3-27-2014 12:00 AM

Statistical methods for the analysis of RNA sequencing data

Man-Kee Maggie Chu, *The University of Western Ontario*

Supervisor: Dr. Wenqing He, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree in Statistics and Actuarial Sciences

STATISTICAL METHODS FOR THE ANALYSIS OF RNA SEQUENCING DATA

(Thesis format: Monograph)

by

Man-Kee Maggie Chu

Graduate Program in Statistical and Actuarial Sciences

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

Abstract

The next generation sequencing technology, RNA-sequencing (RNA-seq), has an increasing popularity over traditional microarrays in transcriptome analyses. Statistical methods used for gene expression analyses with these two technologies are different because the array-based technology measures intensities using continuous distributions, whereas RNA-seq provides absolute quantification of gene expression using counts of reads. There is a need for reliable statistical methods to exploit the information from the rapidly evolving sequencing technologies and limited work has been done on expression analysis of time-course RNA-seq data.

Functional clustering is an important method for examining gene expression patterns and thus discovering co-expressed genes to better understand the biological systems. Clustering-based approaches to analyze repeated digital gene expression measures are in demand. In this dissertation, we propose a model-based clustering method for identifying gene expression patterns in time-course RNA-seq data. Our approach employs a longitudinal negative binomial mixture model to postulate the over-dispersed time-course gene count data. The effectiveness of the proposed clustering method is assessed using simulated data and is illustrated by real data from time-course genomic experiments.

Due to the complexity and size of genomic data, the choice of good starting values is an important issue to the proposed clustering algorithm. There is a need for a reliable initialization strategy for cluster-wise regression specifically for time-course discrete count data. We modify existing common initialization procedures to suit our model-based clustering algorithm and the procedures are evaluated through a simulation study on artificial datasets and are applied to real genomic examples to identify the optimal initialization method.

Another common issue in gene expression analysis is the presence of missing values in the datasets. Various treatments to missing values in genomic datasets have been developed but limited work has been done on RNA-seq data. In the current work, we examine the performance of various imputation methods and their impact on the clustering of time-course RNA-seq data. We develop a cluster-based imputation method which is specifically suitable for dealing with missing values in RNA-seq datasets. Simulation studies are provided to assess the performance of the proposed imputation approach.

Keywords: RNA-seq data, time-course, cluster analysis, missing values

This thesis is dedicated to my family
for their love, support and encouragement.

Acknowledgements

I would like to thank my supervisor Dr. Wenqing He for his continuous guidance during my studies. This research would not have been possible without his support and encouragement. I am also very grateful of everyone in the Department of Statistical and Actuarial Sciences at the University of Western Ontario for their academic and logistic support throughout the years. I want to extend my appreciation to the faculty members for their academic advices and the friendly staff Jane, Jennifer and Lisa for their assistance. I would also like to thank my colleagues in the department for their friendship and support.

I want to thank my parents for their unconditional love for me. Without their encouragement and support, I would never have been able to pursue my interests and achieve this accomplishment. Also, my heartfelt love and thanks go to my fiance Thaddeus for his love and support during my time in graduate school.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Background

1.1 Motivation

In order to have a better understanding of complex conditions such as heart disease and cancers, there is a need for advancements in identifying genes related to common chronic diseases. This can be achieved by furthering our knowledge of gene functions through statistical models, such as performing statistical analysis of gene expression profiles. Since the mid-1990s, DNA microarrays have been the technology of choice for studying gene expression levels. However, these methods have several limitations which restrict their uses in genome research.

Recently, next-generation sequencing technologies have increased sequencing capacity at a fast rate such that ultra-high-throughput sequencing is emerging as the preferred approach over hybridization-based microarrays for characterizing and quantifying entire genomes. As the cost of sequencing continues to fall, the more powerful sequencing data is expected to replace microarrays for many applications. However, there is a need for reliable and accurate statistical analysis methods to exploit the information carried by the rapidly evolving sequencing technologies.

1.2 Background knowledge

1.2.1 Gene expression analysis

The production of proteins in a biological system is controlled by genes through transcription and translation. This production process is referred to the central dogma of molecular biology and it is illustrated in Figure 1.1. Transcription refers to the process of messenger ribonucleic acid (mRNA) being copied and edited from the deoxyribonucleic acid (DNA) coding of the gene and translation represents the assembly of amino acids from mRNA to form the protein (Parmigiani et al, 2003). Cellular activity is controlled by the amount of mRNA being transcribed or expressed by individual genes, and gene expression can be controlled at different steps (see Figure 1.2). Gene regulation is necessary because cells can prevent resources being wasted by switching off genes that are not needed, and gene activity is controlled first and foremost at the transcription stage. Since the main site of control for most cells is the regulation of transcription, exploring the RNA component of cells, known as the transcriptome, can provide great insight into the biological system as a whole. Some of the techniques available for measuring gene expression levels include serial analysis of gene expression (SAGE), complementary DNA (cDNA) subtraction, differential display, cDNA-sequencing, multiplex quantitative reverse transcription polymerase chain reaction (RT-PCR), and microarrays. The most popular methods used in gene expression investigations are microarrays and cDNA library sequencing.

1.2.2 Microarrays

Gene expression microarray technology enables high-throughput profiling by simultaneously monitoring the expression levels of thousands of genes in a single experiment through the hybridization of RNA from tissues or cells onto high-density arrays of thousands of probes. Each probe on the array chip contains many copies of the same sequence, which is a variant

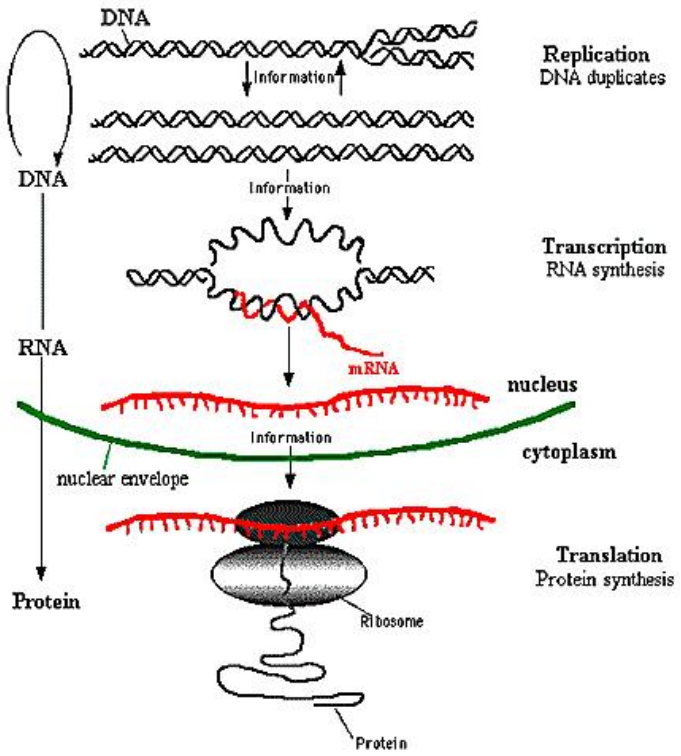

Figure 1.1: The central dogma: Transcription of DNA to RNA to protein

This dogma forms the backbone of molecular biology and is represented by four major stages. (1) The DNA replicates its information in a process that involves many enzymes: replication. (2) The DNA codes for the production of messenger RNA (mRNA) during transcription. (3) In eucaryotic cells, the mRNA is processed (essentially by splicing) and migrates from the nucleus to the cytoplasm. (4) Messenger RNA carries coded information to ribosomes. The ribosomes "read" this information and use it for protein synthesis. This process is called translation. Proteins do not code for the production of protein, RNA or DNA. They are involved in almost all biological activities, structural or enzymatic. (Public domain image from Access Excellence @ the National Health Museum, http://www.accessexcellence.org/RC/VL/GG/images/central.gif)

of a specific DNA, RNA or cDNA. The target sequences labeled with fluorescence tags are hybridized to the microarray chip and the hybridization between the probes and the target transcripts in the sample are detected by fluorescence shown in each probe or other imaging method. The luminescent intensity of each probe indicates the hybridization intensity, which gives the relative quantity of the transcripts that are represented and indicates gene expression levels (Allison et al., 2006; Matukumalli and Schroeder, 2009). Figure 1.3 illustrates the workflow of a microarray experiment for two conditions of interest.

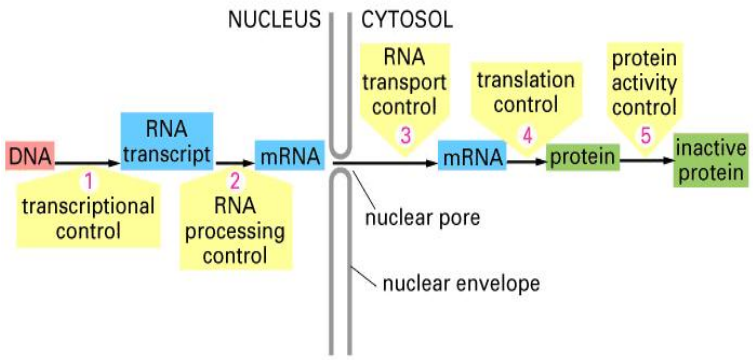

Figure 1.2: Control of gene expression

Eukaryotic gene expression can be controlled at several different steps. Examples of regulation at each of the steps are known, although for most genes the main site of control is step 1- transcription of a DNA sequence into RNA. (Public domain image from Access Excellence @ the National Health Museum, http://www.accessexcellence.org/RC/VL/GG/ecb/ecb_images/08_03_gene_expression.jpg)

1.2.3 Gene expression by sequencing

Another widely used method for exploring transcriptomes is by the advanced DNA-sequencing technology. Microarray has been the technology of choice for gene expression investigations since the mid-1990s, but the Sanger sequencing biochemistry (Sanger et al., 1977) quickly set the stage for sequencing to become an attractive alternative technology for biological research.

In the early to mid-1990s, the DNA molecules being sequenced were viral, organelle or bacterial genomes. The most major step in the sequencing of genome of higher organisms began in 1990 with the Human Genome Project (HGP), which was an international scientific research project between sequencing centers in the United States, Europe and Japan. In competition with the publicly funded project, the privately owned Celera Genomics began to sequence the human genome using a sequencing method called the whole genome random shotgun method. The sequencing process by HGP and Celera both benefited from each other, since the Celera assembly incorporated the HGP DNA sequence data and HGP adopted Celera's paired-end sequencing method (details on methods and processes used during both projects are summarized in Chial (2008)). The two projects were completed in 2000, and the draft human genome se-

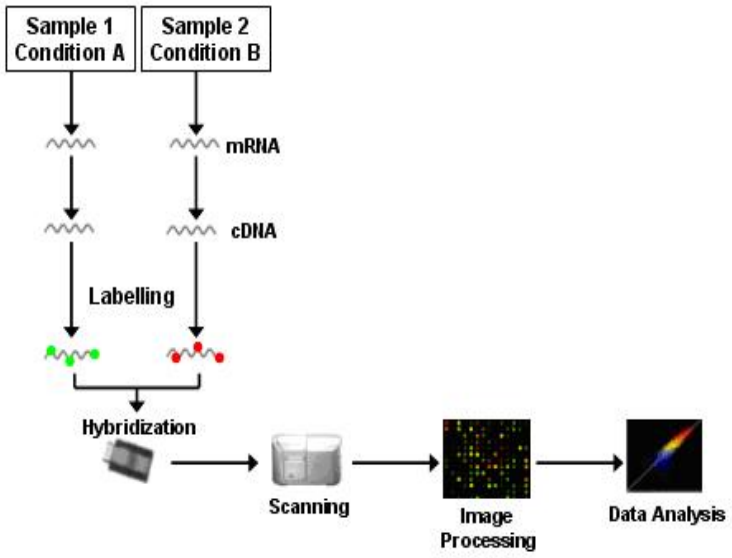

Figure 1.3: Microarray experiment workflow

(Public domain image from Genomic Research Laboratory, Geneva, http://www.genomic.ch/pict/workflow.png)

quences from both the publicly funded project and Celera were published in 2001 (Lander et al., 2001; Venter et al., 2001).

With the success of HGP, very large-scale sequencing of genome is now the approach for analyzing many problems concerning biology, disease and the environment. As an advancement from the 'first generation' Sanger method, next generation sequencing (NGS) methods were developed to produce an enormous amount of data rapidly and cheaply. These NGS methods, including systems from 454 Life Sciences (Roche) (Margulies et al., 2005), Illumina GA (formerly Solexa) (Bennett et al. 2005), and Applied Biosystems' SOLiD (www.appliedbiosystems.com), can generate billions of bases in a single run, allowing thousands of megabases of DNA to be sequenced in a matter of days. The various NGS platforms differ in sequencing biochemistry but the processing pipeline is generally the same and they are based on parallelizing the sequencing process (Hutchison, 2007; Pettersson et al., 2009; Shendure and Ji, 2008). These new methods have shorter read lengths and slower sequence

extraction from each feature as compared to the Sanger method, but the parallelized nature of the process provides much higher total throughput with lower cost by generating thousands of bases per second. Also, by over sampling the single fragments during sequencing, these novel methods may offer greater coverage and increased total accuracy when attempting to build the original sequence (Pettersson et al., 2009).

The development of ultra-high-throughput sequencing technologies with decreased cost in recent years allow for numerous applications in biological research (Shendure and Ji, 2008). Gene expression analysis with whole-transcriptome sequencing (RNA-Seq) can be performed to determine quantitative differences between samples and for annotation of splice junctions and transcript boundaries. Other applications of sequencing technologies include detecting the presence of an event such as the binding of a transcription factor, full-genome resequencing for discovery of mutations or polymorphisms, mapping of copy number variation, analysis of DNA methylation and genome-wide mapping of DNA-protein interactions (ChIP-Seq analyses).

1.2.4 RNA-seq

In RNA-seq experiments, a sample of purified RNA is first sheared and converted into cDNA, then sequenced on a high-throughput platform such as Illumina, SOLiD or Roche454. The platforms differ in their biochemistry and processing steps, but they all generate millions of short reads either taken from one end or from both ends of each cDNA fragment as results. An overview of a general RNA-seq experiment is described in Figure 1.4a, and the data analysis process illustrated in Figure 1.4b is general for gene expression analysis or discovery of novel gene and alternative splicing.

The raw data resulted from a RNA-seq experiment consists of a list of short sequences,

and the steps for detecting differences in gene expression levels between samples in RNA-seq would then follow the processing pipeline outlined in Oshlack et al. (2010) and illustrated in Figure 1.5. To measure gene expression (or transcript abundance), the sequencing reads obtained are aligned to a known reference genome sequence, and the proportion of reads matching a given transcript is used as quantification of its expression level and followed by statistical testing of difference in quantification values between samples (Oshlack et al, 2010, Bloom et al, 2009).

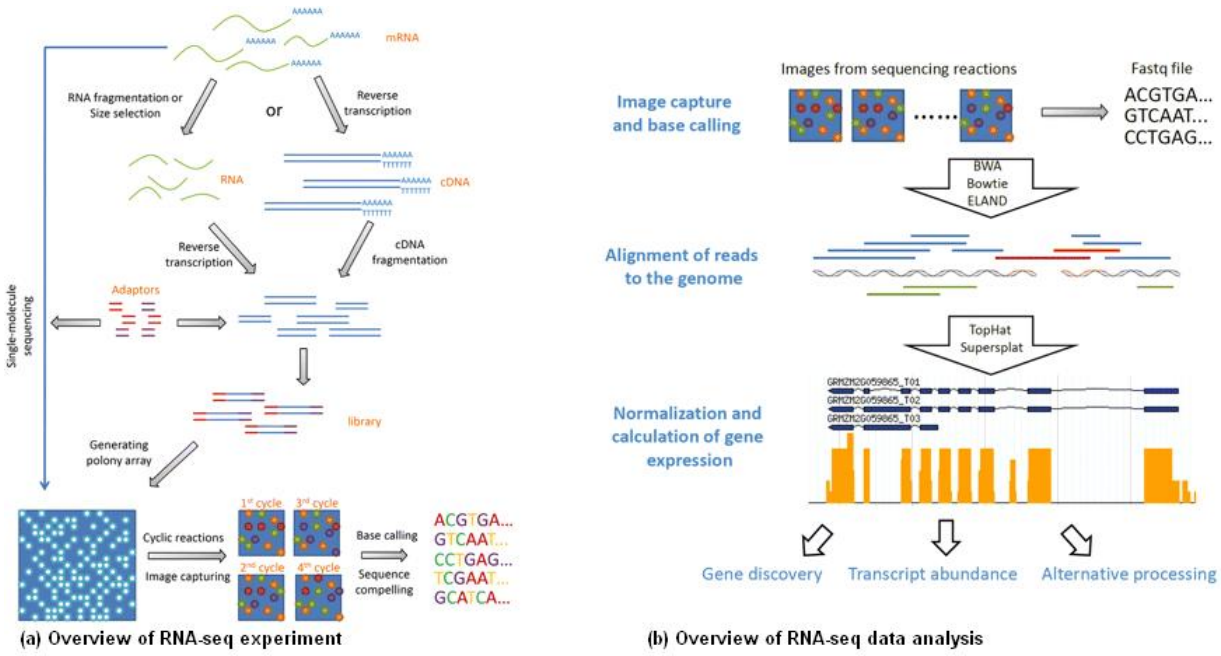

Figure 1.4: Overview of RNA-seq experiment and data analysis process (Wang et al., 2010)

The first step in a typical RNA-seq pipeline for differential expression (DE) analysis is to map the short reads to the reference genome or transcriptome. This step aims to match the short reads to the reference sequence, taking into consideration the sequencing errors and structural variations. Ideally, there would be a unique location where a short read is identical to the reference. However, in practice, the reference is often not a perfect representation of the biological source of RNA being sequenced since it would have sample-specific attributes such

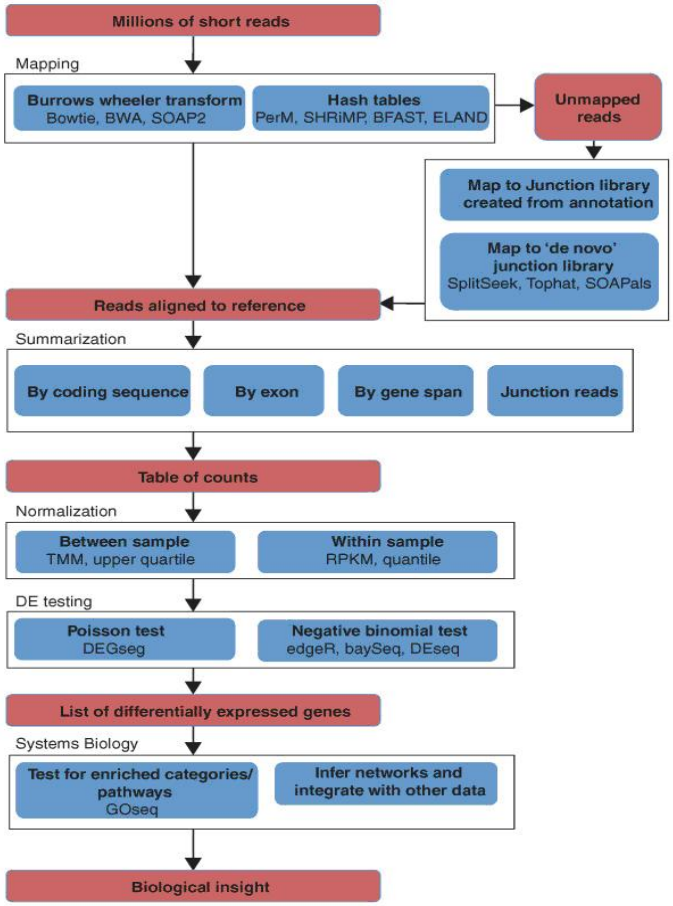

Figure 1.5: Overview of RNA-seq analysis pipeline for detecting differential expression (Oshlack et al., 2010)

as single-nucleotide polymorphisms (SNPs, variations occurring in individual nucleotides) and insertions or deletions (indels). Also, the short reads can sometimes be aligned to multiple locations on the reference and the presence of sequencing errors also needs to be accounted for (Oshlack et al., 2010).

Depending on the aim of the experiment, the mapped reads should be aggregated over some biologically meaningful units so the next step is to summarize the mapped reads for each sample into gene-, exon- or transcript-level summaries. The most common and simplest approach is to count the number of reads overlapping the exons in a gene (Marioni et al., 2008; Mortazavi et al., 2008), but this would exclude the proportion of reads mapped to genomic regions

outside the annotated exons (Oshlack et al., 2010). There are other alternative approaches of summarization such as including reads along the whole gene length to incorporate reads from introns or including only reads that map to coding sequences (Trapnell et al., 2009). With these various options of summarization methods, the count for each gene may change substantially but little research has been done to determine the optimal method for DE analysis.

The follow step involves normalizing the summarized data, which has been shown to be crucial in DE analysis with RNA-seq data (Anders and Huber, 2010; Bullard et al., 2010; Robinson and Oshlack, 2010). Different normalization methods can be used in order for accurate within- and between-sample comparisons of expression levels. To quantify the expression levels of genes within the sample, a widely used approach is to use RPKM (reads per kilobase of exon model per million mapped reads). It is known that longer transcripts are associated with higher read counts at the same expression level, so normalization using RPKM would take into account this gene length effect by dividing the summarized counts by the length of the gene (Marioni et al., 2008; Mortazavi et al., 2008).

When testing for DE in genes between samples, there is no need to worry about technical biases such as gene length and nucleotide composition as they will mainly be cancelled out since the underlying sequence used for summarization is the same between samples (Oshlack et al., 2010). A common method of between-sample normalization is to adjust by the total number of reads in the library (Marioni et al., 2008; Robinson and Smyth, 2007), which accounts for the fact that longer samples being sequenced would associate with more reads. On the other hand, there are other biases that need to be taken into account, such as the composition effects (Bullard et al., 2010), or the issue that DE of genes with low expression levels are more difficult to detect by sequencing than by arrays (Robinson et al., 2010). Some methods that have been proposed to deal with these concerns include using scaling factors within the statistical models that test for DE (Anders and Huber, 2010; Bullard et al., 2010; Robinson and

Oshlack, 2010), performing quantile normalization or employing a method which usess matching power law distributions (Balwierz et al., 2009). However, the latter two methods may not be appropriate for testing DE since these non-linear transformations remove the count nature of the data and that quantile normalization cannot improve DE detection to the same extent as an appropriate scaling factor (Bullard et al., 2010).

The last step of a DE analysis often involves performing statistical testing between samples of interest using the table of summarized count data for each library after normalization. In general, the Poisson distribution may be used to model the RNA-seq count data, but the Poisson assumption does not account for biological variability in the data (Nagalakshmi et al., 2008; Robinson and Smyth, 2007). Ignoring this issue on datasets with biological replicates will result in false positive rates due to underestimation of sampling error (Anders and Huber, 2010). The negative binomial distribution, which requires an additional dispersion parameter to be estimated, is often used to deal with the biological variability in the data. Variations of negative-binomial-based DE analysis of count data have been proposed (Anders and Huber, 2010; Hardcastle and Kelly, 2010; Robinson and Smyth, 2008), along with models which extend the Poisson model by including over-dispersion (Srivastava and Chen, 2010). It is worth noting that these current strategies target data from simple experimental designs. For analysis of more complex designs such as paired samples or time-course experiments, further research is required to develop such methods in the context of RNA-seq data.

1.2.5 Microarrays vs. RNA-seq

Microarrays and sequence-based methods are both often used in gene expression studies, with an increasing popularity of the use of RNA-seq over microarrays in transcriptome analyses. Statistical methods used for DE analysis with these two technologies are different because the array-based technology measures intensities using continuous distributions, whereas RNA-seq

gives discrete measurement of reads for each gene. Although the underlying methods for measuring expression are different, studies have demonstrated that the estimated expression levels have good agreement between the two technologies with correlations ranging from 0.62-0.75 and little variation within methods (Fu et al., 2009; Marioni et al., 2008; Mortazavi et al., 2008).

Transcriptome studies are switching to rely on sequencing-based methods rather than microarrays since RNA-seq has higher sensitivity and dynamic range, with lower technical variation and thus higher precision than microarrays (Bradford et al., 2010; 't Hoen et al., 2008; Oshlack et al., 2010). In comparison to analysis of array data at the same false discovery rate, more differentially expressed genes could be identified by using sequenced data (Marioni et al., 2008). Sequencing-based methods have another advantage of quantifying expression levels in digital, rather than analog, measurements, as the absolute read counts they provide would allow for highly reproducible comparison of transcripts among and within samples or technical replicates (Matukumalli and Schroeder, 2009; Mortazavi et al., 2008).

Microarrays rely on nuclei-acid hybridization and this leads to several limitations, such as cross-hybridization artifacts, dye-based detection issues and other design constraints (Matukumalli and Schroeder, 2009). The probe design of a microarray gene expression assay requires all gene sequences to be known, but the bacterial cloning of the cDNA input for many non-model organisms may not be available for successive designing of the probe (Oshlack et al., 2010). Cross-hybridization of the microarray probes may differ considerably in their hybridization properties, thus affects expression measures in a non-uniform way, and hybridization results from a single sample may not be generalizable to expression of different transcripts (Marioni et al., 2008; Oshlack et al., 2010). The binding affinity constraint also makes it difficult to design reliable probes targeted at specific sequences, causing part of the genome to be inaccessible (Bradford et al., 2010). Hybridization processes are hard to standardize thus interlaboratory comparability is low compared to sequenced data ('t Hoen et al., 2008). Also,

hybridization techniques rely on specific signal-to-noise ratio threshold of the array fluorescence to be established in order to detect rare transcripts, and background hybridization levels especially limit the precision of expression measurements for transcripts which are in low abundance (Marioni et al., 2008; Matukumalli and Schroeder, 2009; Mortazavi et al., 2008).

In regards to the detection of previously unmapped genes and RNA splice events, dense whole-genome tiling microarrays can be used but it is not an attractive method due to the requirement of large amounts of input RNA and other limitations that affect direct splice detection (Mortazavi et al., 2008). On the other hand, sequencing technology is superior over microarrays due to its ability to provide details on novel transcribed regions, alternative splicing and editing of RNA, and allele-specific expression. Microarray probes are only designed to cover small portion of a gene so it is not possible to detect novel transcribed regions, whereas RNA-seq does not rely on pre-determined probes and can be performed on any species lacking both the genome sequence and gene content, allowing it to be used for detecting novel transcription at previously uncharacterized loci (Bradford et al. 2010; Marioni et al., 2008; Matukumalli and Schroeder, 2009; Mortazavi et al., 2008). Sequencing-based methods can not only explore novel gene content, they can also characterize splicing events by capturing reads that span exon-exon junctions, and examine splice variants and rare transcripts (Bradford et al., 2010; Fu et al., 2009; Marioni et al., 2008; Matukumalli and Schroeder, 2009).

However, RNA-seq is not without its limitations and biases, and the main drawback is the presence of gene-length bias. For sequencing-based methods, sequence reads density varies along the length of a transcript and so they have higher statistical power to detect differential expressions for longer transcripts than for shorter transcripts (Marioni et al., 2008; Mortazavi et al., 2008). This association between DE and gene length is not present in microarray data and need to be adjusted for in the RNA-seq analysis. Various sequencing protocols and sequencing errors can introduce other different biases into the resulting reads, such as having a sequence in

its mutated form matching to another existing sequence in the genome, leading to false-positive results in RNA-seq (Mortazavi et al., 2008; Oshlack et al., 2010). Other weaknesses of RNA-seq include ambiguous mapping for paralogous sequences and GC content bias (Bloom et al., 2009; Bullard et al., 2010; Mortazavi et al., 2008). Paralogous genes (diverged genes after a duplication event) often occur in large genomes and the process of mapping reads need to take into account of the multiple matching sites in the genome. The sensitivity and accuracy of sequencing-based methods rely on having significant read coverage over the genome with enough detail for low abundance transcript, but less information is available for genes at a low expression level thus the methods are often biased (Bloom et al., 2009; Bradford et al., 2010; Mortazavi et al., 2008).

Technical constraints, cost, throughput and ease of data analyses are all aspects of genome analyses which need to be well-balanced. For different transcript-profiling platforms, their sensitivity, accuracy and coverage all need to be taken into consideration when trying to select the optimal technology for the problem of interest. A major concern of RNA-seq has been the costs of the experiments. However, the expenses of sequencing methods will continue to decrease as technology improves, with increasing number of reads being generated from a single sequencing run and allowing for multiple samples to be processed simultaneously. RNA-seq will continue to gain strength as a transcriptome profiling tool, for it is a cost-effective approach which brings qualitative and quantitative improvements to gene expression analyses.

1.3 Example datasets

1.3.1 Fibroblast data

Dudley et al. (2002) studied the progressive development of human vertebrate limb through measuring the gene expression of the response of fibroblasts to fetal bovine serum. Microar-

ray experiments have been conducted and the data were analyzed to investigate growth control and cell cycle progression throughout the limb development. The fibroblast responses were measured over time after serum stimulation by using cDNA microarray to obtain gene expression levels. The sample consisted of 8,613 distinct human genes and 14 sampling time points throughout 0 to 24 hours. After some data preprocessing, the final data analyzed consisted of 6,153 genes with six selected sampling time points and data transformation was performed to obtain discrete measures of the relative gene expression levels. Given the dataset from this experiment, cluster analysis can be used to identify the underlying grouping of the gene expression and further describe the effects of fetal bovine serum stimulation on the human vertebrate limb development.

1.3.2 Fruit flies data

The Lawrence Berkeley National Laboratory under the U.S. Department of Energy consists of many different research departments in various divisions. The Genome Dynamics department in the life sciences division uses a variety of technologies to examine the structure and function of genes from numerous organisms. The Drosophila Transcriptome project (Graveley et al., 2011) performed RNA-seq analysis in hope to obtain a comprehensive characterization of the Drosophila melanogaster (fruit fly) transcriptome. In order to examine developmental stages spanning the life cycle of fruit flies, RNA-seq analysis was performed on poly(A)+ RNA from male and female adult fruit flies. Both single- and paired-end sequencing were used on the Illumina GAII platform and Illumina processing pipeline, then the sequencing reads were aligned to the genome sequence using either Tophat (Trapnell et al., 2009) or Bowtie (Langmead et al., 2009). The sample consisted of 542 genes and each of the 12 sampling time point corresponds to two hours. The SAMtools software package (Li et al., 2009) was used to find coverage for each nucleotide position on the genome and then extracted coverage information for the relevant genes based on some gene set information. To account for the different gene lengths (to

avoid gene-length bias), RPKM (reads per kilobase of exon model per million mapped reads) values were calculated and used as the gene expression measure for all genes. A RPKM value for one gene was calculated as the number of mapped reads to the gene divided by the length of transcript in kilobase (transcript length divided by 1000), then divided by the total number of of reads in million. Functional clustering of the dataset with the RPKM values for these genes can be used to identify gene expression patterns and lead to a better understanding of the life cycle of fruit flies.

1.4 Existing methodology for gene expression analysis of RNA-seq data

Gene expression analysis of RNA-seq data often focus on comparing read counts between different biological conditions or genetic variants. The key point in testing for differential expression is to examine whether the observed difference in read count is significant in a given gene. A significant difference is noted if it is greater than what would be expected if it is only due to natural random variation (Anders and Huber, 2010).

The read counts would follow a multinomial distribution if they resulted from reads independently sampled from a population with fixed fractions of genes, thus they can be readily approximated by Poisson distributions. Several researchers have tested for differential expression using Poisson distributions (Bullard et al., 2010; Marioni et al., 2008; Wang et al., 2010). The R package DEGseq (Wang et al., 2010) was developed based on the Poisson model, with Fisher's exact test and likelihood ratio test incorporated into the software. The model assumes that the log-ratios of the different biological samples data have a normal distribution, conditional on the log geometric mean of the data. The software also includes MA-plot-based methods, where MA-plots are statistical tools which are widely used for detecting and visual-

izing intensity-dependent ratio of microarray data. The proposed MA-plot-based methods can be used to identify expression differences when applied to analyze read counts and technical replicates in the data.

However, it has been noted that an over-dispersed model may be more suitable for modeling such gene count data. Nagalakshmi et al. (2008) and Robinson and Smyth (2007) pointed out that the Poisson assumption of equivalent mean and variance ignores the extra variation arises from the differences in replicate samples. Although it has been shown that RNA-seq experiments have low technical background noise (Marioni et al., 2008), the difference between biological replicates exceeds the noise level much more and thus need to be taken into account by the statistical model (Nagalakshmi et al., 2008). When analyzing data with over-dispersion from replicated samples, the tight assumption of equal mean and variance from the Poisson model would result in an underestimation of variations, thus leading to uncontrolled probability of false discoveries (Type I error).

Srivastava and Chen (2010) focus on the number of sequence reads starting from each position of a gene (position-level read counts) and showed that a Poisson model cannot properly explain the non-uniform distribution of these read counts across the same gene. Their approach involves using a two-parameter generalized Poisson model, with one parameter to represent the transcript amount for a gene and another parameter to reflect bias arising from sample preparation and sequencing process. It was noted that the bias parameter is dependent on biological samples but unrelated to library preparations, and the model reduces to a Poisson model when this bias parameter is zero. Normalization of the data was done using a scaling factor which is the ratio of total amount of RNAs between the two samples, and a likelihood ratio test was developed to identify differentially expressed genes.

The Bioconductor software package edgeR (Robinson et al., 2010) has been developed to

examine replicated gene count data using an over-dispersed Poisson model. edgeR performs tests for differential expression for pairwise comparisons and may be applied to count data from different sources other than RNA-seq experiments. With the aim to increase the power to detect differential expression and decrease false discoveries, the statistical model borrows information between probes and uses an empirical Bayes procedure to moderate the degree of over-dispersion across genes (Robinson and Smyth, 2007). The negative binomial parameterization, which essentially corresponds to an over-dispersed Poisson model, is able to separate biological from technical variation. An exact test method has been derived to accommodate over-dispersed data and is used in the software package to assess differential expression in each gene (Robinson and Smyth, 2008).

Anders and Huber (2010) extended the over-dispersed model used in edgeR by modifying the relationship between the mean and variance and developed the Bioconductor package DESeq. Following the notion that data from different genes have similar variability patterns as described by Robinson and Smyth (2007), the model used in DESeq also employs the idea of borrowing information, such as distributional parameters, across genes. For edgeR, the model assumes the mean and variance are related by a single proportionality constant which is the same throughout the data, but the variance assumption in the model used by DESeq incorporates both the raw variance and a noise term. This allows DESeq to be more flexible when encountered with changes in raw squared coefficient of variation (the ratio of the variance to the mean squared) over the large dynamic range in RNA-seq.

The exact test proposed by Robinson and Smyth (2007, 2008) has also been extended by Di et al. (2011) to a test that does not require the constant variation assumption across all genes. Di et al. (2011) noted that the NBP parameterization of the negative binomial distribution allows for a non-constant dispersion parameter within the modelling process, thus accounting for the count variability between biological replicates in RNA-seq experiments. This NBP-based

statistical test has been implemented into the R package NBPSeq for assessing differential gene expression using RNA-seq data.

The statistical approach used by the R package sSeq (Yu et al., 2013) is another method which makes use of the negative binomial distribution for differential expression analysis. The proposed method accounts for over-dispersion in RNA-seq data by incorporating aspects from the different existing methods (edgeR, DESeq) into a simpler model with fewer assumptions. It allows for the differentially expressed genes to have different variances across conditions and estimates dispersions by a shrinkage approach. The advantages of the shrinkage estimation method is that no extra modelling assumptions are required for the differential expression test and it performs well in sensitivity and specificity when sample size is small. Wu et al. (2013) also noted that the assumption of constant dispersion may not be true across all genes, and proposed an empirical Bayes method to shrink the dispersion parameter to better estimate gene-specific dispersion and thus improving the detection of differential genes.

To address the over-dispersion in RNA-seq data, the package BBSeq (Zhou et al., 2011) has been developed for comparing expression levels across different samples using a beta-binomial generalized linear model, which is an extension of a mean-variance modeling approach. For this approach, the observed RNA-seq counts are assumed to be Bernoulli random variables with intrinsic probabilities that are allowed to vary according to a beta distribution. The beta-binomial model is expected to fit similarly as a negative binomial model when library size is large, and it would provide direct interpretation of over-dispersion in the data by describing the unexplained variation in the sequence read probabilities. In this model framework, the dependence of expression on experimental factors or other covariates can be taken into account by using a logistic regression with generic design matrices for flexibility. Length bias in RNA-seq data is assumed to be a constant feature in model since BBseq focuses on comparing expression levels within genes, across experimental conditions.

Since RNA-seq provides quantification of gene expression in read counts, the Poisson distribution and negative binomial models have been used to model the discrete count data and the over-dispersion observed in RNA-seq datasets. However, Esnaola et al. (2013) noted that the negative binomial model may not be adequate when the data consists of the zero-inflation or heavy-tail properties and would lead to erroneous identification of differential genes. To overcome this problem, an analysis approach has been proposed based on the Poisson-Tweedie family of distributions, a more flexible family of count data distributions. The statistical tests based on the Poisson-Tweedie family have been included in the Bioconductor package tweeDEseq as another analysis method for researchers.

Most of the approaches described above have been limited to pairwise comparisons. Recently, a DE analysis pipeline has been implemented as an addition to the Bioconductor edgeR to allow for analyzing RNA-seq data from complex experiments with blocking variables and multiple-treatment comparisons. McCarthy et al. (2012) developed a modelling framework with generalized linear models which would account for the over-dispersion in read counts from multifactor experiments and the algorithm employed an empirical Bayes approach for allowing gene-specific variation to be modelled. Following the idea of sharing information among genes, the statistical method acknowledges gene-specific variation even in situations with only a small number of biological replicates.

In PoissonSeq proposed by Li and others (2012), it uses a log-linear model with a new normalization method to analyze RNA-seq data. This model is equivalent to the ones proposed in Bullard et al. (2010), Marioni et al. (2008) and Wang et al. (2010) when used for a two-class outcome, but allows for the modelling of multiple-class comparisons when needed. The model is first fitted assuming no association between genes and the outcome, and then an additional term is added to the model to accommodate differential expression. A Poisson goodness-of-fit

statistic is used to estimate the set of genes which are not differentially expressed and for updating the iteration estimates of the model.

Hardcastle and Kelly (2010) developed an empirical Bayesian approach which allows for analyzing data from more complex experimental designs. Their baySeq approach assumes a negative binomial distribution for the read counts and uses empirical Bayes methods to examine differential expression patterns within the data. With the goal of increasing the pattern prediction accuracy, the method borrows information across the genes and defines a set of models for patterns of differential expression based on similarity and difference between samples. The prior distribution for each model can be empirically determined from the entire dataset with different prior distributions being assumed for samples behaving differently, and then posterior probabilities for the models are established.

Some evaluations of differential gene expression analysis methods for RNA-seq data have recently been performed. Overall no single analysis algorithm has been found to be favourable across the comparisons, but it was noted that the power of the statistical tests can be improved by increasing the biological replicates (Rapaport et al., 2013; Robles et al., 2012). Robles et al. (2012) evaluated the detection of differential expression using the three packages edgeR, DESeq and NBPSeq through simulations with varying sequencing depth, experimental designs and biological replications and found that DESeq performs more conservatively than the other two algorithms. The statistical tests based on negative binomial distributions (DESeq, edgeR and baySeq) had notably good control of false positive errors with comparable specificity and sensitivity resulted from the tests (Rapaport et al., 2013).

In terms of time-series gene expression analysis using RNA-seq, Oh et al. (2013) have examined and proposed some statistical algorithms to account for the time-dependence structure in the datasets. The two approaches commonly used in time-series data, autoregressive

time-lagged regression models and hidden Markov models, have been suggested as ways for analyzing differential expression in time-course RNA-seq data. When these two approaches are incorporated with the Poisson and negative binomial distributions, they can be applied to identify and infer temporal dynamics from the time-series read counts in RNA-seq experiments. Oh et al. (2013) also proposed the use of statistical evolutionary trajectory index to model the relationship between expression profiles over time. The method consists of computing the autocorrelations of expression profiles across time points and fitting a smooth spline regression to reflect the temporal patterns of gene expressions. This research has been one of the few which focus on temporal dynamics in RNA-seq data and more statistical methods need to be developed to explicitly model temporal RNA-seq data.

1.5 Existing methodology for treatments of missing values

1.5.1 Imputation methods for missing values in gene expression data

Many imputation methods have been proposed for dealing with missing values in gene expression data and reviewed in literature (Aittokallio, 2009; Brock et al., 2008; Hourani and El Emary, 2009; Sahu et al., 2011). The simplest strategy in dealing with missing observations would be to replace missing values with zeros (ZEROimpute), such as in Alizadeh et al. (2000) where missing $\log_2$ transformed gene expression ratios were replaced by zeros. Another simple option would be to impute a missing value using the row average (the mean of intensity of a gene across different experiments), but this approach (ROWimpute) assumes that expression levels of a gene during different experiments (or at different time points in time-course experiments) are constant, which may not be true. These methods ignore any correlation structure present in the data and may reduce the variance of the variables, leading to biased estimates (Schafer and Graham, 2002). Some classic approaches of imputation include hot deck (imputing with a value from a gene with similar expression levels), cold deck (imputing with a value from external data from similar studies), model-based (using a statistical model to predict the

missing values using the non-missing data) and multiple imputation (obtain multiple estimates for each missing value and apply downstream analyses separately for each complete dataset, then combine these multiple inferences to produce the final result).

