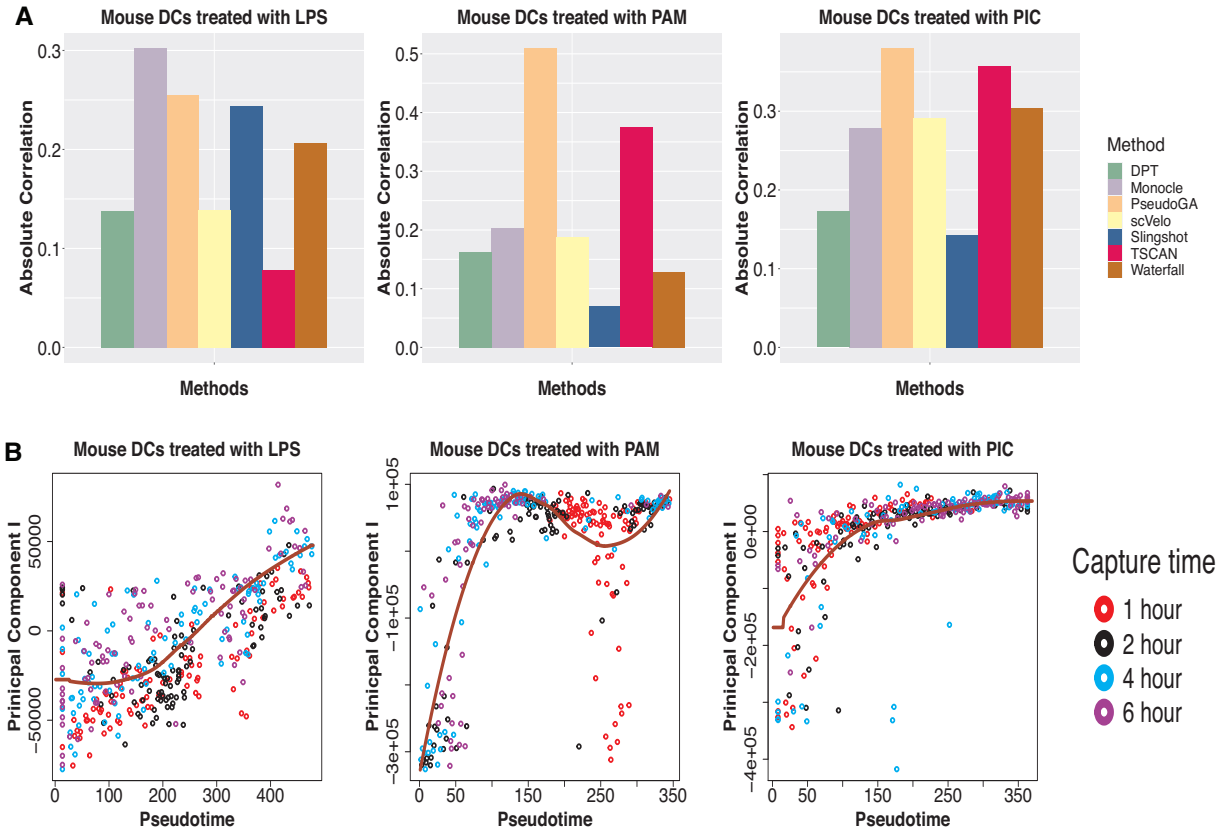


Figure 4.18: (A) Absolute Spearman's rank correlation between capture time of dendritic cells and pseudotime produced by different methods for the entire dataset under stimulation by LPS, PAM and PIC. PseudoGA shows overall best performance. (B) Plot of PC I with pseudotime estimated by PseudoGA for three different types of simulation. In the stimulation by LPS, PC I shows linear pattern whereas in the other two datasets, PC I shows bursting type pattern with pseudotime.

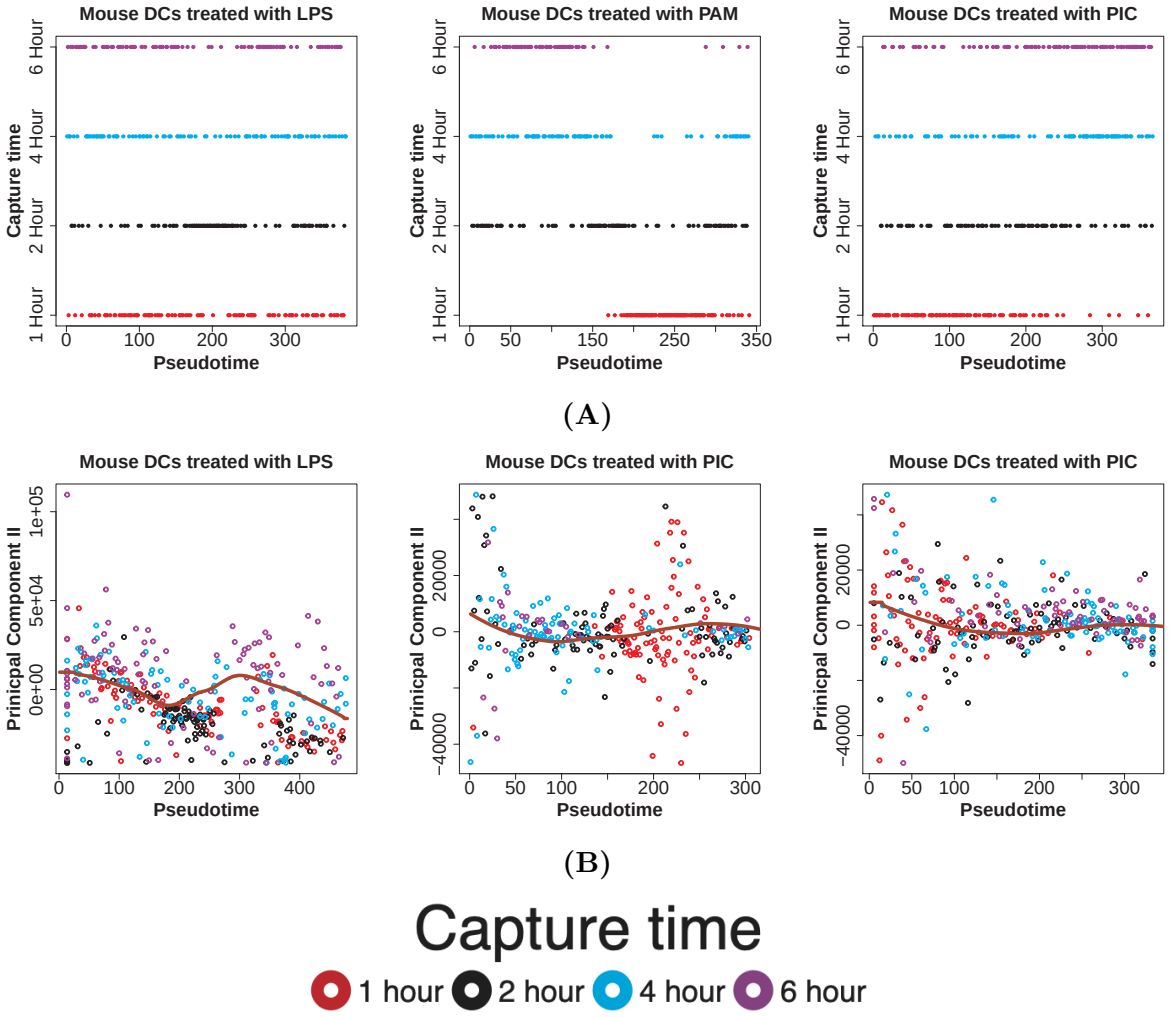


Figure 4.19: (A) Pseudotime estimated by PseudoGA on the dataset where mouse dendritic cells are treated with LPS, PAM and PIC respectively. (B) Functional relationship between PC II and the pseudotime estimated by PseudoGA on the same datasets.

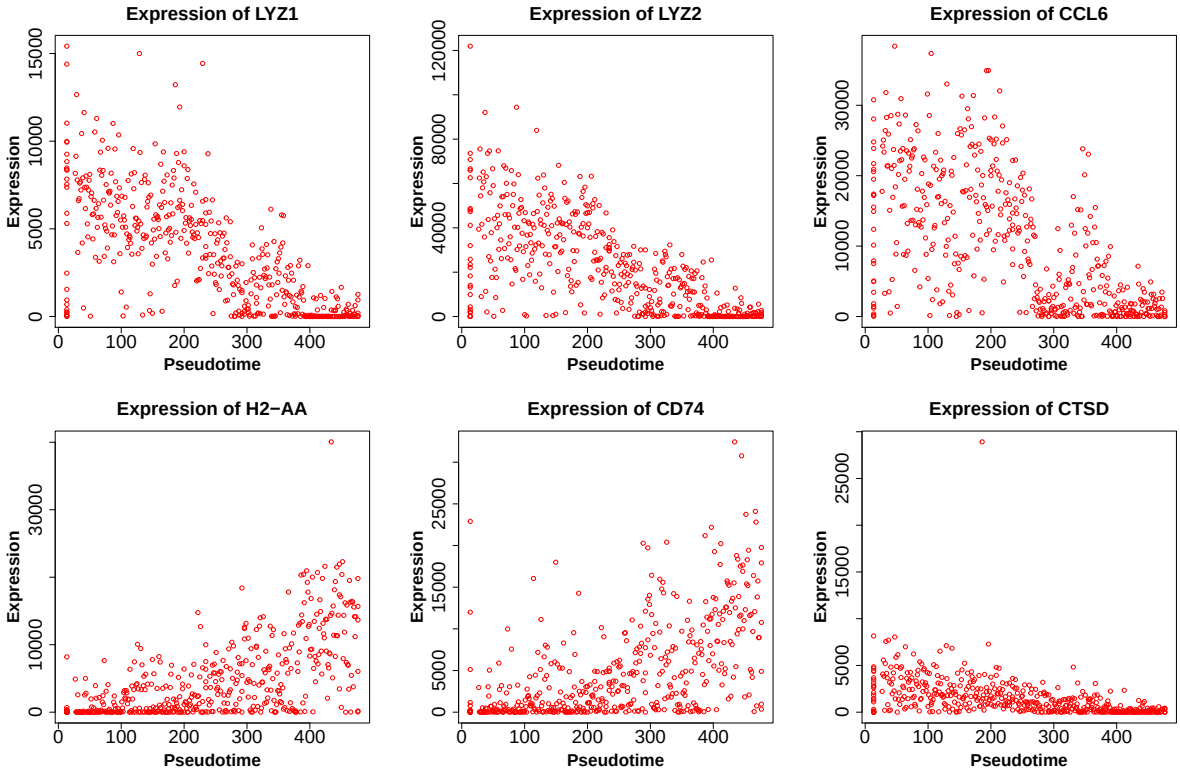


Figure 4.20: Change of expression pattern with pseudotime when dendritic cells are treated with LPS stimulus showing top 6 genes with highest correlation.

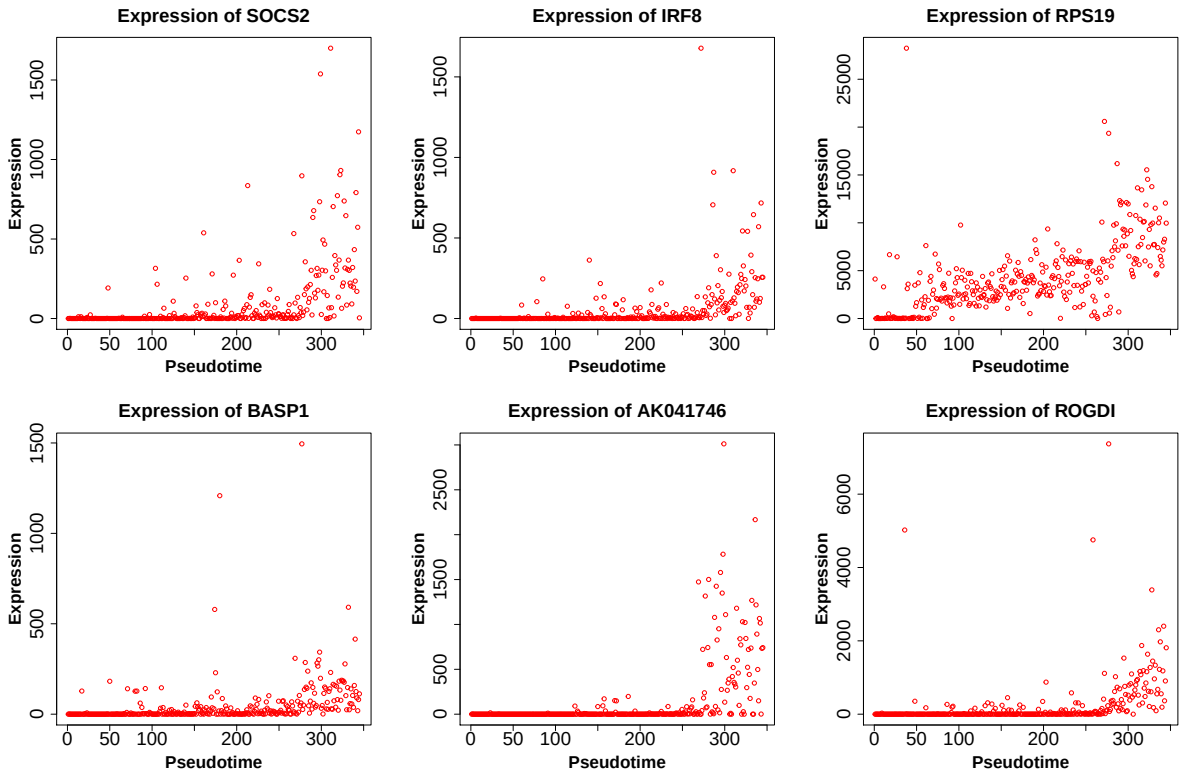


Figure 4.21: Change of expression pattern with pseudotime when dendritic cells are treated with PAM stimulus showing top 6 genes with highest correlation.

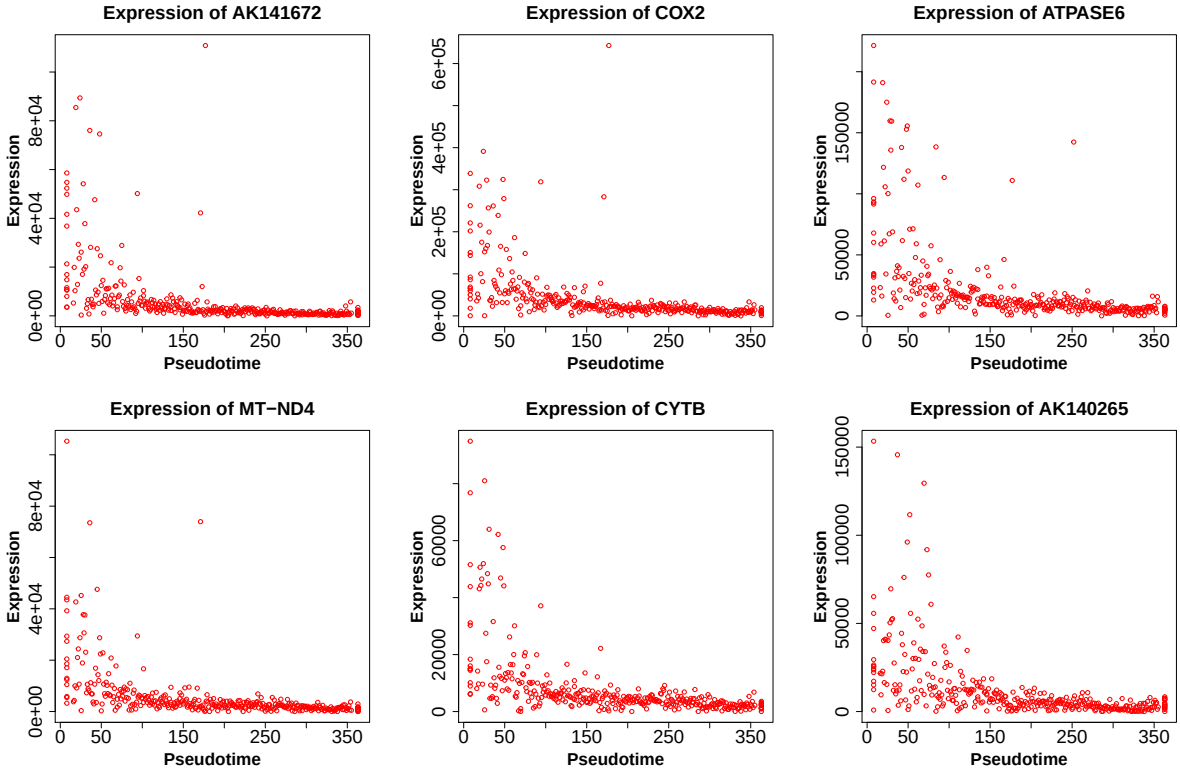


Figure 4.22: Change of expression pattern with pseudotime when dendritic cells are treated with PIC stimulus showing top 6 genes with highest correlation.

The third dataset consists of microfluidic single-cell RNA-seq on 185 individual mouse lung epithelial cells at four different stages (E14.5, E16.5, E18.5, adult) of development (GSE52583) [121]. The transcriptome data present transcriptional states that define the developmental and cellular hierarchy of distal mouse lung epithelium. Cells were assigned to two groups, prenatal and postnatal cells. Here the developmental stage can be considered as the underlying pseudotime. The plot of PC I shows transcriptional bursting pattern whereas the plot of second principal component shows monotonic pattern (Figure 4.23A, 4.24).

Monocle shows the highest correlation with actual pseudotime whereas PseudoGA turns out to be second best (Figure 4.23A). Top 6 genes having highest correlation with pseudotime and their common functions are shown in Table 4.6. The change of expressions with

Cells treated with LPS		
Cluster id	Genes with highest correlation	Common function
Entire dataset	LYZ1, LYZ2, CCL6, H2-AA, CD74, CTSD	Immune response

Table 4.3: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA and their common function in dendritic cells dataset with LPS stimulus.

Cells treated with PAM		
Cluster id	Genes with highest correlation	Common function
Entire dataset	SOCS2, IRF8, RPS19, BASP1, AK041746, ROGDI	Apoptosis(except AK041746)

Table 4.4: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA and their common function in dendritic cells dataset with PAM stimulus.

Cells treated with PIC		
Cluster id	Genes with highest correlation	Common function
Entire dataset	AK141672, COX2, ATPASE6, MT-ND4, CYTB, AK140265	Mitochondrial functions

Table 4.5: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA and their common function in dendritic cells dataset with PIC stimulus.