The KNN imputation (KNNimpute) method, which is an improved hot deck imputation method, was first applied to gene expression data by Troyanskaya et al. (2001). KNNimpute consists of first identifying a set of k predictor genes which have profiles similar to the gene with missing values, with similarity measured using Pearson's correlation or Euclidean distance. From this set of predictor genes, the final estimate of the missing value is obtained as a distance-weighted average over the k genes. The estimation ability of KNNimpute depends on the number of nearest neighbours in KNNimpute and it needs to be determined empirically without any theoretical foundation. KNNimpute performs poorly when k is too large or too small: the choice of a small k may overemphasize a few dominant genes in the estimation process, while a large k may lead to the inclusion of prediction genes that are significantly different from the genes with missing values and thus producing erroneous estimates (Sehgal et al., 2005; Yoon et al., 2007).

The SVD imputation (SVDimpute) is also used in literature. Troyanskaya et al. (2001) used SVD to identify mutually orthogonal expression patterns from the genomic data and linearly combined them to obtain approximate expression of all genes in the dataset. Since SVD requires a complete matrix, missing values are originally imputed with zeros or the row averages. The SVD identified mutually orthogonal patterns are referred to as eigengenes and the k most significant eigengenes are used to estimate the missing values in gene profiles. SVDimpute is not robust to the amount of missingess, as the performance of SVDimpute was found to deteriorate quickly as proportion of missingness in the dataset increases (Troyanskaya et al., 2001).

The multiple imputation idea has been incorporated into some imputation methods for gene expression datasets, for example, the two local approaches: the collateral missing value imputation (CMVE) method and the Gaussian mixture clustering (GMCimpute) method. CMVE generates multiple parallel estimations of missing values to obtain final estimates of missing values (Sehgal et al., 2005). To avoid bias toward any one estimate from the multiple estimation, the final prediction of the missing value is generated using equal weighting to the three estimate matrices. The other method, GMCimpute proposed by Ouyang et al. (2004), assumes microarray data is generated by a Gaussian mixture and uses model averaging to obtain the estimates for missing values. For a specific missing entry, an estimate is made from each of the mixture components, and then using a linear combination of the estimates with mixing proportions (the probabilities that the gene belongs to the components) as the weights to obtain a weighted-average to calculate the final prediction of the missing value. Although the goal of imputation is not to improve clustering but to provide accurate estimates that would prevent biased clustering results, it was shown that *k*-means clustering results can be enhanced by first apply GMCimpute before the clustering procedure (Ouyang et al., 2004).

Two other methods also make use of data clustering technique. The CMI method developed by Zhang et al. (2008) imputes a missing entry with plausible values from genes in the same cluster. It makes use of the *k*-means clustering algorithm with kernel nonparametric regression for filling in the missing values in each cluster. Instead of assuming that a gene belongs to only one cluster at any time, the FCMimpute method (Luo et al., 2005) uses the fuzzy C-means clustering, which is a soft clustering algorithm such that each gene has a weighting associated with its chance of belonging into each cluster.

Most of the generic methods have been developed for static data, and although some have been evaluated on time-series datasets, the methods might not be suitable for time-dependent data. When dealing with gene expression data from time-course experiments, it is important

for statistical models to account for the dependence among values for a gene at different time points. Some imputation approaches have been proposed for missing values in time-series microarray data, including the use of cubic splines (Bar-Joseph et al., 2003), dynamic time warping (Tsiporkova and Boeva, 2007), impulse models (Chechik and Koller, 2009) and autoregressive models (Choong et al., 2009).

Impact of imputation on gene analysis

Most literature on missing data focused on comparing imputation methods using the accuracy measure root mean squared errors (RMSE) between the original values and the imputed missing entries. However, researchers perform genomic studies with the interest on the results from the downstream analyses. As Oh et al. (2011) pointed out, "there is no guarantee that performance evaluations by RMSE measures are consistent with evaluations by biological impacts in downstream analyses, which is the ultimate concern in microarray data analysis", it would be more interesting to know how imputation can affect the performance of the downstream analyses such as identifying differential expressions or gene classification and clustering.

One type of downstream analysis of gene expression data that is commonly performed is cluster analysis. The stability of gene clusters formed by hierarchical clustering can be affected by different imputation methods and the magnitude of the gene misallocations may also depend on the aggregation algorithm used in the hierarchical clustering procedure. The impact of missing values on gene cluster stability is critical even when missing rate is low (de Brevern et al., 2004). It was noted that missing entries should be imputed by some methods since the effect on cluster stability when values are not imputed would be severe compared to when imputed datasets are used. Based on cluster stability, the imputation method KNNimpute is more efficient than the simple ZEROimpute method (de Brevern et al., 2004), but compared to other sophisticated imputation approaches, KNNimpute seems to be less powerful (Celton et al., 2010).

These studies have examined the impact of missingness and imputation methods on downstream analyses with gene expression data. It is of interest to examine the missing value problem in RNA-seq, the newer type of technology that is widely used in genomic studies. To our knowledge, there have not been any studies looking at the effect of missing values on the clustering of RNA-seq data. As previously pointed out, imputation can have a major effect on clustering ability and it would be of interest to examine how it can affect analyses on RNA-seq data, especially in the time-course experimental setting. The results from an evaluation of imputation methods on RNA-seq analyses would give insight to the optimal approach for handling missing values in RNA-seq data.

1.6 Objectives and statement of problems

1.6.1 Functional clustering for time-course genomic data

Data clustering allows for the grouping of similar data points in order to discover and explain relationships among the data. Functional clustering of genomic data can identify co-expressed genes with similar functions and help explain the complexities of biological systems (Eisen et al., 1998). Exploring the patterns shown in genomic data from time-course experiments can provide us with important information on changes in expression levels over time. Some of the major applications of time course genomic experiments include (Androulakis et al., 2007): understanding the dynamics of biological systems such as cell cycles, examining the development of processes such as cell differentiation for organisms, analyzing response dynamics by monitoring how gene expression changes according to varying drug dosages, and studying disease progression over time.

Clustering methods have been widely applied to time-course microarray data to discover

co-expressed genes (Cooke et al., 2011; Grün et al., 2012; Ng et al., 2006; Schliep et al., 2003; Yuan and He, 2008) and described in comprehensive reviews (Androulakis et al., 2007; Bar-Joseph, 2004; Möller-Levet et al., 2003; Wang et al., 2008). However, existing clustering methods used for microarray data are not appropriate for the discrete-type RNA-seq data. We will give a more detailed review on existing clustering methods for microarray data in a later section. Clustering methods for static data have been applied to RNA-seq datasets (Jäger et al. 2011; Pauli et al., 2012) but the approaches ignored the sequential property of time-course data. To the best of our knowledge, statistical methods which thoroughly defines a statistical model for analyzing RNA-seq count data with time-dependence nature and over-dispersion property are very limited. There is a tremendous need for developing novel clustering methods that are suitable for temporal RNA-seq data. In this dissertation, the **first research topic** involves developing an efficient data clustering method to identify patterns on gene expression data from time-course RNA-seq experiments. The goal is to use a model-based clustering approach to identify co-expressed genes and their expression patterns from gene expression levels measured by read counts over time.

1.6.2 Initialization procedures for finite mixture models

Finite mixture models are commonly used in cluster analyses of genomic data and the expectation-maximization (EM) algorithm (Dempster et al., 1977) is often used as the method for maximum likelihood (ML) estimation. The EM algorithm is the most suitable method for parameter estimations in situations with incomplete-data, such as missing data, truncated distributions and censored or grouped observations (McLachlan and Krishnan, 2008). The incompleteness of the data may not be natural or evident, and it would then depend on the statistician to formulate the incompleteness in an appropriate manner to facilitate the application of the EM algorithm. When finite mixture distributions are used to model heterogeneous data, it is a classic example of a problem with incomplete data since the goal is to estimate the proportions in which

the components of the mixture occur along with the component densities parameters. Another application of the EM algorithm is when the likelihood is analytically intractable, then the statistician may simplify the likelihood function by assuming the values of additional parameters as missing data. In this case, the incompleteness of the data is not natural but then it is formulated such that the application of the EM algorithm is appropriate for the optimization of the likelihood function.

The EM algorithm is an iterative procedure where each iteration consists of two steps: the Expectation (E-step) and the Maximization step (M-step). During the E-step, the algorithm finds the expected value of the complete-data log-likelihood with respect to the unknown data, given the observed data and the current parameter estimates. The M-step then consists of maximizing the expected log-likelihood obtained in the first step and update the parameter estimates. Starting from some initial values, the E- and M-steps are repeated until some convergence criterion is satisfied. Each iteration is guaranteed to increase the log-likelihood and thus the algorithm nearly always converges to a local maximum of the ML function (McLachlan and Krishnan, 2008). However, reliable global convergence is not certain and the performance of the EM algorithm can be improved by using good starting values. Different initialization procedures for EM algorithm have been proposed and investigated for ML estimations in Gaussian mixtures models (Biernacki et al., 2003) and also in mixtures of regression models with respect to time-course microarray data in a model-based clustering setting (Scharl et al., 2010). It is of interest to identify a reliable initialization strategy for clusterwise regression specifically for time-course discrete count data from RNA-seq experiments. This leads to the **second research topic** that will be addressed in this dissertation. It can be stated as the following: what is an effective initialization procedure for model-based clustering of time-course RNA-seq data?

1.6.3 Missing value imputation methods for clustering of time-course genomic data

High-throughput analyses such as microarrays and sequencing technologies combined with statistical data analyses provide researchers with the ability to explore and understand complex biological processes. However, technical limitations might lead to the presence of missing values in the data, such as from corrupted spots on microarray through damaged or suspicious spots being filtered during the image analysis phase. Missing value imputation methods have been reviewed and evaluated on their impact on gene expression profiles analyses (e.g. Liew et al., 2010; Oh et al., 2011). Celton et al. (2010) performed an extensive comparison of the effects of imputation methods on cluster analysis of microarray data and noted that data with even a low missing rate would affect gene cluster stability. Noting the difference in data types between microarray and RNA-seq data, there is a need to evaluate imputation methods for the discrete count data, especially in the time-course experiments setting.

Therefore, the **third research topic** that will be investigated in this dissertation is the biological impact of missing value imputation on clustering analyses of genomic data from time-course sequencing experiments. Limited research has been done on the impact of missingness on sequenced data, thus it is desirable to further explore into this area and answer the following questions:

1. Are genomic data produced from next generation sequencing technologies often peppered with missing values?

2. What are the key issues that need to be addressed when dealing with missingness in time-course sequenced data with respect to the time-dependence nature of the data?

3. How can clustering of temporal RNA-seq data be improved by imputation methods when missingness is present?

1.7 Organization of dissertation

The main research topics discussed in this dissertation include data clustering, performance evaluation of initialization strategies, and impact of missing value imputation on time course genomic data. Chapter 2 addresses the first research topic: clustering of time-course gene expression profiles. A detailed review on existing clustering methods, for both static and time-course genomic data, is given. A novel model-based clustering algorithm specific for longitudinal discrete RNA-seq data is proposed and the performance of the algorithm is assessed through simulation study. Results obtained by application of the proposed method are presented to demonstrate its utility in biological research.

The second research topic: initialization procedure for finite mixture models will be addressed in chapter 3. We review the existing initialization strategies suitable for mixture models and investigate their performance with regards to the proposed model-based clustering of mixture of regression models for longitudinal discrete genomic data. The procedures are applied to both synthetic and real gene expression profiles to illustrate their performance. Chapter 4 focuses on the third research topic: missing value imputation methods for cluster analysis of genomic data. Issues regarding missingness in genomic data from sequencing-based technologies are discussed and existing imputation methods for longitudinal data are reviewed. We demonstrate the impact of the different imputation methods on gene clustering of time-course RNA-seq data using simulated and real datasets. Chapter 5 summarizes the original contributions of the dissertation research, discusses future research topics/problems, and provides conclusions for the presented research work.

Chapter 2

Clustering of time-course RNA-seq data

2.1 Introduction

Genomic data from microarray and sequencing experiments such as RNA-seq can be used to describe transcriptional dynamics of a biological system by measuring the levels of gene expression of thousands of genes simultaneously. The genomic experiments can be classified into two types: static and time-series. Static experiments measure gene expression levels from a number of samples at a single time point, for example, comparing gene expression levels in tissue samples taken from individuals with and without a certain disease of interest. In time-series (or temporal or time-course) experiments, gene expression levels from the sample are measured at a number of time points to monitor the change in expression patterns over time.

With time-course expression data, specific biological systems can be monitored over time to understand gene expression response over time (for example, drug dosing over time) and the development of organisms by studying sequences of cell growth and differentiation. It also offers the opportunity to investigate disease progression as gene expression over time may reveal the evolution of pathological conditions. We can use clustering methods based on similarities between the gene expressions to divide the set of genes being studied into smaller sets of genes

with similar expression patterns. By doing this we can discover co-expressed genes and better understand the complexities of organisms. Genes that are co-expressed are often co-regulated, thus clustering assists researchers in identifying regulatory mechanisms of the cells.

Numerous clustering methods have been proposed for both static and time-series microarray data. A majority of the microarray analysis have focused on static data, as described in reviews (Belacel et al., 2006; Chipman et al., 2003; Gollub and Sherlock, 2006). More recently, the detection and analysis of expression in time-course microarray experiments have become more popular because researchers can discover a profoundly different type of information by monitoring the changes in expression levels over time. Here we will give an overview of clustering analysis approaches, then discuss and summarize existing methods developed for clustering time-course microarray data. We propose a model-based clustering method which will account for the time-dependence and over-dispersion properties of time-course RNA-seq data.

2.1.1 Cluster analysis

Data clustering is the search for an optimized grouping of related observations in a dataset based on similarity. Clustering algorithms can be classified in two broad categories: discriminative (or distance/similarity-based) or generative (model-based) approaches. For each category, the clustering methods include hierarchical clustering and partitional clustering (Zhong and Ghosh, 2003). Hierarchical clustering methods start by assuming each data point contributes to its own individual group. Then the method proceeds successively by either merging smaller clusters which are similar or splitting larger clusters, and such algorithms differ from each other in regards to the merging/splitting decision rules they employ. As a result, the hierarchical algorithm will produce a dendrogram which is a tree of clusters illustrating the nested clusters and the relationships of clusters (similarity levels at which groupings change).

On the other hand, partitonal clustering methods partition the data points into a pre-specified K groups according to some partitioning-optimization criterion defined either locally (on a subset of the patterns) or globally (on all of the patterns). These partitional techniques have an advantage over hierarchical techniques when the dataset is large and it would be computationally intensive to construct a dendrogram. However, the choice of the number of desired output clusters is a key design decision when using partitional clustering algorithms.

Discriminative approaches

In discriminative approaches, data are grouped together based on pairwise distance or similarity measures between data points. The goal is to quantify the distance/similarity between pairs of data and to cluster those with measures falling within a certain pre-specified threshold (Androulakis et al., 2007). It is often difficult to define a good similarity measure for a complex data type, as the measure would be very much data-dependent and sometimes require background knowledge on the data. Another disadvantage of discriminative approaches is that the algorithms are usually computationally inefficient since they require the calculation of similarity measure for all pairs of data points (Zhong and Ghosh, 2003).

The function measuring similarity can be defined as pointwise, shape-based, or feature-based similarity measures. The most commonly used straightforward pointwise similarity functions include norm-based distances and correlation metrics such as the Pearson correlation coefficient. These measures are simple to apply and understand, but they may not be appropriate for time-course data since they do not account for the dependency between time points. Information is lost when the algorithm simply consider the pointwise distances at each time point and ignores the temporal content of time-series data. Some clustering algorithms look at the internal structure of time-series data by making use of a shape-based similarity function.

For example, the relative change of gene expression measurement can be characterized by the direction of the change of expression (up and down patterns) over time and it is possible to quantify the similarity between two curves by comparing their slopes or the piecewise linear functions between time points (Möller-Levet et al., 2003). However, the slopes are calculated based on specific time interval of one so this approach does not account for variable time intervals and the temporal order of the slopes.

Another type of similarity measure is based on general features extracted from the data. The gene profiles are first transformed into feature vectors (sequences of events, nominal values or symbols) based on the important aspects of the expression profiles (such as different states or trends) and then analyzed with respect to their similarities to each other. One example of this type of measure is to replace the expression levels by -1,+1, or 0 with respect to three states of gene expression: the gene is under-expressed, over-expressed, or not differentially expressed relative to its baseline measurement (Di Camillo et al., 2005). The advantages of using feature-based measures are that qualitative features of expression profiles could be a more informative proxy for the gene expression information, and that the transformation of expression data into a sequence of symbols effectively reduces the dimension of the time-series data thus making the analysis more robust to noise (Androulakis et al., 2007, Wang et al., 2008). However, discretization/categorization of expression data will inevitably lead to loss of information; for example, some expression patterns might be lost if gene expressions are oversimplified.

Model-based approaches

Model-based clustering approaches assume that data are generated from a finite set of models with specified model types, such as Gaussian or hidden Markov models (HMMs). These algorithms use the data to estimate model parameters of the underlying models by methods like maximum likelihood estimations, and use model selection techniques to obtain the model

structure, such as the number of states in a HMM. The likelihood or posterior probability derived from the model is used in the optimization criterion or merging/splitting decision rule for either partitional or hierarchical algorithms respectively. Often the model-based approaches are preferred over the discriminative techniques because resulting model for each cluster from model-based methods directly characterizes the cluster, thus resulting in better interpretability of the data (Zhong and Ghosh, 2003). Similarities among genes in a give cluster can be easily explained by examining the model corresponding to the cluster.

The model-based clustering approaches are usually based on variants of finite mixture models, where each component probability distribution corresponds to a cluster. The general idea is that each data point can be viewed as arising from a finite number of populations with unknown parameters to be determined by maximum likelihood estimation based on the available data. The data is assumed to follow a set of pre-specified distributions (e.g. Gaussian models for static data or autoregressive models for time-series data), and the emphasis on the specified underlying models make the model-based clustering approaches more robust to noisy data compared to distance-based approaches (Androulakis et al., 2007).

The EM algorithm is the most popular method to be applied to model-based partitional clustering problems. In partitional clustering algorithms, the cluster membership for data points can be treated as missing data and the EM algorithm can estimate the model parameters by maximizing the incomplete-data likelihood. By applying the EM algorithm, the data points can be iteratively partitioned into the different clusters to achieve maximized likelihood. The E-step would consist of computing the cluster membership of each data point and then followed by the estimation of model parameters in the M-step. The EM clustering algorithm can be formulated with a fuzzy/soft clustering nature, which means that the method assigns probabilities of membership in more than one cluster for each data point (i.e. membership of a data point into any cluster may range from zero to one). This is different from hard clustering algorithms,

which allocates each data point into one and only one cluster (i.e. membership of a data point into any cluster can only be zero or one). Soft clustering is favourable for time-course genomic data since gene clusters often overlap and soft clustering algorithms are more robust to noise compared to hard clustering methods (Futschik and Carlisle, 2005).

2.1.2 Clustering of time-course microarray data

There are many types of clustering analyses that can be performed in order to examine coherent patterns seen in time-course gene expression data. The goal of clustering is to group similar data points together to identify genes exhibiting similar responses to signals, that is, the subsets of genes that behave similarly along time under the set of conditions. In this section, we focus on existing clustering algorithms proposed for clustering gene expression monitored over time, with emphasis on mixture-model-based clustering approaches.

In discriminative approaches, clustering algorithms require the use of a similarity-based measure in order to group the similar time-series data together, which would indicate co-expressed genes in gene expression data. However, there is not a clear definition of "similar expression pattern" in literature. Standard approaches using pairwise similarity measures are not appropriate since they disregard temporal information by assuming observations for each gene are independent and identically distributed (Cooke et al., 2011). Time-series similarity measures should be able to account for three basic issues: scaling and shifting of expression levels, unevenly distributed sampling time points, and the shape or internal structure of the time-series measurements which describes the relative change of expression levels over time (Möller-Levet et al., 2003). Clustering algorithms such as k-means clustering and self-organizing maps (SOM) are partitional clustering methods which use distance-based measures to decide the partitions of data points. A major drawback of these clustering algorithms is that by considering each profile as a vector of independent data points, they do not account for

the sampling intervals and the temporal relationship among observations between time points thus leading to skewed results (Bar-Joseph; 2004). Both hierarchical and partitional clustering methods using similarity measures have been proposed and applied in the analysis of time-course microarray data (Das et al., 2009; Eisen et al., 1998; Kim and Kim, 2007; Minas et al., 2011; Tamayo et al., 1999; Tavazoie et al., 1999), and many other methods have been summarized in comprehensive reviews (e.g. Androulakis et al., 2007; Bar-Joseph; 2004; Möller-Levet et al., 2003; Wang et al., 2008).

For model-based clustering methods, the underlying assumption is that the time-course data can be well characterized and represented by parametric models. Several models have been commonly used in regards to time-series data: normal mixture, hidden Markov, autoregressive and splines models. In a normal mixture model-based approach, gene profiles are assumed to arise from a mixture of multivariate normal distributions with different parameterizations. Densities of multivariate normal are computationally tractable and would ensure invariant clustering with regards to shifts in location and scale (Ng et al., 2006). Such an approach have been applied to time-course microarray data (Ghosh and Chinnaiyan, 2002; Yeung et al., 2001), but it is not an effective method for time-series expression measurements since it completely disregards the temporal structure of the data. Another popular model used in model-based clustering approaches is HMM which is a type of stochastic signal model and probabilistic functions of Markov chains. For clustering algorithms of this type on microarray expression data, each gene expression profile is assumed to be generated by a Markov chain with certain probability (Schliep et al., 2003; Yuan and Kendziorski, 2006; Zeng and Garcia-Frias, 2006). A general issue with using HMM for time-series data is that they ignore the length of sampling time intervals thus reducing its effectiveness for data with non-uniform sampling time intervals (Möller-Levet et al., 2003).

An autoregressive (AR) model is often applied for time-series data, as it attempts to find the

relationship between successive data in order to model the change of a current data value from its value from a previous period in time. This model applied to time-series data can capture the ways in which expression measurements depend on one another. An example of a hierarchical clustering approach with the AR model is the microarray data analysis performed by Ramoni et al. (2002), where they applied an AR model of order one which is able to account for time delays to the time-series data. The autoregressive model is limited by the assumption of stationary time-series data (i.e. the data is assumed to be generated by time invariant system). Also, similar to HMMs, the AR model does not consider how samples are distributed in time thus leading to the possibility of missing potentially significant patterns in the data (Möller-Levet et al., 2003).

Mixtures of linear models or of linear mixed models with splines as covariates have also been used to model time-course genomic data (Bar-Joseph et al., 2003; Celeux et al., 2005; Luan and Li., 2003; Ma et al., 2006; Ng et al., 2006). B-splines, representing each point as a linear combination of a set of basis polynomials, with splines coefficients coming from different distributions can be used to model gene expressions in different clusters. As opposed to models used by Celeux et al. (2005) and Luan and Li (2003) which require the independence assumption among all pairs of genes, Ng et al. (2006) employed a random-effects model for clustering of correlated genes by accounting for correlations among gene profiles within a cluster. The use of splines models with properly defined knots is appropriate for handling the temporal structure of time-series data since the models can account for the shape of the expression profiles and the sampling interval lengths (Möller-Levet et al., 2003). Also, the relationship between dependent and independent variables do not need to be defined before model fitting when linear additive models are used (Grün et al., 2012).

One downside of this type of clustering algorithm is that the application of spline representations is questionable for short time-series data since spline models are more suitable for

fitting data with large number of time points (Liu et al., 2005). Since most of the temporal microarray datasets consist of fewer than ten time points (Ernst et al., 2005), splines models might not be the optimal choice for analyzing time-course microarray data. Another issue with using splines model is that the fitting of splines require the pre-specification of the number and length of piecewise segments of the splines. Grün et al. (2012) attempted to solve this problem by proposing a finite mixture of linear additive models with splines as covariates for time-course microarray data. The linear additive model is fitted using regularized estimation as the model coefficients are penalized until the model fit and curve smoothness are in agreement. This model with regularized estimation allows for a data-driven way to estimate the flexibility of the spline functions so a priori specification is not needed and ensures that a suitable model is fitted.

Some other mixture-model-based clustering approaches have also been proposed for time-series genomic data. When it is not desirable to make assumptions about the cluster distributions, a semi-parametric clustering method (Yuan and He, 2008) can be used, where the densities of mixtures are modelled and estimated nonparametrically with unimodal distributions as constraints. Kim et al. (2008) proposed a mathematical model by incorporating Fourier series approximations into a mixture-model-based likelihood function for functional clustering of time-course gene expression data. Cooke et al. (2011) used a mixture model with Gaussian process regression in a Bayesian hierarchical clustering algorithm to analyze time-series microarray data. With the agglomerative hierarchical clustering approach, the algorithm can obtain the optimal number of clusters automatically. This method also explicitly models a proportion of the data as outlier measurements to account for noisy genomic data.

2.1.3 Clustering of time-course RNA-seq data

Since RNA-seq has been widely adopted as an attractive alternative to microarrays for studying gene expression, the development of clustering algorithms suitable for RNA-seq data becomes

an important area of research since they allow for analysis of multiple groups rather than simple two-group analyses. RNA-seq data uses counts of read to quantify gene expression levels and the discrete nature of the data differs from the continuous expression measurements resulting from microarray experiments. The difference in data types makes it problematic to directly apply statistical tools developed for microarrays onto RNA-seq data.

One approach to overcome this issue is to use data transformations. Li et al. (2010) performed log-transformation onto gene expressions from RNA-seq experiment and identified differentially expressed genes using K-means clustering algorithm. Jäger et al. (2011) standardized the RNA-seq count values from their experiment and performed hierarchical clustering on the normalized data to obtain gene groups with similar expression. In another time-course experiment focusing on the early zebrafish development, the RNA-seq data were also normalized and clustered using K-means (Pauli et al., 2012). These heuristic approaches have the advantage of easy implementation; however, they have not been evaluated for RNA-seq data analysis and the employed clustering methods ignore the time-dependence among the time-series data. There are also complications when analyzing transformed count data. Transformation of count data cannot be well approximated by continuous distributions, and it is particularly problematic for data with small sample sizes and lower count ranges (Oshlack et al., 2010). Data with very small counts after transformation are far from normally distributed and count data usually contain a mean-variance relationship that is not addressed by normal-based analyses (McCarthy et al., 2012).

Genome analyses using count models can better distinguish biological from technical variability than analyzing transformed data with the use of continuous distributions (Robinson and Smyth, 2008). Count models have also been shown to have higher power in detecting differential expression than approximate normal models (Robinson and Oshlack, 2010). Following this notion, Si et al. (2014) proposed the use of model-based clustering algorithms with Pois-

son or negative binomial distributions to analyze RNA-seq data and assessed the model-based clustering methods compared to K-means and SOM. To our knowledge, this has been the first statistical research to focus on clustering methodology for RNA-seq data. However, the analysis for time-course RNA-seq has not been explored.

The clustering of time-course data can be viewed as identifying developmental trajectories (or temporal pattern) within a dataset. The semi-parametric group-based trajectory modeling approach (Nagin, 1999; Nagin, 2005) is an example of model-based clustering method for longitudinal data. The method models the data as a mixture of distinct groups/clusters defined by their trajectories and by clustering data with similar trajectory, differences that may explain individual- (or sample-) level variability can be expressed in terms of cluster differences (Nagin, 2005). For analyzing time-course genomic data, the group-based trajectory model can: (1) determine the optimal number of distinct expression patterns and identify those patterns/trajectories, (2) estimate the proportion of samples that is believed to have produced the expression pattern of each group, (3) relate the cluster assignments to covariates of characteristics of the genes, and (4) use the cluster membership probabilities for purposes such as creating summaries of expression patterns of clustered genes.

The model was first applied by Nagin and Land (1993) as a mixed Poisson model to criminal career data, then Roeder et al. (1999) followed the idea with a mixture of zero-inflated Poissons to handle situations where the data contains more zero's than expected from Poisson distributions. The group-based estimation model has then been implemented as a SAS-based procedure, Proc Traj, by Jones, Nagin and Roeder (2001). The procedure uses mixtures of zero-inflated Poissons, censored normals, and Bernoulli models for longitudinal count data, scale data and binary data respectively. Recently, KmL (Genolini and Falissard, 2011), a K-means clustering algorithm for longitudinal data has been proposed and compared to Proc Traj on the modeling of time-course data. While producing similar results as Proc Traj when applied to

real dataset with count data, KmL suffers from the limitation that all clusters are assumed to have the same variance, which would affect its ability to correctly identify cluster if the underlying subpopulations have different variances (Genolini and Falissard, 2011).

The objective of this research is to develop a clustering algorithm suitable for time-course RNA-seq data. Since RNA-seq data suffers from the over-dispersion problem (Nagalakshmi et al.,2008; Robinson and Smyth, 2007), a common approach is to model the count data using negative binomial distributions to accommodate over-dispersion. Here we develop an efficient model-based clustering method with mixtures of negative binomials to cluster time-course RNA-seq data using the semi-parametric group-based modeling approach proposed by Nagin (1999). By identifying the clusters of genes with similar expression patterns, differences that may explain individual-level variability can be expressed in terms of cluster differences. The parameters of this model can be estimated by a direct maximization method, such as the general Quasi-Newton procedure, but the use of this procedure is highly dependent on the starting values (Roeder et al., 1999). To avoid this problem, we present an EM algorithm for maximum likelihood estimation of the parameters in our group-based approach.

2.2 Model

2.2.1 Mixture models

We consider a population consists of g subgroups in some unknown proportions $\pi_1, ..., \pi_g$. We are interested in some random feature $\boldsymbol{Y}$ which is heterogeneous across and homogeneous within the subgroups. Following Frühwirth-Schnatter (2006), we say that $\boldsymbol{Y}$ arises from a finite mixture distribution, with the probability density function $f(\boldsymbol{y})$ taking the mixture density

$$f(\boldsymbol{y}) = \pi_1 f_1(\boldsymbol{y}) + ... + \pi_g f_g(\boldsymbol{y}),$$

where $\pi_i, (i = 1, ..., g)$ represents the mixing proportions (or component weights) with $\pi_i \geq 0, \pi_1 + ... + \pi_g = 1$. Let $Y_1, ..., Y_n$ be a random sample of size n, and $\boldsymbol{y} = (y_1, ..., y_n)^T$ denote the observed random sample where y_j is the realization of the random variable Y_j. Then a standard g-component mixture model can be expressed in the form

$$f(y_j; \boldsymbol{\psi}) = \sum_{i=1}^{g} \pi_i f_i(y_j; \theta_i),$$

where $f_i(y_j; \theta_i)$ is the component density for component i, which is the conditional density function of Y_j given component membership of the i^{th} component with component parameter θ_i, and $\boldsymbol{\psi} = (\pi_1, ..., \pi_g, \theta_1, ..., \theta_g)$ is the set of model parameters from the different mixture components. The corresponding likelihood is given by

$$\begin{aligned} L(\boldsymbol{\psi}) &= \prod_{j=1}^{n} f(y_j; \boldsymbol{\psi}) \\ &= \prod_{j=1}^{n} \sum_{i=1}^{g} \pi_i f_i(y_j; \theta_i). \end{aligned}$$

The mixing proportions $\boldsymbol{\pi} = (\pi_1, ..., \pi_g)$ and the component parameters $\boldsymbol{\theta} = (\boldsymbol{\theta_1}, ..., \boldsymbol{\theta_g})$ are unknown and need to be estimated from the data. We denote $\boldsymbol{S} = (S_1, ..., S_n)^T, S_j \in \{1, ..., g\}$, to indicate the component allocation of individual sample j. If we assume that component allocation is observed, then we can estimate parameters $\boldsymbol{\psi}$ based on the complete data $(\boldsymbol{y}, \boldsymbol{S})$. However, in the clustering context we do not observe the group allocation $\boldsymbol{S}$. We define an unobserved or missing data vector $\boldsymbol{z} = (\boldsymbol{z}_1^T, ..., \boldsymbol{z}_n^T)^T$, where $\boldsymbol{z}_j = (z_{1j}, ..., z_{gj})^T$ is a vector of indicator variables reflecting the component membership of individual sample j. We define $z_{ij} = I(S_j = i)$, indicating that $z_{ij} = 1$ if sample j belongs to component i and $z_{ij} = 0$ otherwise, and we have $\sum_{i=1}^{g} z_{ij} = 1$.

The joint probability function of $\mathbf{Y}_j$ and $\mathbf{z}_j$ is

$$f(\mathbf{y}_j, \mathbf{z}_j) = f(\mathbf{y}_j|z_{ij})f(z_{ij}).$$

We have $Pr(Z_{ij} = 1) = \pi_i$ so the marginal probability function of Z_{ij} is $f(z_{ij}) = \prod_{i=1}^{g} \pi_i^{z_{ij}}$ with $\sum_{i=1}^{g} \pi_i = 1$, $\sum_{i=1}^{g} z_{ij} = 1$, and the density

$$\begin{aligned} f(\mathbf{y}_j, \mathbf{z}_j) &= f(\mathbf{y}_j|z_{ij})f(z_{ij}) \\ &= \prod_{i=1}^{g} f_i(\mathbf{y}_j; \theta_i)^{z_{ij}} \prod_{i=1}^{g} \pi_i^{z_{ij}} \\ &= \prod_{i=1}^{g} \pi_i^{z_{ij}} f_i(\mathbf{y}_j; \theta_i)^{z_{ij}}. \end{aligned}$$

The mixture likelihood is in the form

$$L(\psi) = \prod_{j=1}^{n} \prod_{i=1}^{g} \pi_i^{z_{ij}} f_i(\mathbf{y}_j; \theta_i)^{z_{ij}}$$

and the complete-data log-likelihood can be written as

$$\log L(\psi) = \sum_{i=1}^{g} \sum_{j=1}^{n} z_{ij} \log \pi_i + \sum_{i=1}^{g} \sum_{j=1}^{n} z_{ij} \log f_i(\mathbf{y}_j; \theta_i)$$

2.2.2 Mixture models: Discrete longitudinal count data

When working with time-course RNA-seq data, we denote $Pr(\boldsymbol{Y}_j)$ as the probability of observing a specific time-series sequence of read counts on gene j over time. The goal is to obtain a set of estimates for the parameters such that the likelihood is maximized. These parameters define the shapes of the expression pattern curves for the clusters and the probability of cluster

memberships. The shape of each expression pattern curve (or trajectory) is described by a statistical model, and a separate set of parameters is estimated for each group to allow the shapes of curves to differ across groups.

Let $y_j = (y_{j1}, ..., y_{jm})$ be the observed read counts at m time points for each gene j. Utilizing the flexibility provided by polynomial functions, we assume a quadratic relationship between time and read counts, such that $\mathbf{x}_{jt} = (1, time_{jt}, time_{jt}^2)'$; and conditional on being in cluster i, each gene has independent observations over time. The cluster parameter $\boldsymbol{\theta}_i$ includes $\boldsymbol{\beta}^i$ and s_i, where $\boldsymbol{\beta}^i = (\beta_0^i, \beta_1^i, \beta_2^i)$ determines the shape of the trajectory and the parameter s_i describes the dispersion of the genes in cluster i, and these parameters are allowed to differ across clusters.

We use a negative binomial model for the read counts and conditional on being in group i, a gene j is assumed to have independent read counts over the m sampling time points, so we have

$$f_i(\boldsymbol{y}_j; \boldsymbol{\beta}^i, s_i) = \prod_{t=1}^{m} \left(\frac{\Gamma(y_{jt} + s_i)}{y_{jt}! \Gamma(s_i)} p_i^{s_i} (1 - p_i)^{y_{jt}} \right)$$

with mean

$$\lambda = exp(\boldsymbol{\beta}^i \mathbf{x}_{jt}) = exp(\beta_0^i + \beta_1^i time_{jt} + \beta_2^i time_{jt}^2)$$

and s_i being the dispersion parameter for the group i and probability

$$p_i = \frac{s_i}{s_i + \lambda}.$$

The mixture likelihood for the entire sample of n genes is

$$\begin{aligned} L(\boldsymbol{\psi}) &= \prod_{j=1}^{n} \sum_{i=1}^{g} \pi_i f_i(\boldsymbol{y}_j; \boldsymbol{\beta}^i, s_i) \\ &= \prod_{j=1}^{n} \sum_{i=1}^{g} \pi_i \prod_{t=1}^{m} \left(\frac{\Gamma(y_{jt} + s_i)}{y_{jt}! \Gamma(s_i)} p_i^{s_i} (1 - p_i)^{y_{jt}} \right) \end{aligned}$$

and the corresponding log-likelihood is

$$l(\boldsymbol{\psi}) = \sum_{j=1}^{n} \log \left[\sum_{i=1}^{g} \pi_i \prod_{t=1}^{m} \left(\frac{\Gamma(y_{jt} + s_i)}{y_{jt}!\Gamma(s_i)} p_i^{s_i} (1 - p_i)^{y_{jt}} \right) \right]. \tag{2.1}$$

The maximum likelihood estimates, $\hat{\boldsymbol{\psi}} = (\hat{\pi_1}, ..., \hat{\pi_g}, \hat{\boldsymbol{\beta}^1}, ..., \hat{\boldsymbol{\beta}^g}, \hat{s}_1, ..., \hat{s}_g)$, can be obtained by maximizing the above log-likelihood.

Defining the missing data vector $\boldsymbol{z} = (\boldsymbol{z}_1^T, ..., \boldsymbol{z}_n^T)^T$ with $\boldsymbol{z}_j = (z_{1j}, ..., z_{gj})^T$ reflecting the component membership of gene j, the complete-data likelihood for a sample of n genes can be written as

$$\begin{aligned} L_c(\boldsymbol{\psi}) &= \prod_{j=1}^{n} \prod_{i=1}^{g} \pi_i^{z_{ij}} f_i(\boldsymbol{y_j}; \boldsymbol{\beta}^i, s_i)^{z_{ij}} \\ &= \prod_{j=1}^{n} \prod_{i=1}^{g} \pi_i^{z_{ij}} \left[\prod_{t=1}^{m} \left(\frac{\Gamma(y_{jt} + s_i)}{y_{jt}!\Gamma(s_i)} p_i^{s_i} (1 - p_i)^{y_{jt}} \right) \right]^{z_{ij}}, \end{aligned}$$

and the corresponding complete-data log-likelihood is

$$\begin{aligned} l_c(\psi) &= \sum_{j=1}^{n} \sum_{i=1}^{g} z_{ij} \log \pi_i + \sum_{j=1}^{n} \sum_{i=1}^{g} z_{ij} \log f_i(\boldsymbol{y_j}; \boldsymbol{\beta}^i, s_i) \\ &= \sum_{i=1}^{g} \sum_{j=1}^{n} z_{ij} \log \pi_i \\ &+ \sum_{i=1}^{g} \sum_{j=1}^{n} z_{ij} \left(\sum_{t=1}^{m} \left[\log(\Gamma(y_{jt} + s_i)) - \log(y_{jt}!) - \log(\Gamma(s_i)) + s_i \log(p_i) + y_{jt} \log(1 - p_i) \right] \right). \end{aligned}$$

Since z_{ij} is not observed, the above log-likelihood cannot be directly maximized. The EM algorithm can be used to obtain maximum likelihood estimates, $\hat{\boldsymbol{\psi}}$, from the above complete-data log-likelihood, by treating z_{ij} as missing data.

2.2.3 Estimation

The maximum likelihood estimates of the parameters in the mixture models can be found iteratively by application of the EM algorithm. EM is the preferred approach in situations with incomplete-data problems, such as missing data, truncated distributions and censored or grouped observations (McLachlan and Krishnan, 2008). For mixture models where the likelihood is analytically intractable, the likelihood function can be simplified by assuming the values for additional parameters as missing and the optimization can be achieved using the EM algorithm. In the model-based clustering method, each observation is assumed to have arisen from one of the mixture components and the indicator variable denoting the component membership is taken to be missing. The complete-data log-likelihood is formed on the basis of the sampling distribution of the complete data, regarded as a function of a combination of the missing membership variables and the observed data.

Each iteration of the EM algorithm consists of two steps: the Expectation step (E-step) and the Maximization step (M-step). During the E-step, the algorithm finds the expected value of the complete-data log-likelihood with respect to the missing data, given the observed data and the current parameter estimates. The M-step of the algorithm would then maximize the expected log-likelihood obtained from the E-step and update the parameter estimates. The E- and M-steps are alternated repeatedly until some convergence criteria are satisfied. Each iteration is guaranteed to increase the log-likelihood and thus the algorithm is guaranteed to converge to a local maximum. The justification for this EM property is provided in Section 2.2.4.

For our model-based clustering approach, the EM algorithm is implemented by treating the unknown component membership of the mixture population as missing data, so that the data is augmented with indicators of component membership. In the EM framework, starting from some initial value for $\boldsymbol{\psi}$, say $\hat{\boldsymbol{\psi}}^{(k)}$, the E-step involves the calculation of the expectation of the complete-data log-likelihood, conditional on the observed data and the current estimate

$\hat{\boldsymbol{\psi}}^{(k)}$. Since $\boldsymbol{y}$ and $\hat{\boldsymbol{\psi}}^{(k)}$ are treated as known, the complete-data log-likelihood is linear in the membership variables so the conditional expectation depends only on the expectation of Z_{ij}. The **E-step** in the $(k+1)^{th}$ iteration involves the evaluation of

$$\begin{aligned} E(Z_{ij}|\boldsymbol{y}_j;\boldsymbol{\psi}^{(k)}) &= \frac{\pi_i^{(k)} f_i(\boldsymbol{y}_j;\boldsymbol{\beta}^{i(k)}, s_i)}{f(\boldsymbol{y}_j;\boldsymbol{\psi}_i^{(k)})} \\ &= \frac{\pi_i^{(k)} f_i(\boldsymbol{y}_j;\boldsymbol{\beta}^{i(k)}, s_i)}{\sum_{i=1}^{g} \pi_i^{(k)} f(\boldsymbol{y}_j;\boldsymbol{\beta}^{i(k)}, s_i)} \\ &= \hat{z}_{ij}^{(k)}. \end{aligned} \tag{2.2}$$

This step is simply replacing the missing membership variables by the current values of their conditional expectations, i.e. the resulting estimate is the posterior probability that gene j belongs to cluster i. On the **M-step**, the value of $\boldsymbol{\psi}$ that maximizes the complete-data log-likelihood with each z_{ij} replaced by the corresponding posterior probability is evaluated, and the estimate of $\boldsymbol{\psi}$ is updated by

$$\boldsymbol{\psi}^{(k+1)} = \arg\max_{\psi} E[\log L(\boldsymbol{\psi}|\boldsymbol{y};\boldsymbol{\psi}^{(k)})],$$

which is given by

$$\hat{\pi}_i^{(k+1)} = \frac{1}{n}\sum_{j=1}^{n} \hat{z}_{ij}^{(k)}$$

$$\hat{\boldsymbol{\beta}}^{i(k+1)}, \hat{s}_i^{(k+1)} = \arg\max_{\boldsymbol{\beta}^i, s^i} \sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \left(\sum_{t=1}^{m} \left[\log(\Gamma(y_{jt} + s_i)) - \log(y_{jt}!) - \log(\Gamma(s_i)) + s_i \log(p_i) + y_{jt} \log(1 - p_i) \right] \right). \tag{2.3}$$

Starting from the initial parameter value $\hat{\boldsymbol{\psi}}^{(0)}$, the E- and M-steps are repeated until convergence. There is no closed form solution to the evaluation of $\boldsymbol{\beta}$ and s in the M-step so numerical maximization, such as optimization procedures including Newton-type methods, will be needed. After EM convergence is reached, a probabilistic clustering of the genes into g clusters

are obtained through the posterior probabilities of component membership by assigning gene j to cluster k if $\hat{z}_{kj} = max(\hat{z}_{1j}, ..., \hat{z}_{gj})$.

There are some limitations to the application of EM algorithms. One major drawback is that the covariance matrix of the estimated parameters are not produced as an end-product of the algorithm. Although there are methods for obtaining approximate standard errors from EM algorithms (Louis, 1982; McLachlan and Krishnan, 2008; Meng and Rubin, 1991), they may require extensive derivation or numerical differentiation methods when working with complex likelihood functions. Another issue with the application of EM algorithm is that the speed of convergence can be very slow in some situations, for example, when the proportion of missing data is high. Researchers have developed modified versions of the EM algorithm in attempt to solve these problems. To speed up the estimation procedure, hybrid algorithms such as combining the EM algorithm with Newton-type method have been proposed for application to mixture models (Aitkin and Aitkin, 1996; Redner and Walker, 1984; Wang and Puterman, 1998). Such hybrid algorithm consists of first performing EM algorithm for parameter estimation and then switching to the Newton-type method to speed up convergence. The combination of EM and Newton-type algorithms take advantage of the guaranteed convergence of EM from arbitrary starting values and the fast convergence property of Newton-type method by switching to such a method when it gets closer to the maximum likelihood solution. The approximate standard error of the estimates can be obtained as a by-product of the Newton-type optimization approach. This hybrid estimation approach overcomes the limitations of a) sensitivity to starting values when using Newton-type method for direct maximization, b) slowness of EM convergence, and c) the absence of standard error estimates from EM estimation.

The Quasi-Newton method is one of the Newton-type methods for numerically evaluating

roots of complex functions. The Newton sequence is

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - H^{-1}(\boldsymbol{\theta}^{(k)})S(\boldsymbol{\theta}^{(k)}),$$

where $\boldsymbol{\theta}^{(k)}$ is the ML estimate at the k^{th} iteration, $H^{-1}(\boldsymbol{\theta}^{(k)})$ is the inverse of the Hessian matrix, and S is the score function. The Hessian matrix H is the matrix of the second derivatives of the log-likelihood, $\frac{\partial^2 l}{\partial \boldsymbol{\theta}^2}$, and is the negative of the observed information matrix. The Quasi-Newton method uses an approximation to the Hessian to update the linear iteration sequence so that the calculation of the second order derivatives of the log-likelihood can be avoided by approximating the inverse of the Hessian matrix directly from the first derivative information at each step of the iteration (Nash, 1990 pg 187). This greatly reduces the amount of computation needed to obtain the Hessian matrix and its inverse because the second order derivatives do not need to be calculated, and we can obtain the estimates of the standard errors of the MLE's as a by-product from this optimization process.

We propose a hybrid estimation algorithm for our model-based clustering approach. For a fixed number of components g, we use a combination of the EM algorithm and the Quasi-Newton algorithm to obtain MLE's of parameters in our mixture model. Redner and Walker (1984) noted from their study that 95% of the change in log-likelihood from initial evaluation to the maximum value generally occurred within the first five EM iterations, thus we propose an estimation procedure which starts with running five EM iterations to approach the near-neighbourhood of the ML estimates, and then switches to the Quasi-Newton method for rapid convergence. Following Aitkin and Aitkin (1996), the algorithm returns to the previous EM estimates and run EM for another five iterations if the Hessian obtained after Quasi-Newton estimation is not positive definite.

Algorithm: (EMQN5)

Step 0: Select initial values $\boldsymbol{\psi}^{(0)}$.

Step 1: (E-step) Compute $\hat{\boldsymbol{z}}_j = (\hat{z}_{1j}, ..., \hat{z}_{gj}), (1 \leq j \leq n)$, using 2.2.

Step 2: (M-step) Obtain values of $\hat{\boldsymbol{\beta}}^i$ and $\hat{s}_i$, $i = 1, ..., g$ from 2.3, using the quasi-Newton method.

Step 3: Return to step 1 until five iterations have been performed.

Step 4: Maximize the mixture log-likelihood function $l(\boldsymbol{\psi})$ in 2.1 using the Quasi-Newton algorithm with $\hat{\boldsymbol{\beta}}^i$ and $\hat{s}_i$, $i = 1, ..., g$, as initial values.

Step 5: Stop and obtain MLE's $\hat{\boldsymbol{\psi}}$ if at least one of the following conditions is true; Otherwise, return to step 1 for a further five iterations.

1. Hessian obtained from Quasi-Newton method is positive definite.
2. Estimation procedure has already switched back to EM five times.

2.2.4 The Property of the Proposed Method

Proposition: The proposed method is guaranteed to increase the log-likelihood of the data and it will converge to a locally optimal solution.

The goal of the maximum likelihood estimation is to maximize the log-likelihood of data $\boldsymbol{y}$

$$l(\boldsymbol{y}; \boldsymbol{\psi}) = \sum_{j=1}^{n} \log f(\boldsymbol{y}_j; \boldsymbol{\psi}).$$

In the context of our mixture model of the form, we have

$$f(y_j; \boldsymbol{\psi}) = \sum_{i=1}^{g} \pi_i f_i(\boldsymbol{y}_j; \boldsymbol{\theta}_i),$$

where $f_i(\boldsymbol{y}_j;\boldsymbol{\theta}_i)$ is the component density for component i, and $\boldsymbol{\psi} = (\pi_1, ..., \pi_g, \theta_1, ..., \theta_g)$ is the set of model parameters from the different mixture components. We define an unobserved or missing data vector $\boldsymbol{z} = (\boldsymbol{z}_1^T, ..., \boldsymbol{z}_n^T)^T$, where $\boldsymbol{z}_j = (z_{1j}, ..., z_{gj})^T$ is a vector of indicator variables reflecting the component membership of individual sample j. The parameter $z_{ij} = I(S_j = i)$, indicating that $z_{ij} = 1$ if sample j belongs to component i and $z_{ij} = 0$ otherwise, and $\sum_{i=1}^{g} z_{ij} = 1$. The complete-data log-likelihood can be written as

$$\log L(\boldsymbol{y}, \boldsymbol{z}; \boldsymbol{\psi}) = l(\boldsymbol{y}, \boldsymbol{z}; \boldsymbol{\psi}) = \sum_{i=1}^{g}\sum_{j=1}^{n} z_{ij} \log(\pi_i f_i(\boldsymbol{y}_j;\boldsymbol{\theta}_i))$$

The E-step of the k^{th} iteration of the EM algorithm evaluates the expected complete-data log-likelihood

$$E[l(\boldsymbol{y}, \boldsymbol{z}; \boldsymbol{\psi})|\boldsymbol{y}, \boldsymbol{\psi}^{(k)}] = \sum_{i=1}^{g}\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \log(\pi_i f_i(\boldsymbol{y}_j;\boldsymbol{\theta}_i)).$$

This expectation is over the component memberships given the data $\boldsymbol{y}$ and the current parameters $\boldsymbol{\psi}^{(k)}$ of mixture distribution so that

$$\begin{aligned} E(Z_{ij}|\boldsymbol{y}_j, \boldsymbol{\psi}^{(k)}) &= \frac{\pi_i^{(k)} f_i(\boldsymbol{y}_j;\boldsymbol{\psi}_i^{(k)})}{f(\boldsymbol{y}_j;\boldsymbol{\psi}_i^{(k)})} \\ &= P(Z_{ij} = 1|\boldsymbol{y}_j, \boldsymbol{\psi}^{(k)}) \\ &= \hat{z}_{ij}^{(k)}, \end{aligned}$$

which is the posterior membership generated from current parameters $\boldsymbol{\psi}^{(k)}$. Once we have the posterior memberships, they do not change as a function of $\boldsymbol{\theta}$. This implies that we changed from an incomplete data problem to a complete data problem, so the M-step of the algorithm would maximize over the expected log-likelihood with respect to $\boldsymbol{\theta}$ while keeping other parameters fixed.

Let $\boldsymbol{q} = \{q_{ij}\}$ be any set of distributions over the underlying component memberships, not

necessarily the posterior memberships $\hat{z}_{ij}^{(k)}$, the auxiliary objective the EM algorithm uses is

$$l(\boldsymbol{y}, \boldsymbol{q}; \boldsymbol{\psi}) = \sum_{i=1}^{g} \sum_{j=1}^{n} q_{ij} \log(\pi_i f_i(\boldsymbol{y}_j; \boldsymbol{\theta}_i)) + \sum_{j=1}^{n} H(\boldsymbol{q}_j)$$

where $H(\boldsymbol{q}_j) = -\sum_{i=1}^{g} q_{ij} \log q_{ij}$ is the entropy of membership distribution $\boldsymbol{q}_j$.

With $\boldsymbol{\psi}^{(k)}$ as the current setting of the parameters and letting $\boldsymbol{q}^{(k)}$ be the membership distributions that are really the posteriors, i.e. $\hat{q}_{ij}^{(k)} = \hat{z}_{ij}^{(k)}$, the EM is now:

- E-step: $\boldsymbol{q}^{(k)} = argmax_q \quad l(\boldsymbol{y}, \boldsymbol{q}; \boldsymbol{\psi}^{(k-1)})$

- M-step: $\boldsymbol{\psi}^{(k)} = argmax_\psi \quad l(\boldsymbol{y}, \boldsymbol{q}^{(k)}; \boldsymbol{\psi})$

The E-step generates the posterior memberships, $\hat{q}_{ij}^{(k)} = \hat{z}_{ij}^{(k)}$, and the M-step would maximize

$$l(\boldsymbol{y}, \boldsymbol{q}^{(k)}; \boldsymbol{\psi}) = \sum_{i=1}^{g} \sum_{j=1}^{n} \hat{z}_{ij} \log(\pi_i f_i(\boldsymbol{y}_j; \boldsymbol{\theta}_i)) + \sum_{j=1}^{n} H(\boldsymbol{z}_j^{(k)}).$$

This maximization is the same as before since the entropy term is fixed when we are optimizing $\boldsymbol{\psi}$. With each of these steps being a maximization step, $l(\boldsymbol{y}, \boldsymbol{q}^{(k)}; \psi)$ has to increase monotonically. We can show that the auxiliary objective equals the log-likelihood after any E-step, i.e.

$$l(\boldsymbol{y}, \boldsymbol{q}^{(k)}; \boldsymbol{\psi}^{(k)}) = l(\boldsymbol{y}; \boldsymbol{\psi}^{(k)}).$$

$$
\begin{aligned}
l(\boldsymbol{y}, \boldsymbol{q}^{(k)}; \boldsymbol{\psi}^{(k)}) &= \sum_{i=1}^{g}\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \log(\pi_i^{(k)} f_i(\boldsymbol{y}_j; \boldsymbol{\theta}_i)) + \sum_{j=1}^{n} H(\boldsymbol{z}_j^{(k)}) \\
&= \sum_{i=1}^{g}\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \log(\pi_i^{(k)} f_i(\boldsymbol{y}_j; \boldsymbol{\theta}_i)) - \sum_{j=1}^{n}\sum_{i=1}^{g} \hat{z}_{ij}^{(k)} \log \hat{z}_{ij}^{(k)} \\
&= \sum_{i=1}^{g}\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \log \frac{\pi_i^{(k)} f_i(\boldsymbol{y}_j; \boldsymbol{\theta}_i)}{\hat{z}_{ij}^{(k)}} \\
&= \sum_{i=1}^{g}\sum_{j=1}^{n} P(Z_{ij}|\boldsymbol{y}_j, \boldsymbol{\psi}^{(k)}) \log \frac{\pi_i^{(k)} f_i(\boldsymbol{y}_j; \boldsymbol{\theta}_i)}{P(Z_{ij}|\boldsymbol{y}_j, \boldsymbol{\psi}^{(k)})} \\
&= \sum_{i=1}^{g}\sum_{j=1}^{n} P(Z_{ij}|\boldsymbol{y}_j, \boldsymbol{\psi}^{(k)}) \log f(\boldsymbol{y}_j; \boldsymbol{\psi}^{(k)}) \\
&= \sum_{j=1}^{n} \log f(\boldsymbol{y}_j; \boldsymbol{\psi}^{(k)}) \\
&= l(\boldsymbol{y}; \boldsymbol{\psi}^{(k)})
\end{aligned}
$$

We can now state that

$$
l(\boldsymbol{y}; \boldsymbol{\psi}^{(k)}) = l(\boldsymbol{y}, \boldsymbol{q}^{(k)}; \boldsymbol{\psi}^{(k)}) \overset{\text{M-step}}{\leq} l(\boldsymbol{y}, \boldsymbol{q}^{(k)}; \boldsymbol{\psi}^{(k+1)}) \overset{\text{E-step}}{\leq} l(\boldsymbol{y}, \boldsymbol{q}^{(k+1)}; \boldsymbol{\psi}^{(k+1)}) = l(\boldsymbol{y}; \boldsymbol{\psi}^{(k+1)}) \overset{\text{M-step}}{\leq} \cdots
$$

which shows that the EM algorithm increases the log-likelihood at each iteration and equality of the above would hold only at convergence.

2.3 Simulation study

We conducted simulation studies to assess the performance of the model-based clustering approach with different estimation methods. The group-based semi-parametric model was proposed for identifying trajectories using the EM algorithm to fit mixtures of negative binomial distributions to time-course count data. We also proposed the use of a hybrid EM/quasi-Newton method (EMQN5) to speed up the EM convergence. We performed simulation studies to investigate the properties of EM-based algorithms under a variety of parameter combinations in mixtures with two and four components.

2.3.1 Data generation

Simulations were designed to compare three estimation algorithms when the sample of genes consists of two mixture components. The values of dispersion parameter and trajectory shape parameters were varied to create different simulation settings, and 100 datasets were independently simulated for each setting. We have considered the combinations of trajectories as shown in Figure 2.1, with sampling time plotted against the read counts (measuring expression levels) for genes. For each combination of trajectories, the corresponding parameter values for the trajectories are displayed and we simulated data for each combination with the dispersion parameters being 0.5, 2, 5, and 10 to obtain a total of 16 simulation settings. We generated data involving five sampling time points ($time_{j1} = 1, ..., time_{j5} = 5$), and assumed that the gene counts (y_{jt}) were independent across time. For each dataset, we simulated 300 genes from a mixture with mixing proportions $\pi_1 = 0.3$ and $\pi_2 = 0.7$ (leading to roughly 90 genes in cluster one and 210 genes in cluster two).

2.3.2 Estimation algorithms

For the two-group model, we have the complete-data log-likelihood as:

$$\begin{aligned} l_c(\boldsymbol{\psi}) &= \sum_{j=1}^{n}\Big[z_j \log\pi + (1-z_j)\log(1-\pi)\Big] + \sum_{j=1}^{n}\Big[z_j \log f_1(y_j;\boldsymbol{\beta}^1, s_1) + (1-z_j)\log f_2(y_j;\boldsymbol{\beta}^2, s_2)\Big] \\ &= \mathbf{z}\log\pi + (1-\mathbf{z})\log(1-\pi) + \mathbf{z}\log f_1(y_j;\boldsymbol{\beta}^1) + (1-\mathbf{z})\log f_2(y_j;\boldsymbol{\beta}^2) \end{aligned}$$

with $z_j = z_{1j}$ and $\pi = \pi_1$ (we can make this simplification since we have $z_{1j} + z_{2j} = 1$ and $\pi_1 + \pi_2 = 1$).

We considered three different estimation algorithm for our model-based clustering approach:

1. EM: Given specific initial values, the estimation was performed using the EM algorithm

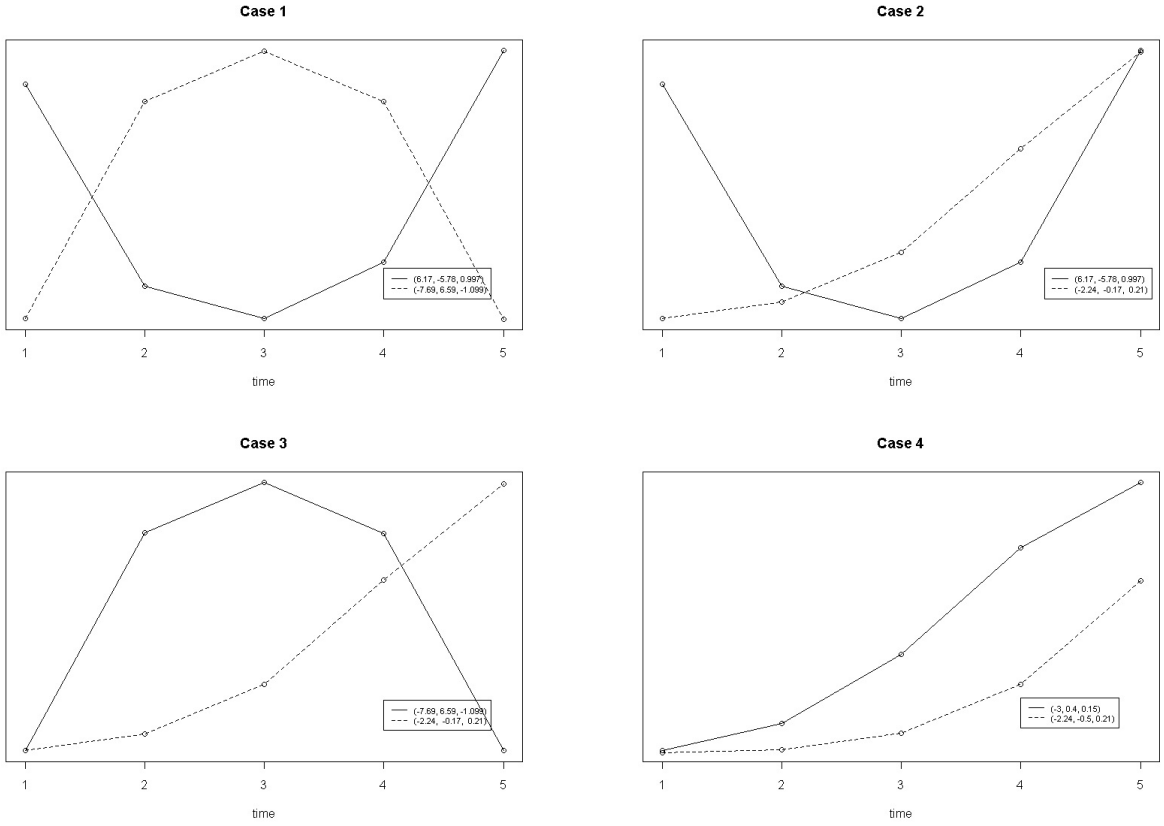

Figure 2.1: Simulation cases: Combinations of trajectories.

with stopping criterion being when log-likelihood stops increasing (a difference in successive values of log-likelihoods of 10^{-4}) or if it reaches a specified maximum number of 1000 iterations.

2. EMQN5: Given specific initial values, the algorithm will perform five EM iterations and then switch to the quasi-Newton method implemented using *nlminb* function in R. It will switch back to EM if the Hessian matrix obtained is not positive definite. (Described in Section 2.5)

3. EMQN10: Given specific initial values, the algorithm will perform ten EM iterations and then switch to the quasi-Newton method (similar to EMQN5 with ten EM iterations instead of five). It will switch back to EM if the Hessian matrix obtained is not positive definite.

To analyze each case of the simulated data, the same set of initial values was used for the different algorithms for a fair comparison and to assess their sensitivity to starting values. We specified the starting values for $\boldsymbol{\beta}$ and π to reflect equal mixing of two very similar trajectories: $\beta^1 = (2, -0.5, 0)$, $\beta^2 = (2, -0.48, 0)$, $\pi_1 = 0.5$, and $s_1 = s_2 = 0.1$. Lower and upper bounds were specified for the parameter estimates in the quasi-Newton estimation procedure (see Table 2.1).

Parameter	Lower bound	Upper bound
β^i	(-50,-50,-50)	(50,50,50)
s_i	0.1	20
π	0	1

Table 2.1: Lower and upper bounds specified for quasi-Newton estimation.

2.3.3 Results

Two-component mixtures

In general, the problem of identifiability of mixture models affected all three estimation methods that we have investigated. In situations with non-identifiable models in some of the simulation datasets, the three algorithms were unable to accurately estimate the trajectory models. These situations were indicated by the undefined standard error estimates of the parameters and we have excluded these datasets from our simulation results. For some of the datasets, the estimated dispersion $\hat{s}$ is close to or equal to the pre-specified upper bound of the parameter estimation and we have also excluded those datasets (with $\hat{s}$ above the threshold of 19.5) from the results. Table 2.2 shows the number of datasets included in the analysis for each case and it is noted that simulation cases with dispersion parameter equals to 10 have the largest number of excluded datasets, especially in case 4 with only 48 datasets included in the results from the EMQN10 algorithm. For result analysis, we have reported the mean parameter estimates (averaged over the included datasets for the specified settings) and the relative bias of mean

estimates is used as an accuracy measure for our parameter estimates, which is calculated by

$$\text{relative bias} = \frac{|\text{mean estimate - true value}|}{\text{true value}} \text{ x } 100.$$

S	Method	Case 1	Case 2	Case 3	Case 4
10	EM	98	97	94	59
	EMQN5	99	76	76	72
	EMQN10	97	76	76	48
5	EM	100	100	100	74
	EMQN5	100	100	100	100
	EMQN10	100	100	100	100
2	EM	100	100	100	94
	EMQN5	100	100	100	100
	EMQN10	100	100	100	100
0.5	EM	100	100	100	72
	EMQN5	100	100	100	100
	EMQN10	100	100	100	80

Table 2.2: Number of datasets included in analysis.

Table 2.3 displays the relative bias of the mean estimated mixing proportions $\hat{\pi}$ for the different estimation methods across the different simulation settings. There is no obvious trend in the estimation precision in terms of change in dispersion parameter from $s = 10$ to $s = 0.5$. The performances of algorithms seem to be the worst for the datasets with lowest dispersion across all four cases of trajectory combinations. When we focus on the relative bias of estimates for datasets with $s = 0.5$, EM algorithm produced the highest relative bias in three of the cases compared to the other two hybrid algorithms. Comparing across the different cases of trajectory combinations displayed in Figure 2.1, the clusters in case 1 are the easiest to classify because of the very distinct trajectory curves and the trajectories in case 4 are the hardest to distinguish. This means it would be the most difficult to correctly classify genes into the two clusters in case 4, and as expected, the algorithms had the highest relative bias in estimating the mixing proportions for the case 4 datasets compared to the other cases. The high relative bias produced by EM algorithm with dispersion $s = 0.5$ in case 4 might be a concern since it

is much larger than the two relative biases produced by the other algorithms in this situation. Overall, the performances for all three estimating algorithms are comparable with no one algorithms appeared to be superior over the others. The differences between mixing proportion estimates produced by EMQN5 and EMQN10 are not striking, but EMQN5 resulted in lower relative biases than EMQN10 in most of the simulation settings.

S	Method	Case 1	Case 2	Case 3	Case 4
10	EM	0.05	0.28	0.08	1.21
	EMQN5	0.01	2.53	0.70	0.17
	EMQN10	0.01	1.60	0.53	0.79
5	EM	0.02	0.48	0.09	0.86
	EMQN5	0.13	2.47	0.03	2.46
	EMQN10	0.13	2.94	0.43	3.10
2	EM	0.05	0.27	0.33	3.67
	EMQN5	0.19	2.08	1.71	4.04
	EMQN10	0.23	5.88	0.01	5.30
0.5	EM	1.78	5.49	0.45	9.66
	EMQN5	0.44	1.31	2.16	3.85
	EMQN10	1.77	3.02	2.21	1.18

Table 2.3: Relative bias of mean estimated mixing proportions.

Method	β^1			β^2			s	s_1	s_2
	6.17	-5.78	0.99	-7.69	6.59	-1.10			
EM	0.52	0.77	0.79	0.29	0.26	0.29	10	9.99	1.02
EMQN5	0.01	0.02	0.02	0.13	0.01	0.09		4.33	5.63
EMQN10	0.11	0.24	0.26	0.82	0.64	0.62		6.02	2.66
EM	0.49	0.53	0.49	0.61	0.48	0.46	5	5.91	2.84
EMQN5	0.20	0.51	0.54	3.23	2.94	3.20		11.00	0.87
EMQN10	0.13	0.04	0.03	0.12	0.03	0.02		7.69	2.04
EM	0.57	0.71	0.68	0.26	0.34	0.38	2	1.44	1.60
EMQN5	0.19	0.94	1.22	1.11	0.65	0.47		1.90	0.31
EMQN10	0.27	0.06	0.03	0.09	0.02	0.02		2.92	1.96
EM	1.07	0.72	0.38	1.39	1.77	1.83	0.5	8.94	3.87
EMQN5	3.96	2.46	2.26	1.99	1.17	1.36		1.81	1.73
EMQN10	0.23	0.03	0.08	0.68	0.59	0.59		2.26	2.07

Table 2.4: Relative bias of mean parameter estimates for Case 1.

Method	β^1			β^2			s	s_1	s_2
	6.17	-5.78	0.99	-2.24	-0.17	0.21			
EM	0.82	0.77	0.71	1.59	3.95	0.09	10	5.61	5.62
EMQN5	1.41	0.94	0.93	3.84	34.80	3.97		2.80	12.46
EMQN10	1.22	0.66	0.67	2.89	31.84	3.93		5.01	13.91
EM	0.70	0.74	0.66	5.14	35.57	3.55	5	10.07	2.74
EMQN5	3.14	2.12	1.79	2.90	24.83	2.93		2.11	0.46
EMQN10	3.78	1.59	1.14	8.17	74.66	9.03		7.55	4.51
EM	0.36	0.19	0.07	4.94	38.27	4.09	2	6.04	1.58
EMQN5	0.48	0.66	0.91	6.80	57.72	6.49		8.19	8.28
EMQN10	2.33	2.36	2.07	1.87	13.97	1.27		17.35	9.18
EM	3.98	2.42	1.82	0.37	11.25	1.42	0.5	4.17	2.02
EMQN5	5.80	6.59	5.60	8.00	65.85	8.14		7.06	3.32
EMQN10	6.62	6.26	5.08	12.83	112.03	13.55		13.56	3.34

Table 2.5: Relative bias of mean parameter estimates for Case 2.

The mean relative bias of the parameter estimates for cases 1 to 4 are displayed in Tables 2.4 to 2.7 respectively. As a general trend, the EM algorithm produced lower relative biases than those from the hybrid algorithms for the parameter estimates in most of the simulation settings. Focusing on the algorithms' performances for case 1 (Table 2.4), it is noticeable that EMQN10 performed well for all the simulations with low relative bias for the β parameter estimates. The other two methods had comparable estimation performances with some difficulties in estimating the dispersion parameters correctly, with EM having high relative bias for s_1 estimates when $s = 10$ and $s = 0.5$ and EMQN5 having the highest relative bias when estimating $s_1 = 5$. The trajectories in case 2 are more difficult to estimate than those simulated in the case 1 datasets since the trajectories have very similar shape from time points 2 to 5 (see Figure 2.1), thus as shown in Table 2.5, the resulting parameter estimates from case 2 simulations have higher relative biases than those from case 1. The algorithms struggled to obtain good estimates for the trajectory with an increasing trend ($\boldsymbol{\beta}^2$), and the linear coefficient $\beta_2^2 = -0.17$ was the hardest to estimate as all three algorithms produced high relative bias for this particular parameter estimation across the different dispersion parameter settings.

Method	β^1			β^2			s	s_1	s_2
	-7.69	6.59	-1.10	-2.24	-0.17	0.21			
EM	0.36	0.23	0.20	0.30	6.92	0.97	10	12.23	5.54
EMQN5	0.86	0.81	0.77	6.37	37.98	3.56		5.60	6.00
EMQN10	3.12	2.54	2.57	7.04	46.94	4.74		28.52	16.56
EM	0.56	0.59	0.76	1.45	12.17	1.40	5	4.20	1.36
EMQN5	1.43	0.84	0.61	2.31	19.64	2.42		8.04	0.89
EMQN10	3.29	2.27	2.40	2.36	0.85	1.37		7.76	0.16
EM	1.37	1.15	1.17	2.89	22.02	2.39	2	3.41	1.62
EMQN5	4.71	3.91	3.93	2.30	5.14	0.07		15.14	3.86
EMQN10	4.71	3.91	3.93	2.30	5.14	0.07		15.14	3.86
EM	1.65	1.16	1.30	0.67	1.45	0.68	0.5	4.68	1.60
EMQN5	3.54	2.67	2.29	0.81	36.51	6.58		5.90	1.34
EMQN10	3.80	3.05	2.76	0.50	48.62	8.76		6.61	1.22

Table 2.6: Relative bias of mean parameter estimates for Case 3.

For simulations in both case 2 and case 3, the EM algorithm produced the lowest relative biases for the parameter estimates compared to the two hybrid methods, and EMQN5 resulted in lower relative biases than EMQN10 in most of the simulations. Table 2.6 shows the relative bias of parameter estimates in case 3 simulations and it is interesting to note that for simulations with $s = 2$, the two hybrid estimation algorithms produced the exactly the same results. This might have resulted from the EM estimation reaching a local maximum after 5 EM iterations and thus even after 10 EM iterations, it still remained in the same parameter neighbourhood and so the quasi-Newton procedures in the two algorithms obtained the same estimation results. Going beyond 10 EM iterations, the EM algorithm might have been able to escape from the local maximum point and so the EM algorithm obtained different parameter estimates compared to the two hybrid methods. However, since there is no guarantee for the EM algorithm to reach the global maximum point, the parameter estimates did not necessarily had lower relative bias than those produced by the hybrid methods.

Classification of genes and parameter estimations should be the most difficult in simulation datasets with case 4 trajectory combination since the trajectories have very similar shapes.

Method	β^1			β^2			s	s_1	s_2
	-2.24	-0.50	0.21	-3.00	0.40	0.15			
EM	0.60	2.84	1.34	1.06	5.33	1.90	10	27.73	5.45
EMQN5	7.69	21.73	3.26	17.96	74.86	26.04		8.40	3.92
EMQN10	10.80	18.40	5.16	8.95	28.61	7.88		2.39	7.31
EM	7.85	18.16	4.84	1.33	5.64	1.88	5	19.10	3.63
EMQN5	1.74	25.89	1.93	4.72	19.32	6.47		18.26	0.99
EMQN10	14.53	33.26	8.72	6.91	24.90	7.88		9.24	2.17
EM	8.21	15.45	2.17	0.65	2.97	1.23	2	36.95	3.91
EMQN5	0.45	24.31	2.40	22.75	97.52	33.72		21.45	5.10
EMQN10	26.82	58.47	6.07	4.49	11.83	1.68		21.30	8.11
EM	23.51	50.11	5.48	0.51	10.85	3.41	0.5	40.71	0.92
EMQN5	14.53	83.50	18.32	9.76	43.06	17.24		18.66	5.12
EMQN10	25.90	50.79	12.74	0.43	2.66	1.47		77.73	2.16

Table 2.7: Relative bias of mean parameter estimates for Case 4.

As shown in Table 2.7, the algorithms generally produced estimates with the highest relative biases out of the four trajectory combination cases. Across the different dispersion parameter settings, the EM algorithm performed well when estimating the β parameters but produced higher relative biases than the two hybrid methods when estimating the dispersion parameter s_1. EMQN5 and EMQN10 had comparable performance, with EMQN5 having more difficulty with estimating trajectory 2 (with $\boldsymbol{\beta}^2$) in some simulations and EMQN10 produced higher relative biases when estimating trajectory 1 (with $\boldsymbol{\beta}^1$) in some other simulations. Noting the high relative biases produced for the parameter estimates in the case 4 simulation datasets, it might be a concern as to how different the estimated trajectories are compared to the true trajectories. Figure 2.2 shows the true and estimated trajectories for case 4 with dispersion parameter $s = 2$. This is one of the simulation settings where EMQN5 and EMQN10 produced high relative bias for parameter estimates (such as 97.52 and 58.47). We can see from the plot that even with such high relative biases, the resulting estimated trajectories are not far from the true trajectories. This shows that all three of our estimation algorithms were able to identify the underlying true trajectories or gene expression patterns in each of the simulation settings since even estimations with such high relative biases would result in estimated trajectories very close

to the true patterns.

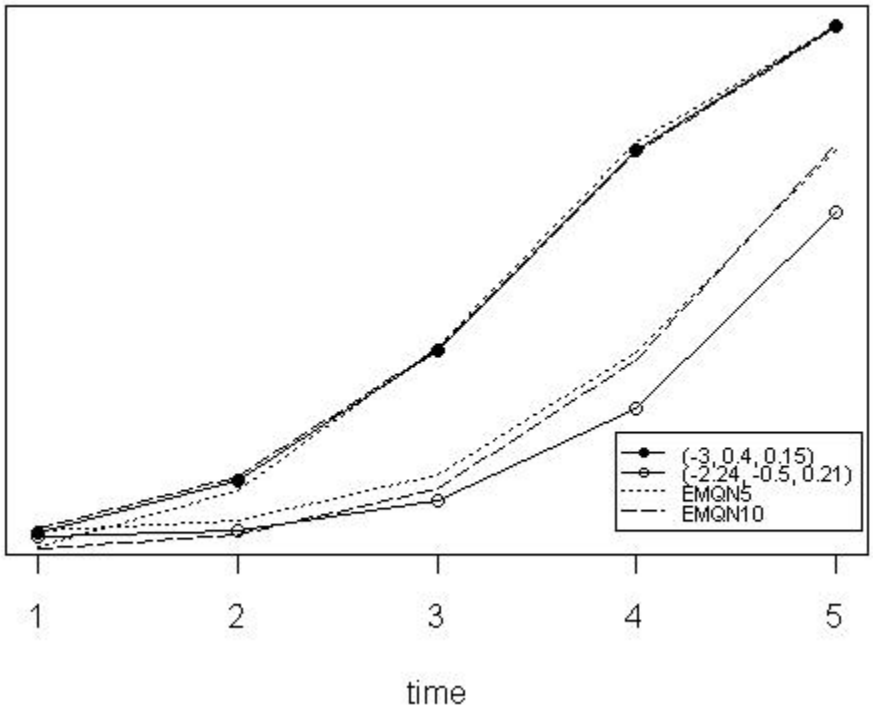

Figure 2.2: True and estimated trajectories for Case 4 ($s = 2$).

Four-component mixtures

In realistic problems, the data often involves more than two clusters and complexity of mixture models increases as the number of components increases. To further examine our model-based clustering approach, we have evaluated the proposed methods with data simulated from a more complex mixture model with four components. The combination of the four trajectories is shown in Figure 2.3, and 500 datasets were independently simulated with dispersion parameter being 2, 5, and 10. For each dataset, 500 genes were simulated with mixing proportions $\boldsymbol{\pi} = (0.1, 0.2, 0.3, 0.4)$ (leading to the number of genes in each cluster being roughly 50, 100, 150 and 200 respectively). We considered three estimation algorithms: EM, EMQN5 and EMQN10. The starting values for the parameters are: $\beta^1 = (2, -0.5, 0)$, $\beta^2 = (2, -0.48, 0)$, $\beta^3 = (2, -0.46, 0)$, $\beta^2 = (2, -0.52, 0)$, $\pi_1 = \cdots = \pi_4 = 0.25$, and $s_1 = \cdots = s_4 = 0.1$. The same lower and upper bounds have been specified for quasi-Newton estimation procedure within the algorithms as in those shown in Table 2.1.

The four-component mixture model is more complex than the two-component model, thus the evaluation for the clustering should not simply depend on the estimate of mixing proportions $\boldsymbol{\pi}$. In order to compare two partitions of objects, there are different measures that can be used to quantify the comparison. Rand (1971) proposed the Rand index as a criteria for evaluating clustering methods. Suppose that there are two different partitions of n objects, $U = u_1, ..., u_g$ and $V = v_1, ..., v_k$, where U is the set of true cluster memberships of the objects and V is a clustering result. Let a represent the number of pairs of objects classified into the same cluster in U and in the same cluster in V, b be the number of pairs of objects classified into the same cluster in U but not in the same cluster in V, c be the number of pairs of objects classified into different clusters in U but placed in the same cluster in V, and d represent the number of pairs of objects which are classified into different clusters in both partitions U and V. The Rand index is then calculated as $\frac{a+d}{a+b+c+d}$ and lies between 0 and 1, with the index equals to 1 when the two cluster partitions agree perfectly.

One problem with the Rand index is that the expected value of the Rand index when comparing two random cluster partitions does not equal to a constant value (e.g. zero). The Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) is a modification of the measure as it is corrected for chance with respect to a reasonable null hypothesis, so ARI has an expected value being zero and is bounded between ± 1. The ARI calculation assumes that the partitions U and V are selected at random such that the number of objects in the clusters are fixed, which corresponds to the generalized hypergeometric distribution for the randomness. Let n_{ij} represent the number of objects classified into both clusters u_i and v_j and we can form the contingency table of the two partitions as shown in Table 2.8.

Cluster	v_1	v_2	...	v_k	Sums
u_1	n_{11}	n_{12}	...	n_{1k}	$n_{1.}$
u_2	n_{21}	n_{22}	...	n_{2k}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
u_g	n_{g1}	n_{g2}	...	n_{gk}	$n_{g.}$
Sums	$n_{.1}$	$n_{.2}$	...	$n_{.k}$	n

Table 2.8: Contingency table for comparing two partitions U and V.

Under the generalized hypergeometric model of randomness, the ARI can be simplified to (Hubert and Arabie, 1985):

$$\frac{\sum_{i,j}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}$$

The ARI has been implemented in the function *adjustedRandIndex* from the R package *mclust*. We have reported the ARI and the percent of genes being correctly classified into the clusters for comparing the different estimation methods.