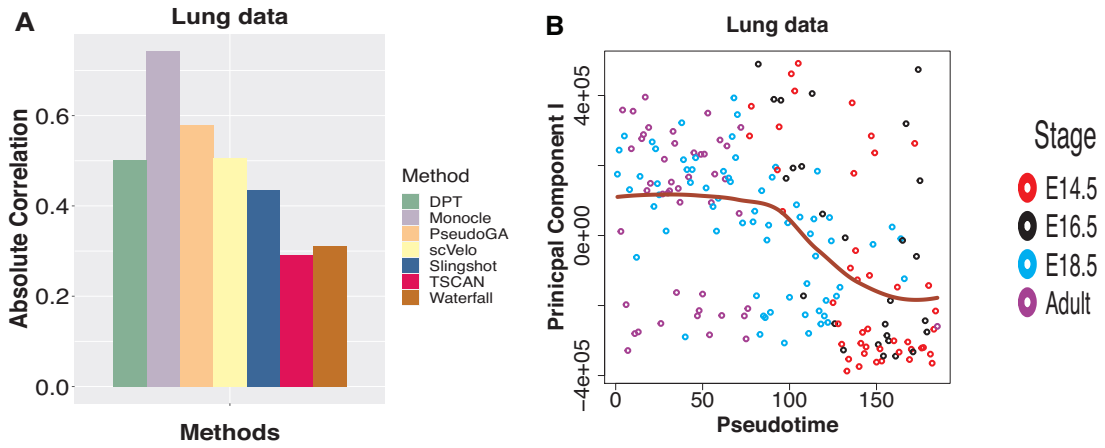


Figure 4.23: (A) Absolute Spearman's rank correlation between developmental stage and pseudotime assigned by different methods on lung data. Monocle shows the highest correlation followed by PseudoGA. (B) Plot of PC I with pseudotime estimated by PseudoGA. PC I shows decreasing pattern with pseudotime though it can also be viewed as bursting.

Cluster id	Genes with highest correlation	Common function
Entire dataset	AGER, THEM123, EMP2, AKAP5, HOPX, TIMP3	Lung functioning

Table 4.6: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA and their common function in mouse lung dataset.

pseudotime along pseudotemporal path is shown in Figure 4.25.

In the fourth dataset, we study the gene expression patterns at single cell level across three different cell cycle stages each containing 96 mouse embryonic stem cells (E-MTAB-2805) [16]. Single-cell RNA-seq was performed on mouse embryonic stem cells (mESC) that were stained with Hoechst 33342 Flow cytometry and sorted for G1, S and G2M stages of cell cycle. PseudoGA is able to separate cells with respect to G1, G2M and S stages from a mixture of cells. Thus it provides a potential ordering of cells across cell cycle. Only for visualization purpose, when we plot the first two PCs, it seems that PC I across pseudotime shows an increasing pattern whereas PC II indicates linear pattern (Figure 4.26B, 4.27).

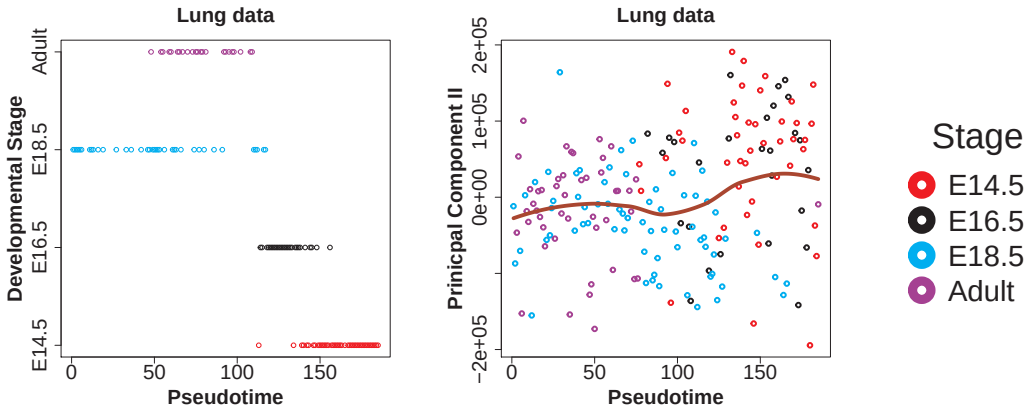


Figure 4.24: Pseudotime ordering by PseudoGA on mouse lung dataset and the functional relationship between PC II and the pseudotime estimated. The plot shows monotonic pattern of PC II with pseudotime.

We consider maximum correlation among all possible permutations of G1, S and G2M stages as the order of cell cycle can be rearranged. PseudoGA shows the highest correlation among all methods (Figure 4.26A). Top 6 genes having highest correlation with pseudotime and their common functions are shown in Table 4.7. The change of expressions with pseudotime along pseudotemporal path in these three scenarios are shown in Figure 4.28.

We consider another dataset where gene expression profiles of human ventral midbrain cells were studied at different developmental stages after gestation ranging between 0 to 11 weeks (GSE76381) [63]. This dataset is much larger than the four other datasets discussed earlier. Single-cell RNA sequencing was performed on 4029 cells from different stages. So, the developmental stage of a cell can be considered as the inherent pseudotime in this case. We performed the pseudotime analysis on the entire dataset. PseudoGA estimate shows the highest correlation with the actual pseudotime (Figure 4.29A). The first two principal components show quadratic or cubic functional relationship with the pseudotime estimated by PseudoGA (Figure 4.29B, 4.30). Top 6 genes with highest correlation with the estimated pseudotime have either cubic or bursting type pattern (Figure 4.31, Table 4.8). This result establishes the consistency of performance as well as robustness of PseudoGA when applied to a large number of cells. However, it is to be noted that if we want to see any possible

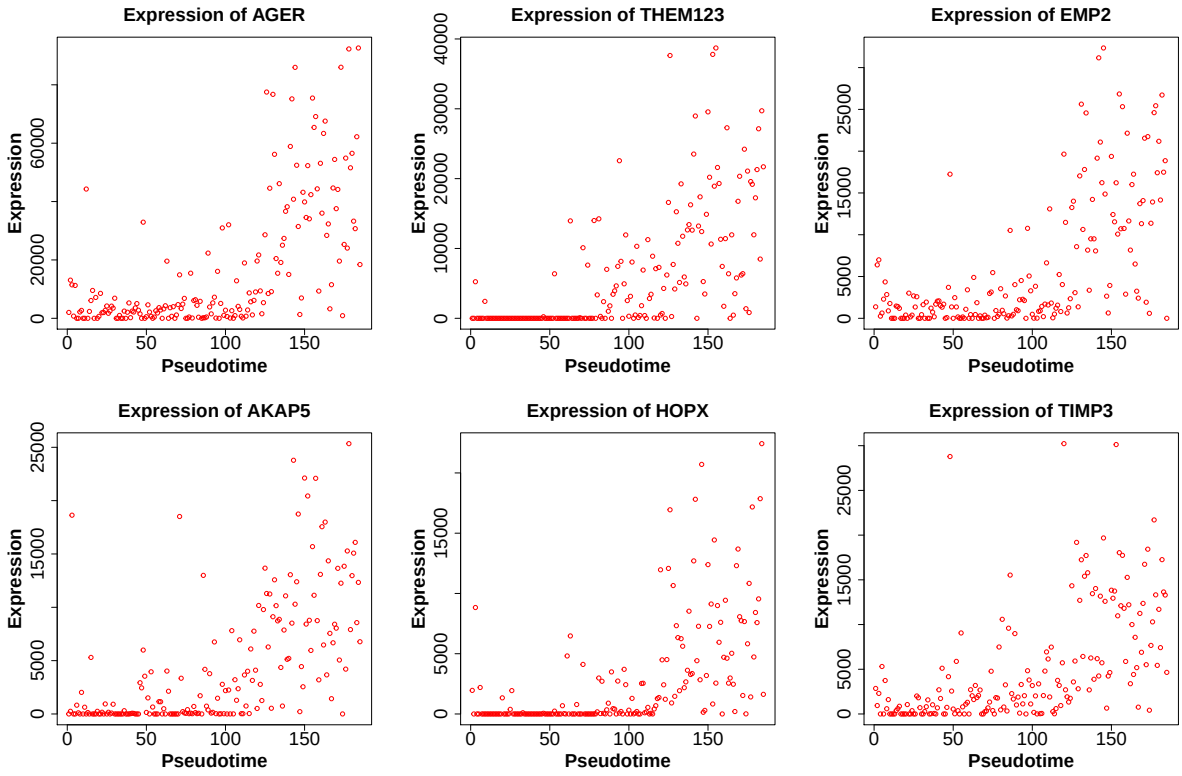


Figure 4.25: Change of expression pattern with pseudotime in mouse lung dataset showing top 6 genes with highest correlation.

Cluster id	Genes with highest correlation	Common function
Entire dataset	Gm13341(Pseudogene), mt-Co1(mtRNA), Gm14303(Pseudogene), mt-Rnr2(mtRNA), RetrogenDB(Pseudogene), mt-Cytb (mtRNA)	Unknown

Table 4.7: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA and their common function in mouse cell cycle dataset.

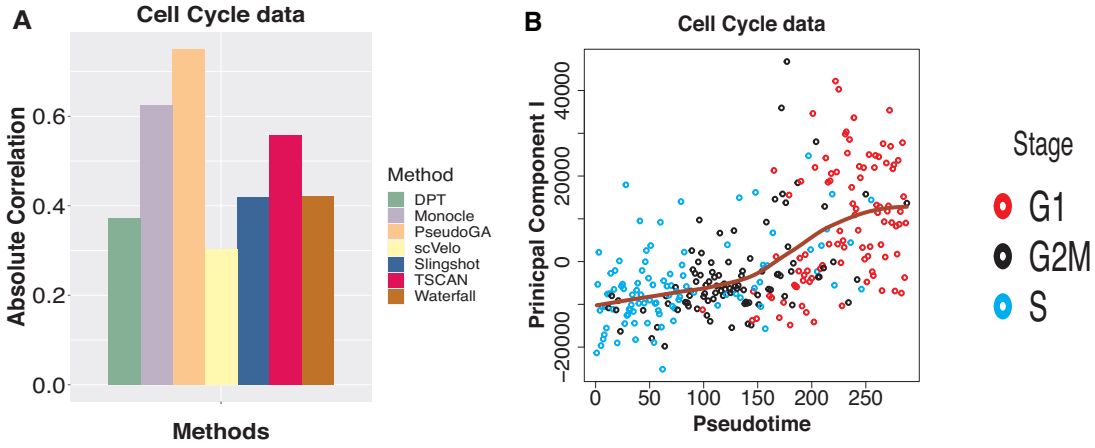


Figure 4.26: (A) Absolute Spearman's rank correlation between developmental stage and pseudotime assigned by different methods on cell cycle data. Maximum correlation among all possible permutations of G1, S and G2M stages has been shown. PseudoGA again shows the highest correlation followed by Monocle. (B) Plot of PC I with pseudotime estimated by PseudoGA. The plot shows increasing pattern of PC I.

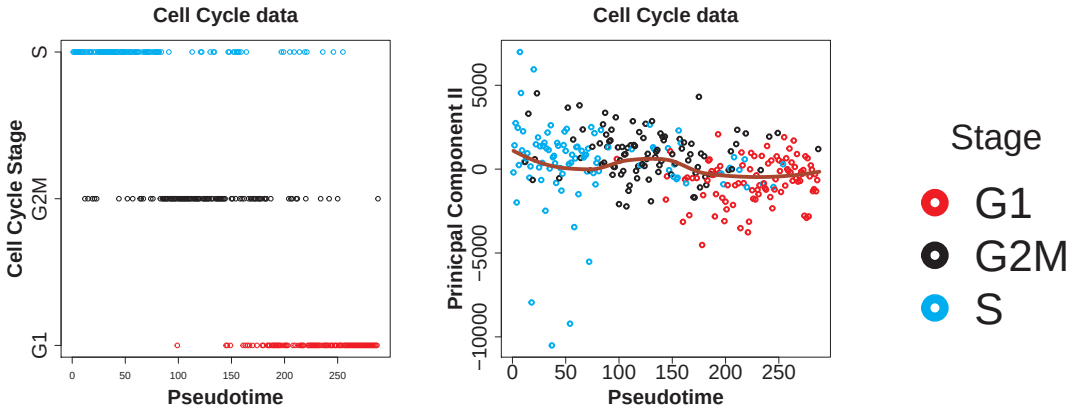


Figure 4.27: Pseudotime ordering by PseudoGA on mouse cell cycle dataset and the functional relationship between PC II and the pseudotime estimated. PC II shows linear pattern with the pseudotime estimated.

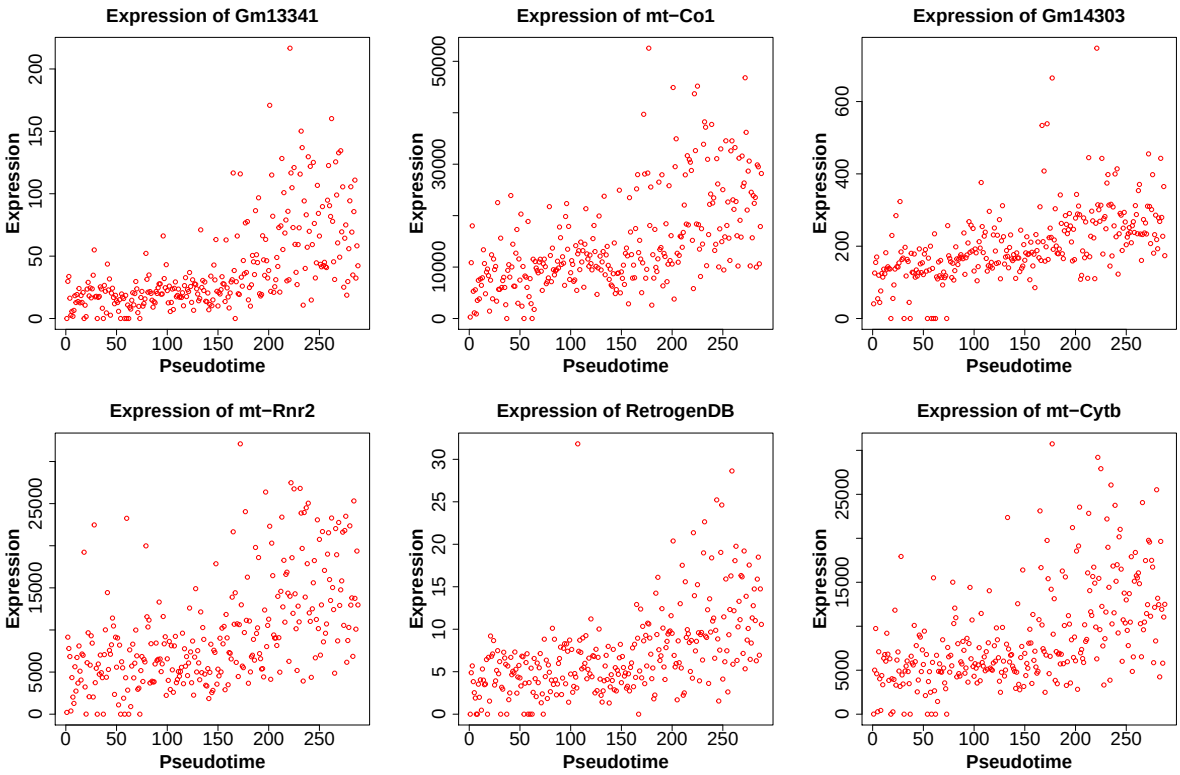


Figure 4.28: Change of expression pattern with pseudotime in mouse cell cycle dataset showing top 6 genes with highest correlation.

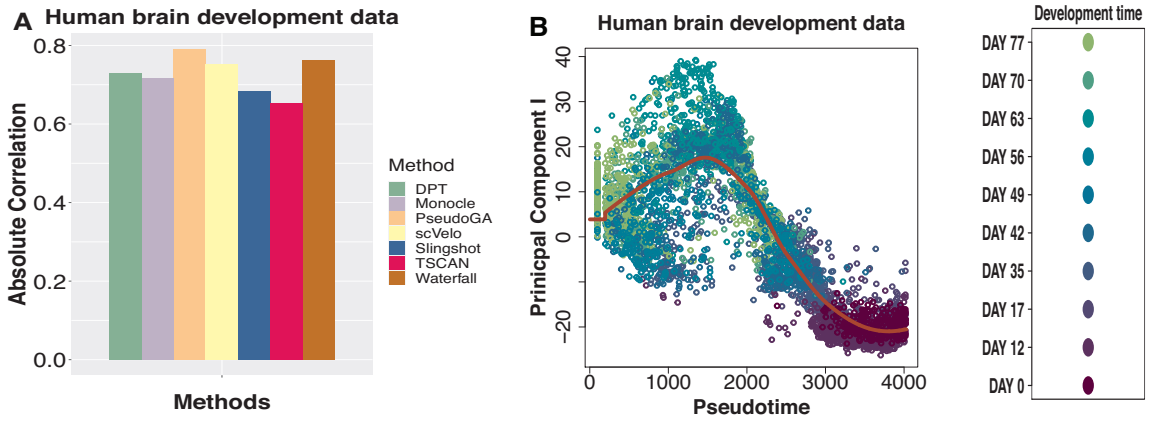


Figure 4.29: (A) Absolute Spearman's rank correlation between developmental stage and pseudotime assigned by different methods on human brain development data. PseudoGA shows the highest correlation followed by Waterfall. (B) Plot of PC I with pseudotime estimated by PseudoGA. PC I changes as quadratic polynomial with respect to PseudoGA estimate.

Cluster id	Genes with highest correlation	Common function
Enntire dataset	LIN28A, RPL23A, RPS19, NPM1, SEPT7, SEPT2	Development

Table 4.8: Top 6 genes having highest correlation with pseudotime estimated by PseudoGA and their common function in human brain development dataset.

branching in pseudotime, we have to cluster the data and apply our algorithm on each cluster which afterwards would be merged to give a consolidated structure.

4.3.2 Pseudotime using simulated data

We simulate datasets to evaluate different methods and compare their performance to PseudoGA. Simulations were performed with two different frameworks: simulation with our own simulation model and three other simulation schemes available in Bioconductor

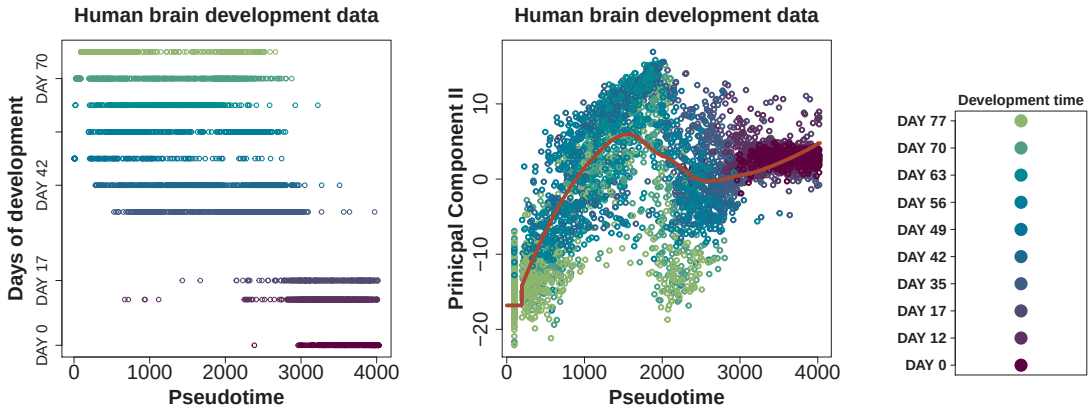


Figure 4.30: Pseudotime ordering by PseudoGA on human brain development dataset and the functional relationship between PC II and the pseudotime estimated. PC II shows cubic relationship with the estimated pseudotime.

package 'Splatter' [133]. Our simulation model can generate single-cell level expression profiles with known inherent pseudotime within a homogeneous population.

Note that all genes are not expressed in all cells. So conditional on the fact that a gene is expressed in a cell, the read count corresponding to this gene is generated from a Poisson distribution whose mean follows a Gamma distribution. The shape parameter of the Gamma distribution for each expressed gene lies on a pseudotime curve. If a gene is not expressed in a cell, we consider its read count to be identically equal to zero. We consider different gene sets where a particular trend is being followed for a set. Thus expression values within a given set may follow increasing or decreasing linear trend, quadratic or sinusoidal trend. To add more generality, we also consider few genes whose expressions are independent of pseudotime.

We know that the abundance of technical zeros or dropouts [39, 81] is a common feature in single-cell RNA-seq data. So we also add zero values for gene expressions that would naturally inflate the left tail of the Gamma-Poisson distribution. Regarding generation of dropouts, we consider three different scenarios. In the first case, we introduce lower rate of dropouts that are mainly due to smaller mean gene expression levels whereas in the second scenario, it is independent of mean expression values. In the third case, we assume relatively higher amount of dropouts that occur independently of pseudotime.

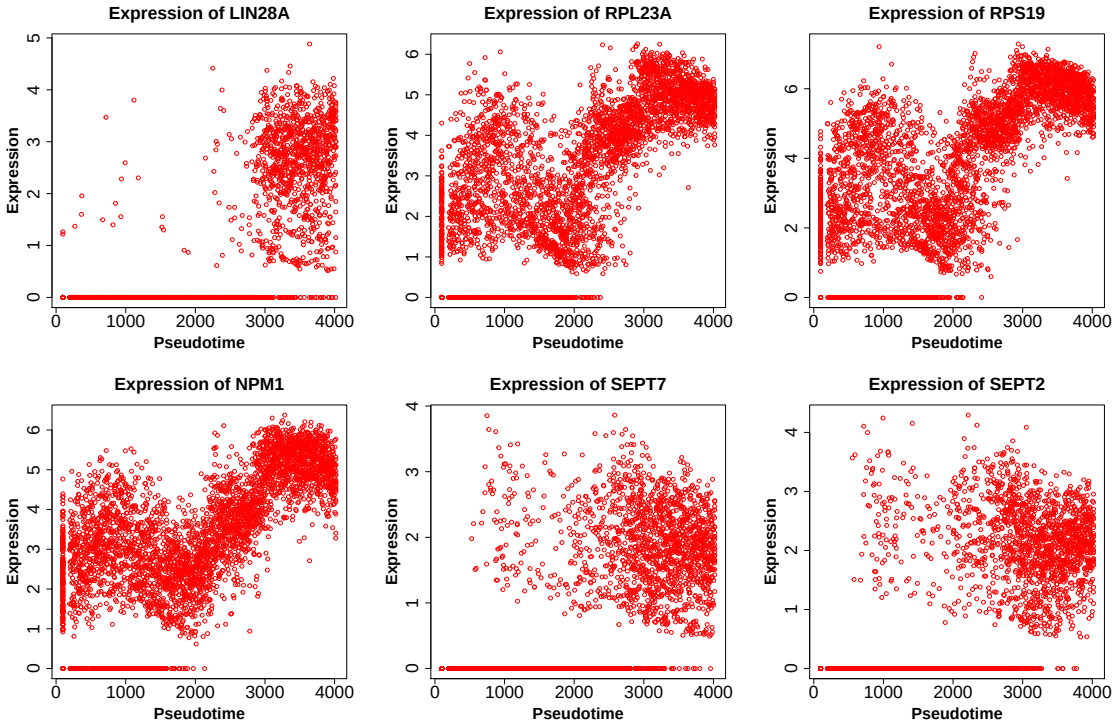


Figure 4.31: Change of expression pattern with pseudotime in human brain development dataset showing top 6 genes with highest correlation. These genes show either cubic or bursting type patterns with the estimated pseudotime.

4.3.3 Simulation model

We assume that read counts follow Gamma-Poisson distribution and we allow the parameters of the Gamma distribution to change with pseudotime. Let us assume that there are n cells with ordering $i_1, i_2, \ldots, i_n$ where $(i_1, i_2, \ldots, i_n)$ is a permutation of $(1, 2, \ldots, n)$. The following data generation model has been used for simulating read counts from a set of genes:

Step 1: We construct $\lambda_{kj} = \alpha_j + \beta_j k$ which is the shape parameter of the Gamma distribution in the k-th cell for j-th gene, α_j and β_j are gene specific parameters. For any given gene with index j, λ_{kj} changes with the position of cells on the pseudotime trajectory and exhibits linear trend along pseudotime ordering. Next, we simulate X_{kj} where $X_{kj} \sim Gamma(\lambda_{kj}, r_j)$, and λ_{kj} is the shape parameter and r_j is a gene specific scale parameter. Then, we generate Z_{kj} that indicates whether j-th gene is expressed in k-th cell or not. We assume $Z_{kj} \sim Bernoulli(\Phi(c_j + \alpha X_{kj}))$, c_j is a gene specific constant, $\alpha \geq 0$. Note that, in scenario I, the probability that a gene is expressed is an increasing function of the Gamma variable X_{kj} ($\alpha > 0$) whereas in scenarios II and III, they are independent ($\alpha = 0$). Finally, we generate $(Y_{kj}|Z_{kj} = 0) = 0$ and $(Y_{kj}|Z_{kj} = 1) \sim Poisson(X_{kj})$ where Y_{kj} denotes the read count of j-th gene in k-th cell. Let $Y_{.j} = (Y_{1j}, \ldots, Y_{nj})$.

Step 2: After Step 1, the set of expression values for any particular gene has increasing linear trend along pseduotime trajectory if β_j is taken positive. If we want j-th gene to have decreasing mean expression level with pseudotime, we modify the set of expression values for j-th gene as $Y'_{.j} = \text{reverse}(Y_{.j})$ where $\text{reverse}(x_1, x_2, \ldots, x_n) = (x_n, x_{(n-1)}, \ldots, x_1)$. To include genes with both increasing and decreasing trend, we apply $Y'_{.j} = \text{reverse}(Y_{.j})$ for some genes and keep $Y'_{.j} = Y_{.j}$ for the rest.

Step 3: To construct a gene with approximate quadratic trend with respect to pseudotime, we modify $Y'_{.j}$ from Step 2 in the following manner:

$$Y''_{.j} = Y'_{.j}[1, 3, 5, \ldots, (2n-1), (2n), (2n-2), \ldots, 2]$$

To construct a gene with approximate cubic trend or sinusoidal trend with respect to pseudotime, we modify $Y'_{.j}$ in the following manner:

$$\begin{aligned} u &= (1, 3, 5, \ldots, (2n-1)) \\ v &= (2n, (2n-2), (2n-4), \ldots, 2) \\ Y''_{.j} &= Y'_{.j}[u[(\tfrac{n}{2}+1) : n], v, u[1 : (\tfrac{n}{2})]] \end{aligned}$$

To generate genes that are not associated with pseudotime, we take $Y''_{.j}$ as a random permutation of $Y'_{.j}$.

To include variety of trends in our final gene expression matrix, we consider four types of genes: genes with quadratic trend, genes with cubic trend, genes with linear trend from Step 2 that we keep unchanged in Step 3 and genes with no association with pseudotime. In scenarios I and II, we assume relatively high values of $c_j (c_j \sim N(1, \frac{1}{4}))$ implying lower occurrence of zeros. In scenario III, relatively lower values of $c_j (c_j \sim N(0, \frac{1}{4}))$ are considered to incorporate higher incidence of dropouts.

The expression values for the j-th gene corresponding to the ordering $(i_1, i_2, ..., i_n)$, $s_{.j}$ is obtained by equating $(s_{i_1 j}, s_{i_2 j}, \ldots, s_{i_n j}) = Y''_{.j}$

Note that if a gene regulation has a prominent relation with pseudotime, it would show a trend, at least approximately. In order to capture this trend, it is expected that enough information should be available for that gene. Hence it is natural to believe that the amount of technical zeros, which is a common characteristic of single-cell data, should be relatively small.

First two scenarios are more rational when the dataset is of good quality or genes with lower dropout rates are filtered successfully before pseudotime estimation. Scenario III is relevant when the data contain too many technical zeros and in addition to that, either gene filtering cannot separate out genes with lower dropout rates or no filtering is applied.

We apply PseudoGA and other commonly available methods to these simulated datasets. Entire study is based on 100 replicates under each simulation scheme. We assess the accuracy of each method using two criteria, (1) absolute rank correlation coefficient between estimated pseudotime with the actual one, and (2) number of genes that show functional relationship with the estimated pseudotime.

We present boxplots of the two criteria for all methods under consideration. Results of our simulation study indicate that our PseudoGA shows superior performance compared to other methods for the first two scenarios while for the third its accuracy is at least as good as other methods (Figure 4.32, 4.33). Thus PseudoGA looks promising in identifying pseudotime trajectory in a variety of situations for single-cell data.

4.3.4 Simulation with Splatter

Our simulation method is very general and has very little (or no) bearing with PseudoGA. However, to see its performance in wider scenarios, we also simulate expression data using Bioconductor package 'splatter' under three different methods: PROSSTT [89], Splat [133] and PhenoPath [21].

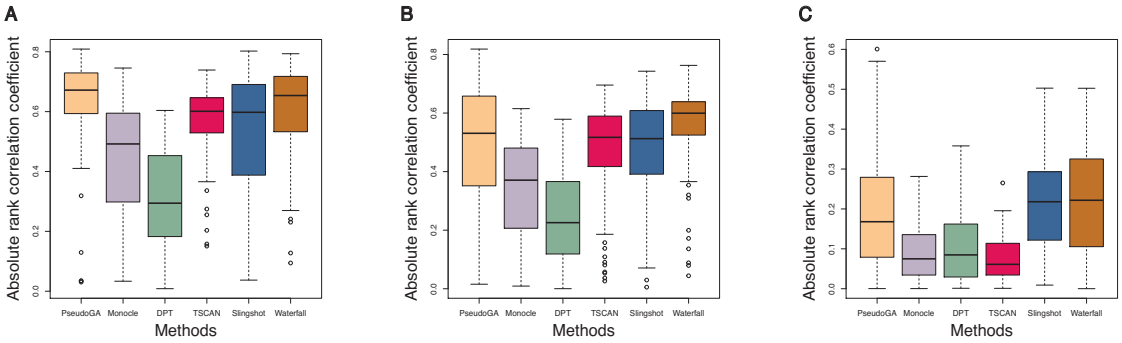


Figure 4.32: Absolute correlation coefficient with the actual pseudotime in (A) Scenario 1: lower dropout rate and dropout probability depends on mean expression level, (B) Scenario 2: lower dropout rate and dropout probability is independent of mean expression level, and (C) Scenario 3: higher dropout rate and dropout probability is independent of mean expression level. PseudoGA shows overall consistent performance across all three scenarios.

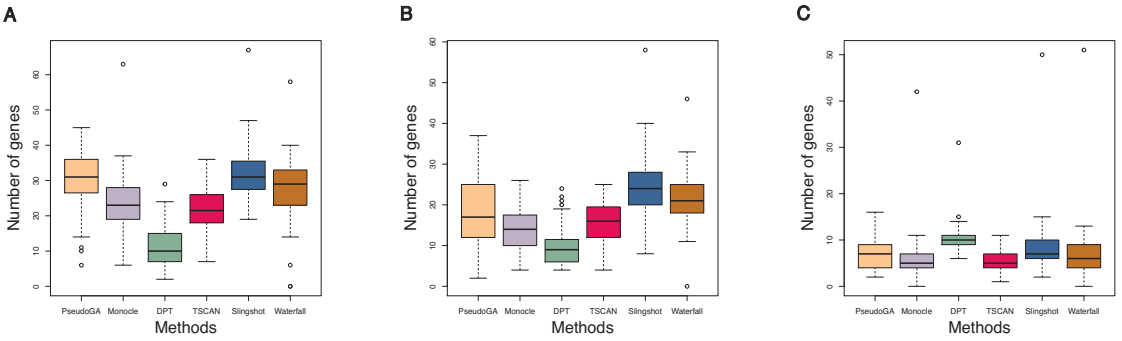


Figure 4.33: Number of genes that have functional relationship with the pseudotime estimated in (A) Scenario 1: lower dropout rate and dropout probability depends on mean expression level, (B) Scenario 2: lower dropout rate and dropout probability is independent of mean expression level, and (C) Scenario 3: higher dropout rate and dropout probability is independent of mean expression level. PseudoGA shows consistent behavior across all three scenarios.

In each dataset generated by Splat, the expression values of the gene with the highest variance were permuted randomly among the corresponding cells. This helps in assessing the performances of different methods on datasets with selected genes containing few misspecified genes or few genes that behave like outliers from the rest. Since PhenoPath simulates log-normalized expressions, anti-log transformation was applied on the data before benchmarking with different methods. Lognormal assumption generates genes with very high variance in expression values. Simulations with Splat and PROSSTT were performed with 300 cells and 100 genes.

Using the Bioconductor package Splatter [133], single-cell RNA-seq data were generated using three different methods: PROSSTT [89], Splat [133] and PhenoPath [21]. Under PROSSTT simulation method, the datasets of the specified size were generated with 100 steps keeping all other parameters as default.

Similarly, datasets with the specified number of features and cells were generated by Splat simulation method with 100 steps keeping all other parameters as default. Let i-th gene has the highest variance of expression values. The expression values of the i-th gene X_i is changed into $\sigma(X_i)$ where $\sigma(.)$ represents a random permutation of $(1, 2, \ldots, n)$, n being the number of cells.

Similarly, simulations using PhenoPath with the specified size were performed with all default parameters. PhenoPath generates expression values comparable to log normalized expressions. To convert them into actual expression values, the following transformation is applied on expression vectors X_i for the i-th gene: $Y_i = 10^{X_i - min(X_i) - 3}$. Y_i values for all 100 genes are used for estimation by different methods. The distribution of Y_i can be thought of as left truncated unimodal log-normal, which is known to be a good approximation of normalized single-cell expression data (4).

To verify whether our subsampling based approach performs comparably with other methods, we simulate 3000 cells and100 genes using PhenoPath under the third scenario.

We compare the accuracies of different methods using the same two criteria as before. Absolute rank correlation coefficient is a measure of concordance between estimated pseudotime and the actual pseudotime, whereas number of functionally related genes is an evidence based measure of concordance. The performance of PseudoGA is consistent across all three situations (Figure 4.34, 4.35). Monocle marginally outperformed PseudoGA in PROSSTT simulation. In Splat simulation, even with only one perturbed gene, accuracy of all methods except PseudoGA is downgraded due to the presence of uncorrelated genes with high variance. In PhenoPath simulation, where the distribution of gene expression differs from usual negative binomial assumption, PseudoGA turns out to be more robust than other methods. Thus, performance of PseudoGA under PhenoPath and Splat simu-

lation indicates that it maintains its accuracy and robustness in presence of outliers and highly variable genes.

We also check the performance of PseudoGA in case of a large dataset and effectiveness of the subsampling based approach. We simulate a trajectory with $10,000$ cells with expression values following Gamma-Poisson distribution and construct trajectory using PseudoGA with only 1% of the data. The remaining cells are added afterwards using nearest neighbour approach as proposed in Material and Methods Section. Based on 30 replications, the median absolute correlation between the actual pseudotime and the pseudotime generated by 100 cells was found to be 0.85 whereas the median absolute correlation with all $10,000$ cells was found to be 0.98. The comparison between the actual pseudotime and the estimated pseudotime based on one subsample is shown in Figure 4.36.

4.3.5 Scalability

Runtime of any genetic algorithm depends on population size in each generation and the number of generations. In general, increasing the values of these parameters will improve the accuracy of an algorithm and at the same time will increase the runtime. In this article, the cost function has been evaluated on 400 permutations in each generation and a minimum of 30 generations have been considered in all simulation and real data analyses. The value of ϵ was taken to be a pre-assigned small positive number. Different algorithms scale differently with number of cells and number of features [133]. PseudoGA approximately scales the same linearly. Using subsampling based approach, we have proposed a method, to tackle the increasing volume of single-cell data with large number of cells.

To assess scalability of PseudoGA, we benchmark PseudoGA runtime against runtime of other methods. We consider two types of count data generated by Splatter: one with 300 cells and 10000 features and the other with 3000 cells and 1000 features. In the second scenario, we run PsedoGA with three subsamples each of size 100 coupled with nearest neighbor matching and principal curve fitting. The boxplots of the runtimes based on 100 replicates are shown in Figure 4.37. For large number of cells, PseudoGA gains time efficiency by using subsampling approach (Figure 4.37B). Since pseudotime estimation on subsamples can be performed independently, parallelization with respect to different subsamples leads to further time efficiency of this approach. Figure 4.37 indicates that PseudoGA is time efficient both with respect to a large number of genes as well as a large number of cells.

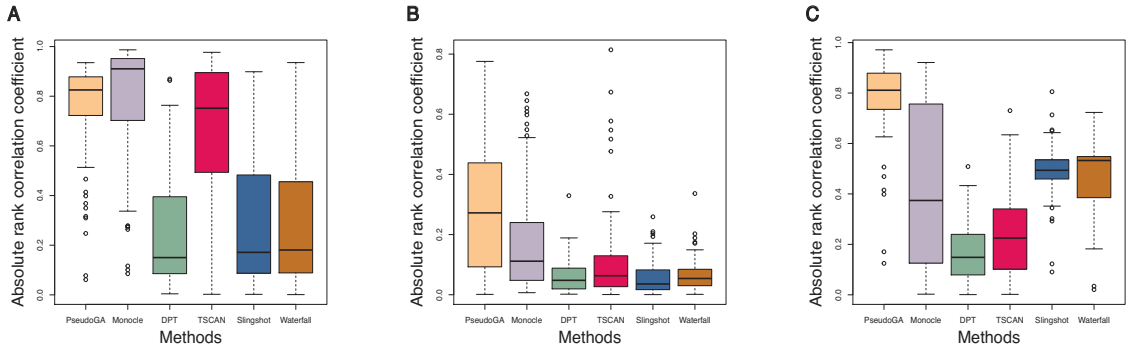


Figure 4.34: Absolute correlation coefficient with the actual pseudotime in simulations with (A) PROSSTT, (B) Splat, and (C) PhenoPath. PseudoGA performs best in (B) and (C).