Our main focus is to assess the clustering ability of the algorithms so the parameter esti-

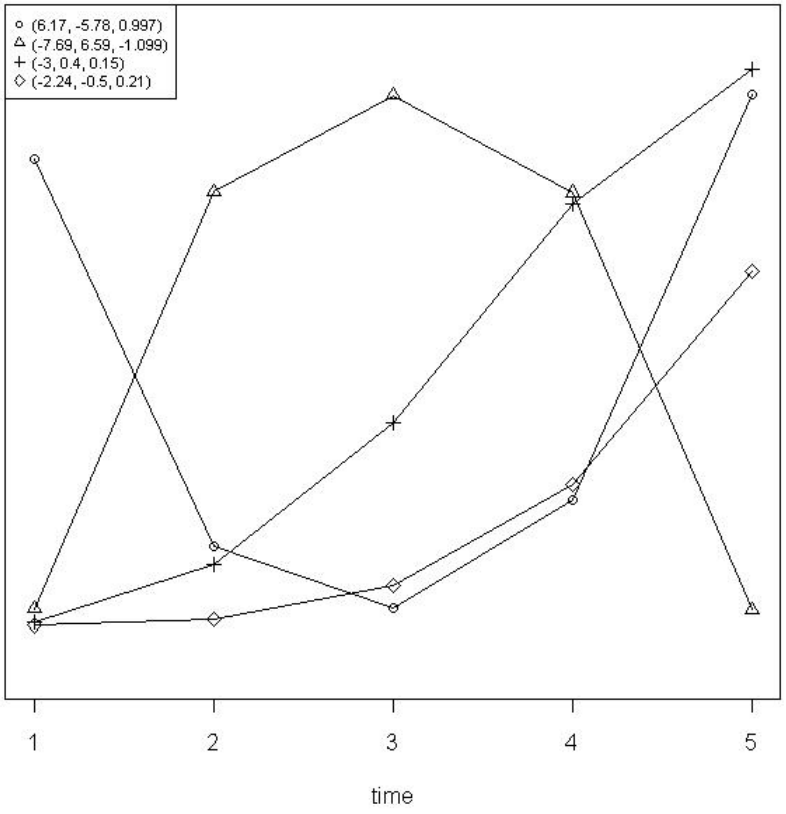

Figure 2.3: Simulated trajectories for the four-component mixtures.

mates for the trajectories and dispersion ($\boldsymbol{\beta}$ and s) are not shown, but in general the algorithms were able to obtain trajectories similar to the true curves. The estimated trajectories obtained by the EM algorithm when dispersion parameter is equal to 10 are shown in Figure 2.4, and similar trajectories have been obtained by both algorithms across all the situations with different dispersion values.

To compare the clustering ability of the algorithms, Figures 2.5 and 2.6 show the boxplots of ARI and the percent of genes being correctly classified into the clusters across the different values of dispersion parameter from 2 to 10. In general, the algorithms have comparable performances with similar ARI and percent of correctly classified genes. From both figures, it is noted that as the dispersion value decreases, the clustering accuracy decreases and variability

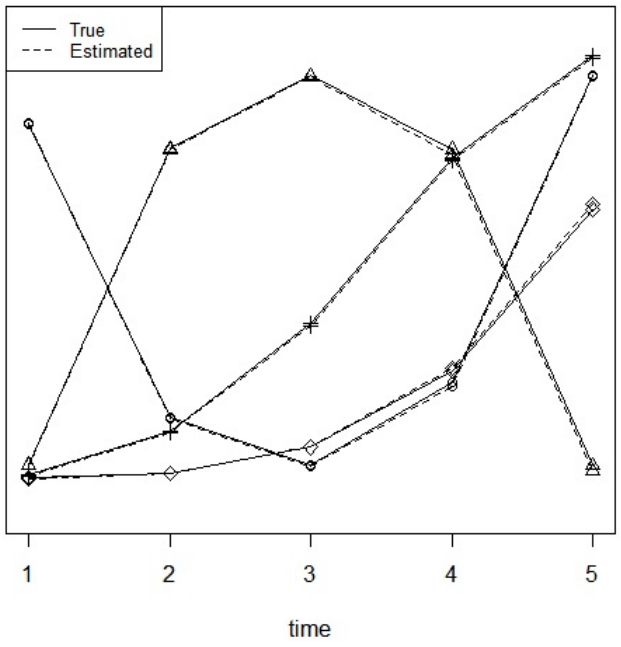

Figure 2.4: Four-component mixtures: Estimated trajectories by EM (dispersion=10).

increases. The increase in variability is more noticeable in the ARI values, but the large variability in percent of correctly classified genes by EMQN5 when dispersion equals to 0.5 may be of concern (not shown here). EM appeared to be a better clustering approach in this situation both in terms of higher and less variable ARI and percent correctness in gene classification.

The results from the two- and four-component simulation cases show that the proposed EM and hybrid algorithms are able to correctly estimate the trajectory shapes with acceptable clustering ability. Our algorithms were able to obtain mean ARI values within the range of 0.6 to 0.8, which is similar to the ARI values obtained by Scharl et al. (2010) when they evaluated the clustering of time-course gene expression data in low-noise setting. Since the parameter estimations in our simulation study were performed with specific initial values, it would be of interest to investigate whether initialization of the parameters in the estimation algorithms would affect the results, in terms of parameter estimation and clustering correctness. This motivates our second research objective and the issue will be addressed in the following chapter.

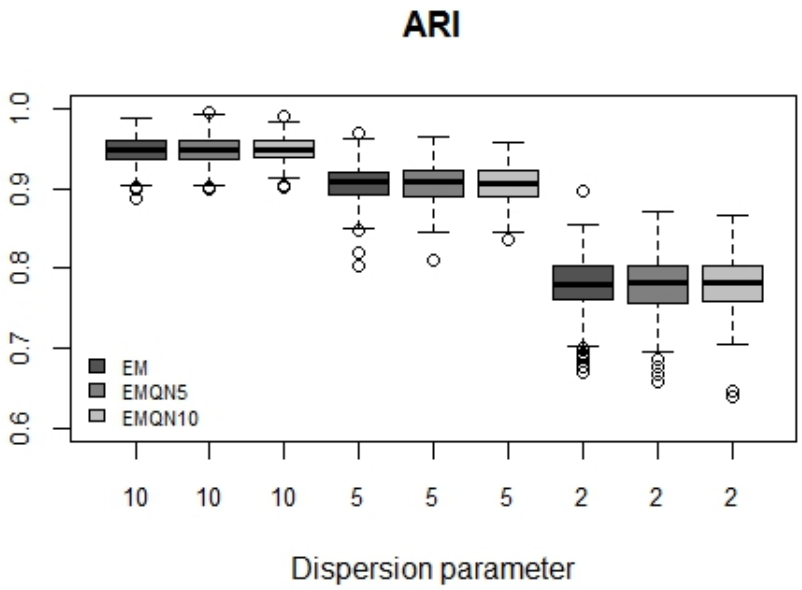

Figure 2.5: Four-component mixtures: Adjusted Rand Index (ARI) (with varying dispersion parameter s).

Model selection

For model-based clustering methods, often the number of components or clusters in the finite mixture model is unknown. The problem of overfitting occurs when we fit too many components to the data and would lead to the gene expression curves showing only random variation. On the other hand, if too few clusters are included in the mixture model then it would not be able to approximate the true underlying distribution. The method we use to determine the appropriate number of clusters in the model is the Bayesian Information Criterion (BIC) (Schwarz, 1978). It measures the goodness-of-fit based on the log-likelihood of the fitted model while penalizing for the model complexity and the sample size. The BIC is defined as

$$BIC = \log(L) - 0.5k\log(n)$$

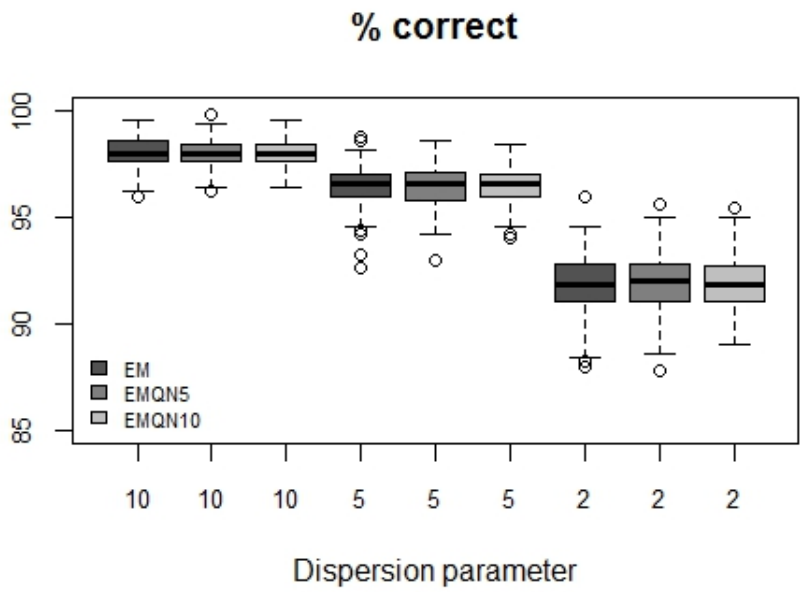

Figure 2.6: Four-component mixtures: Percent of correctly classified genes (with varying dispersion parameter s).

where k is the number of free parameters in the model and n is the sample size. The optimal finite mixture model is then determined as the model with the maximum BIC value. For our EM clustering approach, we choose a model which maximizes BIC among the different component models. Here we show an example of how the BIC values are used to determine the optimal number of clusters for the data, which also demonstrates how our model-based clustering approach can correctly identify the correct number of clusters in the data.

Number of groups	$s = 10$	$s = 5$	$s = 2$
2	-4123	-4185	-4158
3	-3844	-3933	-4040
4	**-3734**	**-3837**	**-3989**
5	-3796	-3906	-4047

Table 2.9: BIC values: EM estimation for four-component mixtures

Table 2.9 displays the mean BIC values obtained when model-based clustering was performed on the data simulated from the four-component mixtures, with the parameter estimated by the EM algorithm. We have tried clustering the data using mixture models with two- to

five-components and the various BIC values were calculated respectively. It is noted that the model-based clustering approach was able to identify the correct number of clusters in the simulated data across the varying dispersion parameter settings. From the results, we found that when data are not as dispersed, i.e. when dispersion parameter equals to 10 and 5, the clustering approach was able to find the correct number of components from the data 84% and 86% of the time respectively. However, the optimal number of components becomes more difficult to identify when dispersion parameter equals to 2, as the clustering approach was only able to identify the correct number of clusters 64% of the time. Some of the reasons contributing to the bad performance in model selection may be due to the fact that model identification becomes a more difficult issue when $s = 2$ and in some cases the five-component models were fitted with one mixing proportion estimated as very close to zero, which indicates that four-component models were actually fitted instead of five-component mixtures. In these cases, the four- and five-component models fitted would actually lead to the same clustering of the data but the BIC may lead to the wrong conclusion that the five-component models are more suitable. In general, the proposed model-based clustering approach is able to identify the correct number of components in the underlying mixture distribution of the data.

Model misspecification

One issue that we want to examine is the importance of modelling the over-dispersion in the data. Given that RNA-seq data has extra variation arising from the differences in replicate samples, it is more suitable to analyze the data using an over-dispersed model than using a Poisson model with equal mean and variance assumed. However, it is of interest to see how much would the model misspecification affect the proposed model-based clustering algorithm when a Poisson model is used instead of negative binomial distribution. For this purpose, the Poisson-based clustering method has been implemented with EM estimation to obtain parameter estimates. We evaluated the Poisson-based clustering approach with data simulated from a four-component mixture model. The simulated data followed the trajectories shown in Figure

2.3, and 500 datasets were independently simulated based on the negative binomial distribution with dispersion parameter s = 2,5 and 10. Since the Poisson-based clustering method would ignore the over-dispersion in the data, the results from this evaluation with negative binomial-based data would demonstrate the effects of model misspecification on the clustering ability.

The clustering ability of the algorithm is assessed by the ARI measure and the ARI values resulted from the Poisson-based clustering method are shown in the boxplot in Figure 2.7. As shown in the plot, the Poisson-based method obtained ARI values that are more variable than the ARI values resulted from correct model specification when datasets were generated under the same conditions (see Figure 2.5). When dispersion parameter in the negative binomial distribution increases, the data becomes more like Poisson. This explains why when the Poisson-based method was used, the clustering ability was higher in data with dispersion $s = 10$ than when $s = 2$. As shown in Figure 2.7, the ARI values generated from the Poisson-based clustering approach for data generated with dispersion $s = 2$ ranged between 0.4 to 0.75 and this indicated that the algorithm was not able to cluster the data well. When data simulated under the same conditions was clustered using the proposed negative binomial-based approach, the ARI values resulted are shown in Figure 2.5 and we note that the ARI values ranged from approximately 0.65 to 0.9. This simulation demonstrated that the over-dispersion in the data should not be ignored and needs to be accounted for in the algorithm when performing model-based clustering analysis. If the mean and variance are falsely assumed to be equal, the clustering accuracy of the proposed method would decrease and greatly affect the implications of the results.

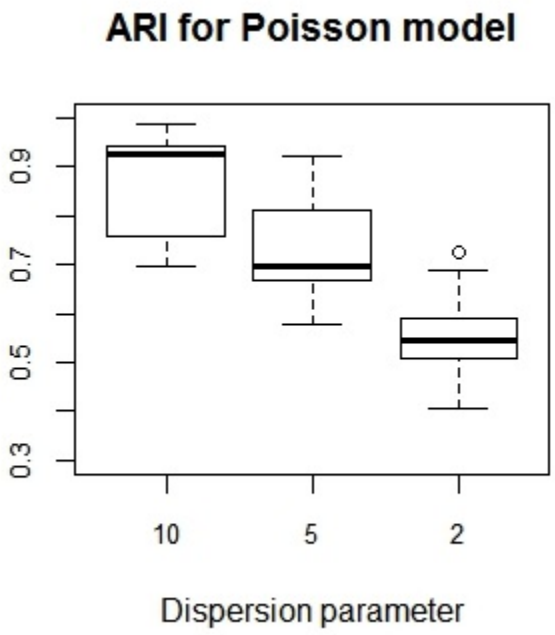

Figure 2.7: Four-component mixtures: Adjusted Rand Index (ARI) when model is misspecified and data generated from negative binomial distributions with varying dispersion parameter

2.4 Real data analysis

2.4.1 Fibroblast data

The proposed clustering approach was applied to the fibroblast data (described in Chapter 1) from the experiments on the progressive development of human vertebrate limb (Dudley et al., 2002). The analysis aims to identify gene expression patterns shown by fibroblast measurements in response to serum stimulation over time. We estimated several trajectory models based on the sample and each model used a quadratic term in sampling time to describe the relationship between time after serum stimulation and gene expression levels. We applied the EM, EMQN5 and EMQN10 algorithms to fit two-, three-, four-, five- and six-component models to see the clustering effects. We used BIC for model selection to obtain the optimal model (maximum BIC value) for each algorithm, where BIC is defined as

$$BIC = \log(L) - 0.5k\log(n)$$

with k being the number of free parameters in the model and n as the sample size. Initial values of the parameters were set to reflect constant trajectories (β_1 and β_2 set to be zero) with equal cluster proportions (for example, $\pi_i = 0.25$ for the four-component models).

	EM	EMQN5	EMQN10
2 groups	-2315	-2315	-2315
3 groups	-764	-764	-764
4 groups	**-642**	**-706**	**-648**
5 groups	-761	-765	-762
6 groups	-687	-2402	-2406

Table 2.10: Fibroblast data: BIC values for models.

The results from different models fitted using the EM algorithm are displayed in Figures 2.8 to 2.12. The BIC values, shown in Table 2.10, indicate that the four-component model was the best fitting model chosen when using all three algorithms for the model-based clustering. EMQN5 and EMQN10 produced almost equivalent cluster proportions and parameter estimates for all the different component models. The two- and three-component models produced by all three algorithms were essentially analogous, with the same expression patterns and very similar cluster proportions. The four-, five- and six-component models showed expression patterns that can be grouped into three main general patterns as shown in the three-component models. For example, in the six-component models shown in Figure 2.12, if we combine clusters 1 and 2, also clusters 3 to 5, and cluster 6; these three main expression patterns reflect the clusters shown in the three-component models (Figure 2.9) as clusters 1, 2 and 3 respectively. The cluster proportions vary between the three combined main groups in the six-component models and the clusters in the three-component models, but still reflected that the main proportion of the genes had a gradually increasing gene expression level pattern. Cluster 1 in the three-component model consisted of 87% of genes, which is similar to the 84% of the gene included in clusters 1 and 2 of the six-component model produced by the EM algorithm. However, the six-component models produced by EMQN5 and EMQN10 showed that a proportion

of those genes belonged to the other cluster with a relatively low-level expression pattern (cluster 3 with 25%) and the gradually increasing patterns consisted of only 65% of the genes. The cluster proportion of cluster 5 in each of the six-component models is 0% (with actual mixing proportions being 10^{-5} or less), which shows that the algorithms tried to separate the low-level expression pattern (clusters 3 to 5) into three clusters without success.

The optimal model chosen when using the EM algorithm was the four-component model showed in Figure 2.10 and it showed that a majority of the genes (over 84%) followed a gradually increasing expression pattern overtime. The models fitted by the EMQN5 and EMQN10 algorithms showed that the majority of the genes (over 76%) had a lower expression pattern overtime compared to the one showed in the model fitted by the EM algorithm. The model fitted by EM showed that clusters 2 and 3 had very similar expression patterns, but the models fitted by the two hybrid algorithms showed almost equivalent patterns for clusters 1 and 3. The large clusters indicate that majority of the genes have common roles in gene regulations and the smaller clusters represent smaller proportions of genes with different characteristics in the regulation process.

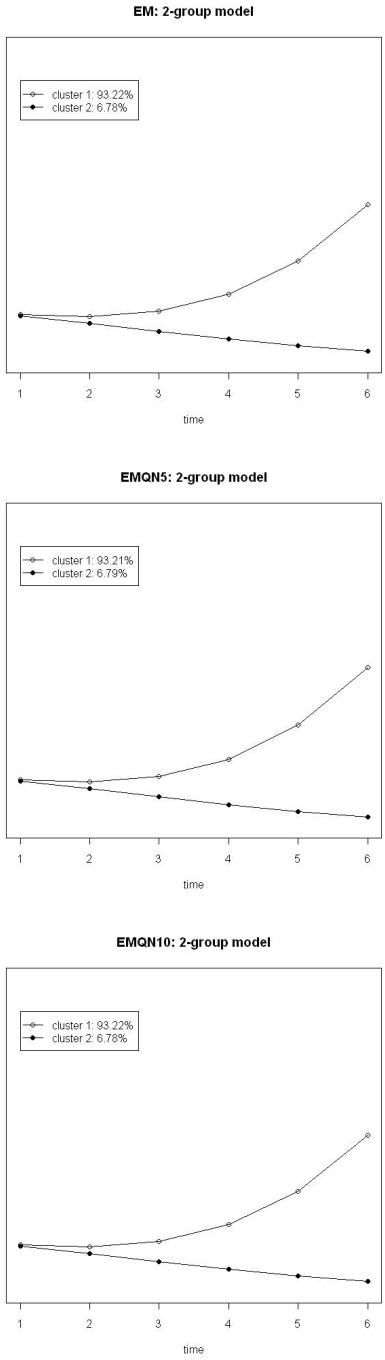

Figure 2.8: Fibroblast data: two-component models.

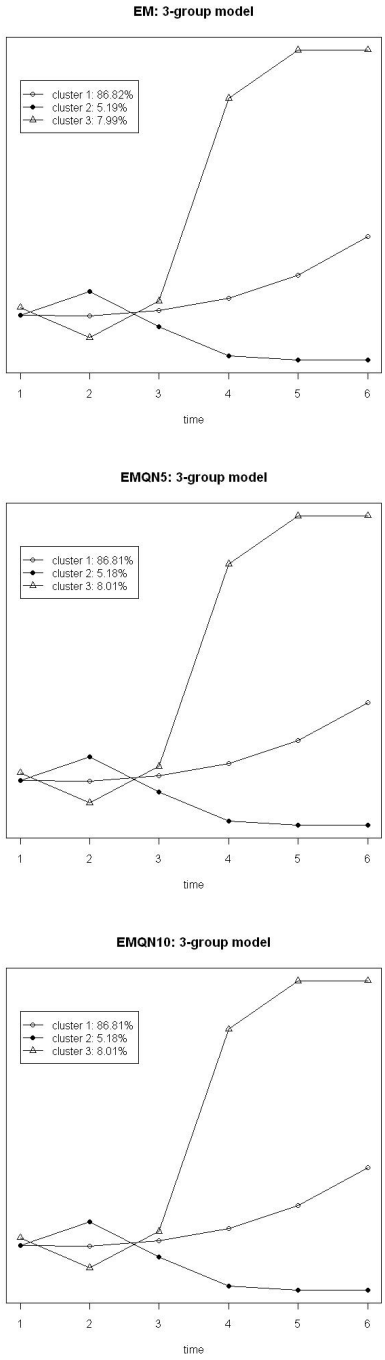

Figure 2.9: Fibroblast data: three-component models.

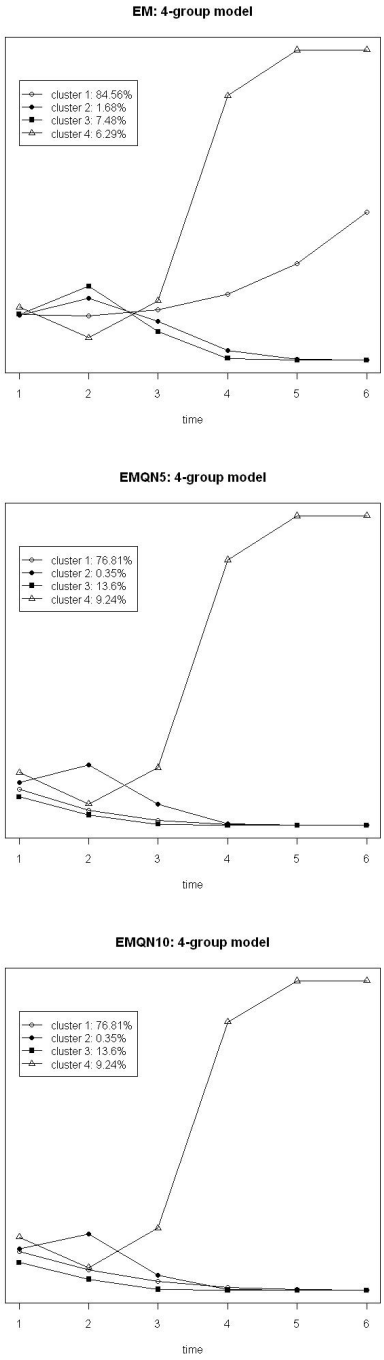

Figure 2.10: Fibroblast data: four-component models.

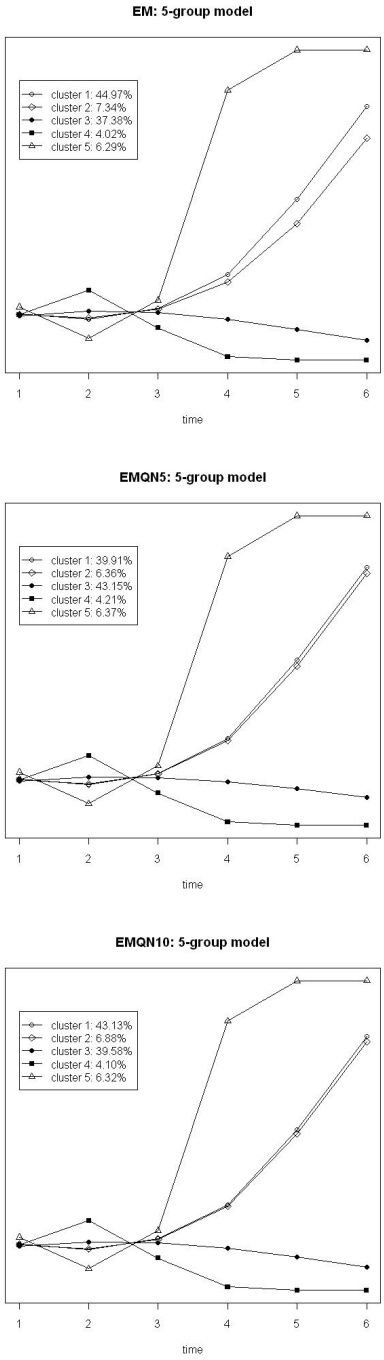

Figure 2.11: Fibroblast data: five-component models.

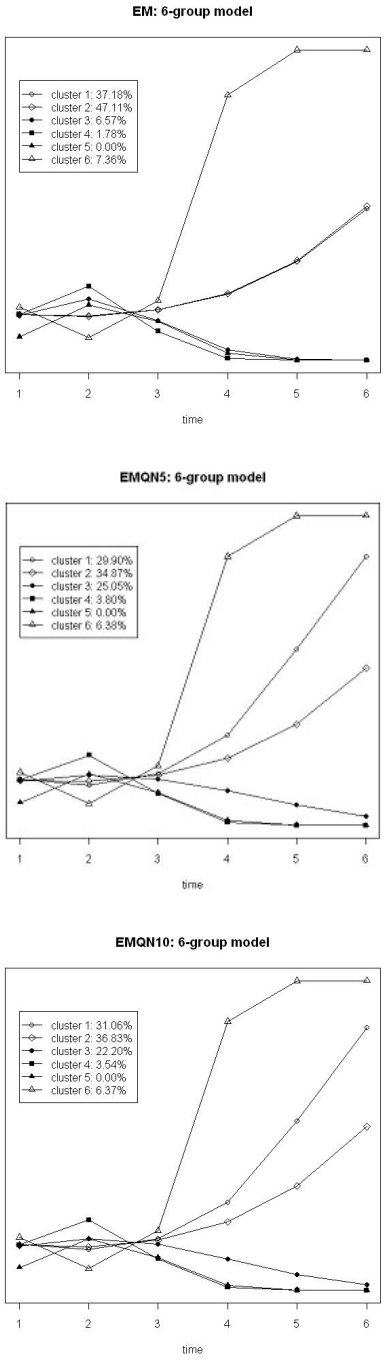

Figure 2.12: Fibroblast data: six-component models.

2.4.2 Fruit flies data

Another application of the proposed clustering algorithm consists of estimating trajectory models using the fruit flies data described in Chapter 1. The goal is to identify gene expression patterns shown by the data from the Drosophila transcriptome project (Graveley et al., 2011). We applied the clusering algorithms to fit two- to six-component models to see the clustering effects and used BIC to select the optimal model. The BIC values obtained from the different models fitted are shown in Table 2.11 and the maximum BIC values for the different algorithms are in boldface. The BIC values from the EMQN5 and EMQN10 algorithms suggested that more components might be fitted to the data, however, approximate standard error estimates were not obtained from EMQN5 on the five- and six- group model and from EMQN10 on the six-group model due to non-identifiable models. This suggests that we may focus on the two- to four-component models as the higher component models may have been non-identifiable mixture models being fitted to the data and resulted in unreliable estimates of parameters.

	EM	EMQN5	EMQN10
2 groups	-2356	-2321	-2321
3 groups	-2135	-2065	-2065
4 groups	**-2015**	-1977	-1941
5 groups	-2016	-1873	-1909
6 groups	-2146	**-1828**	**-1882**

Table 2.11: Fruit flies data: BIC values for models.

The two- to six-component models estimated by the EM, EMQN5 and EMQN10 algorithms are displayed in Figures 2.13 to 2.17. For the two-component models, all three algorithms estimated almost exactly the same cluster trajectories and proportions. The genes were divided into two large groups at approximately 57% and 43% with two expression patterns estimated which showed decreasing expression levels over time. The three-component models showed similar results for all three algorithms as well, with almost equivalent estimates for the trajectories and the proportions. Almost half of the genes (46%) were classified into one

cluster while the remaining genes were divided into two equal-sized clusters. All three clusters of genes still showed decreasing expression patterns over time. However, the algorithms did not obtain the same models for the four-component models. EM and EMQN10 both estimated very similar trajectories and cluster proportions, with three main groups of genes with decreasing expression patterns and one small group (7.6%) of genes having an almost bell-shaped expression pattern over time. The small cluster of genes showed an increasing expression level and then decreased back down to the initial starting level. On the other hand, EMQN5 estimated four groups at different cluster proportions with decreasing expression patterns. This might be due to the EMQN5 algorithm being trapped at a local maximum point during the ML estimation while EM and EMQN10 were able to find the global maximum point with more EM iterations performed than EMQN5.

The five-component models estimated by the three algorithms showed more distinctively different results than the previous models. Besides the four decreasing trajectories shown in the four-component model, the five-group model estimated by EMQN5 contained a small group (7.36%) of genes having the bell-shaped expression pattern just like the one in the four-group models estimated by EM and EMQN10. The five-component model estimated by EM showed three groups with distinctively decreasing patterns, one group (16.36%) with a slightly decreasing pattern and a small group (6.34%) of genes having a slightly curved pattern instead of the bell-shaped pattern in the four-component model. EMQN10 estimated some different trajectories: two distinctively decreasing patterns, one group (19.01%) with a slightly decreasing pattern, one small group (4.08%) with a slightly curved pattern and another small group (5.87%) of genes with an almost bell-shaped pattern.

In terms of the six-component models, EM and EMQN5 estimated similar expression patterns and cluster proportions. EM obtained a model with trajectories similar to those in the five-component model and an additional cluster of genes with a bell-shaped pattern. EMQN5

also obtained the same trajectory shapes at similar cluster proportions (for example, 31.19% in first cluster estimated by EM and 29.23% in first cluster estimated by EMQN5 while both clusters have the same expression pattern). The clusters in the model estimated by EMQN10 showed different expression patterns and while the EM and EMQN5 estimated models each had two cluster proportions being under 8%, the EMQN10 model had three such small clusters (at 3.56% to 5.39%). Also, in the model estimated by EMQN10 there is no cluster with a bell-shaped pattern, but instead there is a cluster (5.39%) of genes which initially had moderate expression level then peaked at around the middle of the sampling time frame before decreasing to a level lower than the initial moderate measurement.

Since the approximate standard error estimates were unable to be obtained by EMQN5 and EMQN10 in the five-component and six-component models and that BIC values of the EM models identified the four-component as the optimal model, we would focus on the four-component model. If we compare the BIC values in Table 2.11 across the three different algorithms for the four-group model, we see that the EMQN10 model had the maximum BIC value. The expression patterns and cluster proportions estimated by the EMQN10 were the same as those obtained by the EM algorithm. There are three large groups of genes which showed decreasing expression levels over the sampling time frame, while one small group of genes expressed in bell-shaped pattern. We see that the majority of genes being studied have similar roles in developmental stages of fruit flies while a small number of genes may have distinctively different responsibility in the regulation process.

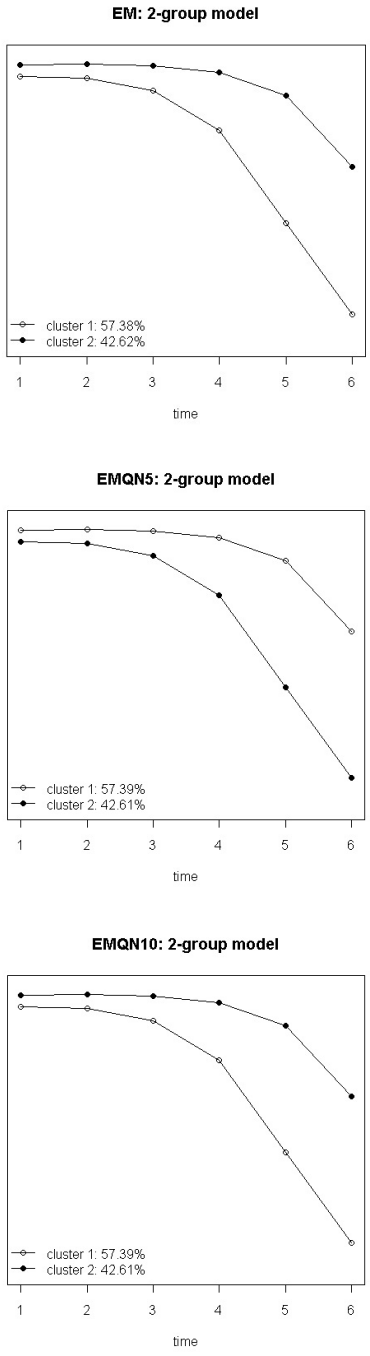

Figure 2.13: Fruit flies data: two-component models.

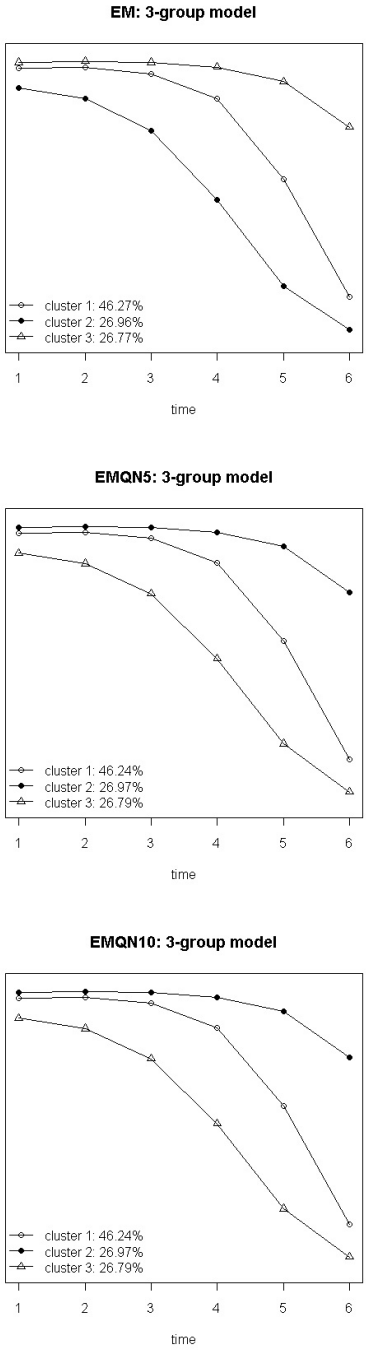

Figure 2.14: Fruit flies data: three-component models.

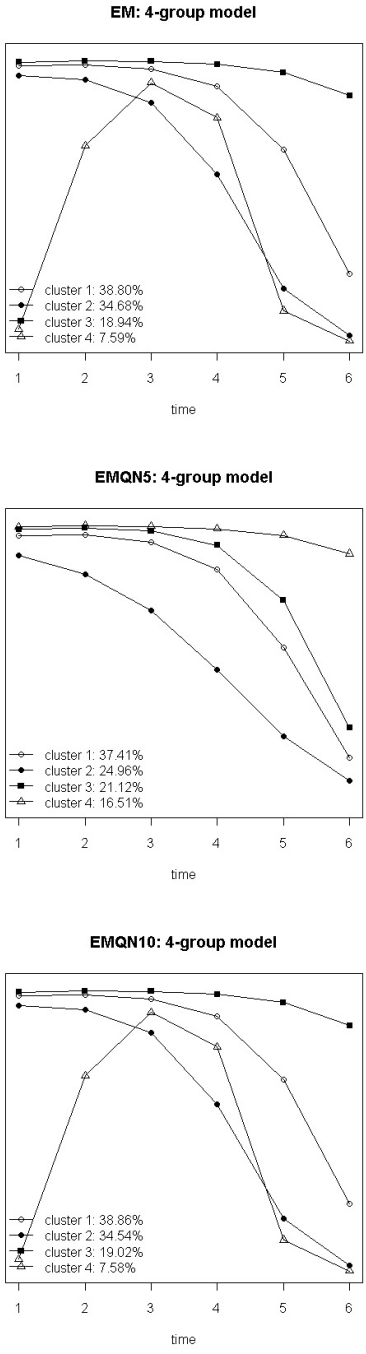

Figure 2.15: Fruit flies data: four-component models.

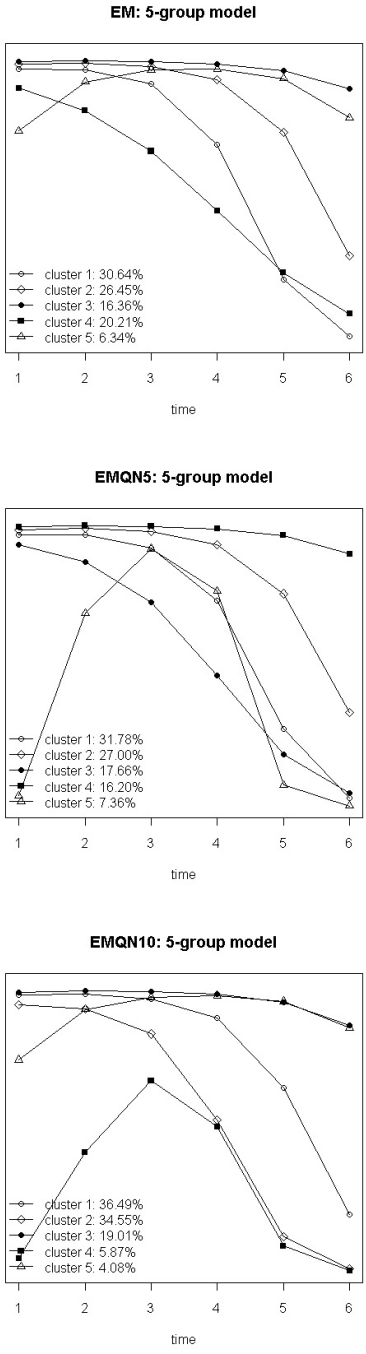

Figure 2.16: Fruit flies data: five-component models.

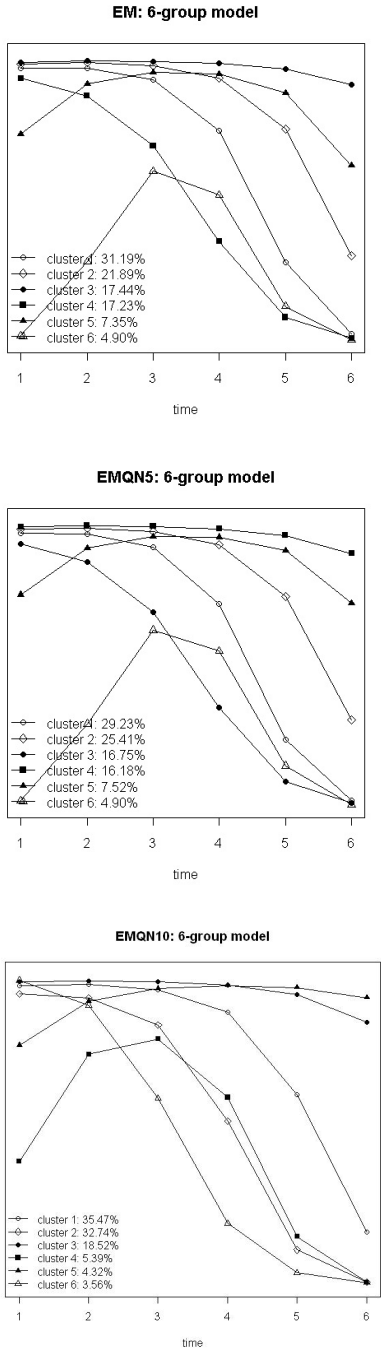

Figure 2.17: Fruit flies data: six-component models.

2.5 Summary

Statistical models for analyzing RNA-seq data has recently become a popular research area in the literature of statistical genetics. To our knowledge, no model framework has been developed for cluster analysis of RNA-seq data focusing on the time-course experiment setting. We propose a clustering algorithm for discovering gene expression patterns in time-series RNA-seq data. The algorithm is based on negative binomial models in the time-course setting and can be applied to RNA-seq data, as well as over types of count data with over-dispersion. We propose an EM clustering method and two EM/quasi-Newton hybrid algorithms to improve on the speed of the EM clustering. We demonstrate through both simulation studies and real data analysis that our proposed algorithms perform well on cluster analysis of time-course count data with over-dispersion. Applications to RNA-seq data illustrate that our model-based clustering approach produces meaningful clustering results that can enhance researchers' understanding about gene expression patterns over time.

Chapter 3

Initialization procedures for EM estimation of finite mixture models

3.1 Introduction

Finite mixture models are often used in model-based clustering approaches, and we have proposed a finite mixture model-based clustering method for analyzing time-course RNA-seq data. The EM algorithm is the main choice of estimation method when working with finite mixture models to perform clustering as it is a very powerful algorithm for learning probabilistic models from data with missing aspects (such as the unobserved labels of cluster components). Starting by guessing initial parameters of the model, the EM algorithm then iterates through two basic steps and finds a maximum likelihood (ML) solution when it reaches convergence. The choice of good starting values becomes an important issue as the type of data gets more complex. It is necessary to investigate the performance of EM for model-based clustering approaches with different initialization procedures.

3.1.1 Starting values for cluster analysis

The choice of starting values are important to iterative algorithms as it can influence the speed of convergence and the ability to find the global maximum in ML estimation. For mixture models with nonparametric ML estimates, Laird (1978) noted that several starting values may be required to avoid being trapped in local maxima and to locate the global maximum. It was suggested that a good set of initial values may be found from performing a grid search on the equally spaced grid between the possible values of the final solution. In cluster analysis, where there is no a priori knowledge about the underlying population in the data, sometimes visualization of the data can provide valuable information. McLachlan (1988) suggested the use of principal component analysis on the two-dimensional scatter plots upon appropriate data transformation to find initial clustering of the data. Such exploratory search relies on two- or three-dimensional scatter plots to exhibit any clustering and may not be appropriate when dealing with more complex data.