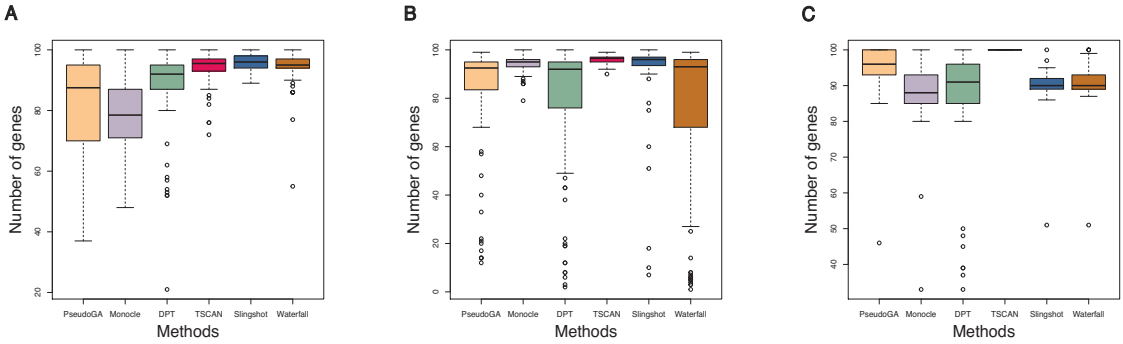
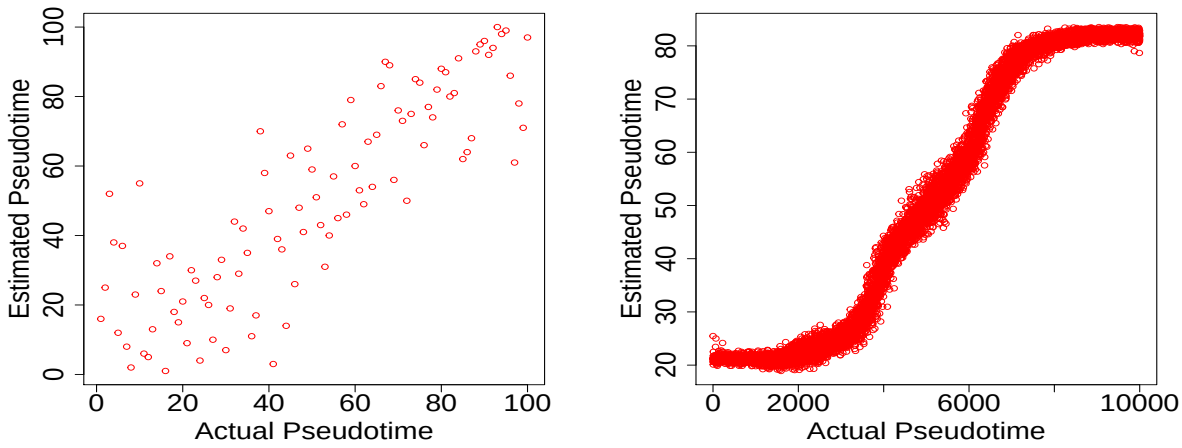


Figure 4.35: Number of genes having functional relationship with pseudotime estimated in simulations with (A) PROSSTT, (B) Splat, and (C) PhenoPath. PseudoGA performance is consistent across all three simulations.



(A) **(B)**

Figure 4.36: (A) An instance of pseudotime ordering by PseudoGA with a subset of cells and (B) Pseudotime ordering of all cells with our suggested algorithm for large data based on the same subset.

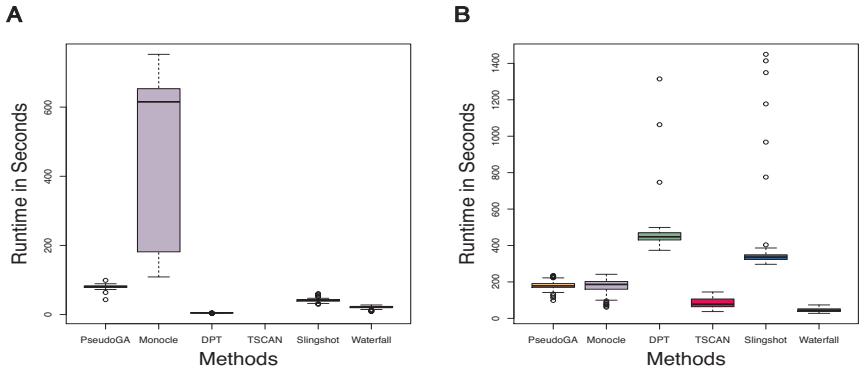


Figure 4.37: Runtime for different algorithms with (A) 300 cells and 10000 features (B) 3000 cells and 1000 features. PseudoGA runtime is comparable with other methods. Because of subsampling approach PseudoGA gains in runtime efficiency in (B).

4.4 Discussion

Several algorithms have been designed to order cells along pseudotime trajectory based on single-cell RNA-seq data. These are mainly based on the philosophy of constructing trajectory on reduced dimensional data. Some genes might show distinct types of functional pattern of expression levels along different transcriptomic stages. Hence, it may be possible that information on gene expression level might be lost to some extent, sometimes substantially, during the dimensionality reduction step. This loss of information may lead to erroneous outcome at the next step while inferring the ordering of cells. Entire pseudotime construction depends on the amount of information captured in reduced dimensionality of data. Moreover, few genes may remain approximately constant over the entire time trajectory, whereas few may be outliers. Presence of such genes might influence the pseudotime construction. We devise an algorithm 'PseudoGA' that searches for the best possible ordering of cells in the set of all permutations and infers the ordering based on actual gene expression levels.

Our proposed method PseudoGA assumes that the dependency structure of gene expression on pseudotime is based on ranks of its values. This allows our method to encompass a large class of functions that gene expression values can assume along a trajectory. To tackle with zeros, we accommodate average ranking to all cells with zero expressions for a given gene. This nonparametric assumption makes our method robust in different types of single-cell expression datasets.

We first cluster the cells in homogeneous subgroups and apply genetic algorithm on each of these homogeneous groups to increase the efficiency, followed by a novel method to concatenate paths from different clusters. This will help us in identifying any lineage or branching structure that may exist in the data with respect to pseudotime. Otherwise, we can always apply our algorithm directly to the entire dataset.

Compared to other existing methods, PseudoGA seems more robust when applied on various real datasets as well as on simulated datasets. PseudoGA has been shown to be consistent in various simulation schemes. In presence of outliers or highly variable genes, methods based on dimensionality reduction could fail but PseudoGA maintains its accuracy and robustness.

Our proposed method can be applied to a variety of datasets with even large number of cells. Our study reveals that even in such a situation, the performance of PseudoGA is extremely well both in terms of accuracy and time. We speculate that further improvements on scalability of the method are possible by implementing a more efficient genetic algorithm and parallelization of the code in case of large datasets. One can use more operators in

addition to the three operators used in PseudoGA and apply them in different manners. Improvement in the genetic algorithm would certainly improve the efficiency of PseudoGA. To the best of our knowledge, this is probably the first application of genetic algorithm in pseudotime estimation with some novel ideas and methods inbuilt in the main algorithm. PseudoGA is a freely available software implemented in R and can conveniently be applied on any single-cell expression data.

4.5 Software availability

Software for PseudoGA is freely available at http://github.com/indranillab/pseudoga .

Chapter 5: SCDI: A fast clustering-based method for Single-cell Data Integration

5.1 Introduction

Single-cell RNA-seq technology has revolutionized our knowledge on different aspects of functional genomics in areas of immune system, cancer, developmental cells etc. [32, 99, 104]. It is often necessary to integrate data from multiple sources, possibly across different studies to make combined inferences [123, 125]. Combining similar types of single-cell datasets across different sources can increase effective sample size and can lay the foundation of meta-analysis. scRNA-seq data usually contain a combination of two types of variability: technical and biological [25, 44]. Data integration might help in removing technical variability from the data to some extent. It may involve integrating data from different batches of same experiment or from different individuals. Many computational tools have been developed to analyze individual single-cell RNA-seq data but comparatively fewer methods exist to address this important issue of data integration. Integrating expression data from different sources may involve confronting with batch effects arising from different sources like choice of platform, laboratory, protocol used, quality of sample, RNA isolation procedure etc. These sources of systematic batch effects need to be adjusted appropriately before performing any analysis with sc-RNA-seq data since batch effect can be confounded with biological variation and may lead to spurious conclusion [17]. Single-cell data integration can be broadly divided into two types: integration with a single modality and integration across multiple modalities. We focus on the data integration of the former type which can also be termed as batch effect correction.

Single-cell RNA-seq data integration is challenging due to several factors. Integrating

two different datasets can also be viewed as removing confounding effects arising due to the assembling of datasets from different sources. The batch effect on individual cells for a particular experiment could be linear as well as nonlinear [120, 30]. Also, cell composition in two different datasets may vary adding complexity to integrate data from multiple sources. Some cell populations may be novel to a particular dataset and also some other cell populations may remain absent in the dataset. In any single-cell transcriptome data, cell types are usually unknown and sometimes there are rare cell types [56]. Matching cell types with data from different sources is a problem that needs to be addressed. High noise level in single-cell data and technical noise from different platforms add complexities to perform any analysis with these data.

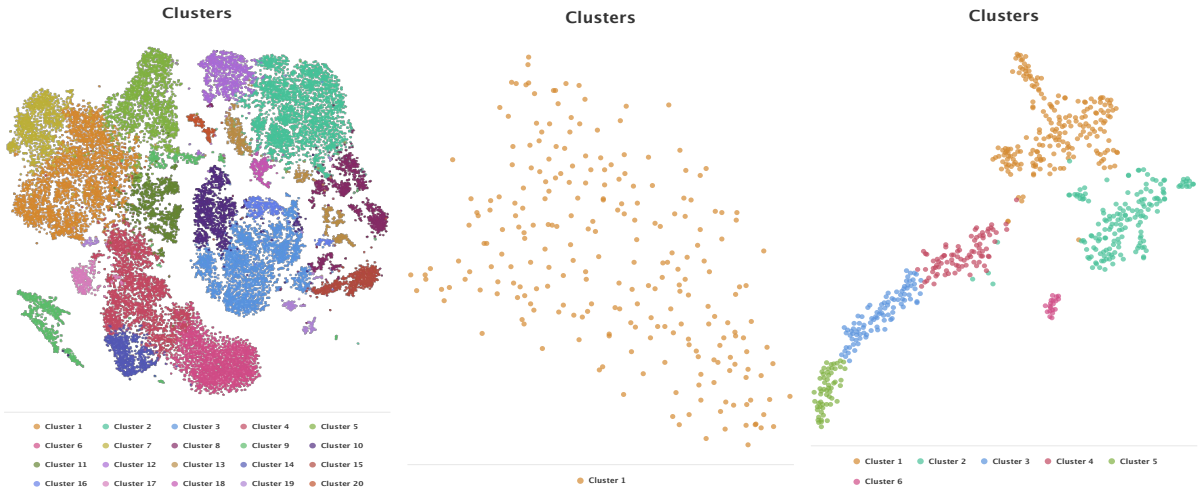
Few approaches are available to integrate sc-RNA-seq data that can be classified into three broad types. Butler et al. [15] use canonical correlation analysis to find linear combinations of expression values across datasets that are maximally correlated. It then uses dynamic time warping to align the expressions with the help of projection vectors obtained from the canonical correlation analysis. The data are then projected into low dimensional space for visualization. However, if there are non-overlapping populations in two different studies, this approach could face difficulties in alignment. A similar approach was used in Rohart et al. [98] where data from different sources were projected on common subspace so that the variation across independent studies is minimized. In Stuart et al. [111], integration from multiple modalities was considered using a similar approach. Haghverdi et al. [46] use a completely different approach by finding mutually nearest neighbors between two studies and correcting for batch effects for individual genes using batch correction vector based on mutual nearest neighbors. The batch corrected data are projected in low dimensional space for visualization. This approach assumes that batch effects are orthogonal to biological signals and the variation due to batch effects is small compared to true biological variability. Since single-cell expression data are noisy, variation due to batch effects could be significant compared to biological variation making this approach fail. This approach also assumes that batch effects are in the same direction across all mutual nearest neighbor pairs for all cell types while applying the batch correction. That could be an unrealistic assumption as well. Polański et al. [92] adopts a similar approach by constructing the network in such a way that cells of the same type are connected across batches. Johansen et al. [57] use an encoder-based neural network to map two different datasets on a common low dimensional space. Lin et al. [70] and Korsunsky et al. [61] take an alike approach by anchoring mutual nearest clusters instead of mutual nearest cells. However, prior to finding mutual nearest clusters, these methods do not attempt to remove batch effects and there might be unwanted variability present in the data while performing common cluster

membership allocation.

We propose a method to align single-cell RNA-seq data from multiple sources. This projects high throughput data in a low dimensional subspace of given dimension and also identifies common clusters across datasets. This low dimensional embedding can be used to perform downstream analyses like finding batch-specific clusters, pseudotime estimation, spatial mapping, etc. Also the common cluster id can be used to perform differential expression analysis. Our approach is similar to Korsunsky et al. [61] but instead of using soft clusters, our method performs hard clustering based on reduced dimensional data adjusted for batch effects. Compared to finding mutual nearest neighbor cells, this approach helps in removing memory usage and time complexity considerably. We propose a modification of Gaussian Process Latent Variable Modeling (GPLVM) [66] that takes care of batch effects arising from two different datasets. This modification takes care of both linear as well as nonlinear batch effects that can act on datasets from different sources. To tackle with large number of cells, where GPLVM can run slow, we propose to apply GPLVM on a subsample of the original data and mapping the rest of the data to a lower-dimensional subspace based on nearest neighbor regression. We advocate the idea that data integration procedure should depend on the purpose of integration. Hence, once we get two batch-corrected datasets, we propose three distinct algorithms for data integration for the purpose of three different downstream analyses. Each algorithm is tailored to the specific objective of downstream analysis.

5.2 Single Cell Data Integration (SCDI) Method

Single-cell expression datasets can be broadly divided into three categories (Figure 5.1). In some data, cells can be classified into distinct clusters. In the second kind of data, cells do not appear in clusters but in a continuum with a single cluster. This type of dataset can also be viewed to possess a single cluster with a continuum. Some datasets could be intermediate between these two with both clusters and continuum of cells. There can be different perspectives behind the integration of two datasets. Sometimes the purpose is to identify common cell types across clusters but the ordering of cells is not of primal importance. In some other datasets, identifying the joint continuum is the main focus. Based on the objective of the integration and the data type, a customized approach for integration might turn out to be more accurate and precise. We propose a novel Single Cell Data Integration (SCDI) method that integrates two datasets after correcting for batch effects. The workflow of SCDI, when the purpose is to find joint clustering or the



(A) **(B)** **(C)**

Figure 5.1: Different types of single-cell transcriptome data: (A) Data with clusters, (B) Data with continuum, (C) Data with a mixture of both continuum and cluster (Source: EMBL-EBI Single Cell Expression Atlas)

data are of clustered type, is based on two steps. First, given individual clustering of two independent single-cell transcriptome data, it constructs a joint clustering of these two data and assigns common membership across datasets. Next, it performs dimension reduction of the combined data considering the joint clustering in the first step. In the first step, SCDI takes individual cluster membership as input. Though there are many standard methods available to perform clustering on a single scRNA-seq data, a user verified clustering is highly recommended to ensure the desired accuracy of SCDI.

After the cluster ids and the raw data are given as input, SCDI applies Batch Corrected GPLVM (BC-GPLVM) on the combined data. Our proposal of BC-GPLVM, which is a modification of usual GPLVM, considers batch-effect correction factor while performing dimensionality reduction. SCDI calculates between cluster distances on reduced dimensions obtained from BC-GPLVM across datasets. If the distance between two clusters is significantly lower than other inter-cluster distances, those two clusters are assumed to be identical. In this step, some clusters in individual datasets may remain unmatched with any other cluster.

After assigning common cluster id across datasets, the next step is to project the data into lower dimensional embedding of any given dimension that can be used for downstream analyses based on combined data. We apply another modification of GPLVM called Clustered GPLVM on all datasets to come up with projection on a lower-dimensional space of a given dimension. To make the approach scalable for large datasets, a random subset of the original data is chosen and the dimension reduction is applied on the subset instead of the whole data. The remaining data outside the subset are projected on the lower dimensional space, based on the projection of the subset and nearest neighbor regression.

When the integration is required for the purpose of pseudotime ordering or the data consist of a single cluster, all cells are assumed to belong to a single population instead of multiple groups. When the data are of mixed type or the purpose of the data integration is not specified, any of these two approaches can be adopted. In the case of differential expression, the data need not be projected into lower dimensional space and the whole data need to be adjusted for batch effects. However, we recommend tackling the problem of batch correction and hence data integration, keeping in mind the next downstream analysis.

SCDI takes two single-cell transcriptomic datasets along with clustering of each of them as input and produces a joint clustering and a reduced dimensional representation of the combined data based on GPLVM. As an optional output, SCDI produces batch corrected data corresponding to all genes based on the cluster-specific linear batch correction. Alternatively, it takes the number of clusters from two datasets as input and produces clustering for individual datasets before applying SCDI. However, it is recommended that individual clustering for each of the datasets is provided as inputs for better accuracy of SCDI. The reduced dimensional representation may be used for data visualization, clustering, pseudotemporal ordering, spatial mapping, etc and the dimension of the output is to be defined by the user. The batch corrected complete dataset can be used for differential expression analysis, marker gene identification, and for many other downstream analyses. SCDI can be easily applied on multiple datasets but for simplicity, we describe it for two datasets only. The generalization of the algorithm to multiple datasets is straightforward. We first perform data integration based on two datasets and then keep on adding the remaining datasets sequentially. This approach has the drawback that the final output depends on the ordering in which the datasets are added. Another approach is to find joint cluster membership based on all datasets with GPLVM applied to the combined data. In that case, a separate batch effect correction factor for all pairs of batches must be used.

5.2.1 Dimensionality reduction

We use two variations of GPLVM together to visualize the combined data in a common frame. First, we propose Batch Corrected GPLVM (BC-GPLVM) that performs GPLVM with a batch correction factor given through the kernel function. Next, we propose another variation called Clustered GPLVM that performs GPLVM with a batch correction factor as well wise cluster-specific correction factor for each cluster. Let us assume that a scRNA-seq dataset contains N cells and D genes. We describe these algorithms along with standard GPLVM below.

Usual GPLVM

In probabilistic principal component analysis (PPCA)[116] setup, given a set of D dimensional variables $\{y_i\}_{i=1}^N$ and a K dimensional latent variable x_i associated with each observation $\{y_i\}$, the model assumption is $y_i = Wx_i + \epsilon_i$ where W is a matrix of coefficients of order $D \times K$. The likelihood of an individual observation can be written as:

$$p(y_i|W, \beta) = \int p(y_i|x_i, W, \beta)p(x_i)dx_i$$

where $p(x_i) = N_K(x_i|0, I)$, and $p(y_i|W, x_i, \beta) = N_D(y_i|Wx_i, \beta^{-2}I)$.
Unlike the usual principal component analysis (PCA), the latent variable x_is here are assumed to be random. If we further assume that x_is are independent, the marginal distribution of y_is are: $y_i \sim N_D(0, WW^t + \beta^{-2}I)$. This can also be viewed as a special case of factor analysis with isotropic variance covariance matrix for ϵ_i.

In Gaussian Process Latent Variable Modeling (GPLVM), we take the reverse approach by assuming x_i's fixed and w_i's random. So here we have,

$$p(y_i|x_i, \beta) = \int p(y_i|x_i, W, \beta)p(W)dW$$

By specifying a prior distribution, $p(w_i) = N_K(w_i|0, \alpha I)$, we obtain a marginalized likelihood for y_i as,

$$p(y_i|x_i, \alpha, \beta) = N_D(0, \alpha x_i^t x_i + \beta^{-2}I)$$

Now, define $X = \begin{bmatrix} x_1^t \\ x_2^t \\ \vdots \\ x_N^t \end{bmatrix}$, and $Y = \begin{bmatrix} y_1^t \\ y_2^t \\ \vdots \\ y_N^t \end{bmatrix}$.

So we have,

$$p(Y|X,\alpha,\beta) = \int (\prod_{i=1}^{N} p(y_i|x_i,W,\beta))p(W)dW = \frac{1}{(2\pi)^{\frac{DN}{2}}} \exp(-\frac{1}{2}tr(K^{-1}YY^t))$$

where $K = \alpha XX^t + \beta^{-2}I$. For each of the coordinates of Y, the variance-covariance matrix for the joint distribution of all observations Y is a constant multiple of inner product of the matrix X with some diagonal elements added. It is customary to assume prior on x_n and usually x_n's are assumed to follow $N(0,I)$ distribution though it can be ignored as well. If $N(0,I)$ prior is assumed for all $x_{i,j}$s, the combined log-likelihood becomes,

$$L = -\frac{DN}{2}\log(2\pi) - \frac{D}{2}\log(|K|) - \frac{1}{2}tr(K^{-1}YY^t) - \frac{Nq}{2}\log(2\pi) - \frac{1}{2}\sum_i \sum_j x_{i,j}^2$$

This is the simplest form of GPLVM where the kernel matrix K is of the $\alpha XX^t + \beta^{-2}I$, i.e. the covariance matrix is induced by a linear kernel $ker_{n,m} = \alpha x_n^t x_m$. A natural extension to this simplest form can be obtained by introducing a nonlinear kernel $ker_{n,m} = \alpha(x_n - x_m)^t(x_n - x_m)$.

In usual GPLVM, we consider a kernel of the form:

$$k_{n,m} = \alpha \exp(-\tfrac{\gamma}{2}(x_n - x_m)^t(x_n - x_m)) + \delta_{nm}\beta^{-2}$$

where $k_{n,m}$ is the (n,m)-th element of K.
To optimize with respect to X, we use gradient descent algorithm as follows.

$$\frac{\partial L}{\partial x_{ij}} = \sum_m \sum_n \frac{\partial L}{\partial k_{n,m}} \frac{\partial k_{n,m}}{\partial x_{ij}} \text{ with } \frac{\partial L}{\partial K} = K^{-1}YY^tK^{-1} - DK^{-1}.$$

and

$$\begin{aligned}
\frac{\partial k_{n,m}}{\partial x_{i,j}} &= 0 \text{ if } m = n \\
&= k_{n,m}(-\gamma(x_{i,j} - x_{m,j})) \text{ if } n = i, n \neq m \\
&= k_{n,m}(-\gamma(x_{i,j} - x_{n,j})) \text{ if } m = i, m \neq n \\
&= 0 \text{ otherwise}
\end{aligned}$$

Similarly, $\frac{\partial L}{\partial \alpha} = \sum_m \sum_n \frac{\partial L}{\partial k_{n,m}} \frac{\partial k_{n,m}}{\partial \alpha}$, $\frac{\partial L}{\partial \beta} = \sum_m \sum_n \frac{\partial L}{\partial k_{n,m}} \frac{\partial k_{n,m}}{\partial \beta}$ and $\frac{\partial L}{\partial \gamma} = \sum_m \sum_n \frac{\partial L}{\partial k_{n,m}} \frac{\partial k_{n,m}}{\partial \gamma}$
where,
$\frac{\partial k_{n,m}}{\partial \alpha} = \exp(-\frac{\gamma}{2}(x_n - x_m)^t(x_n - x_m))$, $\frac{\partial k_{n,m}}{\partial \beta} = -2\delta_{nm}\beta^{-3}$
$\frac{\partial k_{n,m}}{\partial \gamma} = \alpha \exp(-\frac{\gamma}{2}(x_n - x_m)^t(x_n - x_m))(-\frac{1}{2}(x_n - x_m)^t(x_n - x_m))$
$\frac{\partial L}{\partial x_{ij}} = \sum_m \sum_n \frac{\partial L}{\partial k_{n,m}} \frac{\partial k_{n,m}}{\partial x_{ij}} - x_{i,j}$

The parameters can be estimated using gradient descent algorithm with the partial derivatives described above. The initial values of x_{ij}s are taken as the first q principal components of Y, if the desired dimension of the latent space is q.

Batch Corrected GPLVM

Let us assume that the data are collected in two different batches, with the observations denoted by Y_1 and Y_2 respectively and the latent variables denoted by X_1 and X_2 respectively. The original observations are D dimensional whereas the latent space is q dimensional with $D >> q$. If the batch effect acting in these two groups is completely linear, it can easily be removed by adjusting the mean in Y_1 and Y_2. For high dimensional variables, batch effect need not be linear for all coordinates, so it is reasonable to assume that batch effect is nonlinear and hence mean adjustment may not work.

To reduce distance between latent variables from these two groups, we consider the following kernel:

$$k_{n,m} = \alpha \exp(-\frac{\gamma}{2}(x_n - x_m)^t(x_n - x_m)) + \delta_{nm}\beta^{-2} - \sigma_1^2(1 - \delta_{g(m)g(n)})$$

where $g(k)$ is the group the k-th observation belong to and σ_1^2 is the parameter taking care of batch effects.

Here also, we estimate the parameters using gradient descent algorithm exactly like usual GPLVM except for the fact that there is an extra parameter σ_1^2 here to optimize for.

Here, $\frac{\partial L}{\partial \sigma_1} = \sum_m \sum_n \frac{\partial L}{\partial k_{n,m}} \frac{\partial k_{n,m}}{\partial \sigma_1}$ with $\frac{\partial k_{n,m}}{\partial \sigma_1} = -2\sigma_1(1 - \delta_{g(m)g(n)})$.

Clustered GPLVM

Once the common cluster membership across datasets has been found, we can use it to remove cluster-wise batch effects which can be possibly nonlinear. We include a cluster-specific factor for batch effect correction in the usual GPLVM algorithm. To do this, we assume different priors for observations coming from different clusters. Observations coming

from different batches but belonging to the same cluster are assumed to have identical prior distribution. Since the prior distribution assumed is normal, it is enough to assume that the parameters μ and Σ are same for all observations within clusters across batches but different for observations from different clusters.

Let $c(i)$ denote the cluster id for observation i and $g(i)$ denote the group the observation i belongs to. Define $\mu_k = Mean(x_{\{i:c(i)=k\}})$ and $\Sigma_k = Var(x_{\{i:c(i)=k\}})$ which can be described as mean and variance of the latent variables for observations belonging to cluster k. Also, define $\mu_{k,l} = Mean(x_{\{i:c(i)=k,g(i)=l\}})$ and $\Sigma_{k,l} = Var(x_{\{i:c(i)=k,g(i)=l\}})$.
First, we make mean and variance adjustment on latent variables for each of the observations to take care of linear batch effect across batches:

$$x_i' = \Sigma_k^{\frac{1}{2}} \Sigma_{k,l}^{-\frac{1}{2}} (x_i - \mu_{k,l}) + \mu_k$$

We apply GPLVM with X' as the initial value of the latent variable and take μ_k and Σ_k as mean and variance for the prior normal distribution for k-th cluster. Here, the log-likelihood function is,

$$\begin{aligned} L = &- \frac{DN}{2}\log(2\pi) - \frac{N}{2}\log(|K|) - \frac{1}{2}tr(K^{-1}YY^t) \\ &- \sum_k \sum_{i:c(i)=k} [-\frac{q}{2}\log(2\pi) - \frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x_i - \mu_k)^t \Sigma_k^{-\frac{1}{2}} (x_i - \mu_k)]. \end{aligned}$$

In addition to the parameters in batch corrected GPLVM we have additional parameters: μ_k and Σ_k for $k = 1, 2, \ldots, K$ with K as the total number of clusters. Here also, we use gradient descent algorithm to estimate the parameters.

Here, $\frac{\partial L}{\partial x_{ij}} = \sum_m \sum_n \frac{\partial L}{\partial k_{n,m}} \frac{\partial k_{n,m}}{\partial x_{ij}} - \Sigma_k^{-1}(x_{i,j} - \mu_k^j)$
where μ_k^j is the j-th coordinate of μ_k and $c(i) = k$. At the n-th iteration,

$$\hat{\mu}_k^{(n)} = Mean(x_{i:c(i)=n}^{(n)}) \text{ and } \hat{\Sigma}_k^{(n)} = Var(x_{i:c(i)=n}^{(n)}).$$

5.2.2 Joint clustering

If cluster ids for individual datasets are not provided, the algorithm first constructs two separate clusterings for the two datasets. There are many standard algorithms available in the literature that can perform clustering on single-cell RNA-seq data. SC3 [59], CIDR [71], TSCAN [55] are some of the methods available. However, we have observed that K-means clustering or graph clustering applied on reduced dimensions obtained from tSNE

works reasonably well to cluster single-cell transcriptome data. The user can experiment with different methods to arrive at the best possible clustering, based on human judgment with tSNE plot.

Joining cluster centers across data

Once clustering is performed in each of the datasets, similar clusters across datasets are anchored based on inter-cluster distances from the reduced dimensional data obtained from BC-GPLVM. There are many ways to define distance measure between two clusters, though all results in this work are based on average distance between all pairs of points considered from the two clusters. However, one can consider other distance measures as well like distance between cluster centers. Assume for simplicity that there are K_1 clusters in dataset 1 and K_2 clusters in dataset 2 with $K_1 \geq K_2$. Let $\mathcal{D} = \{d_{ij} : 1 \leq i \leq K_1, 1 \leq j \leq K_2\}$ be the set of all pairwise distance between clusters for these two datasets. Let $d_{i\bullet}$ be the vector $(d_{ij})_{1 \leq j \leq K_2}$ and similarly $d_{\bullet j} = (d_{ij})_{1 \leq i \leq K_1}$. Also define $d_{i(-j)} = \{d_{ik} : 1 \leq k \leq K_2, k \neq j\}$. Then cluster x from dataset 1 and cluster y from dataset 2 are joined if either of these two conditions holds:
(1): $d_{xy} = \min(d_{x\bullet})$ and $d_{xy} = min(d_{\bullet y})$.
(2): $d_{xy} = \min(d_{x\bullet})$ and $\frac{(d_{xy} - \text{mean}(d_{x(-y)}))}{\text{sd}(d_{x(-y)})} \geq \lambda$
where λ is a tuning parameter. The parameter λ determines the flexibility with which two clusters from different datasets can be joined. Low value of λ implies more number of matching as well as higher probability of mismatch whereas higher values of λ implies more strictness in joining clusters across datasets but lower chance of mismatch. If cluster j from dataset 2 aligns with two different clusters in dataset 1, namely i_1 and i_2, the final alignment of j is i_1 if $d_{i_1 j} < d_{i_2 j}$ and the alignment is i_2 otherwise. It is possible that some cluster in one dataset does not correspond to any cluster in the other dataset indicating that the cluster is unique to that dataset.

The joining of clusters across two conditions is similar to cell type projection to a reference transcriptome. Similar approach was used to address different problems [101, 69, 130, 4]

Assigning cluster ids to individual cells

After anchoring similar clusters across datasets, SCDI assigns identical cluster id to cells appearing in matched clusters across datasets. All cells appearing in unmatched clusters across datasets are provided unique cluster id specific to the cluster as well as that dataset.

After the assignment of cells to individual clusters, the joint cluster ids are used in the subsequent steps of dimension reduction.

5.2.3 SCDI workflow for common cell type identification

1. Start with D dimensional data from two different sources or batches: call them X_1 and X_2;
2. Apply Batch Corrected GPLVM on the combined data $X = (X_1, X_2)$ and call the q dimensional output as $Z = (Z_1, Z_2)$;
3. Perform joint clustering on Z_1 and Z_2;
4. Perform cluster-specific batch effect correction on Z by adjusting for mean and variance on each of the individual clusters. Call the new q dimensional output Z';
5. Apply Clustered GPLVM on X with Z' as the initial value for the latent variable and the cluster information obtained from step 2. The q dimensional output Z'' in this step can be considered as the batch effect corrected output projected in a lower dimension. tSNE can further be applied on Z'' for better visualization of the data.

5.2.4 SCDI workflow for combined differential expression

1. Start with D dimensional data from two different sources or batches: call them X_1 and X_2;
2. Apply Batch Corrected GPLVM on the combined data $X = (X_1, X_2)$ and call the q dimensional output as $Z = (Z_1, Z_2)$;
3. Perform joint clustering on Z_1 and Z_2;
4. Perform cluster-specific batch effect correction on X by adjusting for gene-wise mean and variance on each of the individual clusters;
5. Perform differential expression analysis based on X.

5.2.5 SCDI workflow for combined pseudotime analysis

1. Start with D dimensional data from two different sources or batches: call them X_1 and X_2;
2. Apply Batch Corrected GPLVM on the combined data $X = (X_1, X_2)$ and call the q dimensional output as $Z = (Z_1, Z_2)$;
3. Apply Clustered GPLVM on X with Z as the initial value for the latent variable and assuming all observations belong to a single cluster. The q dimensional output Z' in this step can be considered as the batch effect corrected output projected in lower dimension;
4. Apply pseudotime estimation algorithm on Z'.

5.2.6 Batch correction with large number of cells

Since GPLVM involves optimization with respect to a large number of parameters, it becomes computationally costly as the number of cells grows. So we choose random subsamples X_1^S and X_2^S from X_1 and X_2 respectively and apply SCDI on $X^S = (X_1^S, X_2^S)$. Let X_1^C and X_2^C denote the part of the dataset that are not included in X^S and denote $X^C = (X_1^C, X_2^C)$. Clustering is performed on the undivided data X_1 and X_2 separately. The cluster ids from the original data are used as inputs implying if y_1 and y_2 are cluster ids of the two original datasets, y_1^S and y_2^S are cluster ids of the two subsets. After applying SCDI, let $Z^S = (Z_1^S, Z_2^S)$ be the output with reduced dimension after the integration.

For observations belonging to the set C, the reduced dimensions are estimated by the randomized nearest neighbor estimator $\hat{z}_i = \hat{f}(x_i) + \hat{\epsilon}_i$ where f is the unknown function mapping the original data X to the reduced dimension Z.

$f(x_i)$ is estimated by $\hat{f}(x_i) = \frac{1}{K} \sum_{j \in N_K(x_i)} z_j$ where $N_K(x)$ is the set of observations in S belonging to the K nearest neighborhood of x_i within the same cluster of x_i. $\hat{\epsilon}_i$ is generated randomly from $N(0, \Sigma)$ distribution where $\hat{\Sigma} = \frac{1}{(K-1)} \sum_{j \in N_K(x)} (z_j - \bar{z})(z_j - \bar{z})^t$

5.3 Performance on simulated datasets

We generate single-cell RNA-seq data using Bioconductor package splatter [133]. Simulations are performed under two different broadly classified categories. In the first type of simulation, batch effects are kept uniform across clusters. In each of the four different

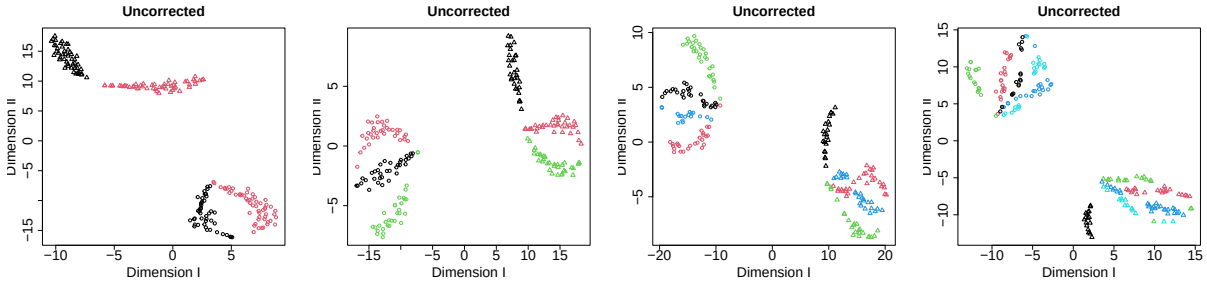
scenarios, two datasets are generated containing 2, 3, 4 and 5 clusters with uniform batch effect across clusters. Cells from the same population falling in different datasets always belong to separate clusters in tSNE plot [76, 72] with the combined data, when tSNE is applied on uncorrected data. Subsequently, SCDI, SMNN and Seurat are applied on these two datasets for batch correction (Figure 5.2). In the second setup, batch effects are varied across clusters indicating interaction between batch effect and clusters (Figure 5.3). We use tSNE to visualize the batch corrected outputs from different methods and measure the mixing accuracy by transfer entropy between two datasets. Barnett and Bossomaier [9] described that log likelihood ratio statistic can be used as transfer entropy for joint distribution of two variables. We compared this statistic for goodness in mixing for different methods.

In both setups, SCDI outperforms the other two methods. SMNN fails to identify cells from the same cluster across datasets in both scenarios. Seurat, though correctly classifies cells in the first type of simulation, fails to do so in the second simulation. SCDI has lowest entropy among all methods under consideration suggesting better performance than others.

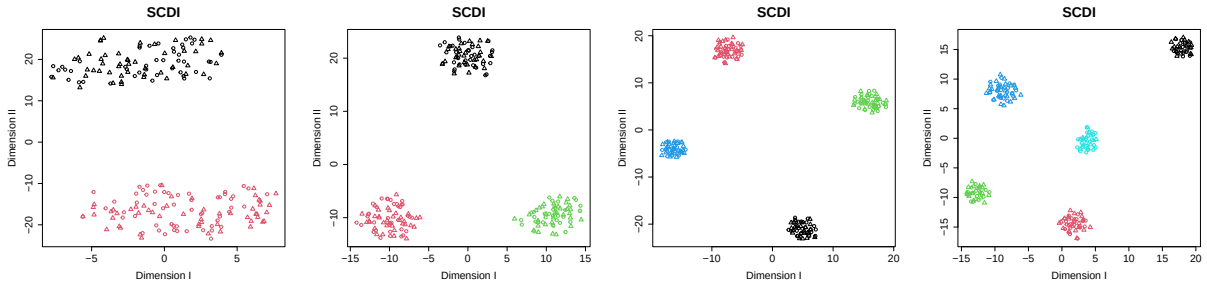
5.4 Performance on real datasets

5.4.1 Integration of pancreatic cells datasets across individuals

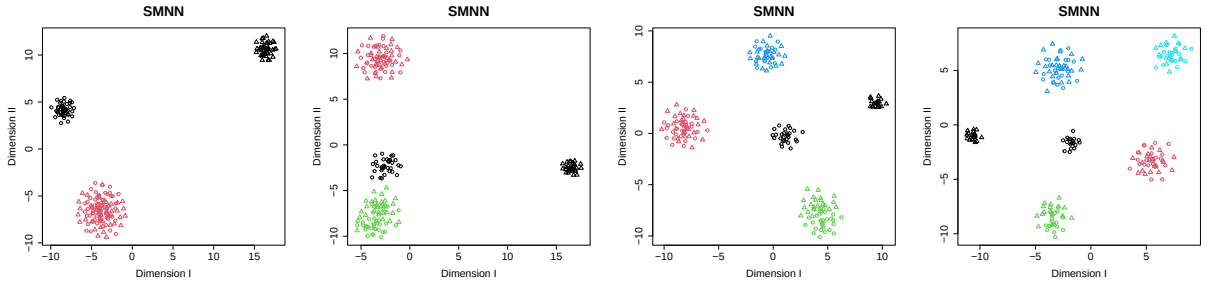
Single-cell RNA-seq data were collected from several human donors to create a comprehensive atlas of human pancreatic cells [8]. There are many studies for bulk level expression analysis for pancreatic cells but cellular level expression profile may help in better understanding of the mechanisms behind functions of pancreas and pancreas related diseases. Integrating data from multiple individuals is necessary for increasing the effective sample size of cells and the creation of exclusive atlas for all cell types. tSNE plot of the combined data assigns cells into wrong clusters if the batch effect is not corrected. We apply all three methods under consideration on these two datasets for the purpose of benchmarking. For visualization purposes, we apply tSNE on the outputs obtained from different methods. Cell types were identified based on other markers. To see the true performance of different methods including ours, we calculate Rand Index between the output cluster id and the cell types. SCDI seems to outperform the other two as revealed from the visualization in lower dimension as well as in terms of Rand index value (Figure 5.4).