Research on initialization strategy with mixture models have mainly focused on or demonstrated through normal or Poisson mixtures of two components. Hosmer (1973) and Hasselblad (1969) examined the importance of starting values in ML estimation of normal mixtures. They indicated that starting values are not that influential to ML estimation, while Fowlkes (1979) concluded otherwise and demonstrated the sensitivity to starting values through simulation results. The contradicting results have been re-examined by Woodward et al. (1984) by repeating the simulation analyses performed by Fowlkes and their results showed that the reason for the susceptibility to starting values which Fowlkes observed was mainly due to the direct maximization procedure that was used for the ML estimation. ML estimation using the EM algorithm is better than direct maximization becaus the EM approach did not exhibit such sensitivity for the choice of starting values.

Woodward et al. (1984) further proposed a similar ad hoc quasi-clustering approach for

dealing with two-component normal mixtures. The initial values would be chosen by selecting the set of values which maximizes a criterion that is based on the mixing proportion of the clusters and the sample medians of the clusters formed from the entire sample data. This initialization strategy has a downside because when the two normal mixtures are overlapped, the right tail in the first population may be truncated as data might be wrongly assigned to the second population and leading to ML estimates from the first cluster being underestimated. Finch et al. (1989) also studied two-component normal mixtures and used a quasi-Newton algorithm as the optimization procedure. They concluded that the crucial parameter in the ML estimation is the mixing proportion since they observed that once the proportion is fixed, the optimal estimates of the normal means and variances can be easily approximated. The arithmetic means and weighted average of the variances calculated from the samples split by the mixing proportion can be used as the choice of parameter estimates.

Another issue related to initialization is large size of the data. Model-based clustering approaches often require a substantial amount of storage and computing time, in proportional to multiples of the dimension of the data (Wehrens et al., 2004). Genomic data are often massive datasets and this imposes limitations on the algorithmic complexity when focusing on clustering gene expression data. One approach of dealing with these limiting factors is to initiate clustering algorithms with a sample or subset of data. The simplest method would be to first cluster a small random sample drawn from the data, then using the estimated model resulting from this sample to initiate clustering on the entire dataset. Banfield and Raftery (1993) employed this approach to analyze medical images by clustering a sample from the image voxels and using the resulting clusters to classify the remaining voxels. They performed discriminant analysis in the model-based clustering framework by simply using a single expectation step in the EM iteration (Fraley and Raftery, 2002).

Unfortunately, the sample may not necessarily be a good representation of all the clusters

since data in small clusters may have a low prior probability of occurrence in the sample. This simple approach based on clustering a sample can be modified in several ways to give improved results, such as by tentatively selecting several models based on multiple samples rather than just one, or by performing several EM steps on the entire dataset rather than one single E step (Wehrens et al., 2004). Bradley and Fayyad (1998) and Fayyad et al. (1998) have also proposed clustering several subsamples of the data as initialization procedures for k-means and EM clustering approaches. The initializing values for the clustering algorithm would be chosen from the cluster centers determined by each subsample clustering. For clustering massive datasets in software metrics and tomography, Maitra (2001) suggested a multistage clustering algorithm which starts by clustering a sample of data, identify from the whole dataset observations which belong to these identified clusters, then iterates the sampling and clustering on the remaining dataset until entire set of observations have been classified. Fraley et al. (2005) extended the sampling method into an incremental model-based clustering process by incrementally adding new clusters which are initialized with poorly fit observations from previous model.

A common strategy for starting EM algorithms has been to select random initial positions and to run the algorithm several times and select the solution which maximizes the likelihood among those several runs. Biernacki et al. (2003) compared random initialization, short runs of EM, and two modified EM algorithms and evaluated their performances on selection of sensible initial parameters for estimation in mixtures of multivariate normals. Karlis and Xekalaki (2003) also compared using random starting values to a number of other initialization methods and found that the performance is poor if the algorithm is started from random points. They advised using a mixed strategy of starting from multiple initial values and running a small number of iterations, then using the point with the largest likelihood after these initial iterations to continue running EM as initial values until final convergence.

3.2 Method: Initialization strategies

We propose initial strategies on modifying the initialization procedures by adding in sampling component and selecting the optimal initial set of parameters from multiple samples. We note that our clustering model may be maximized using a direct maximization step such as a maximization using the Quasi-Newton method. This direct maximization is a much faster estimation process than the EM approach, however, the use of such direct maximizing procedure may lead to some convergence problems, and therefore the EM was proposed as the solution to the problems. We may take advantage of the speed of the direct maximization and use it as an initialization procedure, hoping that the solution from the maximization would be in the neighbourhood of the global maximum so that we can use the solution as good starting values for our EM clustering method.

Various initialization strategies have been proposed and investigated for model-based clustering approaches with mixtures of multivariate Gaussian distributions. Scharl et al. (2010) performed an evaluation of initialization strategies for time-course data from mixtures of regression models with random effects. Their simulation study focused on using smoothing splines and B-splines to fit mixtures of regression models to time-course microarray data.

The purpose of our current research is to obtain a practical initialization procedure for the model-based clustering approach proposed for time-course RNA-seq data described in Chapter 2. The semi-parametric group-based trajectory model we proposed is a mixture of negative binomial distributions for describing discrete counts overtime, thus we would expect the initialization strategies to have different effects on this proposed model as compared to their effects on the mixtures of multivariate normals and mixed-effects models.

We would compare strategies in two main groups: 1) initializing parameters as equal cluster mixing proportions and with true component parameters, and 2) initializing parameters as

equal cluster mixing proportions and with random component parameters. The first case is used in order to investigate the behaviour of the EM algorithm when starting with partially optimal solution, while the second case would more closely reflect the situation when dealing with real data, i.e. no prior knowledge on parameters. The main issue is how should we select the random component parameters (namely the dispersion and β parameters for the model-based clustering approach)? We focus on the following strategies and compare their performances.

- Random initialization: run EM t times with random starting values and select the solution with the maximum likelihood among the t runs.
- Short runs: run EM, hybrid-EM (EMQN5 and EMQN10) or direct maximization t times with random starting values and a lose convergence criterion, followed by a regular run of EM starting from the solution maximizing the likelihood among the t runs.
- Sampling: select a sample and perform EM, hybrid-EM or direct maximization with random starting values, then using the solution as starting values for the EM clustering of the entire dataset.
- Sampling (multiple): select a sample and perform EM, hybrid-EM or direct maximization, repeat this for k times and use the solution with the largest likelihood among the k runs as starting values for the EM clustering of the entire dataset.

3.3 Simulation study

Simulation studies are designed to evaluate the performance of selected initialization strategies on the four component mixture model used in Chapter 2. Artificial datasets are designed to resemble time-course gene expression patterns in a total of 500 genes in four clusters over five time points, with the clusters consisting of 50, 100, 150, and 200 genes respectively. We simulated 500 datasets independently for the evaluation of each initialization method. We

considered datasets simulated with different dispersion parameter values (s = 2, 5 and 10) and different sample sizes for the procedures with sampling component (sample size as 50, 150 and 250). The simulation studies focus on testing the initialization of the component parameters, so we set the initial mixing proportions to be equal for all four clusters ($\pi_1 = \ldots = \pi_4 = 0.25$). The different initialization methods being evaluated are described below.

1. True trajectory shapes (True): use the true values of the $\boldsymbol{\beta}$ and s parameters as starting values for EM run to investigate the performance of EM when starting at optimal solution.

2. Random initialization (Random): run EM 5 times with random β_0 and s parameters each time and select the solution with the maximum likelihood among the 5 runs.

3. Short runs of EM (sEM): run EM 5 times with random β_0 and s parameters and a loose convergence criterion, select the solution with maximum likelihood among those 5 runs and use as initial values to start a regular EM run.

4. Sampling with EM (samEM): select a sample of 50 genes and perform EM with random β_0 and s parameters, then use the solution as initial values for starting a regular EM run.

5. Short runs of direct maximization (sDirect): perform direct maximization 5 times with random β_0 and s parameters and a small number of maximum iterations, select the solution with maximum likelihood among those 5 runs and use as initial values to start a regular EM run.

6. Sampling with direct maximization (samDirect): select a sample of 50 genes and perform direct maximization with random β_0 and s parameters, then use the solution as initial values for starting a regular EM run.

7. Sampling with direct maximization multiple times (samDirectmult): perform direct maximization with random β_0 and s parameters on 5 sets of samples of 50 genes, select the

solution with maximum likelihood among the 5 runs and use as initial values to start a regular EM run.

8. Short runs of EMQN5 (sEMQN5): run EMQN5 5 times with random β_0 and s parameters and a lose convergence criterion, select the solution with maximum likelihood among those 5 runs and use as initial values to start a regular EM run.

9. Sampling with EMQN5 (samEMQN5): select a sample of 50 genes and perform EMQN5 with random β_0 and s parameters, then use the solution as initial values for starting a regular EM run.

10. Sampling with EMQN5 multiple times (samEMQN5mult): perform EMQN5 with random β_0 and s parameters on 5 sets of samples of 50 genes, select the solution with maximum likelihood among the 5 runs and use as initial values to start a regular EM run.

11. Short runs of EMQN10 (sEMQN10): run EMQN10 5 times with random β_0 and s parameters and a lose convergence criterion, select the solution with maximum likelihood among those 5 runs and use as initial values to start a regular EM run.

12. Sampling with EMQN10 (samEMQN10): select a sample of 50 genes and perform EMQN10 with random β_0 and s parameters, then use the solution as initial values for starting a regular EM run.

13. Sampling with EMQN10 multiple times (samEMQN10mult): perform EMQN10 with random β_0 and s parameters on 5 sets of samples of 50 genes, select the solution with maximum likelihood among the 5 runs and use as initial values to start a regular EM run.

The simulations were carried out using parallel computing with 128 cores, programmed in R using the *Rmpi* package (Yu, 2002). For performance comparisons among the various initialization procedures, the measures for evaluation taken into consideration are: percentage of correctly classified genes, adjusted Rand index (ARI), number of EM iterations and total

run-time required. It is desired to identify an initialization procedure which will lead to the optimal classification ability when used with the model-based clustering algorithm.

3.3.1 Results

Varying dispersion parameter

Table 3.1 shows the run-time required for the various starting methods across the three dispersion parameter settings. If we use the run-time of the initialization method 'True' as reference, there seems to be no specific pattern for run-time required as the dispersion decreased. The other initialization methods also showed similar results, with some requiring more time when dispersion is large while some other required more time when dispersion is small. Figure 3.1 shows the mean ARI resulted from the different initialization methods when dispersion parameter varied from $s = 10$ to 2. It is noticeable that the method starting with random starting values ('Random') performed worst, with the lowest mean and most variable ARI obtained. Relative to the results obtained from the 'True' method (using the true parameters as starting values), our clustering method performed best when using starting values from the 'sEM' method. The other initialization methods all had similar performance, with similar mean ARI achieved but much more variable than the 'True' and 'sEM' methods. Overall, all initialization methods had the highest mean ARI when dispersion parameter $s = 10$ and had worse clustering accuracy as the dispersion parameter decreased (ARI from around 0.9 for $s = 10$ went down to ARI around 0.8 for $s = 2$).

Mixture models are often non-identifiable and the estimation methods would not converge when such models are obtained. The parameters in these models would not have reliable estimates and may have unreasonable estimates of standard errors, so the datasets with non-identifiable models were excluded from the analysis. Figure 3.2 displays the mean ARI from the various starting methods while only including datasets with identifiable models (the number

of datasets included in these boxplots are listed in Tables 3.3, 3.5 and 3.7 in later subsections). We can see from the boxplots that removing the non-identifiable models resulted in a lot of the datasets with low ARI (ARI lower than 0.6) being excluded. There are also smaller amounts of outliers for all different initialization methods. Again, the methods 'True' and 'sEM' produced ARI values with smallest variation in all three dispersion parameter settings, and the method 'samEMQN10m' also had better results than the other initialization methods. By looking at the differences between including and excluding the datasets with non-identifiable models, there is reason to believe that the non-identifiable models result in poor clustering/classification ability and we would like to find an initialization method with a good balance between clustering accuracy and chance of obtaining non-identifiable model.

Methods	Run-time (hours)		
	$s = 10$	$s = 5$	$s = 2$
True	14.15	16.78	13.67
Random	70.98	65.99	60.72
sEM	24.51	28.67	29.28
samEM	22.87	21.30	25.92
sDirect	40.84	32.77	22.73
samDirect	19.05	24.16	20.54
samDirectmult	57.92	29.62	17.21
sEMQN5	40.27	36.90	29.76
samEMQN5	14.96	26.43	24.48
samEMQN5mult	20.80	27.55	22.48
sEMQN10	37.78	34.62	28.56
samEMQN10	26.82	21.65	19.82
samEMQN10mult	27.78	29.19	19.06

Table 3.1: Run-time required for the different initialization strategies.

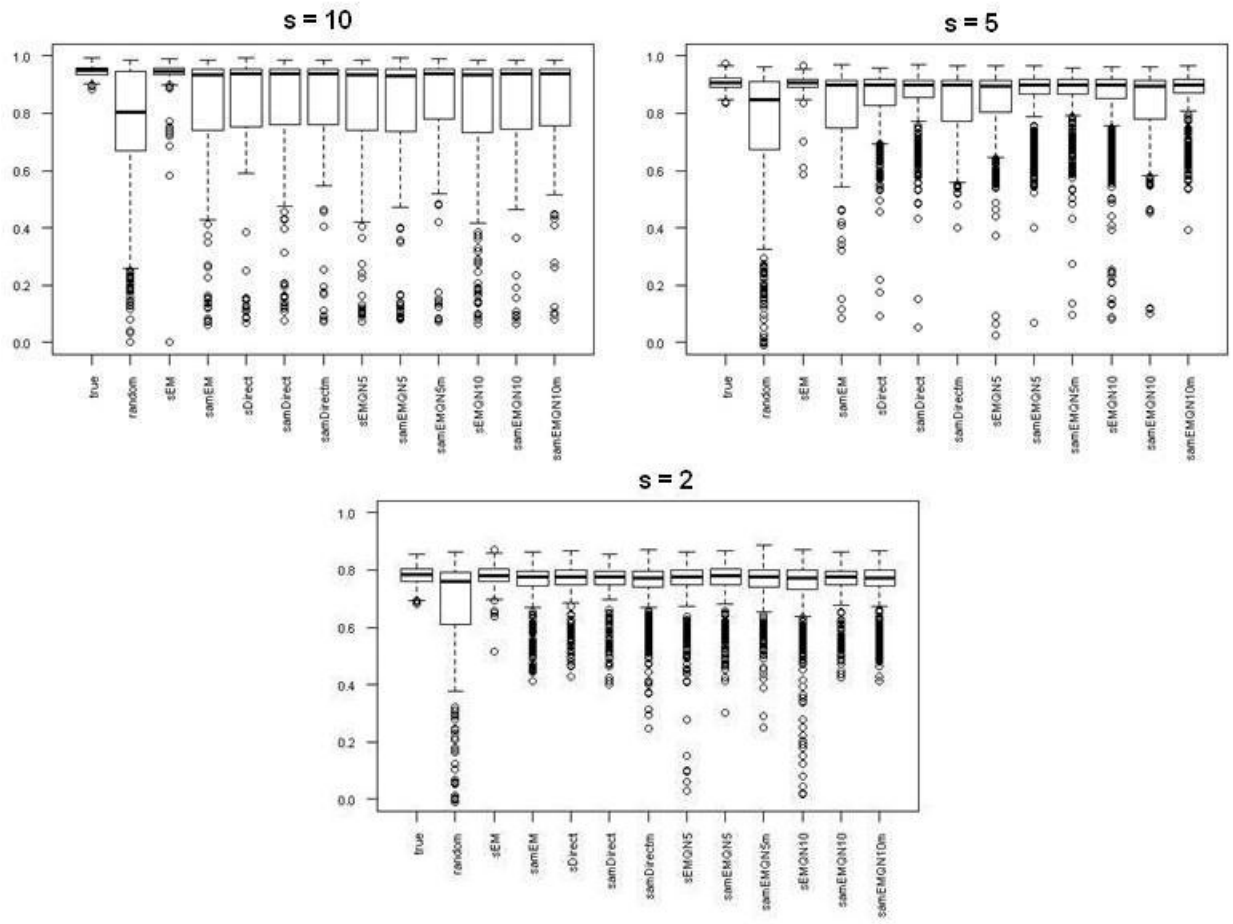

Figure 3.1: Mean ARI of the different initialization strategies across the three dispersion parameter settings.

Dispersion parameter $s = 10$

Table 3.2 shows the mean ARI and mean percent correctness of the different initialization methods for dispersion parameter $s = 10$ when all datasets (total of 500) are included. The estimation using the true trajectory shapes as starting values ('True') had the best performance as expected, with the highest ARI and percent correctness for the classifications. The initialization strategy 'sEM' also achieved good results with the next highest ARI of 0.94 and 97.07% of genes correctly classified into the clusters. The results showed that all the other proposed initialization strategies performed better than the traditional random initialization, as the method 'Random' had the lowest ARI and percent correctness recorded.

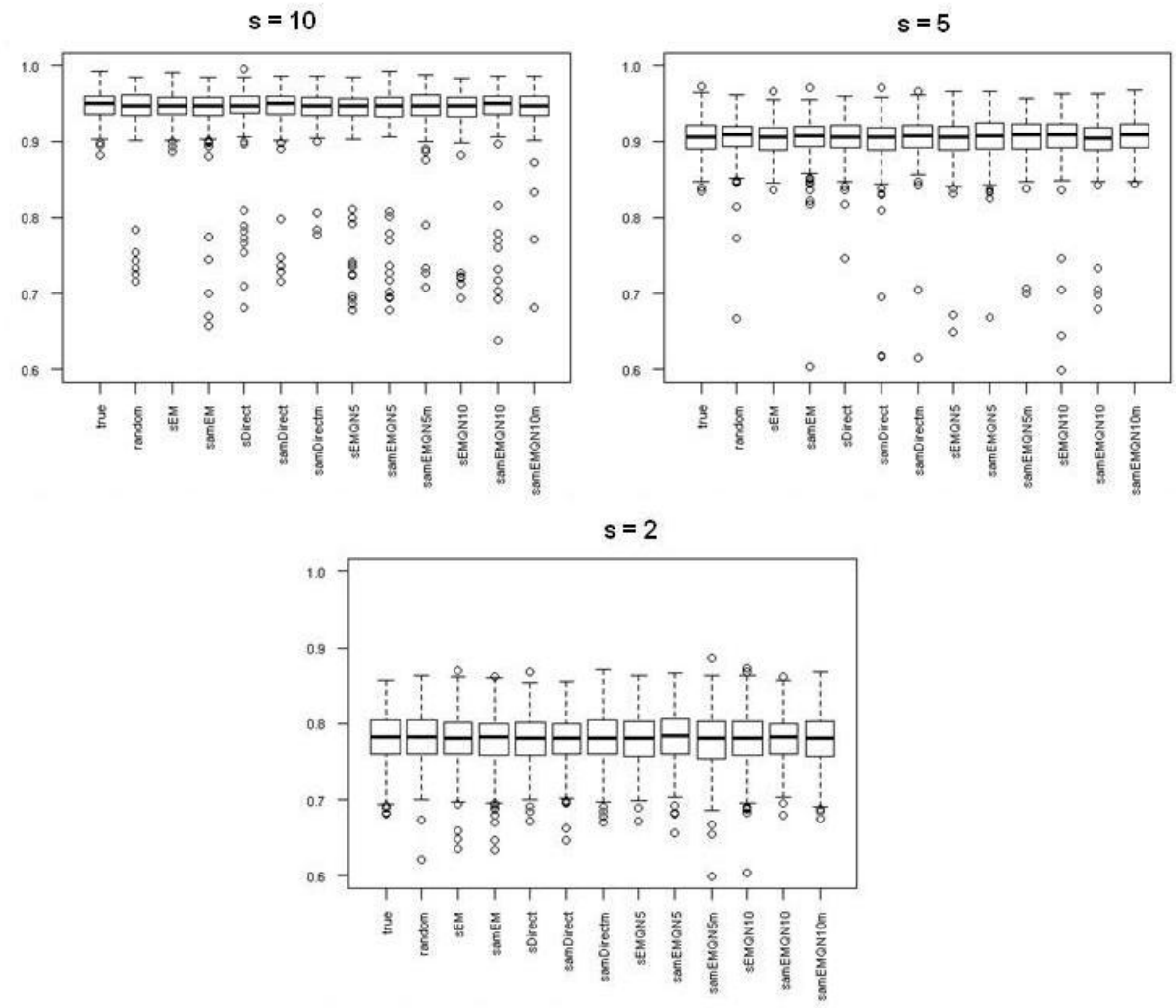

Figure 3.2: Mean ARI of the different initialization strategies across the three dispersion parameter settings (excluding datasets with non-identifiable models).

Including a sampling component into the initialization methods seems to have improved the performance for using direct maximization and EMQN10 to obtain starting values, as the 'samEMQN10' and 'samEMQN10mult' methods were able to obtain higher ARI and percent correctness than using the 'EMQN10' method, and same for 'sDirect', 'samDirect' and 'samDirectmult' respectively. Performing the sampling multiple times improved the results for using EMQN5 to find starting values, since the method 'samEMQN5mult' had a higher ARI of 0.87 and 84.69% correctness compared to the lower ARI and percent correctness of the

Methods	Datasets included	ARI	Percent correctness	EM iterations
True	500	0.95 (0.02)	97.99 (0.66)	90.65
Random	500	0.71 (0.29)	69.22 (29.75)	247.66
sEM	500	0.94 (0.08)	97.07 (6.93)	42.52
samEM	500	0.84 (0.18)	84.32 (23.66)	145.18
sDirect	500	0.85 (0.16)	83.36 (23.72)	197.52
samDirect	500	0.86 (0.16)	83.61 (24.25)	199.39
samDirectmult	500	0.86 (0.15)	83.87 (25.09)	146.31
sEMQN5	500	0.84 (0.17)	79.83 (24.94)	192.65
samEMQN5	500	0.84 (0.18)	80.11 (26.93)	187.17
samEMQN5mult	500	0.87 (0.15)	84.69 (24.94)	164.73
sEMQN10	500	0.83 (0.20)	82.13 (25.63)	162.11
samEMQN10	500	0.86 (0.16)	83.74 (24.73)	205.89
samEMQN10mult	500	0.86 (0.15)	83.31 (26.08)	156.89

Table 3.2: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the different initialization strategies (including all 500 datasets) when $s = 10$.

'sEMQN5' and 'samEMQN5' methods.

The EM iterations required for the EM long run following the different initialization strategies are also shown in Table 3.2. The goal is to find a strategy which can reduce the number of iterations required to find the MLE's for the parameters. It is clear from the results that the 'sEM' method performed the best in this case with the lowest number of EM iterations required for the estimations. Besides the random initialization method 'Random' requiring the highest number of EM iterations, all the other strategies required similar amount of iterations to complete the estimation process.

The run-times required for the different initialization methods are shown in Table 3.1. The method of starting with the true trajectory shapes (the method 'True') should require the least amount of time and the run-time results did support that assumption. For $s = 10$, initialization strategy with random selection ('Random') required approximately five times the run-time of the 'True' method, which makes it not a desirable method considering its long run-time along

with its poor estimation performance (lowest ARI and percent correctness). Using a sampling component should reduce the run-time of the estimation since we try to obtain starting values using a smaller group of genes rather than the entire set of genes. Obtaining starting values using EM with sampling ('samEM') required less time than 'sEM', but the small time difference of 2 hours did not justify for using the 'samEM' method compared to the great performance of 'sEM', which had a higher ARI and higher percent of genes correctly classified.

Methods	Datasets included	ARI (sd)	Percent correctness (sd)	EM iterations
True	500	0.95 (0.02)	97.99 (0.66)	90.65
Random	220	0.94 (0.05)	96.03 (11.12)	204.53
sEM	491	0.95 (0.02)	97.98 (0.64)	40.90
samEM	342	0.94 (0.03)	96.99 (8.12)	90.62
sDirect	342	0.94 (0.03)	96.68 (8.62)	141.54
samDirect	350	0.94 (0.03)	97.19 (6.79)	152.59
samDirectmult	351	0.94 (0.04)	96.92 (8.78)	105.34
sEMQN5	329	0.94 (0.04)	96.48 (11.98)	144.31
samEMQN5	320	0.94 (0.04)	95.89 (11.15)	130.49
samEMQN5mult	362	0.94 (0.03)	97.53 (4.66)	124.66
sEMQN10	338	0.94 (0.04)	97.30 (5.65)	118.61
samEMQN10	350	0.94 (0.04)	96.45 (9.81)	143.26
samEMQN10mult	345	0.95 (0.02)	97.63 (4.42)	107.96

Table 3.3: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the different initialization strategies (excluding datasets with non-identifiable models) when $s = 10$.

Table 3.3 shows the number of datasets (out of a total of 500) included in the performance comparison across the different initialization strategies if we only consider the datasets with identifiable models. The 'sEM' strategy had the fewest number of datasets with non-identifiable models aside from the simulations which used the true trajectory shapes as starting values ('True'). All the other proposed initialization strategies had similar numbers of datasets being excluded, but including the sampling component seems to slightly reduce the number of datasets with non-identifiable models (for simulations with direct maximization, EMQN5 and EMQN10, but not for obtaining starting values with EM method).

Table 3.3 also shows the mean ARI and percent correctness of the different initialization strategies when considering only the datasets with identifiable models, and estimations with the different strategies all obtained similar results. The 'True', 'sEM' and 'samEMQN10mult' produced the three highest estimates of mean ARI and percent correctness with the smallest standard deviations out of all the initialization methods. Compared with the 'samEMQN10mult' method, the 'sEM' method is preferred since there were fewer datasets with non-identifiable models when estimating using the 'sEM' methods (only 9 datasets excluded from the analysis). The method with random initialization ('Random') still appeared to have the worst performance, with the lowest percent of correctly clustered genes and highest standard deviations for the estimates.

Dispersion parameter $s = 5$

Table 3.4 shows the clustering results obtained using different initialization strategies when dispersion parameter $s = 5$. Similar to the results from the case when $s = 10$, it is clear that the methods 'True' and 'sEM' also obtained the best results when dispersion parameter is reduced. These two initialization methods produced the highest ARI and percent correctness with the smallest standard deviations among all the methods, and 'sEM' also required the least number of EM iterations for the EM estimation. The 'Random' method would be a poor choice of initialization strategy since it resulted in the lowest ARI and lowest percent of genes classified correctly. The 'Random' method required more than double of the amount of iterations needed for 'True', five times of the amount needed for 'sEM' and about two-thirds more iterations than needed for the other proposed methods. With its low accuracy in clustering and long runtime requirement, initializing our clustering approach with random starting values seem to be appropriate only if the other methods are not available. Methods implemented with the sampling component seem to have performed better than those without sampling, since they tend to have

produced slightly higher ARI with less EM iterations required.

When we only consider the datasets with identifiable models (Table 3.5), the results showed a similar pattern. The numbers of datasets included for the initialization methods varied a lot, with the methods 'True' and 'sEM' had the highest number of datasets with identifiable models and the best clustering results in terms of ARI and percent of genes being classified correctly. The method 'sEM' also required the least number of EM iterations, while the method 'Random' required the largest number of iterations and produced the worst clustering results with highest amount of variation. Methods which sample datasets multiple times produced higher number of datasets with identifiable models, better clustering ability and required lower amount of EM iterations than the methods without sampling components. When we look at the difference between including and excluding the datasets with non-identifiable models (Tables 3.4 and 3.5), it is noticeable that all measures improved when the non-identifiable models are not considered, with higher and less varied ARI and percent correctness achieved under smaller amounts of EM iterations.

Dispersion parameter $s = 2$

The mean ARI and mean percent correctness among the various initialization procedures for $s = 2$ when all 500 datasets are included are shown in Table 3.6. The mean ARI achieved by all initialization methods ranged between 0.68 to 0.78, which is a lower range than the results obtained when dispersion parameters were 5 or 10. The percents of genes being clustered correctly also showed the same pattern, although the amounts of decrease from $s = 5$ to $s = 2$ are smaller than those observed for the mean ARI measures. The clustering ability comparison across the different initialization methods showed the same pattern as the other two dispersion parameter settings, with 'True' and 'sEM' methods having the highest mean ARI and percent correctness while requiring low numbers of EM iterations. The methods 'Random' and

Methods	Datasets included	ARI	Percent correctness	EM iterations
True	500	0.91 (0.02)	96.49 (0.88)	108.31
Random	500	0.72 (0.27)	70.96 (30.08)	241.41
sEM	500	0.90 (0.03)	96.20 (3.73)	47.63
samEM	500	0.83 (0.14)	85.81 (21.57)	136.51
sDirect	500	0.84 (0.13)	84.77 (22.89)	157.72
samDirect	500	0.85 (0.12)	86.22 (22.29)	180.60
samDirectmult	500	0.84 (0.12)	83.34 (25.81)	151.36
sEMQN5	500	0.83 (0.14)	83.94 (24.44)	168.07
samEMQN5	500	0.85 (0.11)	86.88 (21.66)	155.28
samEMQN5mult	500	0.86 (0.11)	86.59 (22.32)	136.88
sEMQN10	500	0.83 (0.15)	85.11 (23.20)	155.97
samEMQN10	500	0.84 (0.13)	84.27 (23.98)	160.86
samEMQN10mult	500	0.86 (0.10)	86.62 (22.36)	138.55

Table 3.4: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the different initialization strategies (including all 500 datasets) when $s = 5$.

'sEMQN10' obtained the lowest mean ARI and percent correctness with large standard deviations, but variation among percent correctness achieved by the other initialization methods (for example, standard deviation of 23.06 from 'samEMQN10mult') were also much larger than those shown in 'True' and 'sEM' results (standard deviations of 0.03).

Table 3.7 shows the results obtained by the various initialization strategies when the datasets with non-identifiable models were excluded from the analysis. We can see that the numbers of datasets included for all initialization methods were high compared with when dispersion parameters were set to 10 or 5. This may be due to the fact that as dispersion parameter decreases the data tends to be more like the negative binomial distribution than Poisson, which is what we assume for our model-based clustering approach. The results obtained when non-identifiable models were disregarded are much less variable since they excluded a lot of the outliers and we have higher mean ARI values and percentages of genes being clustered correctly. It is reasonable to believe that the EM estimation may be trapped in local points for a large number of iterations when non-identifiable models obtained, and this is supported by the lower average

Methods	Datasets included	ARI (sd)	Percent correctness (sd)	EM iterations
True	500	0.91 (0.02)	96.49 (0.88)	108.31
Random	254	0.91 (0.03)	96.03 (5.87)	188.26
sEM	497	0.90 (0.02)	96.45 (0.82)	44.37
samEM	373	0.90 (0.03)	96.13 (5.32)	84.55
sDirect	377	0.90 (0.02)	96.30 (3.91)	112.58
samDirect	393	0.90 (0.03)	95.91 (6.17)	134.83
samDirectmult	373	0.91 (0.03)	96.15 (5.52)	117.99
sEMQN5	377	0.90 (0.03)	96.12 (4.79)	128.11
samEMQN5	396	0.91 (0.03)	96.49 (1.55)	122.47
samEMQN5mult	402	0.90 (0.04)	95.98 (5.91)	104.60
sEMQN10	383	0.90 (0.03)	95.91 (6.82)	130.97
samEMQN10	377	0.90 (0.03)	95.97 (5.24)	107.19
samEMQN10mult	398	0.91 (0.02)	96.55 (0.83)	87.46

Table 3.5: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the different initialization strategies (excluding datasets with non-identifiable models) when $s = 5$.

EM iterations required when we only consider the datasets with identifiable models.

Varying percent of sampling

Since we evaluated the performance of our clustering approach with some initialization strategies having a sampling component, it is also of interest as to how the percent of sampling would affect the clustering ability. We have simulated 500 datasets independently with each dataset consists of 500 genes from four clusters over five time-points ($\pi_1 = 0.1, \pi_2 = 0.2, \pi_3 = 0.3$ and $\pi_4 = 0.4$) and dispersion parameter $s = 10$. We assessed the performance of the sampling initialization strategies when the sample consists of 50, 150, and 250 genes (10%, 30% and 50% of the entire dataset) respectively. While keeping other parameters constant, we hypothesize that the larger sample size (higher percent of sampling) would lead to more accurate clustering results but may require additional run-time.

The initialization strategies that are evaluated in this subsection are the methods: 'samDi-

Methods	Datasets included	ARI	Percent correctness	EM iterations
True	500	0.78 (0.03)	90.92 (7.45)	81.73
Random	500	0.68 (0.19)	72.88 (29.68)	171.92
sEM	500	0.78 (0.03)	90.61 (8.50)	54.33
samEM	500	0.75 (0.09)	85.80 (17.45)	124.22
sDirect	500	0.76 (0.07)	85.14 (20.46)	122.33
samDirect	500	0.76 (0.08)	85.30 (19.95)	124.67
samDirectmult	500	0.74 (0.10)	82.02 (23.71)	121.59
sEMQN5	500	0.75 (0.11)	83.74 (22.17)	140.83
samEMQN5	500	0.76 (0.08)	84.63 (20.69)	134.24
samEMQN5mult	500	0.75 (0.09)	84.12 (21.47)	108.77
sEMQN10	500	0.73 (0.14)	81.46 (23.64)	130.17
samEMQN10	500	0.76 (0.08)	85.00 (20.61)	135.08
samEMQN10mult	500	0.75 (0.08)	83.38 (23.06)	109.72

Table 3.6: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the different initialization strategies (including all 500 datasets) when $s = 2$.

rect', 'samDirectmult', 'samEM', 'samEMQN5', 'samEMQN5mult', 'samEMQN10' and 'samEMQN10mult'. Table 3.8 shows the run-time required for the different starting methods across the three settings of sampling percentage. While we hypothesized that the higher percent of sampling would lead to longer run-time being required, this theory was supported by results from the 'samEM' and most of the hybrid-EM methods (except for 'samEMQN10') but not all methods. Focusing on the methods with direct maximization as the starting procedures ('samDirect' and 'samDirectmult'), they required the longest time for the clustering to finish when they used 10% of the genes in the sample to search for starting values. The method 'samEMQN10' required more time when using 30% of the genes as the sample than when using 50% of the entire dataset. The run-time required for the clustering approach depends on the time required for the initialization part as well as the regular EM run which follows after selecting the starting values. It is true that initialization procedure performed on a larger sample should require a longer time, however, it may decrease the amount of time needed for the EM estimation if the large sample provided good starting values. This may be the reason why we did not observe in results from all initialization methods the hypothesized additional

Methods	Datasets included	ARI (sd)	Percent correctness (sd)	EM iterations
True	500	0.78 (0.03)	90.92 (7.45)	81.73
Random	340	0.78 (0.03)	91.22 (6.49)	170.03
sEM	499	0.78 (0.03)	90.71 (8.19)	52.43
samEM	437	0.78 (0.03)	91.46 (4.06)	103.75
sDirect	454	0.78 (0.03)	90.84 (7.64)	107.87
samDirect	448	0.78 (0.03)	91.17 (6.11)	110.82
samDirectmult	421	0.78 (0.03)	90.77 (7.98)	111.68
sEMQN5	438	0.78 (0.03)	90.82 (7.75)	120.42
samEMQN5	441	0.78 (0.03)	91.20 (6.32)	117.71
samEMQN5mult	441	0.78 (0.04)	90.70 (7.83)	99.83
sEMQN10	410	0.78 (0.03)	91.34 (5.13)	112.08
samEMQN10	446	0.78 (0.03)	91.03 (6.82)	114.18
samEMQN10mult	430	0.78 (0.03)	91.56 (4.09)	97.80

Table 3.7: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the different initialization strategies (excluding datasets with non-identifiable models) when $s = 2$.

run-time being required as the percent of sampling increased.

Table 3.9 shows the mean ARI and percent correctness obtained, as well as the mean EM iterations required, by the various initialization strategies across the three settings of percent of sampling. In terms of ARI, the method 'samEMQN5mult' seems to have performed the best with the highest mean ARI compared to the other methods. The initialization strategies did not show much improvement in clustering accuracy as the percent of sampling changed from 10% to 50%. Most strategies obtained the highest ARI when 30% of the genes were used in the initial sample. The same pattern was shown in the percent of genes being correctly classified by the methods. Also, most methods required the least number of EM iterations when only 30% of genes were used to obtain initial values.

If we only consider the datasets with identifiable models, the corresponding results are shown in Table 3.10. Sampling 30% of the genes gave us the highest number of datasets with identifiable models for most initialization strategies (except for 'samDirectmult'). There

doesn't seem to be any specific patterns in terms of ARI and percent correctness obtained as the sampling percentage changed. Excluding the datasets with non-identifiable models allowed us to disregards results with bad clustering performance, hence the mean ARI and percent correctness are both higher (with much less variation) when we only consider the datasets with identifiable models. The EM iterations required are also much lower when we exclude the datasets with non-identifiable models. The methods 'samEM' and 'samEMQN10mult' required the smallest number of EM iterations for the regular EM run compared to the other initialization strategies. Some methods required the least number of EM iterations with percent of sampling being 30%, while some other methods required least with 50% genes in initial sample.

Methods	Percent of sampling		
	10 %	30 %	50 %
samDirect	57.92	26.39	33.59
samDirectmult	40.84	27.64	28.62
samEM	22.87	28.28	28.29
samEMQN5	14.96	29.62	39.19
samEMQN5mult	20.80	52.63	66.35
samEMQN10	27.78	38.17	31.54
samEMQN10mult	26.82	41.54	77.34

Table 3.8: Run-time required for the various initialization procedures with sampling components (10%, 30% and 50% of sampling).

Methods	Datasets included			ARI (sd)						Percent correct (sd)						EM iterations		
	10%	30%	50%	10%		30%		50%		10%		30%		50%		10%	30%	50%
samDirect	500	500	500	0.86	(0.16)	0.86	(0.17)	0.85	(0.15)	83.61	(24.25)	84.17	(24.33)	82.50	(24.99)	199.39	179.76	208.97
samDirectm	500	500	500	0.86	(0.15)	0.85	(0.17)	0.84	(0.18)	83.87	(25.09)	82.06	(26.12)	82.77	(25.34)	146.31	166.00	195.33
samEM	500	500	500	0.84	(0.18)	0.86	(0.16)	0.85	(0.18)	84.32	(23.66)	85.61	(22.33)	83.17	(24.28)	145.18	156.15	151.58
samEMQN5	500	500	500	0.84	(0.18)	0.86	(0.16)	0.85	(0.17)	80.11	(26.93)	83.98	(23.90)	83.07	(24.46)	187.17	181.04	190.45
samEMQN5m	500	500	500	0.87	(0.15)	0.89	(0.13)	0.87	(0.16)	84.69	(24.94)	86.14	(23.69)	85.67	(23.79)	164.73	129.88	142.23
samEMQN10	500	500	500	0.86	(0.16)	0.85	(0.16)	0.86	(0.16)	83.74	(24.73)	82.95	(24.79)	83.12	(24.99)	205.89	196.99	166.78
samEMQN10m	500	500	500	0.86	(0.15)	0.89	(0.12)	0.88	(0.13)	83.31	(26.08)	89.30	(20.34)	85.93	(22.64)	156.89	130.62	140.43

Table 3.9: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation) and mean EM iterations required for the different sampling initialization strategies across various sampling percentages.

Methods	Datasets included			ARI (sd)						Percent correct (sd)						EM iterations		
	10%	30%	50%	10%		30%		50%		10%		30%		50%		10%	30%	50%
samDirect	312	319	310	0.94	(0.03)	0.94	(0.03)	0.94	(0.03)	97.10	(7.19)	97.26	(6.35)	96.91	(7.86)	157.71	129.13	164.50
samDirectm	312	319	310	0.94	(0.04)	0.94	(0.04)	0.94	(0.04)	96.80	(9.31)	96.83	(8.79)	96.81	(8.31)	105.25	101.83	139.70
samEM	312	319	310	0.94	(0.03)	0.95	(0.03)	0.94	(0.03)	96.89	(8.50)	97.69	(4.60)	96.44	(9.75)	95.29	88.55	76.49
samEMQN5	312	319	310	0.94	(0.04)	0.94	(0.03)	0.94	(0.03)	95.85	(11.29)	96.89	(7.79)	96.65	(8.43)	130.21	120.41	128.97
samEMQN5m	312	319	310	0.94	(0.03)	0.95	(0.03)	0.95	(0.03)	97.45	(5.01)	97.53	(5.52)	97.14	(7.24)	122.51	94.22	93.94
samEMQN10	312	319	310	0.94	(0.03)	0.94	(0.04)	0.95	(0.03)	96.82	(8.79)	95.82	(11.49)	97.26	(7.18)	127.54	116.21	113.70
samEMQN10m	312	319	310	0.95	90.02)	0.95	(0.03)	0.95	(0.02)	97.78	(3.37)	97.43	(5.98)	97.84	(3.32)	107.45	70.72	76.88

Table 3.10: Number of datasets included, mean Adjusted Rand Index (standard deviation), mean percent correctness (standard deviation) and mean EM iterations required for the different sampling initialization strategies across various sampling percentages (excluding datasets with non-identifiable models).