Uncorrected
Uncorrected
Uncorrected
Uncorrected
Dimension I
Dimension II

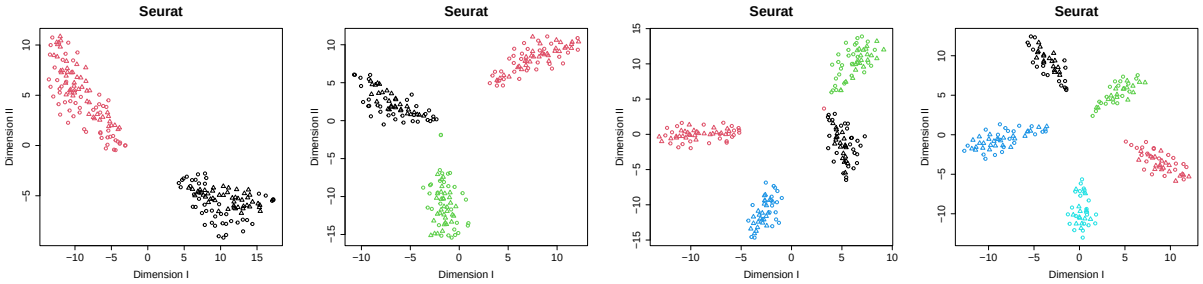
(A)


SCDI
SCDI
SCDI
SCDI
Dimension I
Dimension II

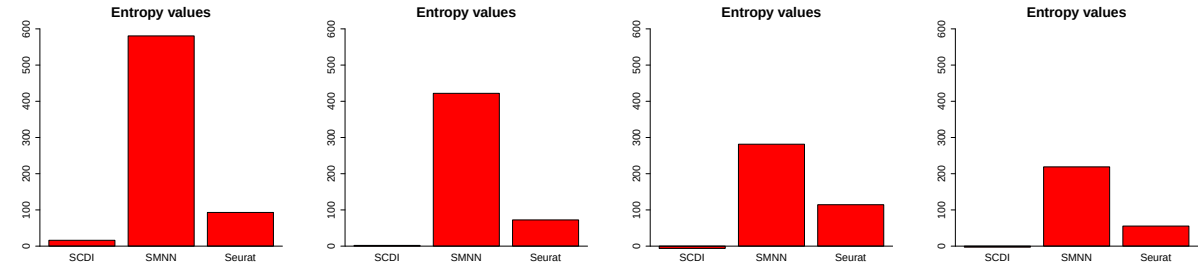
(B)


SMNN
SMNN
SMNN
SMNN
Dimension I
Dimension II

(C)

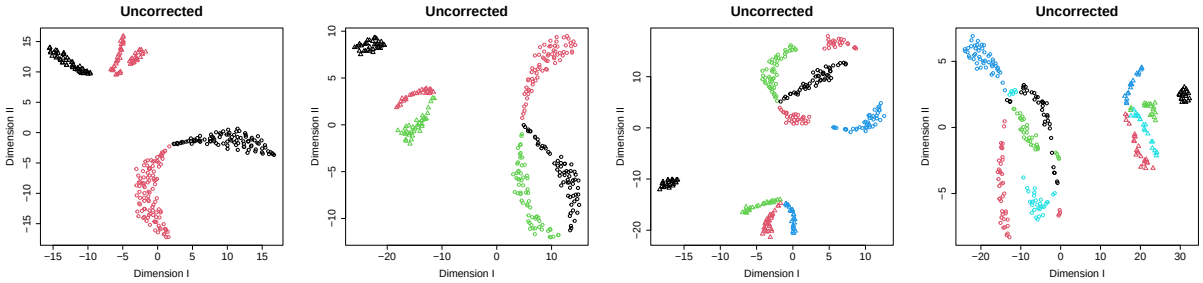


(D)

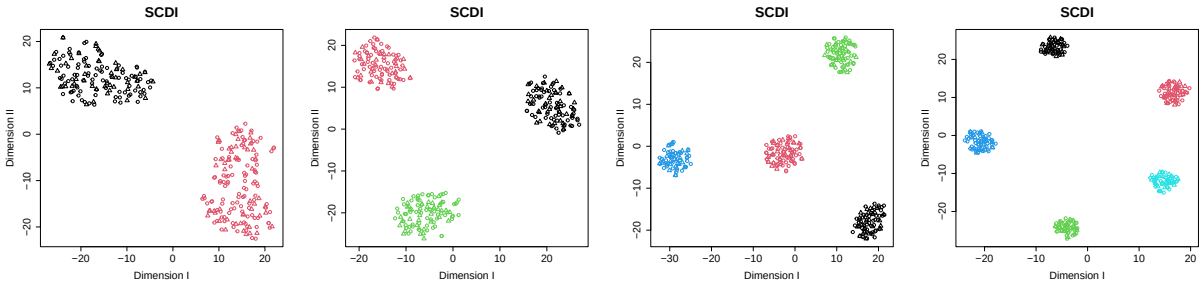


(E)

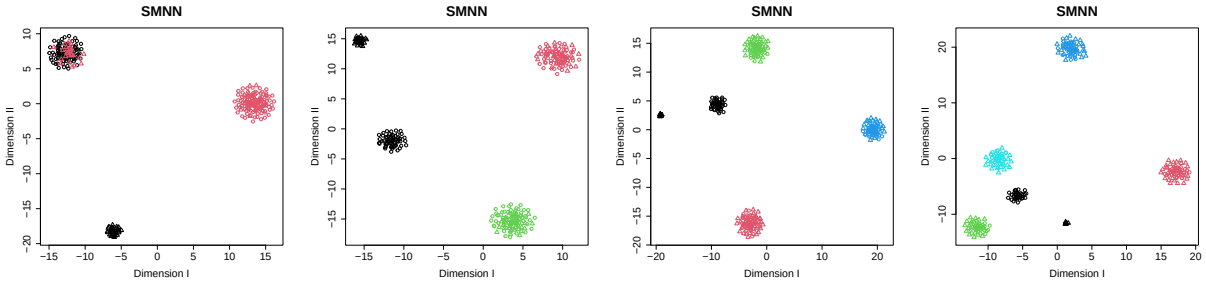
Figure 5.2: Visualization of data integration from different methods when batch effects is uniform: (A) Uncorrected data, (B) SCDI, (C) SMNN, (D) Seurat. Dimensionality reduction was performed using tSNE. (E) Comparison of transfer entropy in four different scenarios.


Uncorrected
Uncorrected
Uncorrected
Uncorrected
Dimension I
Dimension II

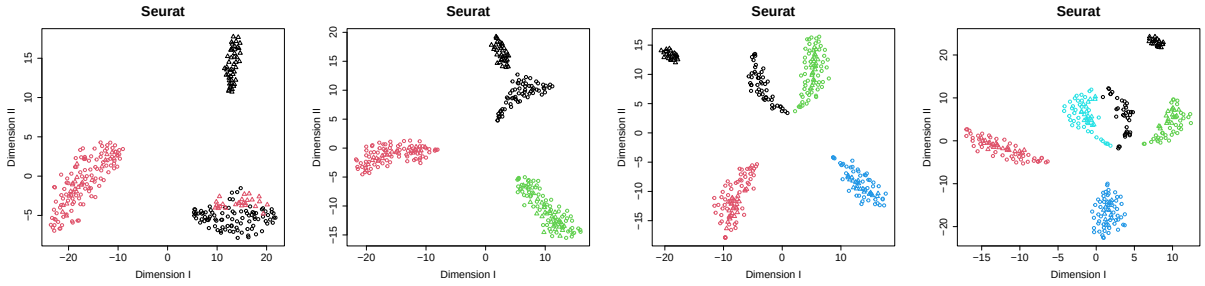
(A)


SCDI
SCDI
SCDI
SCDI
Dimension I
Dimension II

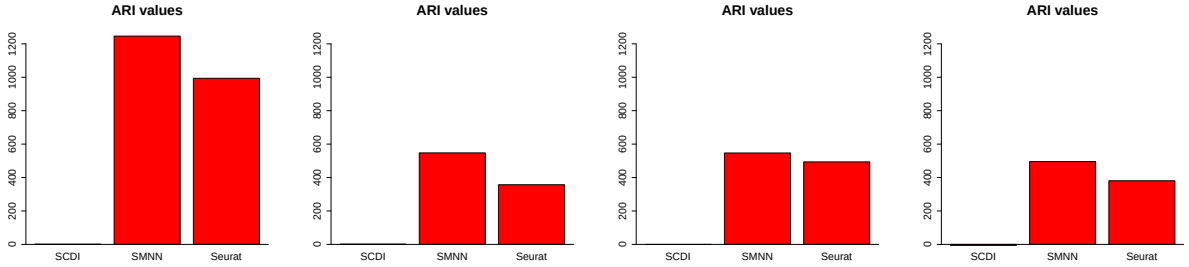
(B)


SMNN
SMNN
SMNN
SMNN
Dimension I
Dimension II

(C)



(D)



(E)

Figure 5.3: Visualization of data integration from different methods when batch effects is non-uniform: (A) Uncorrected data, (B) SCDI, (C) SMNN, (D) Seurat. (E) Comparison of transfer entropy in four different scenarios.

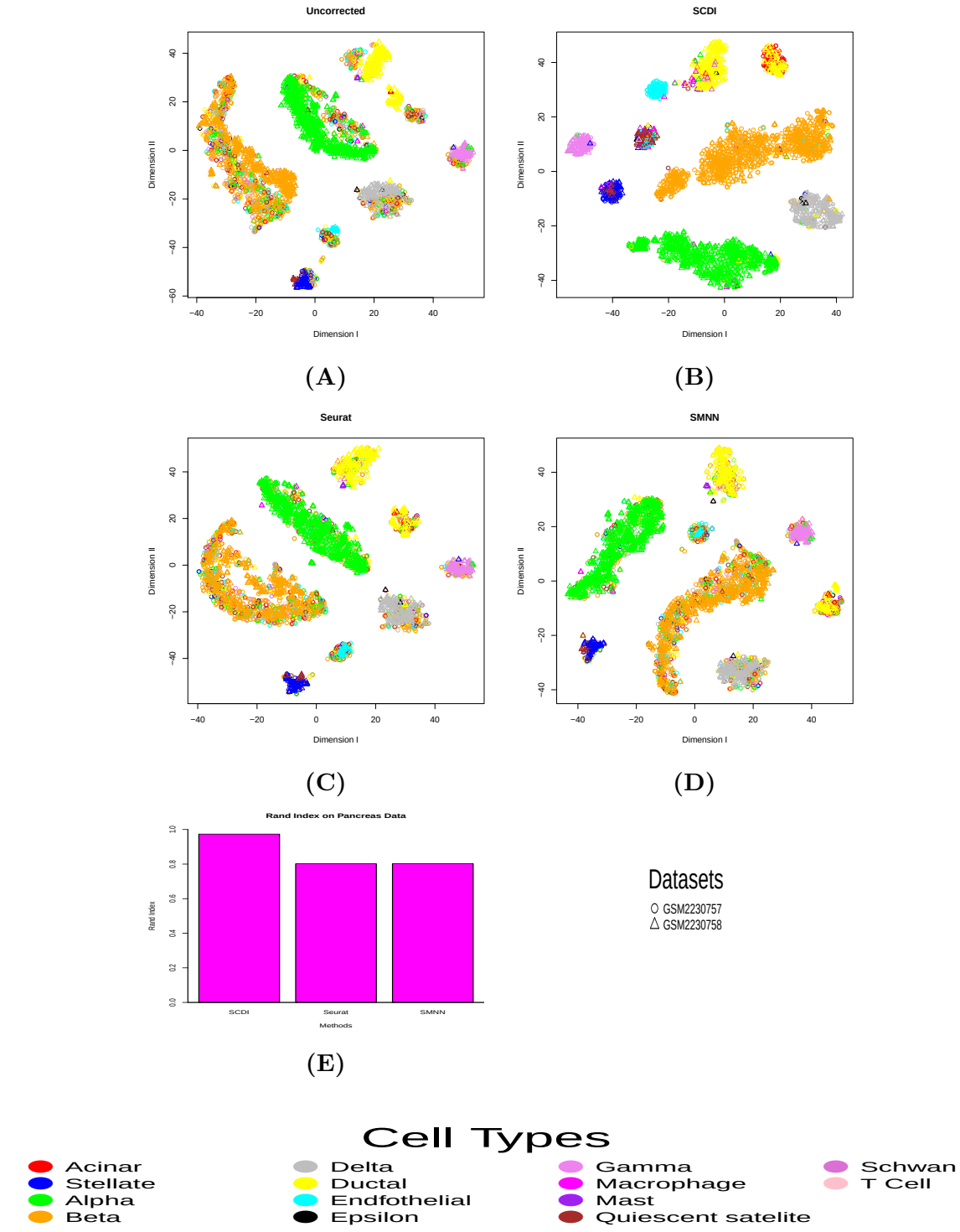


Figure 5.4: (A) tSNE plot of uncorrected data. Integration pancreatic cells by (B) SCDI, (C) Seurat, and (D) SMNN. (E) is the comparison of Adjusted Rand Index (ARI) by different methods. SCDI has the highest ARI among all methods.

5.4.2 Integration of hematopoietic stem cells with datasets from multiple labs using different technologies

Paul et al. [90] studied transcriptomic signature present in different types of hematopoietic progenitor cells and how gene regulatory mechanisms influence the cellular fate of hematopoietic stem cells that give rise to progenitors. Using multicolored FACS, they identified Common Myeloid Progenitors (CMP), Megakaryocite-erithrocyte Progenitors (MEP), and Granulocyte-erithrocyte Progenitors (GMP) using surface markers and profiled transcriptome of 2730 cells using MARS-seq. Early-stage Myeloid progenitor cells were sorted out based on CD41, FIt3+Csf1r+, and CMP Irf8-GFP+MHCII+ markers. A conditional Cebpa knockout model along with a matching control was used to study the structure of Cebpa gene in determining the fate of progenitor cells with the help of Mx1-Cre activation. Similarly, Cebpe knockout model and matching control were used to study the influence of Cebpe gene in cell fate determination.

Nestorowa et al. [86] profiled 1656 transcriptomes encompassing hematopoietic stem cells(HSC) and Myeloid progenitor(MP) cells to study gene regulatory mechanism that drives stem cells into progenitors. Some cells were also found to be of the intermediate stage to form a continuum between HSCs and MPs. Cells were sorted with gates based on c-Kit and Sca1 protein expression. Based on this classification there are three major cell types: HPSC, LT-HSC, and Progenitors. Cells were collected from 10 female 12-week-old mice and were sequenced with SMART-seq2 protocol.

After, performing quality checking and removing cells with low quality scores, tSNE plot of the uncorrected data completely separates the two datasets. Two branches of progenitor cells are created from Hematopoietic Stem Cells (HSC) in SMART-seq2 dataset whereas, in the MARS-seq dataset, there are two clusters one of which can be characterized by the abundance of Cepba knockout cells and the other can be characterized by the abundance of CMP Flt3+ Csf1r+ and CMP Irf8-GFP+MHCII+ cells. Cepba control, Cepbe control, Cepbe knockout, and unsorted myeloid cells are common to both clusters. We apply SCDI with no clustering information i.e. all cells are assumed to belong to a single cluster for both datasets. All three methods identified HSC cells from SMART-seq2 dataset as a separate population (Figure 5.5). However, only SCDI mixed up the progenitor cells from two populations properly. All three algorithms predict the lineage accurately in a qualitative manner but only SCDI successfully removes the batch effect completely to map the progenitor cells from two populations together. Both Seurat and SMNN have a bias towards creating lumps of cells from the same dataset. tSNE plot preserves clustering information but it does not preserve the shape of the data. The relative ordering of different

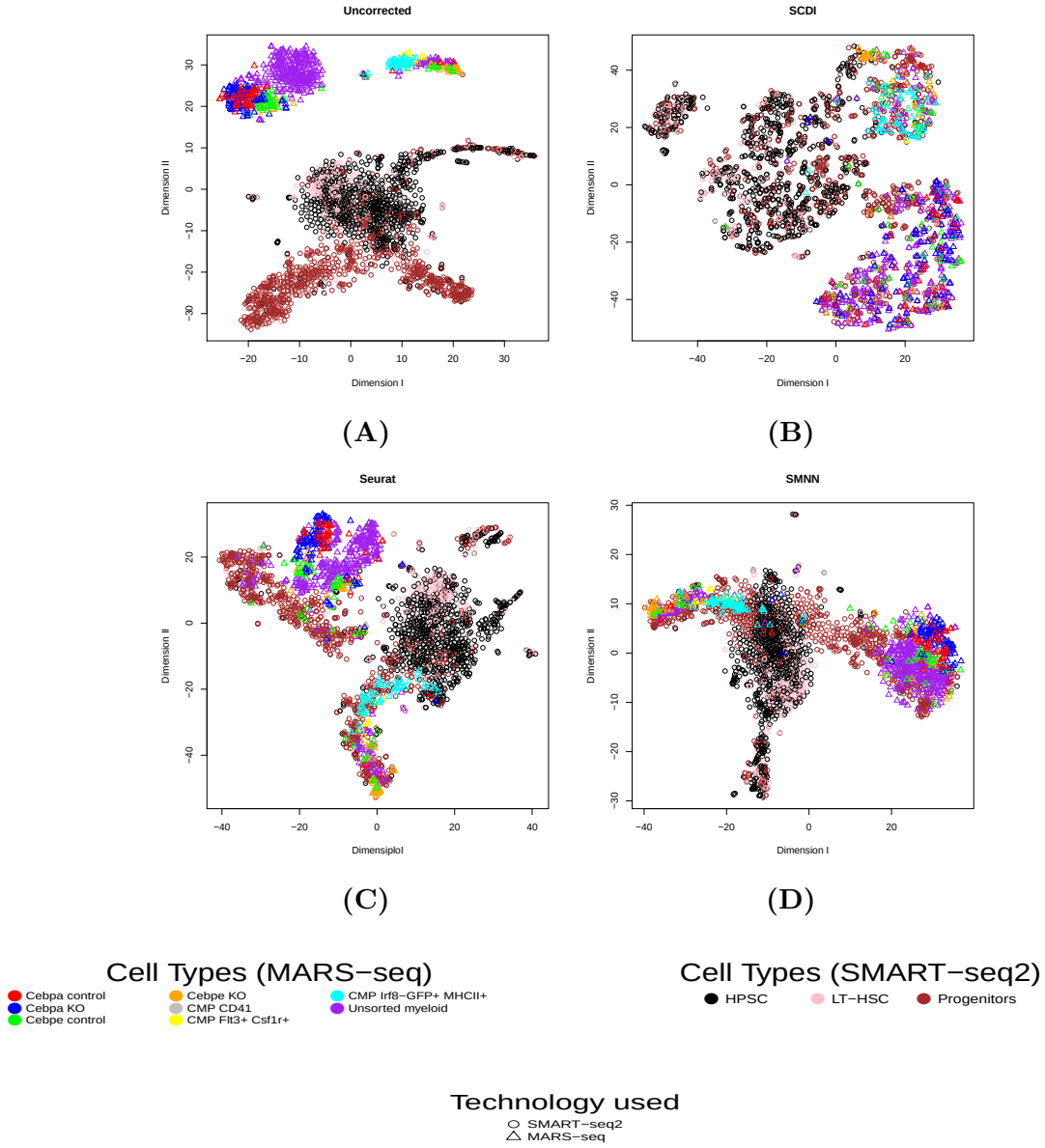


Figure 5.5: (A) tSNE plot of uncorrected data. Integration pancreatic cells by (B) SCDI, (C) Seurat, and (D) SMNN. Only SCDI mixes up progenitor cells from the two populations properly.