3.4 Real data analysis

3.4.1 Fibroblast data

The initialization strategies evaluated in the simulation studies are applied to the real dataset from Dudley et al. (2002). The dataset measured gene expression of the response to fibroblasts to fetal bovine serum in order to investigate the development of human vertebrate limb. The dataset has been described in detail in Chapter 1. For the comparison of initialization strategies, all methods were started using the four-group model since that was the optimal model chosen for this dataset when model selection was performed in section 2.4.1.

Table 3.11 shows the EM iterations used, logliklihood and BIC obtained by the various initialization methods. The goal of the comparison is to find out which initialization method yields the best likelihood or model in terms of BIC. The method 'sEM' gave the best BIC when compared to the other methods with the lowest number of EM iterations required. The methods 'samDirect' and 'samDirectmult' also resulted in good BIC values but required more EM iterations to complete clustering the dataset. When the model-based clustering approach was started with the methods 'sDirect' and 'samEMQN5', the EM algorithm was not able to converge within 1000 iterations and resulted in likelihoods which were far away from the likelihoods obtained by the other methods upon convergence. It should be noted that the method 'Random' required almost five times as much run-time as needed by the other methods because it performed the clustering approach five times with random starting values to find the best solution. This becomes problematic when we are presented with large datasets such as this one, with a sample size of 6,153, it required days to obtain the clustering results. In this sense, the other initialization methods had obvious benefits over the 'Random' methods by being able to achieve reasonable result under much less run-time.

	EM iterations	log-likelihood	BIC
Random	440	-1438.35	-1521.24
sEM	**50**	-1416.60	**-1499.49**
sEMQN5	628	-1794.61	-1877.50
sEMQN10	291	-1438.60	-1521.48
sDirect	1000	-3861.38	-3944.26
samEM	362	-1438.35	-1521.23
samEMQN5	1000	-3684.85	-3767.73
samEMQN10	229	-1438.03	-1520.91
samDirect	336	-1438.11	-1520.99
samEMQN5mult	246	-1438.36	-1521.24
samEMQN10mult	376	-1438.34	-1521.22
samDirectmult	191	-1437.72	-1520.60

Table 3.11: Results of initialization strategies used on the fibroblasts data using mixtures of negative binomials with four components.

3.4.2 Fruit flies data

The initialization strategies were also applied to the RNA-seq dataset described in Chapter 1. The *D. melanogaster* dataset (Graveley et al., 2011) measured gene expression for studying developmental stages spanning the life cycle of fruit flies. The RNA-seq dataset was clustered using our model-based clustering approach with the various initialization methods using mixtures of four components, which was the best component-model discovered for this dataset in section 2.4.2.

The results from the various initialization methods are shown in Table 3.12. Same as the other dataset, we notice that the method 'sEM' produced the model with the largest BIC value with a small number of EM iterations required. Some of the other methods also obtained solutions very similar to the one resulted from 'sEM' but required more EM iterations. The method 'samDirect' only required 14 EM iterations for the algorithm to converge thus needing the smallest amount of time to finish clustering of the dataset, but the likelihood resulted was not close to the maximum likelihood obtained by 'sEM'. The 'samEMQN10' method seems to be the second best initialization method in terms of time requirement and likelihood obtained,

since it required a comparatively low amount of run-time to produce a solution very close to the one resulted from 'sEM'. The 'Random' method required a small amount of run-time because two out of the five runs resulted in non-identifiable models quickly, so only three runs were used to select the best solution.

	EM iterations	Time (hr)	log-likelihood	BIC
Random	70	7.33	-1955.01	-2014.82
sEM	60	11.72	-1955.01	**-2014.82**
sEMQN5	121	7.39	-1955.10	-2014.91
sEMQN10	110	6.24	-1955.03	-2014.84
sDirect	114	5.85	-1955.07	-2014.88
samEM	137	6.24	-1955.11	-2014.92
samEMQN5	69	2.71	-2024.83	-2084.64
samEMQN10	97	4.09	-1955.11	-2014.92
samDirect	**14**	0.68	-2024.17	-2083.98
samEMQN5mult	139	5.96	-2039.73	-2099.54
samEMQN10mult	97	5.27	-1955.11	-2014.92
samDirectmult	114	3.73	-1955.06	-2014.87

Table 3.12: Results of initialization strategies used on the fruit flies data using mixtures of negative binomials with four components.

3.5 Summary

In this chapter, we have proposed and evaluated some initialization strategies suitable for model-based clustering of time-course RNA-seq data. Initialization methods based on short runs of estimation and sampling components were compared against the commonly used approach of multiple random starting points. Simulation results showed that random initialization had poor performance in terms of inaccurate clustering and long run-time. Our results reflected that running short runs of EM and using their best solution as starting values provide good clustering performance with identifiable models, and the tradeoff between the quality of clustering performance and run-time is good. All initialization strategies had similar clustering

ability when regarding only the identifiable models. Sampling 30% of the genes into an initial sample and using hybrid-EM methods to obtain starting values seem to be a good alternative initialization strategy.

Chapter 4

Missing Data in high-dimensional datasets

4.1 Introduction

Recently biological research studies often involve the use of high-throughput technologies such as gene expression microarrays, mass-spectrometry-based proteomic assays, and quantitative protein and genetic interaction screens. The high-dimensional datasets produced from these biotechnologies often suffer from missing values due to various reasons, and how to deal with such missing information becomes a key issue in the data analysis. In this chapter, we evaluate the performance of several imputation methods on time-course genomic datasets and propose a new cluster-based imputation method which is suitable for RNA-seq data. We first give an overview of the problem of missingness in large-scale datasets, the imputation methods that have been proposed to deal with the missing values, and the impact of these imputations on downstream analyses such as cluster analysis.

In proteomic research, researchers can use liquid chromatography-mass spectrometry-based assays to study the profiling of complex peptide mixtures in large samples. High-throughput mass spectrometry can provide identification and quantification of thousands of peptides, but the problem of missing peptide identifications and abundance values often occur, especially in

large-scale clinical proteomic studies (Aittokallio, 2009; Karpievitch et al., 2012). Missing values may arise when a peptide is not identified in all samples in the experiment, as an abundance value would be declared missing if a peptide is not present, or if it is present in the sample but at a concentration below a certain threshold that cannot be detected by the instrument.

In spotted cDNA microarrays, each spot on the array is often assigned to a unique gene, so if a spotting problem corresponds to a missing value then it would lead to the loss of information for that particular gene (Brás and Menezes, 2007). Imperfections in the spots may be due to different mechanisms. Smudges and scratches on the slides or deposition of dust may corrupt signals at the spots during the microarray experiment and the spots will be noted as missing when the array image is digitized. Hybridization failures or bleed-over from neighbouring spots can also affect the spot intensity and cause an increase in the background intensity at those spots when the array image is scanned, producing negative expression levels being measured, which would then be treated as missing (Hourani and El Emary, 2009; Jörnsten et al., 2005). In some large-scale studies, the problem of missing values may be severe as more than 85% of the genes may be affected with at least one missing value (de Brevern et al., 2004; Ouyang et al., 2004). Since many downstream analyses in genomic studies require complete data matrices, the problem of missingness have been an important area of research and literature mainly focus on missing values in microarray data.

4.1.1 Missing mechanism and treatments of missing values

Missing data can arise due to different reasons and the missing mechanism can be categorized into three types (Little and Rubin, 1987): missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR). The missingness is MCAR if the probability that a value is missing is unrelated to the observed or missing values, that is, if missingness on response Y is unrelated to the observed values of Y. Missing data is classified as

MAR if missingness on Y is only related to the observed values of Y, so that the missing data may depend on the observed variables but not on the missing variable itself. MCAR and MAR are sometimes referred to as 'ignorable' missingness and valid statistical methods have been developed to analyze data with such missing mechanisms. The case of MNAR is the most difficult to deal with because the probability that a value is missing depends on the missing value itself. Missingness with MNAR is often termed 'non-ignorable' and statistical analysis on this type of missing data is complicated because it relies on the ability to model the process which generates the missing data. When working with microarray data, one may expect the missingness arises from a random distribution due to hybridization failures, dusts on the slides, etc., at random spots on the array. However, when the signal intensity at a spot is too low to be distinguished from the background it may be declared as a missing value. This represents the situation where the missing pattern actually depends on the spot intensity itself, thus leading to a case of a mixture of MAR and MNAR in the dataset (Brás and Menezes, 2007).

Many downstream analyses require complete gene expression matrices, such as classification methods (e.g. support vector machines), multivariate statistical analysis methods (e.g. principal component analysis, singular value decomposition), and some model-based or hierarchical clustering methods. Thus, in order to carry on proper analyses, there is a need to obtain complete datasets. There are different ways to deal with datasets with missingness. For some studies, it may be possible to repeat the experiments and try to obtain a new set of results with no missingness. However, this is often not feasible for microarray experiments involved in large-scale studies due to economic reasons or biological sample availability.

When working with datasets with missingness, researchers can choose between: eliminating the data objects with missing values, estimating the missing values or ignore the missing values during the data analysis. When a spot on the microarray corresponds to a specific gene, it might lead to severe loss of information if we simply throw away a gene when its expression

is not measured in some of the experiments. If the downstream analysis method require complete data matrices, then it would not be possible to simply ignore the missing values. In such situations, researchers would wish to estimate the missing values using imputation methods or use analysis methods which can accommodate missingness (for example, Schliep and others (2003) have developed a hidden Markov models approach which can handle gene expression data with missing values). However, analysis methods may not be robust to the presence of missing values, and may have reduced efficiency even if it can accommodate missingness in the dataset.

The similar problem of missing values also occur when RNA-seq is used to measure gene expression, thus it is of interest to investigate the impact of missing values in RNA-seq data. Due to the difference between data types in microarray and RNA-seq data, methods suitable for microarray data cannot be directly applied to data from RNA-seq experiments. Efficient imputation and analysis methods for missingness in the discrete count reads from RNA-seq data need to be developed and evaluated.

4.2 Methods

4.2.1 Imputation methods

To evaluate the performances of various imputation procedures on time-course RNA-seq data, we included seven commonly used imputation methods that are readily available for researchers. We have included simple methods (imputing with zeros and row or column averages), straightforward methods tailored for observations over time (imputing with observed values from the previous or the next time point), and imputation methods which account for correlation structures in the datasets (KNNimpute and SVDimpute).

- ZEROimpute: replacing missing values with zeros

- ROWimpute: replacing missing values with the row averages (i.e., mean of observed values from the same gene over time)

- COLimpute: replacing missing values with the column averages (i.e., mean of observed values from different genes at the same time point)

- LOCF (Last observation carried forward): replacing a missing entry with the observed value from the previous time point of the same gene. If the missing value is at the first time point then the method will replace the missing entry with the next observed value from the same gene.

- NOCB (Next observation carried backward): replacing a missing entry with the observed value in the next time point of the same gene. If the missing value is at the last time point then the method will replace the missing entry with the previously observed value from the same gene.

- KNNimpute: replacing missing values with the KNNimpute method (Troyanskaya et al., 2001) using the function *kNNImpute* in the R package *imputation* (Wong, 2013).

- SVDimpute: replacing missing values with the SVDimpute method (Troyanskaya et al., 2001) using the function *SVDImpute* in the R package *imputation* (Wong, 2013).

4.2.2 Datasets and missing data

We used two types of datasets in our evaluation of imputation methods. The first type consists of simulated datasets with discrete counts as outcomes that are similar to the type of data resulting from the RNA-seq technology. The second is a real RNA-seq dataset from *D. melanogaster* (Graveley et al., 2011) that measured gene expression during the life cycle of fruit flies (described in Chapter 1). The first type, the artificial datasets, are designed to resemble time-course gene expression patterns in a total of 500 genes in four clusters over five time points, with the clusters consisting of 50, 100, 150, and 200 genes respectively. We considered

datasets simulated with dispersion parameter $s = 2$ and the expression patterns follow those shown in Figure 2.4 in Chapter 2 of this dissertation.

Since both types of datasets have complete observations, missing entries were artificially introduced to the complete matrices by discarding values. We have considered both the MCAR and MAR scenarios, varying the overall missing probabilities from 5% to 25%. Given that missing entries may be introduced at random (note that read counts of zeros are not missing values), such that we can mimic the situation by randomly removing the specified percentage of entries from a complete data matrix to generate MCAR missingness. However, in some cases, the missing values may not be randomly introduced but may be dependent on the genes. We follow the procedure used by Brás and Menezes (2007) to generate these MAR scenarios: first randomly sample some rows (genes) of the complete matrix, then assign missing values to some entries of the selected rows such that expression values at different time points of the selected genes have a chance of being missing.

For each dataset with the various percentages of missingness, the imputation methods were applied and the accuracy of the methods were evaluated and compared using the normalized root mean squared error (RMSE). Let $\hat{y}_{jt}$ be the estimated expression value and y_{jt} be the original value for gene j at time point t, the accuracy of an imputation method is measured by the RMSE over the entire matrix:

$$\text{RMSE} = \frac{\text{root mean squared difference between}(y_{jt}, \hat{y}_{jt})}{\text{root mean squared}(y_{jt})}$$

This evaluation measure proposed by Ouyang et al. (2004) have been used by other authors for measuring imputation accuracy (Jörnsten et al., 2005; Sehgal et al., 2005). A particular feature of this metric is that the RMSE for ZEROimpute always equals to one so it is easy to compare imputation accuracy and difficulty across datasets. For comparison, the simulation

study included repeating the missing value generation and imputation procedure 500 times for each dataset and each imputation method. Figure 4.1 shows the schematic illustration of the simulation study.

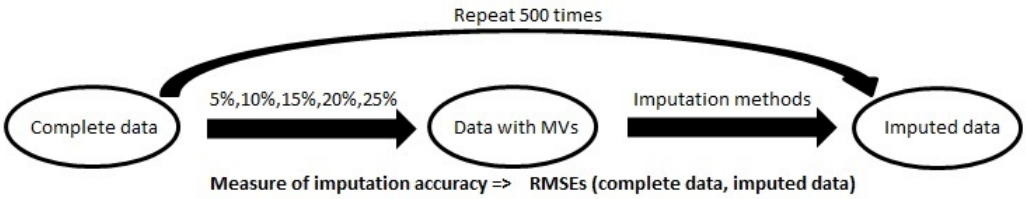

Figure 4.1: Schematic illustration of the simulation study: evaluation by RMSE comparing accuracy of imputation methods.

4.2.3 Results

MCAR simulated data

Each simulated dataset is a 500 x 5 matrix, representing expression patterns of 500 genes over five time points. For KNNimpute and SVDimpute, different numbers were used as the parameter k (number of neighbours in KNNimpute and number of eigenvectors in SVDimpute) to examine the effect of the parameter selection. Figure 4.2 shows the RMSE values obtained when using KNNimpute with k ranging from 5 to 30 across the missing probabilities from 5% to 25%. It is clear that as the missing probability increases the accuracy of KNNimpute decreases, and that more neighbouring genes in the candidate set seems to improve the accuracy of imputation shown by the lower RMSE values. Figure 4.3 displays the effect of k on the imputation accuracy across different missing probabilities. It shows that the performance of KNNimpute would improve as k increases and then become stabilized after k reaches a certain value, such as threshold of $k = 35$ when missing probability was 5% and $k = 60$ for the cases with higher missing probabilities.

Similar evaluation procedures were performed on SVDimpute to examine the effect of k

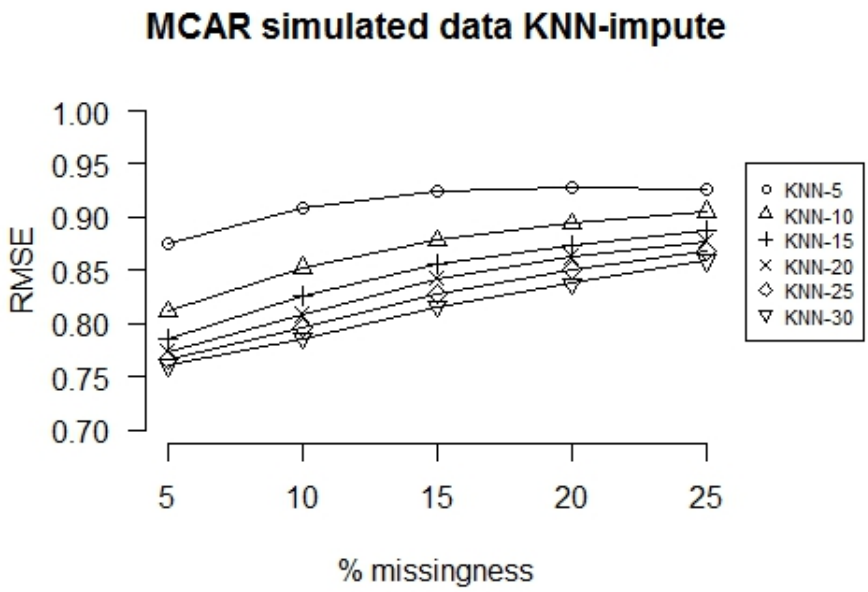

Figure 4.2: MCAR simulated data: RMSE obtained by KNNimpute with various k parameter across different missing probabilities.

on the imputation accuracy. The RMSE obtained from SVDimpute with different numbers of eigenvectors across the datasets with 5% to 25% of values missing are shown in Figure 4.4, and it is clear that SVDimpute with $k = 1$ is not appropriate. The missing percentage does not seem to have an effect on the imputation accuracy for SVDimpute when k ranged from two to five. Figure 4.5 shows that the RMSE values from SVDimpute with $k = 2$ is clearly the lowest compared to other values of k across all five setting of missing probabilities. In general, SVDimpute is more robust to missing probabilities than KNNimpute but KNNimpute was more accurate when only 5% or 10% of entries were missing.

Table 4.1 displays the RMSE values from all the evaluated imputation procedures (with KNNimpute and SVDimpute using the optimal k parameters) across the different missing probabilities. It shows that the simple imputation methods of ROWimpute and NOCB performed worse than just simply replacing missing values with zeros, since they produced RMSE values higher than ZEROimpute (which always give RMSE of one). COLimpute performed surpris-

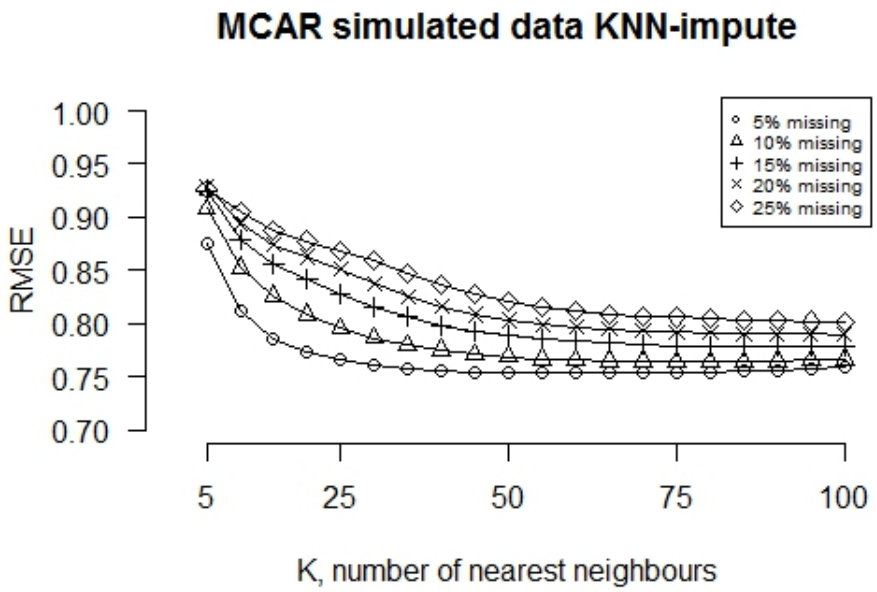

Figure 4.3: MCAR simulated data: RMSE obtained by KNNimpute when k ranges from 5 to 100.

	Missing probabilities				
Methods	5%	10%	15%	20%	25%
ROWimpute	1.09 (0.07)	1.09 (0.05)	1.10 (0.05)	1.11 (0.06)	1.12 (0.05)
COLimpute	0.83 (0.05)	0.83 (0.03)	0.83 (0.03)	0.83 (0.02)	0.83 (0.02)
LOCF	0.99 (0.08)	0.99 (0.06)	0.99 (0.05)	1.01 (0.05)	1.02 (0.05)
NOCB	1.30 (0.22)	1.30 (0.17)	1.29 (0.14)	1.30 (0.13)	1.31 (0.11)
KNNimpute	0.75 (0.07)	0.76 (0.04)	0.78 (0.03)	0.79 (0.03)	0.81 (0.03)
SVDimpute	0.82 (0.08)	0.81 (0.05)	0.82 (0.04)	0.82 (0.03)	0.82 (0.03)

Table 4.1: MCAR simulated data: Comparison of RMSE of the imputation methods (means and standard deviations of 500 simulation runs).

ingly well with RMSE values similar to those obtained by the more sophisticated KNNimpute ($k = 75$) and SVDimpute ($k = 2$). The RMSE values produced by ROWimpute, LOCF and KNNimpute had obvious increases as the missing probabilities increased. Out of all the imputation methods, NOCB produced the highest and most variable RMSE values, indicating that it is not a suitable method for this type of datasets.

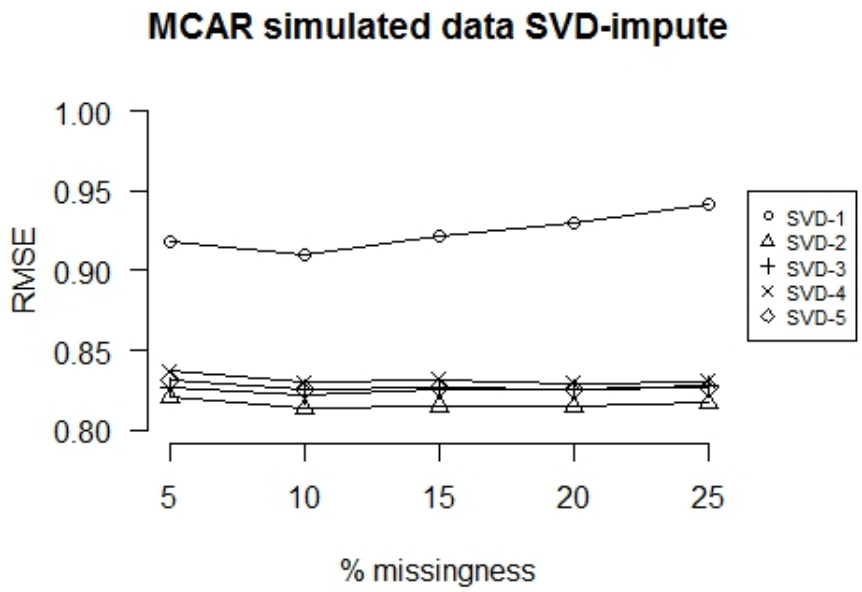

Figure 4.4: MCAR simulated data: RMSE obtained by SVDimpute with various k parameter across different missing probabilities.

MAR simulated data

The MAR simulated data were generated by first randomly selected genes from the 500 genes in the complete matrices, then removed some of the entries from the selected genes to obtain missing probabilities ranging from 5% to 25% in the 500 x 5 matrices. Figure 4.6 shows that the RMSE values decreased from approximately 0.92 to 0.84 as the k parameter for KNNimpute increased from 5 to 30, and the accuracy of KNNimpute decreases slightly when the datasets contained more missing values. KNNimpute appeared to be less sensitive to changes in missing probabilities in the MAR datasets compared to the MCAR datatsets. Similar to the case of MCAR missingness, Figure 4.7 shows that in datasets with MAR missingness, RMSE values decreased when k increases but stabilized after a certain threshold, and the effect of missing rate on imputation accuracy was not obvious when k gets larger than 50. When k is 75 or larger, the RMSE values converged and it reflects that the missing probabilities seemed to have essentially no effects on the performance of KNNimpute.

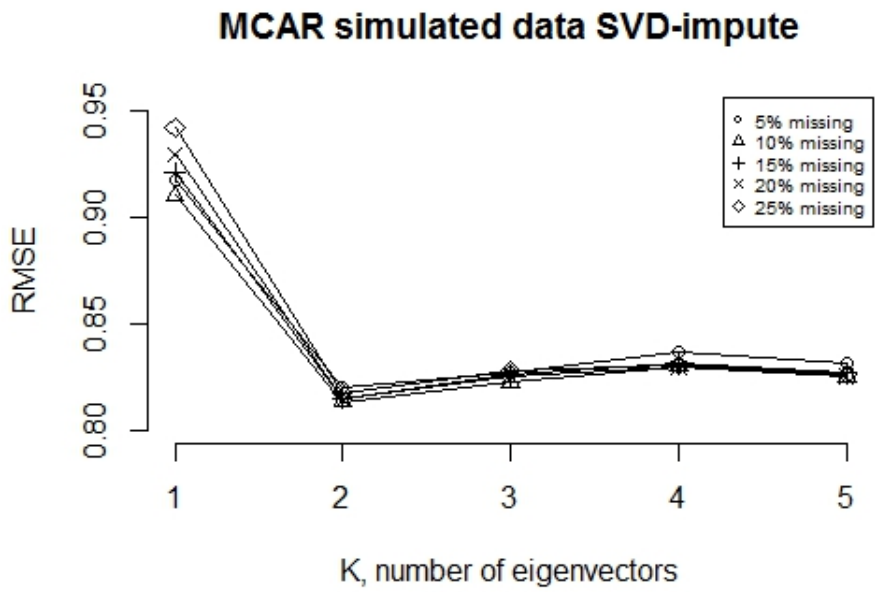

Figure 4.5: MCAR simulated data: RMSE obtained by SVDimpute when k ranges from 1 to 5.

When SVDimpute with various k values were performed on the MAR simulated data, we obtained similar patterns of RMSE values as in the MCAR simulated data. From Figure 4.8 and Figure 4.9, it can be seen that SVDimpute with $k = 2$ produced the most accurate imputation values but the differences between $k = 2, 3, 4$ or 5 were not obvious. SVDimpute with $k = 1$ had the worst performance with mean RMSE of approximately 0.885 across the various missing probabilities settings, while the other parameter settings produced RMSE values between 0.82 to 0.84. SVDimpute outperformed KNNimpute in the case of MAR missingness since KNNimpute was only able to achieve similar accuracy as SVDimpute when KNNimpute was performed with k set to 35 or larger.

Table 4.2 compares the performances of the different imputation methods and it shows that same as for the MCAR datasets, the NOCB method produced the least accurate imputation values for the MAR simulated datasets out of all the methods. The table contained the RMSE values obtained from KNNimpute with $k = 60$ and SVDimpute with $k = 2$, which were the optimal k values for the two methods respectively for these datasets. The simple methods

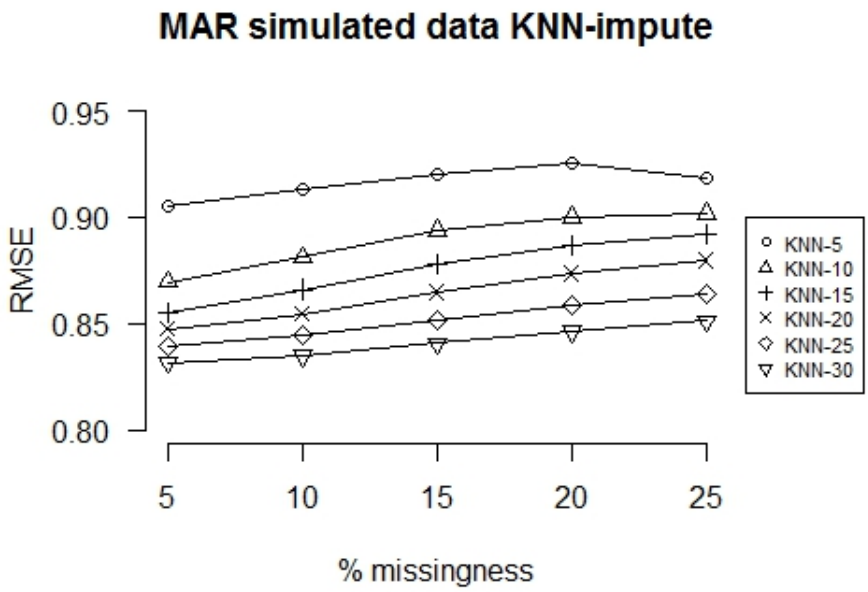

Figure 4.6: MAR simulated data: RMSE obtained by KNNimpute with various k parameter across different missing probabilities.

ROWimpute, LOCF and NOCB all had mean RMSE values above one with large standard deviations, showing that they were less accurate than ZEROimpute. Similar to the MCAR situation, COLimpute estimated missing entries with good accuracy, outperforming other simple imputation methods and had comparable performance as the more sophisticated SVDimpute and KNNimpute. Overall, RMSE values from the imputation methods were higher in the MAR situation compared to the MCAR missingness, showing that the MAR missingness is more difficult to handle than completely random missingness.

MCAR fruit flies data

The fruit flies dataset consists of expression values from 542 genes over six time points, and missing entries were randomly selected from the complete matrix to create MCAR missingness. KNNimpute with different k values were performed for missing probabilities ranging from 5% to 25% and the RMSE values obtained are shown in Figure 4.10. From the graph,

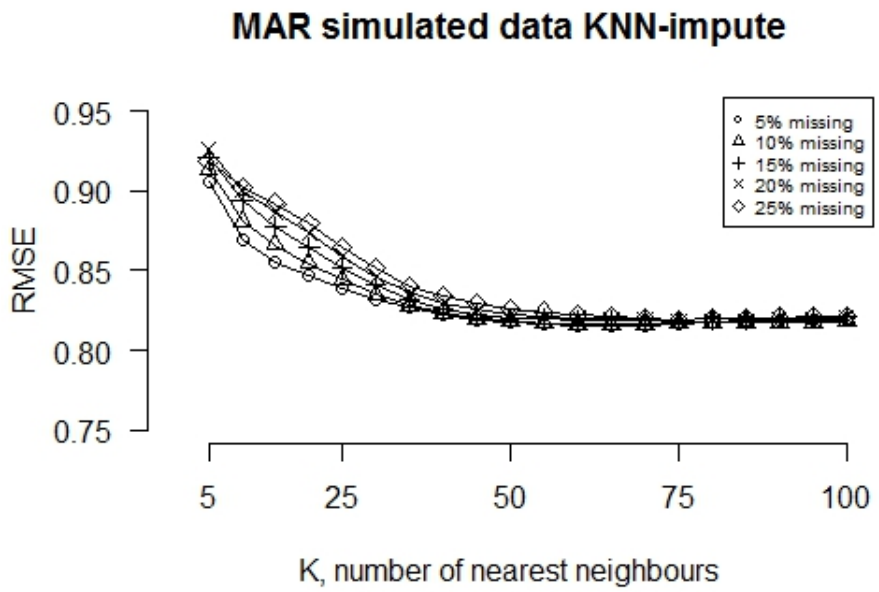

Figure 4.7: MAR simulated data: RMSE obtained by KNNimpute when k ranges from 5 to 100.

	Missing probabilities				
Methods	5%	10%	15%	20%	25%
ROWimpute	1.20 (0.20)	1.20 (0.16)	1.19 (0.11)	1.20 (0.12)	1.20 (0.11)
COLimpute	0.83 (0.05)	0.83 (0.04)	0.83 (0.03)	0.83 (0.02)	0.83 (0.02)
LOCF	1.11 (0.19)	1.12 (0.15)	1.11 (0.11)	1.12 (0.12)	1.12 (0.10)
NOCB	1.34 (0.28)	1.34 (0.21)	1.33 (0.17)	1.34 (0.16)	1.33 (0.14)
KNNimpute	0.82 (0.07)	0.82 (0.04)	0.82 (0.03)	0.82 (0.03)	0.82 (0.03)
SVDimpute	0.83 (0.06)	0.82 (0.04)	0.82 (0.03)	0.83 (0.03)	0.82 (0.02)

Table 4.2: MAR simulated data: Comparison of RMSE of the imputation methods (means and standard deviations of 500 simulation runs).

it is clear that KNNimpute had decreasing accuracy as the missing probability increased. It is also noted that KNNimpute with five nearest neighbours in the candidate set ($k = 5$) produced higher RMSE values than KNNimpute with other k parameters as the missing probability increased. The performances of KNNimpute with k varying from 10 to 30 were very similar, and this is also indicated by Figure 4.11 which showed the changes in RMSE values as the parameter k changes from 5 to 100 across the different missing probability settings. Overall, it appears that $k = 15$ can be seen as the optimal number of nearest neighbours to be included in the candidate set when performing KNNimpute on the MCAR fruit flies dataset. Disre-

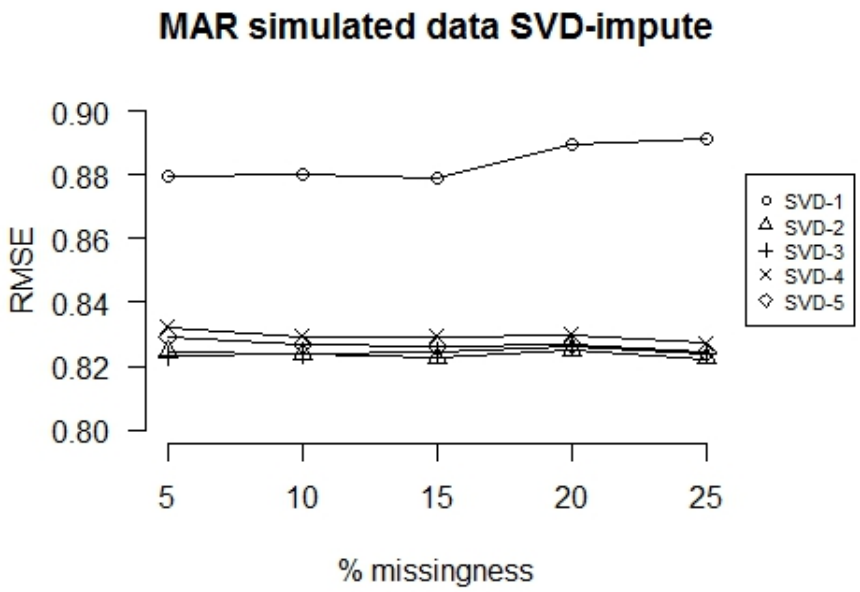

Figure 4.8: MAR simulated data: RMSE obtained by SVDimpute with various k parameter across different missing probabilities.

gard the missing probability, RMSE values obtained by KNNimpute increased as k increased from 20 to 100, indicating that larger candidate sets may not lead to more accurate imputations.

The performances of SVDimpute on the MCAR fruit flies dataset across different missing probability settings and various number of eigenvectors (k) are shown in Figure 4.12 and Figure 4.13. From both graphs, it is clear that $k = 2$ was the optimal parameter setting for SVDimpute across the various missing probabilities. Figure 4.12 shows that SVDimpute with $k = 6$ had the worst accuracy as missing probabilities changed. SVDimpute with $k = 1$ and $k = 6$ produced constant RMSE values regardless the increase in missing probabilities, whereas with the other four parameter settings ($k = 2$ to $k = 5$) SVDimpute obtained higher RMSE values with higher amount of missingness in the dataset. KNNimpute outperformed SVDimpute for this dataset since RMSE values from KNNimpute ranged from 0.25 to 0.35, whereas SVDimpute produced values between 0.3 to 0.6.

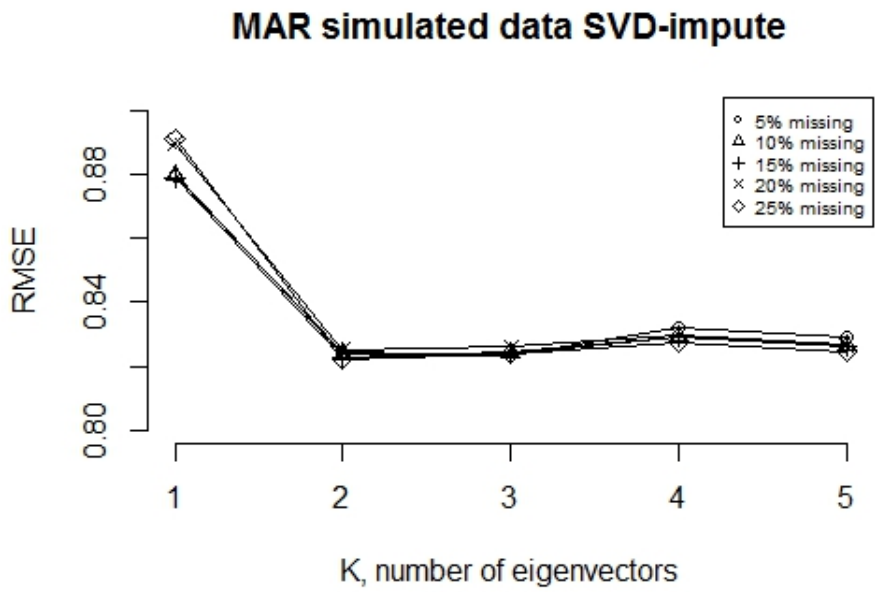

Figure 4.9: MAR simulated data: RMSE obtained by SVDimpute when k ranges from 1 to 5.

Table 4.3 shows the RMSE values obtained by the various imputation methods when missing probabilities ranged from 5% to 25%. The two more sophisticated imputation methods, KNNimpute and SVDimpute, appeared to be more accurate than the other imputation methods, with KNNimpute (k=15) performing slightly better than SVDimpute (k=2) across all cases. The imputation method LOCF made the worst predictions of the missing values with the largest and most variable RMSE values. All imputation methods obtained RMSE values below one, meaning that any of these methods would be an improvement over imputing missing values with zeros.

MAR fruit flies data

The MAR fruit flies data were generated from the same procedure as the MAR simulated data to produce datasets with missing probabilities ranging from 5% to 25% in the dataset. Figure 4.14 shows the performance of KNNimpute when the number of nearest neighbours in the candidate set and the missing probabilities varied. Overall, the imputation method performed

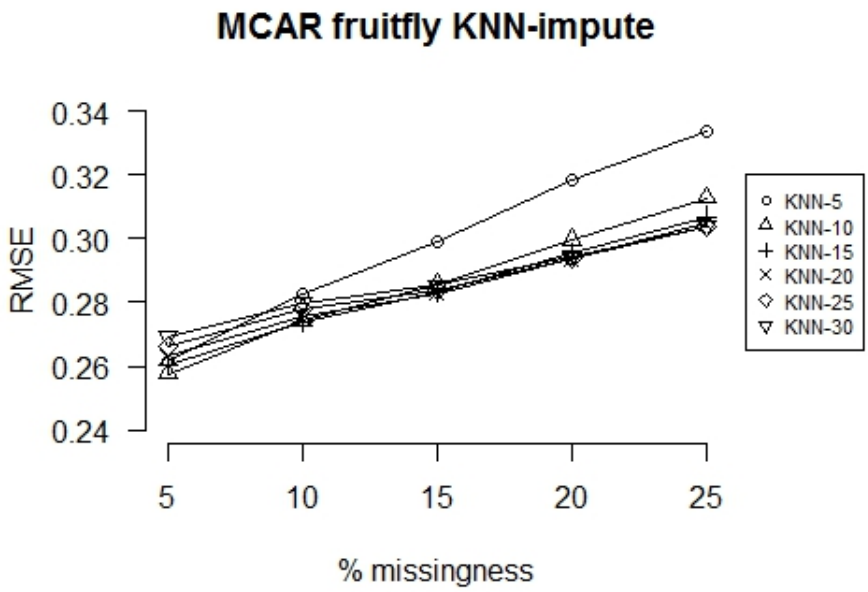

Figure 4.10: MCAR fruit flies data: RMSE obtained by KNNimpute with various k parameter across different missing probabilities.

worse as the missing probabilities increased from 5% to 25%. Across all the different missing probabilities, the RMSE values obtained by KNNimpute decreased as the k parameter ranged from 5 to 30, with very similar results obtained when k is 20 or larger. However, Figure 4.15 shows that a larger candidate set does not always lead to better predictions of the missing values. It appears that $k = 20$ is the optimal parameter value when performances were averaged across the five different missing probabilities. The accuracy of KNNimpute increased sharply when k increased from 5 to 20, then the method's performance worsened as k increased.