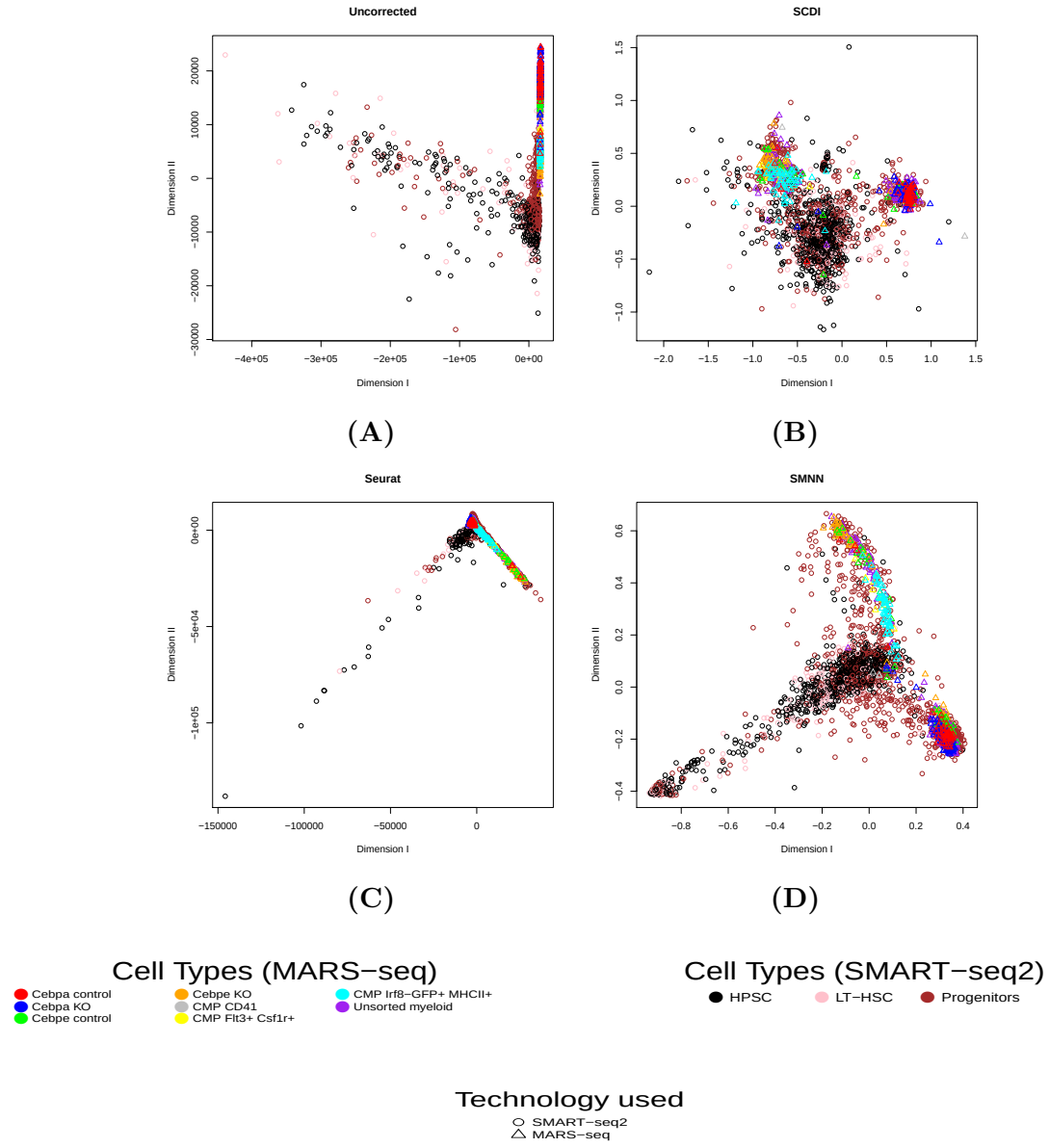


Figure 5.6: PCA plots of (A) uncorrected data (B) output from SCDI (C) output from Seurat (D) output from SMNN. Only SCDI output preserves the clear pseudotime structure.

progenitor cells can be visualized from PCA plot (Figure 5.6) where early-stage progenitors (CD41, FIt3+Csf1r+ and CMP Irf8-GFP+MHCII+ markers) appear at the initiation of branches.

5.5 Discussion

As large amount of single-cell transcriptom data continue to get generated, a suitable tool is necessary to integrate data from different sources across platforms and species. Many challenges are involved including identification of shared as well as isolated cell subpopulations across datasets, working with data of large sample size, integration from a large number of sources, etc. This will serve the ultimate goal of creating a comprehensive reference atlas of all human cells and will help the projects like Human Cell Atlas to build up and grow. That in turn will help in understanding the tissue heterogeneity for complex diseases like cancer, boost our knowledge on the immune response to infectious diseases and revolutionize the field of developmental biology.

We proposed SCDI, a semi-supervised algorithm to integrate single-cell transcriptome data from multiple sources. For datasets with subpopulations, it provides improved accuracy when cluster ids for individual datasets are provided as input. SCDI identifies common subpopulations across clusters and creates a joint clustering of all the datasets combined. We also provide the notion of batch corrected GPLVM that takes care of batch effect when two datasets are projected in lower-dimensional space simultaneously. The joint clustering also helps in more accurate mixing of cells from different datasets. The algorithm also takes care of large datasets efficiently with the help of subsampling and mapping cells to most similar clusters with the help of nearest neighbor regression.

We exhibit, with the help of splatter simulation, that SCDI can take care of batch effects dependent on cell subpopulations while the other algorithms are not properly designed to do so. Comparison on pancreas data shows that SCDI provides better mixing in terms of Rand index compared to other methods. This is probably a consequence of applying batch corrected GPLVM in an intermediate step and batch effect correction at subpopulation level that is not manifested in principles of other algorithms. Analysis of hematopoietic stem cells transcriptome data shows that SCDI mixes similar cells from different population better than other methods when multiple experimental platforms are concerned.

We speculate that the method of data integration should depend on the final objective of the integration. Hence, we propose three different ways of integration based on three different purposes, namely: common cell type identification, pseudotime estimation, and

differential expression analysis. Verification of the two algorithms for purposes other than common cell type identification remains a scope for future work.

Lastly, as technology evolves with time, the scope and scale of single-cell transcriptome data will also continue to grow. Efficient methods might be required to cope up with a large number of datasets. Towards the goal of constructing a comprehensive atlas of all cells, while dealing with a huge volume of data and considerable diversity of data types, the proficient user interface is to visualize and analyze single-cell data is an element future researchers can consider improving upon.

5.6 Code and software availability

Reproducible codes for all figures, data, and software for SCDI are available at:
http://github.com/indranillab/scdi .

Bibliography

[1] Abbas-Aghababazadeh,F., Li,Q. and Fridley,B.L. (2018) Comparison of normalization approaches for gene expression studies completed with high-throughput sequencing. *PLOS ONE*, **13(10)**, e0206312.

[2] Abdoun,O. and Abouchabaka,J. (2011) A comparative study of adaptive crossover operators for genetic algorithms to resolve the traveling salesman problem. *International Journal of Computer Applications (0975 - 8887)*, **31(11)**.

[3] Akter,S., Nahar,N., Hossain,M.S. and Andersson,K. (2019) A new crossover technique to improve genetic algorithm and its application to TSP. *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*.

[4] Andreatta,M., Corria-Osorio,J., Müller,S., Cubas,R., Coukos,G., and Carmona,S.J. (2021). Interpretation of T cell states from single-cell transcriptomics data using reference atlases. *Nat Commun.*, **12(1)**: 2965.

[5] Andrews,T.S., Hemberg,M. (2019) M3Drop: dropout-based feature selection for scRNASeq. *Bioinformatics*. **35(16)**, 2865–2867.

[6] Antolovic,V., Miermont,A., Corrigan,A.M. and Chubb,J.R. (2017) Generation of single-cell transcript variability by repression. *Current Biol.*, **27(12)**, 1811–1817.e3.

[7] Applegate,D.L., Bixby,R.E., Chvátal,V. and Cook,W.J. (2007) The Traveling Salesman Problem: A Computational Study. *Princeton University Press*.

[8] Baron,M., Veres,A., Wolock,S.L., Faust,A.L., Gaujoux,R., Vetere,A., Ryu,J.H., Wagner,B.K., Shen-Orr,S.S., Klein,A.M., et al. (2016) A Single-Cell Transcriptomic Map of the Human and Mouse Pancreas Reveals Inter- and Intra-cell Population Structure. *Cell Systems*, **3(4)**, 346-380.

[9] Barnett,L. and Bossomaier,T. (2012) Transfer Entropy as a Log-Likelihood Ratio. *Phys. Rev. Lett.*, **109(13)**, 138105.

[10] Benjamini,Y. and Yekutieli,D. (2001) The control of the false discovery rate in multiple testing under dependency. *Ann. Statist.*, **29(4)**, 1165-1188.

[11] Van den Berge,K., Perraudeau,F., Soneson,C., Love,M.I., Risso,D., Vert,J.P., Robinson,M.D., Dudoit,S. and Clement,L. (2018) Observation weights unlock bulk RNA-seq tools for zero inflation and single-cell applications. *Genome Biol*, **19**, 24.

[12] Bergen,V., Lange,M., Peidli,S., Wolf,F.A. and Theis,F.J. (2020) Generalizing RNA velocity to transient cell states through dynamical modeling. *Nat. Biotechnol.*, **38(12)**, 1408–1414.

[13] Borcherding,N., Voigt,A.P., Liu,V., Link,B.K., Zhang,W. and Jabbari,A. (2019) Single-cell profiling of cutaneous T-cell lymphoma reveals underlying heterogeneity associated with disease progression. *Clin. Cancer Res.*, **25(10)**, 2999–3005.

[14] Brennecke,P., Anders,S., Kim,J.K., Kolodziejczyk,A.A., Zhang,X., Proserpio,V., Baying,B., Benes,V., Teichmann,S.A, Marioni,J.C. et al. (2013) Accounting for technical noise in single-cell RNA-seq experiments. *Nat. Methods.* **10(11)**, 1093–1095.

[15] Butler,A., Hoffman,P., Smibert,P., Papalexi,E. and Satija,R. (2018) Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol.*, **36**, 411-420.

[16] Buettner,F., Natarajan,K.N., Casale,F.P., Proserpio,V., Scialdone,A., Theis,F.J., Teichmann,S.A., Marioni,J.C. and Stegle,O. (2015) Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. *Nat. Biotechnol.*. **33(2)**, 155–160.

[17] B*ü*ttner,M., Miao,Z., Wolf,A., Teichmann,S.A. and Theis,F.J. (2019) A test metric for assessing single-cell RNA-seq batch correction. *Nat Methods*, **16**, 43-49.

[18] Cacchiarelli,D., Qiu,X., Srivatsan,S., Manfredi,A., Ziller,M., Overbey,E., Grimaldi,A., Grimsby,J., Pokharel,P. and Livak,K.J. et al. (2018) Aligning single-cell developmental and reprogramming trajectories identifies molecular determinants of myogenic reprogramming outcome. *Cell Systems*, **7(3)**, 258–268.e3.

[19] Campbell,K., Ponting,C.P. and Webber,C. (2015) Laplacian eigenmaps and principal curves for high resolution pseudotemporal ordering of single-cell RNA-seq profiles. *bioRxiv*, **18 Sep 2015**, DOI: 10.1101/027219.

[20] Campbell,K.R. and Yau,C. (2017) Probabilistic modeling of bifurcations in single-cell gene expression data using a Bayesian mixture of factor analyzers. *Wellcome Open Res*, **2**: 19.

[21] Campbell,K.R. and Yau,C. (2018) Uncovering pseudotemporal trajectories with covariates from single cell and bulk expression data. *Nat. Communications*, **9(1)**, 2442.

[22] Cannoodt,R., Saelens,W. and Saeys,Y. (2016) Computational Methods for trajectory inference from single-cell transcriptomics. *Eur. J. Immunol.*, **46(11)**, 2496–2506.

[23] Charrad, M., Ghazzali, N., Boiteau, V. and Niknafs, A. (2014) NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set. *Journal of Statistical Software*, **61(6)**.

[24] Chen,H.H., Jin,Y., Huang,Y. and Chen,Y. (2016) Detection of high variability in gene expression from single-cell RNA-seq profiling. *BMC Genomics*. **17(Suppl 7)**, 508.

[25] Chen,G., Ning,B. and Shi,T. (2019) Single-Cell RNA-Seq Technologies and Related Computational Data Analysis. *Frontiers in Genetics*, **10**, 317.

[26] Coifman,R.R. and Lafon,S. (2006) Diffusion maps. *Applied and Computational Harmonic Analysis*, **21(1)**, 5–30.

[27] Contreras-Bolton,C. and Parada,V. (2015) Automatic combination of operators in a genetic algorithm to solve the traveling salesman problem. *PLoS ONE*, **10(9)**, e0137724.

[28] Couturier,C.P., Ayyadhury,S., Le,P.U., Nadaf,J., Monlong,J., Riva,G., Allache,R, Baig,S., Yan,X., Bourgey,M. et al. (2020) Single-cell RNA-seq reveals that glioblastoma recapitulates a normal neurodevelopmental hierarchy. *Nat Commun*, **11**, 3406.

[29] Davis,L. (1985) Applying adaptive algorithms to epistatic domains. *IJCAI'85: Proceedings of the 9th international joint conference on Artificial intelligence*, **1**, 162–164.

[30] Dayton,J,B. (2019) Adversarial Deep Neural Networks Effectively Remove Nonlinear Batch Effects from Gene- Expression Data. *Brigham Young university Scholars Archive*, **Thesis and Dissertations**, 7521.

[31] Dillies,M.-A., Rau,A., Aubert,J., Hennequet-Antier,C., Jeanmougin,M., Servant,N., Keime,C., Marot,G., Castel,D., Estelle,J. et al. (2013) A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis. *Briefings in Bioinformatics*, **14(6)**, 671-683.

[32] Eberwine,J., Sul,J-Y., Bartfai,T. and Kim,J. (2014) The promise of single-cell sequencing. *Nature Methods*, **11(1)**, 25–27.

[33] Editorial (2014) Method of the Year 2013. *Nat Methods*, **11(1)**.

[34] Elowitz,M.B., Levine,A.J., Siggia,E.D. and Swain,P.S. (2002) Stochastic Gene Expression in a Single Cell. *Science*, **297(5584)**, 1183-1186.

[35] Eungdamrong,N.J. and Iyengar,R. (2004) Modeling cell signaling networks. *Biol. Cell*, **96(5)**, 355–362.

[36] Fan,J., Salathia,N., Liu,R., Kaeser,G.E., Yung,Y.C., Herman,J.L., Kaper,F., Fan,J-B, Zhang,K. and Chun,J. (2016) Characterizing transcriptional heterogeneity through pathway and gene set overdispersion analysis. *Nat Methods*, **13**, 241-244.

[37] Farewell,V.T., Long,D.L., Tom,B.D.M., Yiu,S. and Su,L. (2017) Two-Part and Related Regression Models for Longitudinal Data. *Annual Review of Statistics and Its Application*, **4**, 283-315.

[38] Fernández-Val,I. and Weidner,M. (2018) Fixed Effects Estimation of Large-TPanel Data Models. *Annual Review of Economics*, **10**, 109-138.

[39] Finak,G., McDavid,A., Yajima,M., Deng,J., Gersuk,V., Shalek,A.K., Slichter,C.K., Miller,H.W., McElrath,M.J. and Prlic,M. et al. (2015) MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. *Genome Biol.*, **16:278**.

[40] Fujita,K., Iwaki,M. and Yanagida,T. (2016) Transcriptional bursting is intrinsically caused by interplay between RNA polymerases on DNA. *Nat Commun*, **7**, 13788.

[41] Goldberg,D.E. and Lingle, R. Jr. (1985) Alleles, loci and the travelling salesman problem. *Proceedings of the first international conference on genetic algorithms and their applications*, 154–159.

[42] Goldberg,D.E. (1989) Genetic Algorithms in Search, Optimization & Machine Learning. *Addison-Wesley Publishing Company.*

[43] Goodman,L.A. (1954) Kolmogorov-Smirnov tests for psychological research. *Psychol Bull*, **51(2:1)**, 160-168.

[44] Hafemeister,C. and satija,R. (2019) Normalization and variance stabilization of single-cell RNA-seq data using regularized negative binomial regression. *Genome Biol*, **20**, 296.

[45] Haghverdi,L., Büttner,M., Wolf,A., Buettner,F. and Theis,F.J. (2016) Diffusion pseudotime robustly reconstructs lineage branching. *Nat. Methods*, **13(10)**, 845–848.

[46] Haghverdi,L., Lun,A.T.L, Morga,M.D. and Marioni,J.C. (2018) Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat. Biotechnol.*, **36**, 421-427.

[47] Holzmann,H. and Vollmer,S. (2008) A likelihood ratio test for bimodality in two-component mixtures with application to regional income distribution in the EU. *AStA Advances in Statistical Analysis*, **92**, 57-69.

[48] Hou,W., Ji,Z., Ji,H. and Hicks,S.C. (2020) A systematic evaluation of single-cell RNA-sequencing imputation methods. *Genome Biol*, **21**, 218.

[49] Hussain,A., Muhammad,Y.S., Sajid,M.N., Hussain,I., Shoukry,A.M., and Gani,S. (2017) Genetic algorithm for traveling salesman problem with modified cycle crossover operator. *Comput. Intell. Neurosci.*.

[50] Hwang,B., Lee,J.H. and Bang,D. (2018) Single-cell RNA sequencing technologies and bioinformatics pipelines. *Experimental & Molecular Medicine*, **50(8)**, 96.

[51] Iacono,G., Massoni-Badosa,R. and Heyn,H. (2019) Single-cell transcriptomics unveils gene regulatory network plasticity. *Genome Biol.*, **20(1)**, 110.