The performances of SVDimpute with different number of eigenvectors (parameter k) are displayed in Figure 4.16 and Figure 4.17. It is clear that the accuracy of SVDimpute stayed rather constant disregard to changes in missing probabilities. SVDimpute with one or two eigenvectors had similar results, with $k = 2$ having the lowest RMSE values obtained. As k increased from three to six, the accuracy of SVDimpute decreased linearly as larger RMSE values were achieved. Similar to the case of MCAR missingness, SVDimpute produced larger

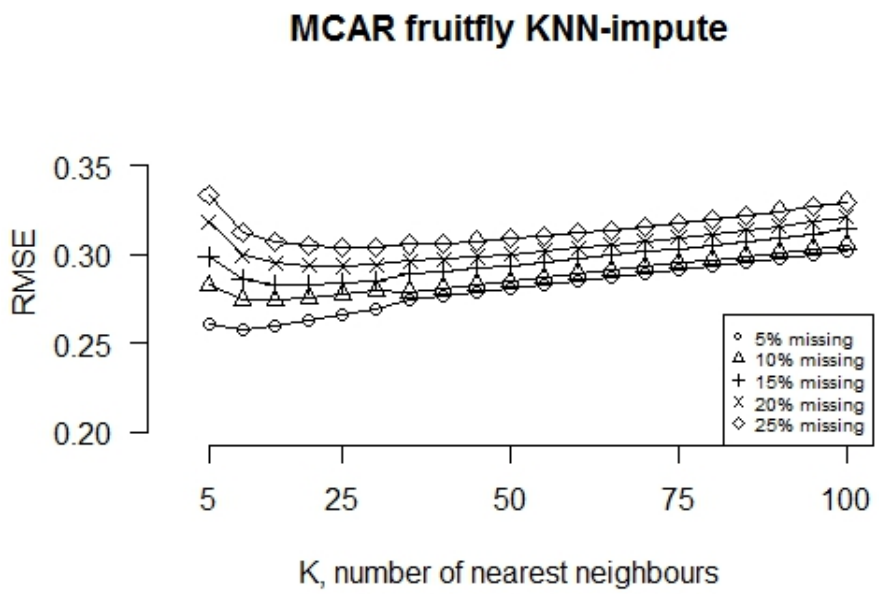

Figure 4.11: MCAR fruit flies data: RMSE obtained by KNNimpute when k ranges from 5 to 100.

	Missing probabilities				
Methods	5%	10%	15%	20%	25%
ROWimpute	0.51 (0.03)	0.52 (0.02)	0.52 (0.02)	0.52 (0.02)	0.53 (0.01)
COLimpute	0.60 (0.03)	0.60 (0.02)	0.60 (0.02)	0.60 (0.01)	0.60 (0.01)
LOCF	0.73 (0.04)	0.72 (0.03)	0.71 (0.03)	0.71 (0.02)	0.70 (0.02)
NOCB	0.63 (0.04)	0.63 (0.03)	0.62 (0.02)	0.62 (0.02)	0.62 (0.02)
KNNimpute	0.26 (0.03)	0.27 (0.02)	0.28 (0.02)	0.30 (0.02)	0.31 (0.01)
SVDimpute	0.30 (0.03)	0.31 (0.03)	0.31 (0.02)	0.32 (0.02)	0.33 (0.02)

Table 4.3: MCAR fruit flies data: Comparison of RMSE of the imputation methods (means and standard deviations of 500 simulation runs).

RMSE values than those obtained by KNNimpute. This indicates that KNNimpute might be more suitable than SVDimpute for the fruit flies dataset.

Table 4.4 compares the performances of the different imputation methods when dataset consisted of various percentages of missingness. The RMSE values obtained by the imputation methods under the case of MAR (Table 4.4) were generally larger and more variable than those obtained in the case of MCAR (Table 4.3). The more sophisticated methods again performed better than the simpler methods, with KNNimpute achieving the most accurate predictions of

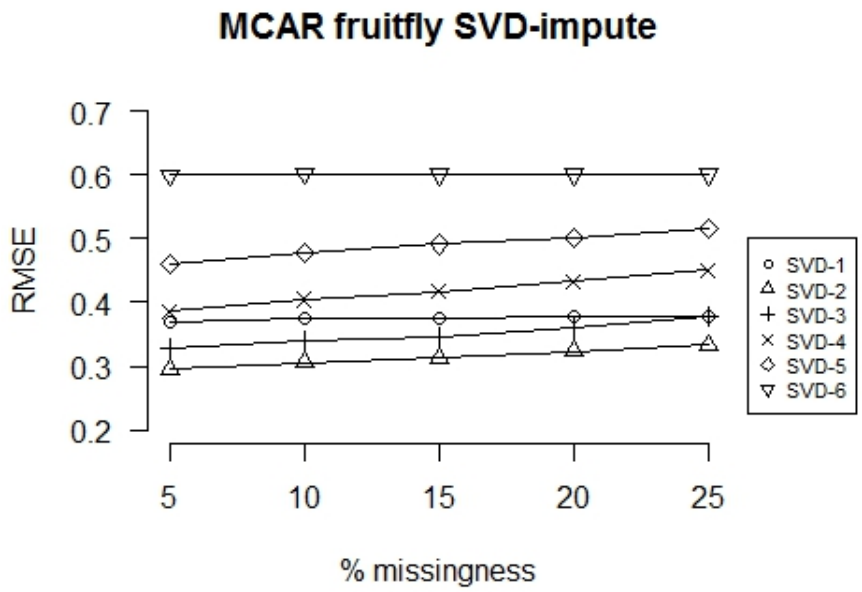

Figure 4.12: MCAR fruit flies data: RMSE obtained by SVDimpute with various k parameter across different missing probabilities.

the missing values. The method LOCF produced the largest and most variable RMSE values across all missing probabilities. When focusing on the effects of MAR and MCAR types of missingness on imputation accuracy, it appears that simple methods had similar performances under both missing mechanisms but KNNimpute and SVDimpute both had notably worse performance in MAR datasets than in the MCAR datasets. Among all the methods, LOCF was the only one which performed better in the MAR dataset than in the MCAR dataset.

	Missing probabilities				
Methods	5%	10%	15%	20%	25%
ROWimpute	0.57 (0.06)	0.56 (0.03)	0.57 (0.03)	0.57 (0.02)	0.57 (0.02)
COLimpute	0.61 (0.05)	0.60 (0.03)	0.60 (0.02)	0.60 (0.02)	0.60 (0.02)
LOCF	0.68 (0.06)	0.68 (0.04)	0.68 (0.04)	0.68 (0.03)	0.68 (0.03)
NOCB	0.62 (0.06)	0.62 (0.04)	0.62 (0.03)	0.62 (0.03)	0.62 (0.02)
KNNimpute	0.32 (0.05)	0.31 (0.03)	0.32 (0.03)	0.33 (0.02)	0.33 (0.02)
SVDimpute	0.38 (0.05)	0.38 (0.04)	0.38 (0.03)	0.38 (0.02)	0.38 (0.02)

Table 4.4: MAR fruit flies data: Comparison of RMSE of the imputation methods (means and standard deviations of 500 simulation runs).

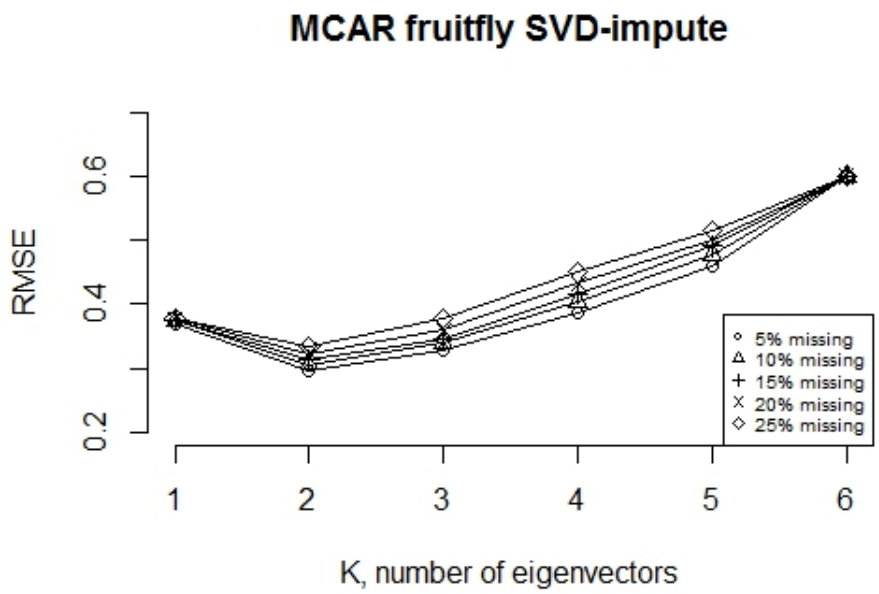

Figure 4.13: MCAR fruit flies data: RMSE obtained by SVDimpute when k ranges from 1 to 6.

4.2.4 Discussion

By imputing the modified simulated and fruit flies datasets with different imputation methods, we note that there are some differences among the performances of imputation approaches on the two types of datasets. On the simulated datasets with artificially created missingness, the imputation methods resulted in higher and more variable RMSE values than those obtained from imputing the fruit flies datasets. Some of the imputation methods even produced RMSE values larger than one from imputing the simulated datasets, indicating that it would be easier and more accurate if one imputed the missing values with zeros. From these results, it shows that it was more difficult to impute the simulated datasets than the real RNA-seq fruit flies datasets with created missing values.

Overall, the results obtained from this section showed the difficulty in selecting the optimal imputation method for any given dataset. The KNNimpute performed well for the fruit flies data while SVDimpute produced better results for the simulated datasets. Also, when

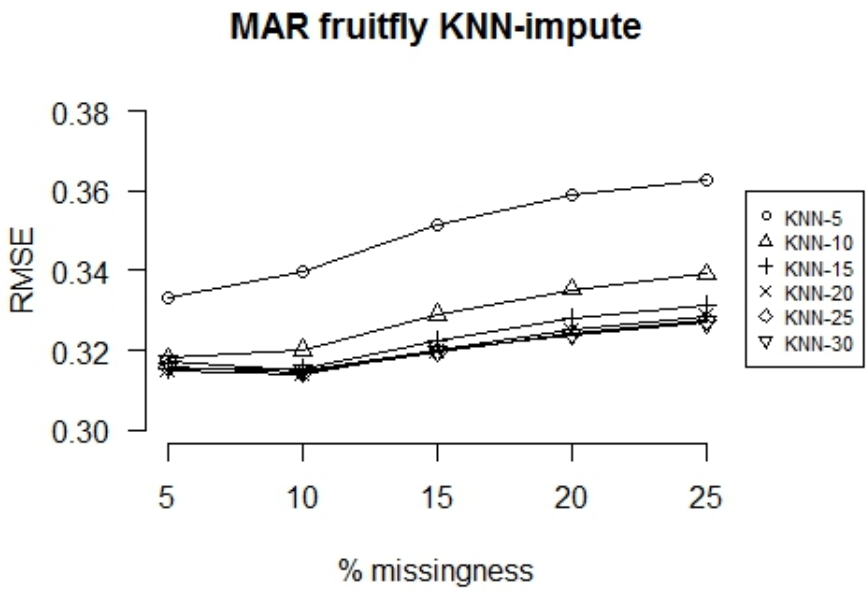

Figure 4.14: MAR fruit flies data: RMSE obtained by KNNimpute with various k parameter across different missing probabilities.

using KNNimpute and SVDimpute for estimating the missing values, the choice of optimal k parameter for either methods poses another issue in the analysis procedure. The best k parameter values for the two methods depend on the characteristics of the datasets and further investigation would be needed to determine the best k parameter for RNA-seq datasets.

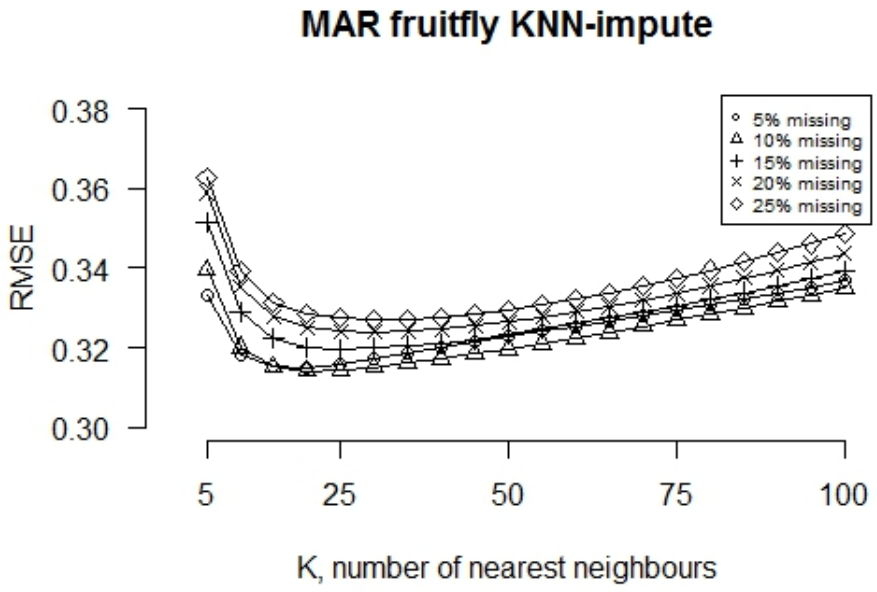

Figure 4.15: MAR fruit flies data: RMSE obtained by KNNimpute when k ranges from 5 to 100.

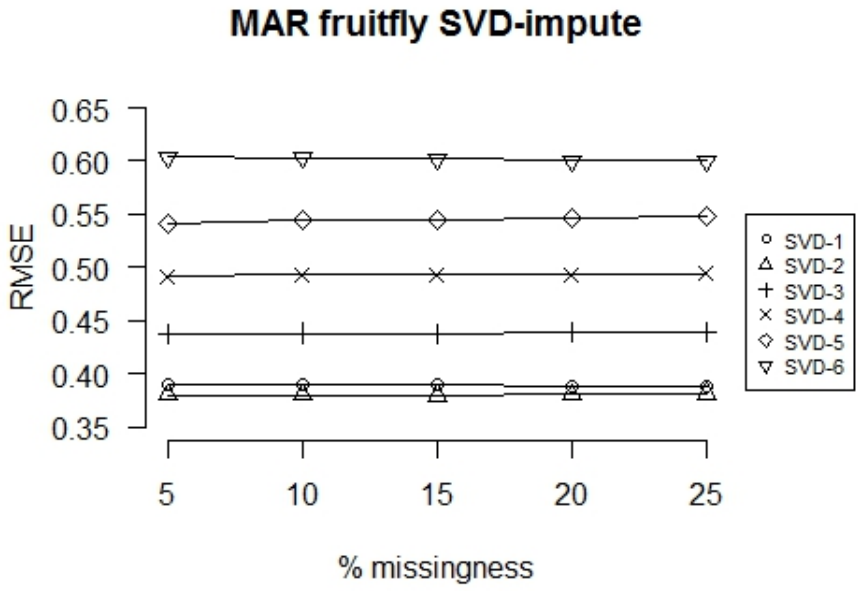

Figure 4.16: MAR fruit flies data: RMSE obtained by SVDimpute with various k parameter across different missing probabilities.

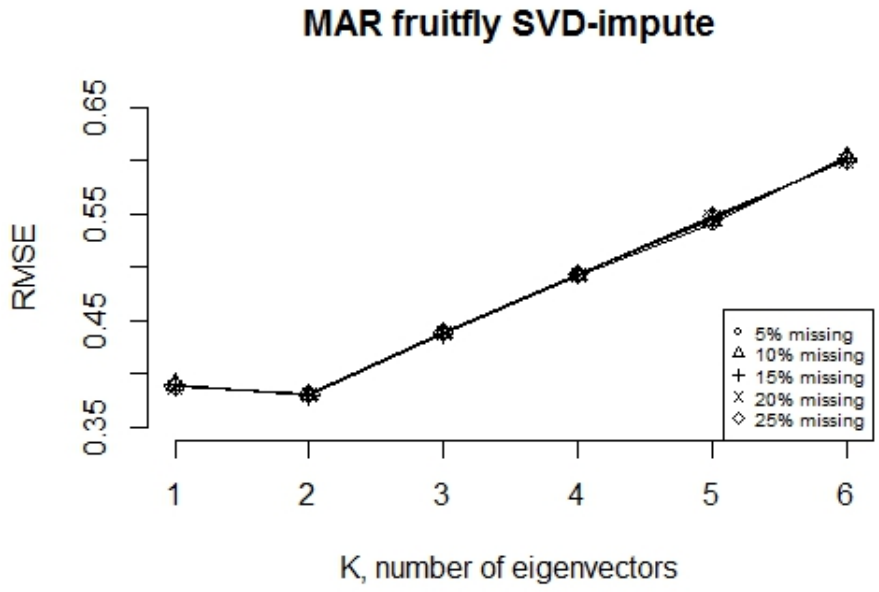

Figure 4.17: MAR fruit flies data: RMSE obtained by SVDimpute when k ranges from 1 to 5.

4.3 Impact on clustering

When we encounter gene expression data with missing values, we can either delete all the genes with missing entries, obtain estimates of missing entries by some imputation methods, or use downstream analysis approaches which can accommodate missing values. As researchers try to preserve all the obtained information from the experiment, imputation is often performed as a pre-processing step prior to cluster analysis. However, imputation as a pre-processing step has its limitations, since the imputed values would remain constant along with the observed values during the analysis process and badly imputed values may bias cluster results (Kim et al., 2007). This problem becomes worse as the missing probabilities increases or if the missing values are mainly localized in one part of the data matrix (Yun et al., 2007). Also, sophisticated imputation approaches are often computationally intensive and may require almost the same amount of computation as the clustering method itself. In such situations, it might be appropriate to use clustering methods which can handle missing values automatically or ones that incorporate the task of imputation into the clustering algorithm, or carefully evaluate the impact of imputation methods on downstream analysis processes if imputation is preferred.

The focus of this section is to investigate not the imputation efficiency of various imputation methods, but their impact on the clustering of simulated genomic data. Artificial datasets are simulated to resemble time-course RNA-seq data with discrete read counts. In each dataset, expression patterns of 500 genes over five time points were generated to reflect genes from four distinct clusters, with 50, 100, 150 and 200 genes in each cluster respectively. These simulated datasets were generated in the same procedure as the ones used in the comparison in Section 4.2.2. Since the simulated datasets were complete matrices, missing entries were generated according to MCAR and MAR missing mechanisms with 1%, 5% and 10% of entries as missing values. The imputation methods being considered include ZEROimpute, ROWimpute, COLimpute, LOCF, NOCB, KNNimpute and SVDimpute (described in Section 4.2.1). Since the results from the Section 4.2.3 showed that KNNimpute with $k = 60$ (number of genes to be in-

cluded in the candidate set) was the optimal setting for KNNimpute and that SVDimpute with $k = 2$ had the best performance for the MAR simulated datasets, these two settings were used in the evaluation in this section. We simulated 100 datasets independently for the evaluation of each imputation method under different missingness settings and the model-based clustering approach proposed in Chapter 2 was used to cluster the datasets. The impact of missingness and imputation methods on clustering ability is measured by the percentage of correctly classified genes, adjusted Rand index (ARI) and number of EM iterations required for the parameter estimations. The clustering results from the various imputation methods would be compared to the clustering results obtained if the datasets have no missing values.

4.3.1 Results from clustering

Datasets with MCAR missingness

We have simulated datasets with 1%, 5% and 10% of entries missing completely at random. These percentages were shown to be commonly encountered missing probabilities in real genomic (microarray) datasets (Brás and Menezes, 2007), and here we examine the impact on clustering when values in simulated RNA-seq data are missing at similar rates. Focusing on the clustering ability, ARI values can reflect how well the clustering algorithm performed and Figure 4.18 contains the boxplots comparing the ARI values obtained across the different missing probabilities. For each case, we independently simulated 100 datasets but in some datasets the clustering resulted in non-identifiable models and so we made note of the actual number of datasets included in the analyses for each case (in Tables 4.5 to 4.7. Only the identifiable models are included in the boxplots. The boxplots with label "no_miss" correspond to the clustering results obtained when the datasets were complete matrices and they are included for easy comparison to the results obtained when missingness is present and entries imputed by different methods.

Clustering ability did not change significantly when 1% missing values were introduced

into the complete datasets. Overall, the boxplots show that the missing probabilities have an effect on the clustering ability since the ARI values obtained from clustering the differently imputed datasets decreased as the missing probabilities increased from 1% to 10%. When 10% of entries are missing, the mean ARI value dropped from approximately 0.8 (clustering of complete datasets) to mean ARI values ranging from 0.45 to 0.7. It appears that ROWimpute had the worst effect on clustering while the other imputation methods all had similar impact on the clustering accuracy. The methods LOCF and KNNimpute performed well under all three missing probabilities but still lowered the clustering ability from the situation where the datasets have no missing entries. Tables 4.5, 4.6 and 4.7 show the results from using different imputation methods to impute the missing values before performing the model-based clustering. The issue of non-identifiable models is a main concern when datasets have 10% of entries as missing values, as shown in Table 4.7 that the NOCB-imputed datasets resulted in more than 50% of the models being non-identifiable.

When only the identifiable models are considered, the increase in missing probabilities from 1% to 10% resulted in lower clustering accuracy (decrease in ARI and percent of genes being correctly classified) and required higher number of EM iterations to perform clustering. The method ROWimpute, although easy to implement, would not be recommended because it had the worst impact on clustering accuracy which can be seen by the low ARI values and percent correctness obtained when datasets have 5% and 10% MCAR missingness. Also, the clustering of ROW-imputed datasets required a larger number of EM iterations for the clustering when compared to other methods in the cases of 1% and 5% MCAR missingness. The method ZEROimpute had a similar effect on clustering as compared to other more sophisticated methods, but this method might be biased towards datasets with small counts. The performances of KNNimpute across all three different missing probability settings seem to be relatively well, resulting with very low number of non-identifiable models, high ARI and percent correctness and required low number of EM iterations for the clustering procedure.

Methods	Datasets included	ARI	Percent correctness	EM iterations
ZEROimpute	100	0.77 (0.03)	89.85 (9.83)	48.44
ROWimpute	99	0.77 (0.04)	89.72 (10.44)	68.71
COLimpute	99	0.77 (0.03)	91.23 (1.35)	42.29
LOCF	98	0.77 (0.03)	90.98 (6.07)	57.26
NOCB	95	0.77 (0.03)	91.37 (1.33)	58.06
KNNimpute	100	0.77 (0.03)	91.46 (1.23)	43.18
SVDimpute	100	0.77 (0.03)	91.29 (1.27)	50.87

Table 4.5: Number of datasets included, mean ARI (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the clustering with various imputation methods for datasets with MCAR at 1%.

Methods	Datasets included	ARI	Percent correctness	EM iterations
ZEROimpute	100	0.71 (0.04)	83.77 (16.69)	67.37
ROWimpute	70	0.63 (0.06)	48.94 (29.52)	138.66
COLimpute	99	0.71 (0.04)	86.47 (11.21)	61.16
LOCF	99	0.73 (0.03)	88.09 (9.55)	70.18
NOCB	83	0.71 (0.04)	88.17 (5.90)	106.57
KNNimpute	99	0.73 (0.03)	88.67 (7.90)	48.76
SVDimpute	96	0.71 (0.04)	86.94 (10.94)	62.88

Table 4.6: Number of datasets included, mean ARI (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the clustering with various imputation methods for datasets with MCAR at 5%.

Datasets with MAR missingness

Simulated datasets with MAR missingness at 1%, 5% and 10% were also generated and imputed by the various imputation methods, and the results obtained from clustering those imputed datasets are displayed in Figure 4.19 and Tables 4.8, 4.9 and 4.10. In general, the clustering results from the MAR datasets showed the same pattern as the MCAR datasets, with decreasing ARI values resulted when missing probabilities increased from 1% to 10%. When only 1% of entries were missing from the datasets, the missingness and the different imputation methods did not have much impact on the clustering ability. However, as the probability of missingness increased, clustering accuracy resulted from clustering the imputed datasets became notably lower than those from the datasets with complete matrices. Same as the results obtained from the MCAR datasets, the method ROWimpute had the largest impact on

Methods	Datasets included	ARI	Percent correctness	EM iterations
ZEROimpute	96	0.62 (0.05)	67.54 (24.64)	92.90
ROWimpute	80	0.48 (0.04)	16.40 (15.35)	83.90
COLimpute	84	0.64 (0.04)	83.58 (11.68)	97.57
LOCF	95	0.66 (0.04)	83.26 (12.91)	119.43
NOCB	43	0.65 (0.04)	82.55 (11.30)	102.23
KNNimpute	99	0.68 (0.04)	83.33 (14.53)	53.61
SVDimpute	90	0.64 (0.05)	77.96 (19.26)	94.99

Table 4.7: Number of datasets included, mean ARI (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the clustering with various imputation methods for datasets with MCAR at 10%.

the clustering accuracy while the other imputation methods had smaller effects and resulted in similar ARI values. Clustering of datasets imputed by ZEROimpute, LOCF and SVDimpute performed slightly better than those imputed by COLimpute, NOCB and KNNimpute in the cases of 5% and 10% missingness.

The results regarding the number of datasets included in the analyses, variability of ARI and percent of genes being correctly classified, and EM iterations required during the clustering process are displayed in Tables 4.8, 4.9 and 4.10 for datasets with MAR at 1%, 5% and 10% respectively. In the case of 1% missingness, most of the datasets resulted in identifiable models and were included in the analyses, but more and more non-identifiable models were obtained from the clustering of imputed datasets with originally 5% and 10% missing entries. The more sophisticated methods KNNimpute and SVDimpute had the least impact on this issue, while the methods NOCB, ROWimpute and COLimpute resulted in large numbers of non-identifiable models when missing probabilities increased. The percent of genes being correctly classified when datasets imputed by different methods were clustered showed a very large variability regardless of the missing probabilities. The method COLimpute performed surprisingly well in this sense as clustering of those COL-imputed datasets had high percent of correctness with very low variability. The methods LOCF and SVDimpute also had relatively higher percent of correctness with lower standard deviations when compared to the other imputation methods.

Focusing on the number of EM iterations required in the clustering process across the three missing probabilities settings, ROW-imputed datasets required the most number of iterations while COL-imputed datasets could be clustered within the smallest number of EM iterations.

Methods	Datasets included	ARI	Percent correctness	EM iterations
ZEROimpute	100	0.77 (0.04)	88.60 (12.54)	52.10
ROWimpute	96	0.76 (0.03)	89.05 (10.04)	52.30
COLimpute	100	0.77 (0.03)	91.30 (1.36)	51.92
LOCF	98	0.77 (0.03)	90.87 (5.77)	69.22
NOCB	96	0.76 (0.03)	90.68 (3.18)	59.92
KNNimpute	100	0.77 (0.04)	89.61 (9.71)	58.41
SVDimpute	99	0.77 (0.04)	90.29 (8.17)	41.77

Table 4.8: Number of datasets included, mean ARI (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the clustering with various imputation methods for datasets with MAR at 1%.

Methods	Datasets included	ARI	Percent correctness	EM iterations
ZEROimpute	98	0.71 (0.04)	82.76 (18.13)	58.99
ROWimpute	65	0.66 (0.04)	65.02 (29.26)	101.29
COLimpute	97	0.70 (0.03)	88.61 (1.39)	46.80
LOCF	78	0.71 (0.03)	85.15 (14.09)	74.67
NOCB	66	0.69 (0.04)	84.62 (14.24)	48.33
KNNimpute	99	0.70 (0.04)	75.67 (23.55)	55.59
SVDimpute	95	0.71 (0.03)	88.97 (1.40)	60.01

Table 4.9: Number of datasets included, mean ARI (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the clustering with various imputation methods for datasets with MAR at 5%.

4.3.2 Discussion

In this section, we examined the impact of different imputation methods on the clustering results of genomic data under various amounts of MAR and MCAR. Overall, the impact of imputing datasets seem to be the same for the two types of missing mechanisms since ARI values resulting from the clustering were within similar ranges regardless of MAR or MCAR missingness. The missing mechanism was also irrelevant to the time required for the clustering

Methods	Datasets included	ARI	Percent correctness	EM iterations
ZEROimpute	87	0.63 (0.04)	58.72 (26.30)	78.23
ROWimpute	75	0.56 (0.04)	38.62 (28.28)	122.39
COLimpute	61	0.62 (0.03)	84.69 (5.40)	67.25
LOCF	71	0.63 (0.04)	77.66 (18.92)	89.23
NOCB	42	0.60 (0.04)	67.32 (25.31)	75.95
KNNimpute	91	0.63 (0.04)	57.02 (26.61)	98.97
SVDimpute	87	0.63 (0.05)	81.62 (14.03)	87.06

Table 4.10: Number of datasets included, mean ARI (standard deviation), mean percent correctness (standard deviation), and mean EM iterations required for the clustering with various imputation methods for datasets with MAR at 10%.

process with both types of datasets requiring similar numbers of EM iterations to complete the model-based clustering. Across the various imputation methods, imputing the datasets with row averages (ROWimpute) had the worst impact on clustering accuracy. The simple imputation methods ZEROimpute and COLimpute produced complete matrices which did not impact clustering results significantly, but ZEROimpute may bias towards datasets with entries of small counts. The more sophisticated methods did not have significantly superior performance over the more simple methods in terms of preserving the dataset characteristics for efficient clustering.

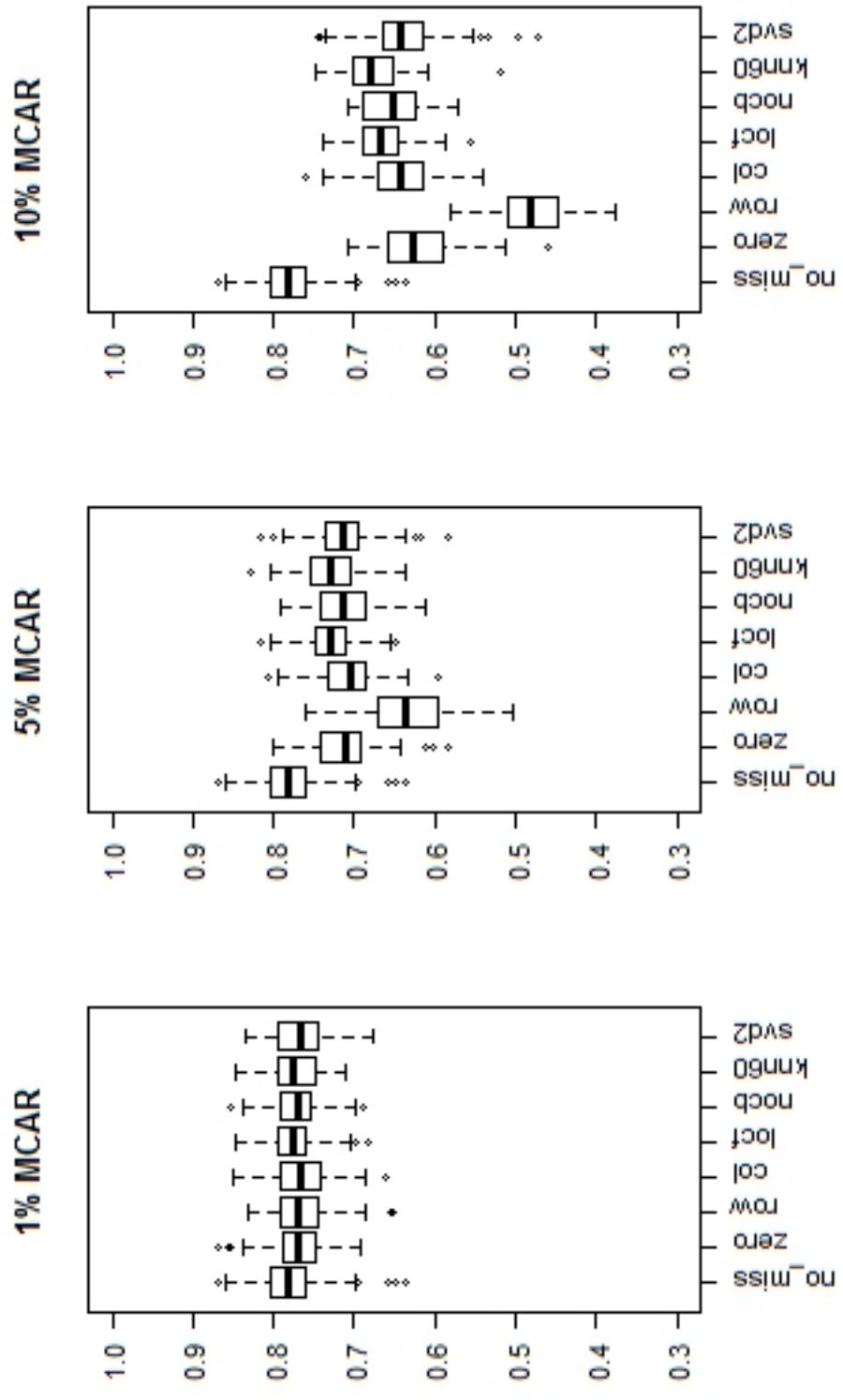

Figure 4.18: MCAR simulated data: mean ARI resulted from clustering datasets with 1%, 5% and 10% missing entries (excluding datasets with non-identifiable models).

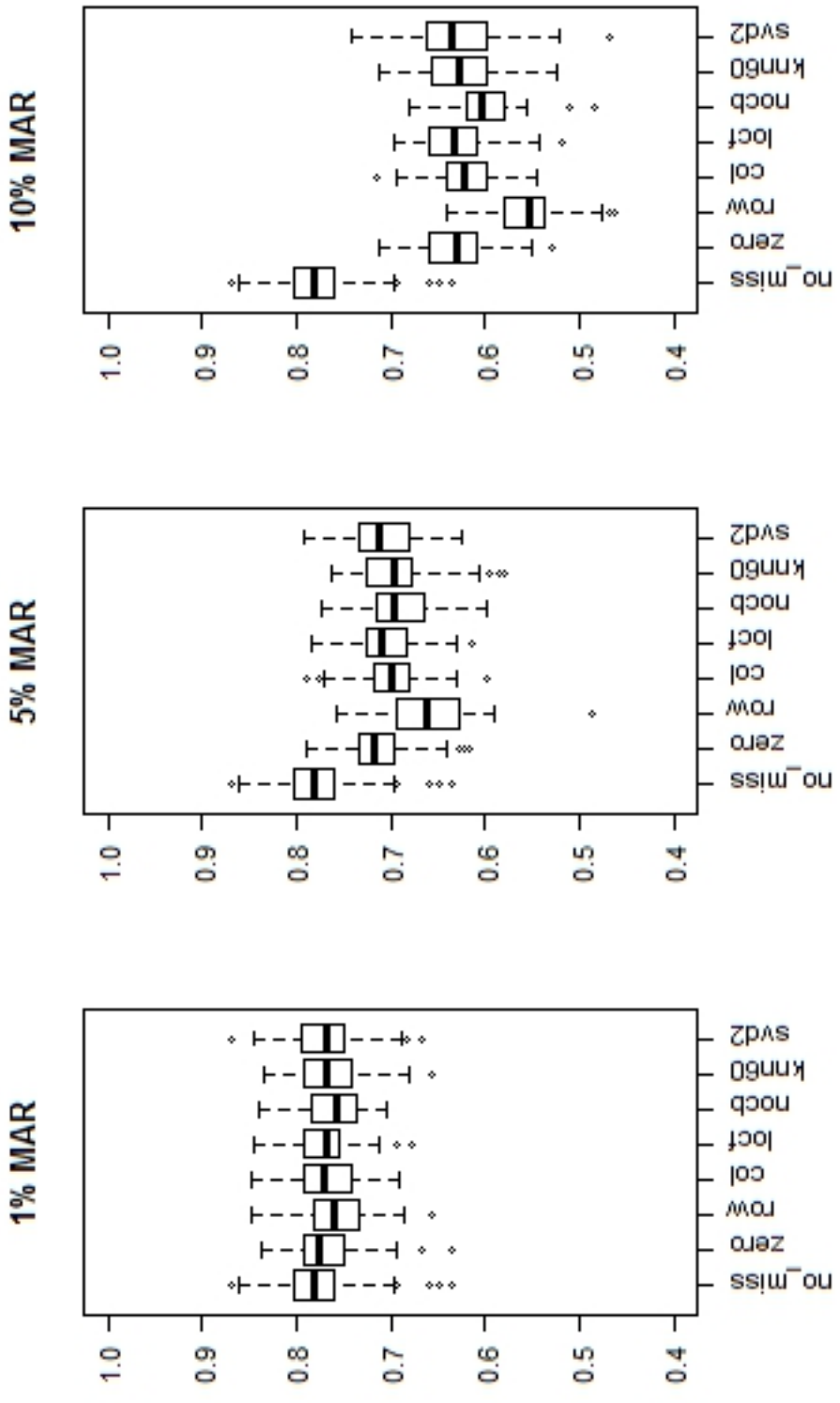

Figure 4.19: MAR simulated data: mean ARI resulted from clustering datasets with 1%, 5% and 10% missing entries (excluding datasets with non-identifiable models).

4.4 Cluster-based imputation method

Rather than using distance-based or correlation-based methods to identify candidate genes to obtain estimations for missing values, an alternate approach is to use data clustering results for imputation. As mentioned in Chapter 1, several clustering-based imputation methods have been proposed and evaluated on genomic data, such as GMCimpute (Ouyang et al., 2004), CMI method (Zhang et al., 2008) and FCMimpute (Luo et al., 2005). The GMCimpute method uses a model-based clustering approach which assumes the microarray data to be generated by a Gaussian mixture, whereas the CMI and FCMimpute methods uses distance/similarity-based k-means and fuzzy C-means clustering techniques respectively. One limitation to the k-means, non-parametric algorithms is that all the clusters are assumed to have the same variance (Genolini and Falissard, 2011). In the case of a group of data where this assumption does not hold, the clustering algorithm might fail to identify the correct clusters. When dealing with RNA-seq data where the count values are over-dispersed, the amount of over-dispersion may not simply be assumed to be the same for all genes so a non-parametric clustering approach may not be appropriate. The use of a model-based method can overcome this problem by separately estimating the characteristics, including variability, for each cluster and so using model-based clustering results to impute the missing values in a RNA-seq dataset would be an efficient imputation algorithm.

4.4.1 Model

We propose an imputation method which is based on mixtures of negative binomial models in the model-based clustering context. The time-course RNA-seq data are assumed to be generated from a finite mixture model and exhibit over-dispersion which can be modelled by the negative binomial distribution. We consider the negative binomial mixture clustering of time-course RNA-seq data from Chapter 2 of this dissertation, now applying the method to missing values imputation. For a missing entry in the dataset, an estimate is made from a linear combi-

nation of the component-wise averages of the values at that time point as the imputation value.

From Chapter 2, we have the following formulation for the model-based mixture clustering of time-course RNA-seq data. In the mixture model, each component is modelled by a negative binomial distribution which takes into account the over-time, repeated measurements. The component density function for component i is

$$f_i(\boldsymbol{y}_j; \boldsymbol{\beta}^i, s_i) = \prod_{t=1}^{m} \left(\frac{\Gamma(y_{jt} + s_i)}{y_{jt}!\Gamma(s_i)} p_i^{s_i} (1 - p_i)^{y_{jt}} \right)$$

with mean

$$\lambda = exp(\boldsymbol{\beta}^i \mathbf{x}_{jt}) = exp(\beta_0^i + \beta_1^i time_{jt} + \beta_2^i time_{jt}^2)$$

and s_i being the dispersion parameter for the component i and probability

$$p_i = \frac{s_i}{s_i + \lambda}.$$

Assuming that there are g number of components in the mixture, the mixture likelihood for the entire sample of n genes would be

$$\begin{aligned} L(\boldsymbol{\psi}) &= \prod_{j=1}^{n} \sum_{i=1}^{g} \pi_i f_i(\boldsymbol{y_j}; \boldsymbol{\beta}^i, s_i) \\ &= \prod_{j=1}^{n} \sum_{i=1}^{g} \pi_i \prod_{t=1}^{m} \left(\frac{\Gamma(y_{jt} + s_i)}{y_{jt}!\Gamma(s_i)} p_i^{s_i} (1 - p_i)^{y_{jt}} \right) \end{aligned}$$

with the mixing proportions $\pi_i \geq 0, \pi_1 + ... + \pi_g = 1$. The EM algorithm can be used to obtain maximum likelihood estimates, $\hat{\boldsymbol{\psi}}$, from the complete-data log-likelihood.

The first step in the cluster-based imputation approach is to initially impute the missing entries using the simple method (COLimpute) to obtain complete matrices. The missing entries would be permanently highlighted so that the algorithm can update the estimates in the

later step. The second step in the cluster-based imputation approach is to perform the model-based clustering to obtain the distinct clusters, where g, the number of clusters, is empirically determined. Finally, with the cluster information we would impute the originally missing entries with the cluster-wise averages at the corresponding time points to obtain the final imputed dataset. Each missing entry is imputed with the weighted average of the cluster-wise averages with the mixing proportions as the weights.