[52] Islam,S., Kjällquist,U., Moliner,A., Zajac,P., Fan,J-B., Lönnerberg,P. and Linnarsson,S. (2011) Characterization of the single-cell transcriptional landscape by highly multiplex RNA-seq. *Genome Res.*, **21(7)**, 1160-1167.

[53] Jakkola,M.K., Seyednasrollah,F., Mehmood,A, and Elo,L.L. (2017) Comparison of methods to detect differentially expressed genes between single-cell populations. *Briefings in Bioinformatics*, **18(5)**, 735-743.

[54] Jennings,P.C., Lysgaard,S., Hummelshϕj,J.S., Vegge,T. and Bliggard,T. (2019) Genetic algorithms for computational material discovery accelerated by machine learning. *npj Computational Materials*, **5**.

[55] Ji,Z. and Ji,H. (2016) TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic Acids Research*, **44(13)**, e117.

[56] Jindal,A., Gupta,P., Jayadeva and Sengupta,D. (2018) Discovery of rare cells from voluminous single cell expression data. *Nat. Commun.*, **9**, 4719.

[57] Johanson,N. and Quon,G. (2019) scAlign: a tool for alignment, integration, and rare cell identification from scRNA-seq data. *Genome Biol.*, **20**, 166.

[58] Kharchenko,P.V., Silberstein,L. and Scadden,D.T. (2014) Bayesian approach to single-cell differential expression analysis. *Nat Methods*, **11**, 740-742.

[59] Kiselev,V.Y., Kirschner,K., Schaub,M.T., Andrews,T., Yiu,A., Chandra,T., Natarajan,K.N., Reik,W., Barahona,M., Green, A.R., Hemberg,M. (2017) SC3: consensus clustering of single-cell RNA-seq data. *Nat. Methods.*, **14**, 483?486.

[60] Korthauer,K.D. , Chu,L-F, Newton,M.A., Li,Y., Thomson,J., Stewart,R. and Kendziorski,C. (2016) A statistical approach for identifying differential distributions in single-cell RNA-seq experiments. *Genome Biol.*, **17**, 222.

[61] Korsunsky,I., Millard,N., Fan,J., Slowikowski,K., Zhang,F., Wei,K., Baglaenko,Y., Brenner,M., Loh,P.-r., Raychaudhuri,S. (2019) Fast, sensitive and accurate integration of single-cell data with Harmony. *Nat Methods*, **16**, 1289-1296.

[62] Kumar,N., Singh,A. and Kulkarni,R.V. (2015) Transcriptional Bursting in Gene Expression: Analytical Results for General Stochastic Models. *PLOS Computational Biology*, **11(10)**, e1004292.

[63] La Manno,G., Gyllborg,D., Codeluppi,S., Nishimura,K., Salto,C., Zeisel,A., Borm,L.E., Stott,S.R.W., Toledo,E.M. and Villaescusa,J.C. et al. (2016) Molecular diversity of midbrain development in mouse, human, and stem cells. *Cell*, **167(2)**, 566-580.e19.

[64] L*ä*hnemann,D., K*ö*ster,J., Szczurek,E., McCarthy,D.J., Hicks,S.C., Robindon,M.D., Vallejos,C.A., Campbell,K.R., Beerenwinkel,N. and Mahfouz,A. et al. (2020) Eleven grand challenges in single-cell data science. *Genome Biol*, **21**, 31.

[65] Larrañaga,P., Kuijpers,C.M..H., Murga,R.H., Inza,I. and Dizdarevic,S. (1999) Genetic algorithms for the travelling salesman problem: a review of representations and operators. *Artificial Intelligence Review*, **13(2)**, 129–170.

[66] Lawrence,N. (2005) Probabilistic Non-linear Principal Component Analysis with Gaussian Process Latent Variable Models. *Journal of Machine Learning Research*, **6(60)**, 1783-1816.

[67] Levsky,J.M., Shenoy,S.M., Pezo,R.C. and Singer,R.H. (2002) Single-cell gene expression profiling. *Science*, **297(5582)**, 836–840.

[68] Li,P. and Chen,S. (2016) A review on Gaussian Process Latent Variable Models. *CAAI Transactions on Intelligence Technology*, **1(4)**, 366-376.

[69] Li,H., Courtois,E.T., Sengupta,D., Tan,Y., Chen,K.H., Goh,J.J.L., Kong,S.L, Chua,C., Hon,L.K., and Tan,W.S. et al. (2017) Reference component analysis of single-cell transcriptomes elucidates cellular heterogeneity in human colorectal tumors. *Nat Genet*, **49**, 708-718.

[70] Lin,Y., Ghazanfar,S., Wang,K.Y.X., Gagon-Bartsch,J.A., Lo,K.K., Su,X., Han,Z.-G., Ormerod,J.T., Speed,T.P., Yang,P. et al. (2019) scMerge leverages factor analysis, stable expression, and pseudoreplication to merge multiple single-cell RNA-seq datasets. *Proceedings of the National Academy of Sciences*, **116(20)**, 9775-9784.

[71] Lin,P., Troup,M., Ho, J.W. (2017) CIDR: Ultrafast and accurate clustering through imputation for single-cell RNA-seq data. *Genome Biol.*, **18**, 59.

[72] Linderman,G.C., Rachh,M., Hoskins,J.G., Steinerberger,S., Kluger,Y. (2019) Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data. *Nat Methods*, **16**, 243-245.

[73] Liu,Y., Fan,X., Wang,R., Lu,X., Dang,Y-L., Wang,H., Lin,H-Y., Zhu,C., Ge,H. and Cross,J.C. et al. (2018) Single-cell RNA-seq reveals the diversity of trophoblast subtypes and patterns of differentiation in the human placenta. *Cell Research*, **28**, 819-832.

[74] Loeffler-Wirth,H., Binder,H., Willscher,E., Gerber,T. and Kunz,M. (2018) Pseudotime dynamics in melanoma single-cell transcriptomes reveals different mechanisms of tumor progression. *Biology (Basel)*, **7(2)**, 23.

[75] Ma,L., Pak,M.L., Ou,J., Yu,J., Louis,P.S., Shan,Y., Hutchinson,L., Li,S., Brehm,M.A., Zhu,L.J. et al. (2019) Prosurvival kinase PIM2 is a therapeutic target for eradication of chronic myeloid leukemia stem cells. *Proceedings of the National Academy of Sciences*, **116(21)**, 10482-10487.

[76] van der Maaten,L. and Hinton,G. (2008) Visualizing Data using t-SNE. *Journal of Machine Learning Research*, **9(86)**, 2579-2605.

[77] Macaulay,I.C., Svensson,V., Labalette,C., Ferreira,L., Hamey,F., Voet,T., Teichmann,S.A. and Cvejic,A. (2016) Single-cell RNA-sequencing reveals a continuous spectrum of differentiation in hematopoietic cells. *Cell Rep.*, **14(4)**, 966–977.

[78] Marco,E., Karp,R.L, Guo,G., Robson,P., Hart,A.H., Trippa,L. and Yuan,G. (2014) Bifurcation analysis of single-cell gene expression data reveals epigenetic landscape. *Proc. Natl. Acad. Sci.*, **111(52)**, E5643–E5650.

[79] Massey,F.J. Jr (1951) The Kolmogorov-Smirnov Test for Goodness of Fit. *Journal of the American Statistical Association*, **46:253**, 68-78.

[80] McDavid,A., Gottardo,R., Simon,N. and Drton,M. (2019) Graphical models for zero-inflated single cell gene expression. *Ann App Stat*, **13(2)**, 848-873.

[81] Miao,Z., Deng,K., Wang,X. and Zhang,X. (2018) DEsingle for detecting three types of differential expression in single-cell RNA-seq data. *Bioinformatics*, **34(18)**, 3223-3224.

[82] Molin,A.D., Baruzzo,G. and Camillo,B.D. (2017) Single-Cell RNA-Sequencing: Assessment of Differential Expression Analysis Methods. *Front Genet.*, **8:62**.

[83] Moratzavi,A., Williams,B.A., McCue,K., Schaeffer,L. and Wold,B. (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat Methods*, **5**, 621-628.

[84] Nawy,T. (2014) Single-cell sequencing. *Nature Methods*, **11(1),** 18.

[85] Nelms,B. and Walbot,V. (2019) Defining the developmental program leading to meiosis in maize. *Science*, **364(6435)**, 52–56.

[86] Nestorowa,S., Hamey,F.K., Sala,B.P., Diamanti,E., Stepherd,M., Laurenti,E., Wilson,N.K., Kent,D.G., Göttgens,B. (2016) A single-cell resolution map of mouse hematopoietic stem and progenitor cell differentiation. *Blood*, **128(8)**, e20-e31.

[87] Oliver,I.M., Smith,D.J. and Holland,J.R.C. (1987) A study of permutation crossover operators on the traveling salesman problem. *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and their application*, 224–230.

[88] Olsen,T.K. and Baryawno,N. (2018) Introduction to Single-Cell RNA Sequencing. *Curr Protoc Mol Biol.*, **122(1)**:e57

[89] Papadopoulos,N., Gonzalo,P.R. and Söding,J. (2019) PROSSTT: probabilistic simulation of single-cell RNA-seq data for complex differentiation processes. *Bioinformatics*, **35(18)**, 3517–3519.

[90] Paul,F., Arkin,Y., Giladi,A., Jaitin,D.A., Kenigsberg,E., Keren-Shaul,H., Winter,D., Lara-Astiaso,D., Gury,M. and Weiner,A. (2015) Transcriptional Heterogeneity and Lineage Commitment in Myeloid Progenitors. *Cell*, **163(7)**, 1663-1677.

[91] Picelli,S., Faridani,O.R., Björklund,Å.K., Winberg,G., Sagasser,S. and Sandberg,R. (2014) Full-length RNA-seq from single cells using Smart-seq2. *Nature Protocols*, **9**, 171-181

[92] Polański,K., Young,M.D., Miao,Z., Meyer,K.B., Tiechmann,S.A. and Park,J.-E. (2020) BBKNN: fast batch alignment of single cell transcriptomes. *Bioinformatics*, **36(3)**, 964-965.

[93] Potvin,J-Y. (1996) Genetic algorithms for the traveling salesman problem. *Annals of Operations Research*, **63(3)**, 337-370.

[94] Qiu,X., Mao,Q., Tang,Y., Wang,L., Chawla,R., Pliner,H.A. and Trapnell,C. (2017) Reversed graph embedding resolves complex single-cell trajectories. *Nat. Methods*, **14(10)**, 979–982.

[95] Qiu,P. (2020) Embracing the dropouts in single-cell RNA-seq analysis. *nat Commun*, **11**, 1169.

[96] Raj,A. and Oudenaarden,A.v. (2008) Nature, Nurture, or Chance: Stochastic Gene Expression and Its Consequences. *Cell*, **135(2)**, 216-226.

[97] Robertson,C.A. and Fryer,G. (1969) Some descriptive properties of normal mixtures. *Scandinavian Actuarial Journal*, **1969**, 3-4.

[98] Rohart,A., Eslami,A., Matigian,N., Bougeard,S. and Cao,K.-A..L. (2017) MINT: a multivariate integrative method to identify reproducible molecular signatures across independent experiments and platforms. *BMC Bioinformatics*, **18**, 128.

[99] Saliba,A.-E., Westermann,A.J., Gorski,S.A. and Vogel,J. (2014) Single-cell RNA-seq: advances and future challenges. *Nucleic Acids Research*, **42(14)**, 8845-8860.

[100] Sarathi,A. and Palaniappan,A. (2019) Novel significant stage-specific differentially expressed genes in hepatocellular carcinoma. *BMC cancer*, **19**, 663.

[101] Schimdt,F., Ranjan,B., Lin,Q.X.X., Krishnan,V., Joanito,I., Honardoost,M.A., Nawaz,Z., Venkatesh,P.N., Tan,J. and Rayan,N.A. et al. (2021) RCA2: a scalable supervised clustering algorithm that reduces batch effects in scRNA-seq data. *Nucleic Acids Res.*, **49(15)**, 8505-8519.

[102] Shalek,A.K., Satija,R., Shuga,J., Trombetta,J.J., Gennert,D., Lu,D., Chen,P., Gertner,R.S., Gaublomme,J.T. and Yosef,N. et al. (2014) Single-cell RNA-seq reveals dynamic paracrine control of cellular variation. *Nature*, **510(7505)**, 363–369.

[103] Shin,J., Berg,D.A., Zhu,Y., Shin,J.Y., Song,J., Bonaguidi,M.A., Enikolopov,G., Nauen,D.W., Christian,K.M. and Ming,G-I. et al. (2015) Single-cell RNA-seq with Waterfall reveals molecular cascades underlying adult neurogenesis. *Cell Stem Cell*, **17(3)**, 360–372.

[104] Shintaku,H., Nishikii,H., Marshall,L.A., Kotera,H. and Santiago,J.G. (2014) On-Chip Separation and Analysis of RNA and DNA from Single Cells. *Anal. Chem.*, **86(4)**, 1953-1957.

[105] Silverman,J.D., Roche,K., Mukherjee,S. and David,L.A. (2020) Naught all zeros in sequence count data are the same. *Computational and Structural Biotechnology Journal*, **20**, 2789-2798.

[106] Sivanandam,S.N. and Deepa,S.N. (2008) Introduction to Genetic Algorithms. Springer, Berlin, Heidelberg.

[107] Slakter,M.J. (2012) A Comparison of the Pearson Chi-Square and Kolmogorov Goodness-of-Fit Tests with Respect to Validity. *Journal of the American Statistical Association*, **60(311)**, 845-858.

[108] Soneson,C, and Robinson,M.D. (2018) Bias, robustness and scalability in single-cell differential expression analysis. *Nature Methods*, **15**, 255-261.

[109] Stone,J.V. (2014) Independent Component Analysis: A Tutorial Introduction. *MIT Press*.

[110] Street,K., Risso,D., Fletcher,R.B., Das,D., Ngai,J., Yosef,N., Purdom,E. and Dudoit,S. (2018) Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. *BMC Genomics*, **19**, 477.

[111] Stuart,T., Butler,A., Hoffman,P., Hafemeister,C., Papalexi,E., Mauck III, W.M., Hao,Y., Stoeckius,M., Smibert,P. and Satija,R. (2019) Comprehensive Integration of Single-Cell Data. *Cell*, **177(7)**, 1888-1902.

[112] Tanay,A. and Regev,A. (2018) Single cell genomics: from phenomenology to mechanism. *Nature*, **541(7637)**, 331–338.

[113] Tang,F., Barbacioru,C., Wang,Y., Nordman,E., Lee,C., Xu,N., Wang,X., Bodeau,J., Tuch,B.B. and Siddiqui,A. et al.(2009) mRNA-Seq whole-transcriptome analysis of a single cell. *Nature Methods*, **6**, 377-382.

[114] Tang,X., Huang,Y., Lei,J., Luo,H. and Zhu,X. (2019) The single-cell sequencing: new developments and medical applications. *Cell Biosci*, **9**, 53.

[115] Thomas,A., Barriere,S., Broseus,L., Brooke,J., Lorenzi,C., Villemin,J-P., Beurier,G., Sabatier,R., Reynes,C., Mancheron,A. et al. (2019) GECKO is a genetic algorithm to classify and explore high throughput sequencing data. *Communications Biol.*, **2:222**.

[116] Tipping,M.E. and Bishop C.M. (1999) Probabilistic Principal Component Analysis *Journal of the Royal Statistical Society, Series B*, **61**, 611-622.

[117] Trapnell,C., Cacchiarelli,D., Grimsby,J., Pokharel,P., Li,S., Morse,M., Lennon,N.J., Livak,K.J., Mikkelsen,T.S. and Rinn,J.L. (2014) The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nat. Biotechnol.*, **32(4)**, 381–386.

[118] Trapnell,C. (2015) Defining cell types and states with single-cell genomics. *Genome Res.*, **25**, 1491-1498.

[119] Trapnell,C. (2019) HSMMSingleCell: Single-cell RNA-Seq for differentiating human skeletal muscle myoblasts (HSMM). *R package version 1.4.0*.

[120] Tran,H.T.N., Ang,K.S., Chevrier,M., Zhang,X., Lee,N.Y.S., Goh,M. and Chen,J. (2020) A benchmark of batch-effect correction methods for single-cell RNA sequencing data. *Genome Biol*m **21**, 12.

[121] Treutlein,B., Brownfield,D.G., Wu,A.R., Neff,N.F., Mantalas,G.L., Espinoza.F.H., Desai,T.J., Krasnow,M.A. and Quake,S.R. (2014) Reconstructing lineage hierarchies of the distal lung epithelium using single-cell RNA-seq. *Nature*, **509(7500)**, 371-375.

[122] Tsuyuzaki,K., Sato,H., Sato,K. and Nikaido,I. (2020) Benchmarking principal component analysis for large-scale single-cell RNA-sequencing. *Genome Biol.*, **21(1)**, 9.

[123] Tung,P.-Y., Blischak,J.D., Hsiao,C.J., Knowles,D.A., Burnett,J.E., Pritchard,J.K. and Gilad,Y. (2017) Batch effects and the effective design of single-cell gene expression studies. *Sci Rep*, **7**, 39921.

[124] Vallejos,C.A., Marioni,J.C. and Richardson,S. (2015) BASiCS: Bayesian analysis of single-cell sequencing data. *PLoS Computat. Biol.*, **11(6)**, e1004333.

[125] Verhulst,S., Roskams,T., Sancho-Bru,P. and Grunsven, L.A.v. (2019) Meta-Analysis of Human and Mouse Biliary Epithelial Cell Gene Profiles. *Cells*, **8(10)**, 1117.

[126] Voelkerding, K.V., dames, S.A. and Durtschi, J.D. (2009) Next-Generation Sequencing: From Basic Research to Diagnostics. *Clinical Chemistry*, **55(4)**, 641-658.

[127] Vu,T.N., Wills,Q.F., Kalari,K.R., Niu,N., Wang,L., Rantalainen,M. and Pawitan,Y. (2016) Beta-Poisson model for single-cell RNA-seq data analyses. *Bioinformatics*, **32(14)**, 2128-2135.

[128] Wagner,A., Regev,A. and Yosef,N. (2016) Revealing the vectors of cellular identity with single-cell genomics. *Nat Biotechnol*, **34**, 1145-1160.

[129] Wang,Y. and Navin,N.E. (2015) Advances and Applications of Single-Cell Sequencing Technologies. *Molecular Cell*, **58(4)**, 598-609.

[130] Wang,L., Catalan,F., Shamardani,K., Babikir,H., and Diaz,A. (2020) Ensemble learning for classifying single-cell data and projection across reference atlases. *Bioinformatics*, **36(11)**, 3585-3587.

[131] Wu,Z., Zhang,Y., Stitzel,M.L. and Wu,H. (2018) Two-phase differential expression analysis for single cell RNA-seq. *Bioinformatics*, **34(19)**, 3340-3348.

[132] Yang,Y., Li,G., Qian,H., Wilhelmson,K.C., Shen,Y. and Li,Y. (2021) SMNN: batch effect correction for single-cell RNA-seq data via supervised mutual nearest neighbor detection. *Briefings in Bioinformatics*, **22(3)**.

[133] Zappia,L., Phipson,B and Oshlack,A. (2017) Splatter: simulation of single-cell RNA sequencing data. *Genome Biol.*, **18:174**.