4.4.2 Data and evaluation

Two types of datasets were used to evaluate the effectiveness of the cluster-based imputation approach. One type of datasets consisted of simulated values to reflect discrete read counts obtained from a time-course RNA-seq experiment, while the other refers to a dataset obtained from a real RNA-seq experiment of *D. melanogaster* (Graveley et al., 2011). To analyze the impact of different mechanisms of missingness on the imputation accuracy, missing entries were generated to reflect 1%, 5%, and 10% of MCAR and MAR missing mechanisms. The procedures for generating the artificial datasets and the various cases of missingness have been provided in Section 4.2.2. The cluster-based imputation method is evaluated and compared against the simple imputation methods ZEROimpute, COLimpute, ROWimpute, LOCF and NOCB, as well as the more sophisticated methods KNNimpute and SVDimpute. The normalized root mean squared error (RMSE) values obtained from the imputed datasets are used for performance assessment of the different imputation methods. The RMSE from ZEROimpute is always equal to one, so it can be used as a standard to assess the difficulty of imputation across different datasets and methods.

4.4.3 Results

The results obtained from the various imputation methods in the different cases of missingness have been summarized in Tables 4.11 to 4.14. The results labelled as 'CLUSTimpute' refer to the proposed cluster-based imputation method, and they are compared to the results obtained

by the other imputation methods.

Figure 4.11 shows the mean RMSE values and the standard deviations obtained by using different imputation methods when there were 1%, 5% and 10% of MCAR missingness in the simulated datasets. First, very notably, the NOCB method performed the worst while COL-impute had surprisingly good performance in terms of accuracy of imputed estimates. The proposed cluster-based imputation method had similar performance as the SVDimpute, and it outperformed the KNNimpute method. The RMSE values obtained by NOCB had the highest variability, and the variability decreased as missing probabilities changed from 1% to 10% for all imputation methods. Overall, the cluster-based imputation approach produced the most accurate estimates of the missing entries whereas KNNimpute had disappointing performance compared to the more simple methods.

	Missing probabilities		
Methods	1%	5%	10%
CLUSTimpute	0.89 (0.28)	0.83 (0.04)	0.82 (0.03)
KNNimpute	1.14 (0.37)	1.04 (0.06)	1.03 (0.03)
SVDimpute	0.91 (0.30)	0.83 (0.05)	0.83 (0.03)
ZEROimpute	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)
ROWimpute	1.19 (0.34)	1.08 (0.07)	1.09 (0.06)
COLimpute	0.91 (0.30)	0.83 (0.05)	0.83 (0.03)
LOCF	1.02 (0.31)	0.98 (0.08)	0.99 (0.05)
NOCB	1.51 (0.84)	1.30 (0.23)	1.31 (0.18)

Table 4.11: MCAR simulated data: mean RMSE (and standard deviations) obtained by cluster-based imputation and other imputation methods across different missing probabilities.

The results obtained when simulated datasets with MAR missingness are imputed are shown in Table 4.12 and all the methods produced RMSE values in the same range as in the case of MCAR datesets, with all the mean RMSE values between 0.8 and 1.6. The change in missing probabilities from 5% to 10% had no impact of the performances of the different methods, but the methods produced higher RMSE values when missing probabilities changed from 1% to 5%. The simple method COLimpute again had surprisingly well performance, with similar

results as those obtained by CLUSTimpute and SVDimpute and outperforming the KNNimpute method. When focusing on the proposed cluster-based method, it obtained similar results as SVDimpute, with lower and less variability in RMSE values when the datasets had 1% of entries as missing. Comparing to the results obtained when datasets had MCAR missingness, the simple methods ROWimpute, LOCF and NOCB had worse performances when applied to the MAR datasets. The performances of the other methods were not affected by the mechanism of missingness, which is a good property since missing data problems in real life are often not the simple MCAR.

	Missing probabilities		
Methods	1%	5%	10%
CLUSTimpute	0.89 (0.24)	0.83 (0.04)	0.83 (0.03)
KNNimpute	1.10 (0.36)	1.00 (0.03)	1.00 (0.01)
SVDimpute	0.91 (0.35)	0.83 (0.05)	0.83 (0.03)
ZEROimpute	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)
ROWimpute	1.41 (0.84)	1.21 (0.24)	1.21 (0.17)
COLimpute	0.91 (0.35)	0.83 (0.05)	0.83 (0.03)
LOCF	1.26 (0.82)	1.12 (0.24)	1.12 (0.16)
NOCB	1.59 (1.07)	1.36 (0.34)	1.35 (0.21)

Table 4.12: MAR simulated data: mean RMSE (and standard deviations) obtained by cluster-based imputation and other imputation methods across different missing probabilities.

To assess the performance of the cluster-based imputation approach on RNA-seq data, we compared its performance on the real dataset obtained from the fruit flies data (Graveley et al., 2001) to the performances of other imputation methods. The fruit flies dataset has first been modified to contain MCAR missingness in 1%, 5% and 10% of the entries and the results of the imputation methods are shown in Table 4.13. It shows that the imputation process has been improved by including the clustering component, with RMSE values as approximately 0.6 obtained by COLimpute compared to approximately 0.37 obtained by CLUSTimpute. All imputation methods had better performances than simply imputing the missing entries with zeros, which also shows that the imputation difficulty of this type of datasets was lower than the simulated datasets. In this type of RNA-seq datasets with MCAR missingness, CLUSTimpute

had a noticeably better performance over the other imputation methods, with the most accurate estimates of the missing entries shown by the lowest mean RMSE values achieved. SVDimpute is the next best option for this type of datasets, while KNNimpute performed worse than the other simple imputation methods.

When we focus on the fruit flies data with simulated MAR missingness in the entries, the results obtained are displayed in Table 4.14. The performances of CLUSTimpute, KNNimpute and SVDimpute in the MAR datasets were worse than their performances in the MCAR datasets. However, CLUSTimpute and SVDimpute were still the preferred imputation methods for this type of datasets since the RMSE values obtained were still lower than those produced by the other methods.

	Missing probabilities		
Methods	1%	5%	10%
CLUSTimpute	0.38 (0.09)	0.37 (0.04)	0.36 (0.02)
KNNimpute	0.86 (0.08)	0.86 (0.03)	0.86 (0.02)
SVDimpute	0.45 (0.07)	0.46 (0.03)	0.48 (0.02)
ZEROimpute	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)
ROWimpute	0.51 (0.07)	0.52 (0.03)	0.52 (0.02)
COLimpute	0.61 (0.07)	0.60 (0.03)	0.60 (0.02)
LOCF	0.74 (0.10)	0.73 (0.05)	0.72 (0.03)
NOCB	0.63 (0.09)	0.63 (0.04)	0.63 (0.03)

Table 4.13: MCAR fruit flies data: mean RMSE (and standard deviations) obtained by cluster-based imputation and other imputation methods across different missing probabilities.

4.4.4 Discussion

In general, the results obtained from our simulation and evaluation showed that the proposed CLUSTimpute is a appropriate imputation method for imputing missing entries present in time-course RNA-seq experiments. The simple method ZEROimpute is easy to implement, but may be biased towards datasets with small counts or many zero entries. The two methods ROWim-

	Missing probabilities		
Methods	1%	5%	10%
CLUSTimpute	0.55 (0.16)	0.51 (0.05)	0.50 (0.04)
KNNimpute	0.89 (0.11)	0.96 (0.03)	0.99 (0.01)
SVDimpute	0.55 (0.11)	0.54 (0.04)	0.54 (0.03)
ZEROimpute	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)
ROWimpute	0.58 (0.13)	0.57 (0.05)	0.56 (0.04)
COLimpute	0.62 (0.12)	0.60 (0.04)	0.60 (0.02)
LOCF	0.71 (0.17)	0.69 (0.07)	0.68 (0.04)
NOCB	0.64 (0.14)	0.62 (0.06)	0.62 (0.04)

Table 4.14: MAR fruit flies data: mean RMSE (and standard deviations) obtained by cluster-based imputation and other imputation methods across different missing probabilities.

pute and COLimpute make use of averages either from the same gene across time or from the measurements of different genes at the same time point. On the other hand, NOCB and LOCF both make use of information only from the individual gene, disregarding the information that may be obtained from the other genes in the dataset. All these mentioned methods are common imputation approaches that may be used for measurements over time, and we have compared these simple methods with the more sophisticated KNNimpute and SVDimpute.

KNNimpute uses local information in imputation, with only a subset of genes (the set of candidate genes) involved when imputing a missing entry. This method requires the datasets to have enough complete data patterns to produce estimates of the missing entries, which may be difficult if datasets have many missing data. SVDimpute makes use of mutually orthogonal expression patterns in the dataset to approximate the data and this allows the imputation approach to work well on time-course data with low level of noise. The proposed CLUSTimpute is based on model-based clustering and weighted averages across the mixture model. This approach uses information from the entire dataset and the cluster structures identified from the data, and the final estimate of each missing entry is calculated from a linear combination of the cluster-wise averages weighted by the probabilities of the individual gene belonging to the clusters.

The KNNimpute, SVDimpute and CLUSTimpute all require a parameter to be determined empirically. Both KNNimpute and SVDimpute require a parameter k to be pre-specified, with k being the number of genes in the candidate gene set in KNNimpute and k being the number of eigenvectors used in the SVD procedure of SVDimpute. There is no known theoretical way of determining the k parameter for these two approaches. For CLUSTimpute, the data is assumed to arise from a mixture model and the number of mixture components/clusters is empirically determined. This parameter can be determined by sub-sampling or model selection criteria such as BIC for the finite mixture models.

The imputation method GMCimpute (Ouyang et al., 2004) based on Gaussian mixture clustering have been proposed for estimating missing entries in microarray data. We incorporated the model-based clustering method with negative binomial models and presented an imputation method suitable for time-course RNA-seq data with discrete read counts. Results from our evaluation showed that CLUSTimpute has good performance when used to impute RNA-seq data with time-dependence nature, outperforming the more commonly used methods such as KNNimpute and SVDimpute.

Chapter 5

Conclusions and future work

The RNA sequencing (RNA-seq) technology has recently replaced microarrays as the approach being used for gene expression analyses. RNA-seq is more reliable than microarrays since RNA-seq has higher sensitivity and dynamic range, with lower technical variation and thus higher precision than microarrays. Statistical methods for analyzing microarray data have been well developed but they are not suitable for RNA-seq data since microarrays measure gene expression in continuous intensities, whereas RNA-seq provides absolute quantification of gene expression using discrete counts of reads. In current literature, limited work has been done on statistical methods has been done on expression analysis of time-course RNA-seq data to account for the time-dependence nature of the count data with over-dispersion property. In this thesis, we propose some statistical approaches for examining longitudinal RNA-seq data.

Functional clustering is an important method for examining gene expression patterns and thus discovering co-expressed genes to better understand the biological systems. To our knowledge, no model framework has been developed for cluster analysis of RNA-seq data focusing on the time-course experiment setting. We propose a model-based clustering method to identify important information on changes in expression levels over time from longitudinal RNA-seq data. The clustering method is based on finite mixture modelling of negative binomial distribu-

tions to accommodate over-dispersion in the data. An EM algorithm for maximum likelihood estimation is incorporated to obtain parameter estimates in our clustering approach. The EM clustering method and two EM/quasi-Newton hybrid algorithms are proposed and evaluated through simulation studies and the results indicate the excellent clustering ability of our approach. Through an application to real RNA-seq dataset, we are able to produce meaningful clustering results that can provide insights on changes in gene expression patterns over time.

The clustering algorithm developed for time-course RNA-seq data makes use of the mixtures of negative binomial models without specifying covariates. Further development of the clustering algorithm may include the covariates information into the model, such as including other characteristics of the genes in addition to the gene expression levels. This may help further stabilize the mixture model and lead to more accurate clustering of the genes. Another way to extend the clustering approach may be to include correlation structure into the mixture model. The current model assumes that given the cluster, genes are independent over time. A correlation structure can be introduced to model the dependence between gene expressions over time points. These extensions to the clustering approach can improve the modelling of the relationship between genes over time for a better characterization of the biological system of interest.

It was noted that for mixture modelling, model identification can be difficult. The estimation methods used in this research did not always converge and sometimes produced non-identifiable models, which are models without reliable parameter estimates. Cases like these result in singular Fisher information matrices and so the standard errors of the estimates cannot be computed for the parameter estimates. Non-identifiable models with unreliable standard errors may also appear when the clustered gene expression patterns are over-parameterized. There is a need for further research on how to efficiently work with the problem of model identifiability in the mixture-based clustering methods being proposed.

The EM algorithm used in our clustering approach is an iterative process which requires the specification of starting values of the parameters for the estimation to begin. Poor choice of initial values of the parameters may affect the estimation process and lead to convergence problems. In order to improve the performance of the clustering algorithm on time-course RNA-seq data, we propose some methods and assess their performances in selecting the initial values to start the parameter estimation. In this work, the different initiation procedures are evaluated for mixtures of negative binomial models through simulation studies and the results show that a good option would be to initiate the clustering approach with values resulted from short runs of the EM algorithm.

For mixture models, there is not one commonly accepted statistical procedure for choosing the optimal number of mixture components in the models. Many different instruments, such as AIC or BIC, can be used to determine the most favourable mixture model and the problem of model selection continues to be an important research area in cluster analysis. Future work can include identifying the ideal model selection statistical tool for the proposed model-based clustering method on RNA-seq data. The clustering approach may also be improved by using a different convergence criterion. Our iterative estimation methods were implemented such that a small difference between two successive log-likelihood values would indicate convergence of the estimation. An extensive evaluation of different convergence criteria will be beneficial for future developments of mixture modelling with EM algorithm.

Another topic which is investigated in this dissertation is the problem of missing values in gene expression analysis. It is of interest to examine the performance of different imputation methods on time-course RNA-seq data, as well as the impact of imputation on the clustering of such datasets. We develop a cluster-based imputation method specifically designed to better deal with the missing values problem in RNA-seq, and the proposed method shows supe-

rior performance over the widely used KNN and SVD imputation methods through simulation studies. This research would be beneficial to future researchers as a new treatment method for missing values in gene expression data. Further studies can be conducted to assess the impact of the cluster-based imputation method on other downstream analyses of genomic data besides functional clustering. Future work can also include extending the imputation approach to incorporate some stochastic characteristcs such that there would be randomness in the estimated imputation values or modify the approach to be a multiple imputation process, which may improve the downstream gene expression analyses.

We note that imputation as a pre-processing step has its limitations, since the imputed values would remain constant along with the observed values during the analysis process and badly imputed values may bias cluster results (Kim et al., 2007). This problem becomes worse as the missing probabilities increases or if the missing values are mainly localized in one part of the data matrix (Yun et al., 2007). Also, sophisticated imputation approaches are often computationally intensive and may require almost the same amount of computation as the downstream analysis method itself. It is of interest to develop efficient RNA-seq analysis methods which can accommodate missing values so that imputation pre-processing is not a necessary step.

Bibliography

Aitkin, M. and Aitkin, I. (1996). A hybrid EM/Gauss-Newton algorithm for maximum likelihood in mixture distributions. Statistics and Computing 6:127-130.

Aittokallio, T. (2010). Dealing with missing values in large-scale studies: microarray data imputation and beyond. Briefings in bioinformatics 11(2):253-264.

Allison, D., Cui, X., Page, G. and Sabripour, M. (2006). Microarray data analysis: from disarray to consolidation and consensus. Nature Reviews Genetics 7:55-65.

Anders, S. and Huber, W. (2010). Differential expression analysis for sequence count data. Genome Biology 11:R106.

Androulakis, I.P., Yang, E. and Almon, R.R. (2007). Analysis of time-series gene expression data: Methods, challenges, and opportunities. Annual Review of Biomedical Engineering 9:205-28.

Balwierz, P., Carninci, P., Daub, C., Kawai, J., Hayashizaki, Y., Van Belle, W., Beisel, C. and van Nimwegen, E. (2009). Methods for analyzing deep sequencing expression data: constructing the human and mouse promoterome with deepCAGE data. Genome Biology 10:R79.

Banfield, J. and Raftery, A. (1993). Model-based Gaussian and non-Gaussian clustering. Biometrics 49:803-821.

Bar-Joseph, Z., Gerber, G., Gifford, D., Jaakkola, T. and Simon, I. (2003). Continuous representations of time-series gene expression data. Journal of Computational Biology 10(3-4):341-356.

Bar-Joseph, Z. (2004). Analyzing time series gene expression data. Bioinformatics 20(16):2493-2503.

Belacel, N., Wang, Q. and Cuperlovic-Culf, M. (2006). Clustering methods for microarray gene expression data. OMICS: A Journal of Integrative Biology 10(4):507-531.

Bennett, S., Barnes, C., Cox, A., Davies, L., and Brown, C. (2005). Toward the 1,000 dollars human genome. Pharmacogenomics 6:373-382.

Biernacki, C., Celeus, G. and Govaert, G. (2003). Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. Computational Statistics and Data Analysis 41:561-575.

Blanchet, J. and Vignes, M. (2009). A model-based approach to gene clustering with missing observation reconstruction in a Markov random field framework. Journal of Computational Biology 16(3):475-486.

Bloom, J., Khan, Z., Kruglyak, L., Singh, M. and Caudy, A. (2009). Measuring differential gene expression by short read sequencing: quantitative comparison to 2-channel gene expression microarrays. BMC Genomics 10:221.

Bø, T., Dysvik, B. and Jonassen I. (2004). LSimpute: accurate estimation of missing values in microarray data with least squares methods. Nucleic Acids Research 32(3):1-8.

Brás, L. and Menezes, J. (2007). Improving cluster-based missing value estimation of DNA microarray data. Biomolecular engineering 24(2):273.

Bradford, J., Hey, Y., Yates, T., Li, Y., Pepper, S. and Miller, C. (2010). A comparison of massively parallel nucleotide sequencing with oligonucleotide microarrays for global transcription profiling. BMC Genomics 11:282.

Bradley, P. and Fayyad, U. (1998). Refining initial points for k-means clustering. Proceedings of the 15th International Conference on Machine Learning 91-99.

de Brevern, A., Hazout, S. and Malpertuy, A. (2004). Influence of microarrays experiments missing values on the stability of gene groups by hierarchical clustering. BMC Bioinformatics 5(1):114.

Brock, G., Shaffer, J., Blakesley, R., Lotz, M., and Tseng, G. (2008). Which missing value imputation method to use in expression profiles: a comparative study and two selection schemes. BMC Bioinformatics 9(1):12.

Bullard, J., Purdom, E., Hansen, K. and Dudoit, S. (2010). Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. BMC Bioinformatics 11:94.

Celeux, G., Martin, O. and Lavergne, C. (2005). Mixture of linear mixed models for clustering gene expression profiles from repeated microarray experiments. Statistical Modelling 5(3):243-267.

Celton, M., Malpertuy, A., Lelandais, G. and de Brevern, A. (2010). Comparative analysis of missing value imputation methods to improve clustering and interpretation of microarray experiments. BMC Genomics 11:15.

Chechik, G. and Koller, D. (2009). Timing of gene expression responses to environmental changes. Journal of Computational Biology 16(2):279-290.

Chial, H. (2008) DNA sequencing technologies key to the Human Genome Project. Nature Education 1(1).

Chipman, H., Hastie, T. and Tibshirani, R. (2003). Clustering microarray data. In: Speed, T. (Ed.), Statistical Analysis of Gene Expression Microarray Data. Chapman and Hall, Boca Raton, FL.

Choong, M., Charbit, M. and Yan, H. (2009). Autoregressive-model-based missing value estimation for DNA microarray time series data. IEE Transactions on Information Technology in Biomedicine 13(1):131-137.

Cooke, E., Savage, R., Kirk, P., Darkins, R. and Wild, D. (2011). Bayesian hierarchical clustering for microarray time series data with replicates and outlier measurements. BMC Bioinformatics 12:399.

Das R., Kalita, J. and Bhattacharyya, D. (2009). A new approach for clustering gene expression time series data. International Journal of Bioinformatics Research and Applications 5(3):310-328.

Dempster, A., Laird, N. and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39(1):1-38.

Di, Y., Shafer, D., Cumbie, J. and Chang, J. (2011). The NBP negative binomial model for assessing differential gene expression from RNA-Seq. Statistical Applications in Genetics and Molecular Biology 10(1):A24.

Di Camillo, B., Sanchez-Cabo, F., Toffolo, G., Nair, S., Trajanoski, Z. and Cobelli, C. (2005). A quantization method based on threshold optimization for microarray short time series. BMC Bioinformatics 6 Suppl 4:S11.

Dudley, A., Ros, M. and Tabin, C. (2002). A re-examination of proximo distal patterning during vertebrate limb development. Nature 418:539544.

Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Sciences 95: 14863-14868.

Ernst, J., Nau, G. and Bar-Joseph, Z. (2005). Clustering short time series gene expression data. Bioinformatics 21(Suppl. 1):I159-168.

Esnaola, M., Puig, P., Gonzalez, D., Castelo, R. and Gonzalez, J. (2013). A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. BMC Bioinformatics 14:254.

Fayyad, U., Reina, C. and Bradley, P. (1998). Initialization of iterative refinement clustering algorithms. Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining 194-198.

Finch, S., Mendell, N. and Thode, H. (1989). Probabilistic measures of adequacy of a numerical search for a global maximum. Journal of the American Statistical Association 84:1020-1023.

Fowlkes, E. (1979). Some methods for studying the mixture of two normal (lognormal) distributions. Journal of the American Statistical Association 74:561-575.

Fraley, C. and Raftery, A. (2002). Model-based clustering, discriminant analysis, and density estimation. Journal of the American Statistical Association 97(458):611-631.

Fraley, C., Raftery, A. and Wehrens, R. (2005). Incremental model-based clustering for large datasets with small clusters. Journal of Computational and Graphical Statistics 14(3):529-546.

Frühwirth-Schnatter, S. (2006). Finite Mixture and Markov Switching Models. New York: Springer.

Fu, X., Fu, N., Guo, S., Yan, Z., Xu, Y., Hu, H., Menzel, C., Chen, W., Li, Y., Zeng, R. and Khaitovich, P. (2009). Estimating accuracy of RNA-Seq and microarrays with proteomics. BMC Genomics 10:161.

Futschik, M. and Carlisle, B. (2005). Noise-robust soft clustering of gene expression time-course data. Journal of Bioinformatics and Computational Biology 3(4):965-988.

Genolini, C. and Falissard, B. (2011). Kml: A package to cluster longitudinal data. Computer methods and programs in biomedicine 104(3):e112-e121.

Ghosh, D. and Chinnaiyan, A. (2002). Mixture modeling of gene expression data from microarray experiments. Bioinformatics 18(2):275-286.

Gollub, J. and Sherlock, G. (2006). Clustering microarray data. Methods in Enzymology 411:194-213.

Graveley, B., Brooks, A., Carlson, J., Duff, M., Landolin, J., Yang, L., Artieri, C. et al. (2011). The developmental transcriptome of Drosophila melanogaster. Nature 471:473-479.

Grün, B., Scharl, T. and Leisch, F. (2012). Modelling time course gene expression data with finite mixtures of linear additive models. Bioinformatics 28(2):222-228.

Hardcastle, T. and Kelly, K. (2010). baySeq: Empirical Bayesian analysis of patterns of differential expression in count data. BMI Bioinformatics 11:442.

Hasselblad, V. (1966). Estimation of parameters for a mixture of normal distributions. Technometrics 8:431-446.

't Hoen, P., Ariyurek, Y., Thygesen, H., Vreugdenhil, E., Vossen, R., Menezes, R., Boer, J., van Ommen, G. and den Dunnen, J. (2008). Deep sequencing-based expression analysis shows major advances in robustness, resolution and inter-lab portability over five microarray platforms. Nucleic Acids Research 36(21):e141.

Hosmer, D. (1973). A comparison of iterative maximum likelihood estimates of the parameters of two normal distributions under three different types of samples. Biometrika 29:761-770.

Hourani, M., and El Emary, I. (2009). Microarray missing values imputation methods: Critical analysis review. Computer Science and Information Systems 6(2):165-190.

Hubert, L. and Arabie, P. (1985). Comparing partitions. Journal of Classification 2:193-218.

Hutchison, C.A., III. (2007). DNA sequencing: bench to bedside and beyond. Nucleic Acids Research 35:6227-6237.

Irigoien, I., Vives, S. and Arenas, C. (2011). Microarray time course experiments: finding profiles. IEEE/ACM Transactions and Computational Biology and Bioinformatics 8:464-475.

Jain, A., Murty, M. and Flynn, P. (1999). Data clustering: A review. ACM Computing Surveys 31(3):264-323.

Jäger, M., Ott, C., Grünhagen, J. et al. (2011). Composite transcriptome assembly of RNA-seq data in a sheep model for delayed bone healing. BMC Genomics 12:158.

Jones, B., Nagin, D. and Roeder, K. (2001). A SAS procedure based on mixture models for estimating developmental trajectories. Sociological Methods and Research 29(3):374-393.

Jörnsten, R., Wang, H., Welsh, W. and Ouyang, M. (2005). DNA microarray data imputation and significance analysis of differential expression. Bioinformatics 21(22):4155-4161.

Karlis, D. and Xekalaki, E. (2003). Choosing initial values for the EM algorithm for finite mixtures. Computational Statistics and Data Analysis 41:577-590.

Karpievitch, Y., Dabney, A. and Smith, R. (2012). Normalization and missing value imputation for label-free LC-MS analysis. BMC bioinformatics 13(Suppl 16):S5.

Keogh, E. and Pazzani, M. (2000). Scaling up dynamic time warping for datamining applications. In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 285-289). ACM.

Kim, B., Zhang, L., Berg, A., Fan, J. and Wu, R. (2008). A computational approach to the functional clustering of periodic gene expression profiles. Genetics 180:821-834.

Kim, D., Lee, K., Lee, K. and Lee, D. (2007). Towards clustering of incomplete microarray data without the use of imputation. Bioinformatics 23(1):107-113.

Kim, J. and Kim, J. (2007). Difference-based clustering of short time-course microarray data with replicates. BMC Bioinformatics 8:253.

Laird, N. (1978). Nonparametric maximum likelihood estimation of a mixing distribution. Journal of the American Statistical Association 73:805-811.

Lander, E., Linton, L., Birren, B., Nusbaum, C., Zody, M., Baldwin, J., Devon, K., Dewar, K., Doyle, M. et al. (2001). Initial sequencing and analysis of the human genome. Nature 409:860-921.

Langmead, B., Trapnell, C., Pop, M. and Salzberg, S. (2009). Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biology 10:R25.
Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R. and 1000 Genome Project Data Processing Subgroup. (2009). The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics 25(16):2078-2079.

Li, J., Witten, D., Johnstone, I. and Tibshirani, R. (2012). Normalization, testing, and false discovery rate estimation for RNA-sequencing data. Biostatistics 13(3):523-538.

Li, P., Ponnala, L., Gandotra, N. et al. (2010). The developmental dynamics of the maize leaf transcriptome. Nature Genetics 42:1060-1067.

Liew, A., Law, N. and Yan, H. (2010). Missing value imputation for gene expression data: computational techniques to recover missing data from available information. Briefings in Bioinformatics 12(5):498-513.

Liu, H., Tarima, S, Borders, A., Getchell, T., Getchell, M. and Stromberg, A. (2005). Quadratic regression analysis for gene discovery and pattern recognition for non-cyclic short time-course microarray experiments. BMC Bioinformatics 6:106.

Louis, T. A. (1982). Finding the observed information matrix when using the EM algorithm. Journal of the Royal Statistical Society B 44:226-233.

Lozano, J., Na, J. and Naga, P. (1999). An empirical comparison of four initialization methods for the k-means algorithm. Pattern Recognition Letters 20:1027-1040.

Luan, Y. and Li, H. (2003). Clustering of time-course gene expression data using a mixed-effects model with B-splines. Bioinformatics 19(4): 474-482.

Luo, J., Yang T. and Wang, Y. (2005). Missing value estimation for microarray data based on fuzzy c-means clustering. In High-Performance Computing in Asia-Pacific Region, 2005. Proceedings. Eighth International Conference on (pp. 6-pp). IEEE.

Ma, P., Castillo-Davis, C., Zhong, W. and Liu, J. (2006). A data-driven clustering method for time course gene expression data. Nucleic Acids Research 34(4):1261-1269.

Maitra, R. (2001). Clustering massive data sets with applications in software metrics and tomography. Technometrics 43(3):336-346.

Maitra, R. (2009). Initializing partition-optimization algorithms. IEEE/ACM Transactions on computational biology and bioinformatics 6(1):144-157.

Margulies, M., Egholm, M., Altman, W., Attiya, S., Bader, J., Bemben, L., Berka, J., Braverman, M., Chen, Y., Chen, Z., et al. (2005). Genome sequencing in microfabricated high-density picolitre reactors. Nature 437:376-380.

Marioni, J., Mason, C., Mane, S., Stephens, M. and Gilad, Y. (2008). RNA-seq: An assessment of technical reproducibility and comparison with gene expression arrays. Genome Research 18:1509-1517.

Matukumalli, L. and Schroeder, S. (2009). Sequence based gene expression analysis. In D. Edwards, J. Stajich and D. Hansen (Eds.), Bioinformatics: Tools and Applications (pp.191-207). New York: Springer.

McCarthy, D., Chen, Y. and Smyth, G. (2012). Differential expression analysis of multifactor RNA-seq experiments with respect to biological variation. Nucleic Acids Research 40(10):4288-4297.

McLachlan, G. (1988). On the choice of initial values for the EM algorithm in fitting mixture models. The Statistician 37:417-425.

McLachlan, G. and Krishnan, T. (2008). The EM algorithm and extensions, Second Edition. New Jersey: John Wiley and Sons.

Meng, X. L. and Rubin, D. B. (1991). Using EM to obtain asymptotic variance-covariance matrices: The SEM algorithm. Journal of the American Statistical Association 86:899-909.

Minas, C., Waddell, S. and Montana, G. (2011). Distance-based differential analysis of gene curves. Bioinformatics 27(22): 3135-3141.

Möller-Levet, C., Cho, K, and Wolkenhauer, O. (2003). Clustering of gene expression time-series data. Technical report, University of Rostock.

Mortazavi, A., Williams, B., McCue, K., Schaeffer, L. and Wold, B. (2008). Mapping and quantifying mammalian transcriptomes by RNA-Seq. Nature Methods 5:621-628.

Nagalakshmi, U., Wang, Z., Waern, K., Shou, C., Raha, D., Gerstein, M. and Snyder, M. (2008). The transcriptional landscape of the yeast genome defined by RNA sequencing. Science 320(5881):1344-1349.

Nagin, D. (1999). Analyzing developmental trajectories: A semiparametric group-based approach. Psychological Methods 4:139-157.

Nagin, D. (2005). Group-Based Modeling of Development. Cambridge: Harvard University Press.

Nagin, D. and Land, K. (1993). Age, criminal careers, and population heterogeneity: Specification and estimation of a nonparametric, mixed Poisson model. Criminology 31:327-362.

Nash, J. (1990). Compact Numerical Methods for Computers: Linear algebra and function minimisation, Second Edition. New York, NY: Adam Hilger.

Ng, S., McLachlan, G., Wang, K., Ben-Tovim Jones, L. and Ng, S. (2006). A mixture model with random-effects components for clustering correlated gene-expression profiles. Bioinformatics 22(14):1745-1752.

Oh, S., Kan, D., Brock, G. and Tseng, G. (2011). Biological impact of missing-value imputation on downstream analyses of gene expression profiles. Bioinformatics 27(1):78-86.

Oh, S., Song, S., Grabowski, G., Zhao, H. and Noonan, J. (2013). Time series expression analyses using RNA-seq: A statistical approach. BioMed Research International 2013.

Oshlack, A., Robinson, M. and Young, M. (2010). From RNA-seq reads to differential expression results. Genome Biology 11:220.

Ouyang, M., Welsh, W. and Georgopoulos, P. (2004). Gaussian mixture clustering and imputation of microarray data. Bioinformatics 20(6):917-923.

Parmigiani, G., Garrett, E., Irizarry, R. and Zeger, S. (2003). The Analysis of Gene Expression Data: An Overview of Methods and Software. In: Parmigiani, G., Garrett, E., Irizarry, R. and S. Zeger (Eds.), The Analysis of Gene Expression Data: Methods and Software (pp.1-36). New York: Springer.

Pauli, A., Valen, E., Lin, M. et al. (2012). Systematic identification of long noncoding RNAs expressed during zebrafish embryogenesis. Genome Research 22:577-591.

Pettersson, E., Lundeberg, J. and Ahmadian, A. (2009). Generations of sequencing technologies. Genomics 93:105-111.

R Core Team (2012). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Ramoni, M., Sebastiani, P. and Kohane, I. (2002). Cluster analysis of gene expression dynamics. Proceedings of the National Academy of Sciences USA 99(14):9121-26.

Rand, W. (1971). Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66:846-850.

Rapaport, F., Khanin, R., Liang, Y., Pirun, M., Krek, A. et al. (2013). Comprehensive evaluation of differential gene expression analysis methods for RNA-seq data. Genome Biology 14:R95.

Redner, R. and Walker, H. (1984). Mixture densities, maximum likelihood and the EM algorithm. SIAM Review 26:195-239.

Rhodes, D., Barrette, T., Rubin, M., Ghosh, D. and Chinnaiyan, A. (2002). Meta-analysis of microarrays interstudy validation of gene expression profiles reveals pathway dysregulation in prostate cancer. Cancer research 62(15):4427-4433.

Roeder, K., Lynch, K. and Nagin, D. (1999). Modeling uncertainty in latent class membership: A case study in criminology. Journal of the American Statistical Association 94(447):766-776.

Robinson, M., McCarthy, D. and Smyth, G. (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26(1):139-140.

Robinson, M. and Oshlack, A. (2010). A scaling normalization method for differential expression analysis of RNA-seq data. Genome Biology 11:R25.

Robinson, M. and Smyth, G. (2007). Moderated statistical tests for assessing differences in tag abundance. Bioinformatics 23:2881-2887.

Robinson, M. and Smyth, G. (2008). Small-sample estimation of negative binomial dispersion, with applications to SAGE data. Biostatistics 9:321-332.

Robles, J., Qureshi, S., Stephen, S. and Wilson, S. (2012). Efficient experimental design and analysis strategies for the detection of differential expression using RNA-Sequencing. BMC Genomics 13:484.

Sahu, M., Swarnkar, M., and Das, M. (2011). Estimation methods for microarray data with

missing values: a review. International Journal of Computer Science and Information Technologies 2(2):614-620.

Sanger F., Nicklen, S. and Coulson, A.R. (1977). DNA sequencing with chain-terminating inhibitors. Proceedings of the National Academy of Sciences of the United States of America 74:5463-5467.

Scharl, T., Grün, B. and Leisch, F. (2010). Mixtures of regression models for time course gene expression data: evaluation of initialization and random effects. Bioinformatics 26(3):370-377.

Schliep, A., Schonhuth, A. and Steinhoff, C. (2003). Using hidden Markov models to analyze gene expression time course data. Bioinformatics 19(Suppl. 1):I283-89.

Sehgal, M., Gondal, I. and Dooley, L. (2005). Collateral missing value imputation: a new robust missing value estimation algorithm for microarray data. Bioinformatics 21(10):2417-2423.

Shendure, J. and Ji, H. (2008). Next-generation DNA sequencing. Nature Biotechnology 26(10): 1135-1145.

Si, Y., Liu, P., Li, P. and Brutnell, T. (2014). Model-based clustering for RNA-seq data. Bioinformatics 30(2):197-205.

Srivastava S. and Chen, L. (2010). A two-parameter generalized Poisson model to improve the analysis of RNA-seq data. Nucleic Acids Research 38(17):e170.

Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., et al. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. Proceedings of the National Academy of Sciences of the United States of America 96:2907-2912.

Tavazoie, S., Hughes, J., Campbell, M., Cho, R., and Church, G. (1999). Systematic determination of genetic network architecture. Nature Genetics 22:281-285.

Trapnell, C., Pachter, L. and Salzberg, S. (2009). TopHat: discovering splice junctions with RNA-seq. Bioinformatics 25(9)1105-1111.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. (2001). Missing value estimation methods for DNA microarrays. Bioinformatics 17(6):520-525.

Tseng, G. and Wong, W. (2005). Tight clustering: A resampling-based approach for identifying stable and tight patterns in data. Biometrics 61:10-16.

Tsiporkova, E. and Boeva, V. (2007). Two-pass imputation algorithm for missing value estimation in gene expression time series. Journal of Bioinformatics and Computational Biology 5(05):1005-1022.

Venter, J., Adams, M., Myers, E., Li, P., Mural, R., Sutton, G., Smith, H., Yandell, M., Evans, C. et al. (2001). The sequence of the human genome. Science 291:1304-1351.

Wang, L., Feng, Z., Wang, X., Wang, X. and Zhang, X. (2010). DEGseq: an R package for identifying differentially expressed genes from RNA-seq data. Bioinformatics 26(1):136-138.

Wang, P. and Puterman, M.L. (1998). Mixed Logistic Regression Models. Journal of Agricultural, Biological, and Environmental Sciences 3(2):175-200.

Wang, X., Wu, M., Li, Z. and Chan, C. (2008). Short time-series microarray analysis: Methods and challenges. BMC Systems Biology 2:58.

Wehrens, R., Buydens, L., Fraley, C. and Raftery, A. (2004). Model-based clustering for image segmentation and large datasets via sampling. Journal of Classification 21:231-253.

Wong, J. (2013). imputation: imputation. R package version 2.0.1.
http://CRAN.R-project.org/package=imputation

Woodward, W., Parr, W., Schucany, W. and Lindsey, H. (1984). A comparison of minimum distance and maximum likelihood estimation of a mixture proportion. Journal of the American Statistical Association 79(387):590-598.

Wu, H., Wang, C. and Wu, Z. (2013). A new shrinkage estimator for dispersion improves differential expression detection in RNA-seq data. Biostatistics 14(2):232-243.

Yeung, K., Fraley, C., Murua, A., Raftery, A. and Ruzzo, W. (2001). Model-based clustering and data transformations for gene expression data. Bioinformatics 17:977-987.

Yoon, D., Lee, E. and Park, T. (2007). Robust imputation method for missing values in microarray data. BMC bioinformatics 8(Suppl 2):S6.

Yu, H. (2002). Rmpi: parallel statistical computing in R. R News 2:10-14.

Yu, D., Huber, W. and Vitek, O. (2013). Shrinkage estimation of dispersion in Negative Binomial models for RNA-seq experiments with small sample size. Bioinformatics 29(10):1275-1282.

Yuan, A. and He, W. (2008). Semiparametric clustering method for microarray data analysis. Journal of Bioinformatics and Computational Biology 6(2):261-282.

Yuan, M. and Kendziorski, C. (2006). Hidden Markov models for microarray time course data

in multiple biological conditions. Journal of the American Statistical Association 101(476):1323-1332.

Yun, T., Kim, S., Hwang, T. and Yi, G. (2007). A Direct Clustering Method for Imperfect Microarray Data without Imputation. In Frontiers in the Convergence of Bioscience and Information Technologies, 2007. FBIT 2007 (pp. 183-187). IEEE.

Zeng Y. and Garcia-Frias, J. (2006). A novel HMM-based clustering algorithm for the analysis of gene expression time-course data. Computational Statistics and Data Analysis 50(9):2472-2494.

Zhang, S., Zhang, J., Zhu, X., Qin, Y. and Zhang, C. (2008). Missing value imputation based on data clustering. In Transactions on computational science I (pp. 128-138). Springer Berlin Heidelberg.

Zhong, S. and Ghosh, J. (2003). A unified framework for model-based clustering. Journal of Machine Learning Research 4:1001-1037.

Zhou, Y-H., Xia, K. and Wright, F. (2011). A powerful and flexible approach to the analysis of RNA sequence count data. Bioinformatics 27(19):2672-2678.

Curriculum Vitae

Name:	Man-Kee Maggie Chu
Education:	Ph.D. Biostatistics, 2010 - 2014 The University of Western Ontario, London, Ontario, Canada
	M.Sc. Biostatistics, 2008 - 2010 The University of Western Ontario, London, Ontario, Canada
	B.CS. Honours Bioinformatics Co-op, 2003 - 2008 University of Waterloo, Waterloo, Ontario, Canada
Honours and Awards:	Ontario Graduate Scholarship, 2011 - 2013 Schulich Graduate Thesis Research Award, 2009 Schulich Scholarship for Medical Research, 2008 - 2009
Related Work Experience:	Teaching Assistant, 2010 - 2014 The University of Western Ontario, London, Ontario, Canada
	Research Assistant, 2008 - 2010 The University of Western Ontario, London, Ontario, Canada

Publications:

[1] **Chu, M.K.** and Koval, J. (2014). Trajectory modelling of longitudinal binary data: Application of the EM algorithm for mixture models. Communication in Statistics - Simulation and Computation. 43:495-519.
[2] **Chu, M.K.** and He, W. (2013). Model-based clustering of time-course RNA-seq data. In Proceedings 59^{th} ISI World Statistics Congress, 25-30 August 2013, Hong Kong.
[3] Sarma, S., Devlin, R., Thind, A. and **Chu, M.K.** (2012). Canadian family physicians' decision to collaborate: Age, period and cohort effects. Social Science and Medicine. 75(10):1811-1819.
[4] Sarma, S., Thind, A. and **Chu, M.K.** (2011). Do new cohorts of family physicians work less compared to their older predecessors? The evidence from Canada. Social Science and Medicine. 72(12):2049-2058.