DEALING WITH MISSING VALUES IN DNA MICROARRAY

CAO YI

NATIONAL UNIVERSITY OF SINGAPORE

2008

DEALING WITH MISSING VALUES IN DNA MICROARRAY

CAO YI

(M.Eng. USTC, CHINA)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2008

Acknowledgements

First and foremost, I would like to thank my supervisor Associate Professor Poh Kim Leng, for his untiring support and guidance throughout my entire candidature. His valuable advice and critical comments on various aspects of the thesis have definitely improved the quality of this work. I would also express my sincere gratitude to Associate Professor Leong Tze Yun for her helpful suggestion on my research topic.

I greatly acknowledge the support from Department of Industrial and Systems Engineering for providing a scholarship, without which it would be impossible for me to complete study. Many thanks also go to members of the Biomedical Decision Engineering Group for many insightful discussions with them. Further, I thank my colleagues in System Modeling and Analysis Lab for the memorable days spent with them.

Family support has been crucial for me in this effort. Thanks to my parents for their constant encouragement and allowing me to pursue my study far away from home all these years. Their unconditional love, care, and attention have been showering on me all along the way. I am very grateful for that and am confident that this effort gives them much joy.

Finally, I wish to express my most loving thanks to my dear and understanding wife, Qu Huizhong, whose keen criticism and advice has contributed to every page of this dissertation, and whose constant, loving support has made its completion possible. A special THANK YOU to you.

Contents

Summary

Microarray data has been used in a large number of studies covering a broad range of areas in biology. Missing values are often encountered when analyzing microarray gene expression data. However, in many microarray data mining methods, a complete data matrix is required. It is essential that the estimates for the missing gene expression values are accurate to make the subsequent analysis as informative as possible.

Although numerous imputation algorithms have been proposed to estimate the missing values, many of them have limitations. Some algorithms perform well only when strong local correlation exists, while some provide better performance when data is dominated by global structure. In this study, we first develop nonparametric regression approach (NPRA) for imputation, which can capture both linear and non-linear relations between genes. NPRA serves the purpose of exploiting local gene-wise relationships.

The study is further extended to take advantage of relations between arrays to improve imputation accuracy. Moreover, one drawback of the existing imputation methods is their lack of robustness in case of outliers in microarray. In order to deal with outliers in microarray, we employ robust regression based on array components. Robust principal component analysis (RPCA) imputation method serves the purpose of utilizing global array-wise relationships.

Furthermore, we construct a missing value imputation framework, which makes use of the gene-wise correlation by means of nonparametric regression on the one hand, and

exploits the array-wise correlation by virtue of robust regression with array components on the other hand. By combining the estimates from NPRA and RPCA respectively, we propose a heuristic algorithm to determine the weighted coefficient for different estimates. As such, we borrow strength from each method and avoid particular types of systematic errors.

Finally, most of the imputation algorithms have been evaluated in terms of prediction error between imputed value and true value, such as normalized root mean squared error (NRMSE), which does not fully demonstrate the impact of missing values and imputation on subsequent data analysis. In this study, we focus on investigating the impact on gene clustering analysis, and justify that clustering accuracy is also a measure to assess imputation methods.

List of Figures

List of Tables

Nomenclature

m	the number of genes in microarray
n	the number of arrays in microarray
k	the number of similar genes
K_{pc}	the number of significant components
k^{clu}	the number of clusters in k-means clustering
$\mathbf{g}_i^T$	the expression profile of the i^{th} gene
$\mathbf{a}_j$	the expression profile of the j^{th} array
$\mathbf{g}_t$	target gene
μ	the weighted coefficient between estimates from $NPRA$ and $RPCA$
MV	missing value
$MCAR$	missing completely at random
MAR	missing at random
$MNAR$	missing not at random
$NRMSE$	normalized root mean squared error
$RNSE$	robust normalized squared error
$NPRA$	nonparametric regression approach
$RPCA$	robust principal component analysis
$MNHD$	mean normalized hamming distance

Chapter 1

Introduction

A worldwide data explosion is just beginning. Driven by rapid progress of process ability, the proliferation of data devices, the amount of data in our lives seems to be increasing. In such a fast changing world, the current convenience and affordability of data storage solutions coupled with the industry awareness have given rise to dealing with more data.

As a result of the Human Genome Project, there has been an explosion in the amount of information available about the DNA sequence of the human genome. The emergence of DNA microarray technology facilitates the identification and classification of this DNA sequence information and the assignment of functions to these new genes in the past decade. DNA Microarray allows the collection of data about the expression levels of thousands of genes simultaneously.

1.1 The Missing Value Problem in Microarray

The explosion in the amount of microarray data confronts the community with new questions, since these static data alone do not give insight into how genes interact with each other. Numerous applications based on gene expression data have been developed in a broad range of areas in biology. For example, regulatory pathway inferring [16, 18, 86]

provides insights into gene regulations and functions in order to gain an understanding of the underlying mechanisms of genetic regulation. Another example is functional gene finding in which the detection of differentially expressed genes is of more interest [69].

There are three main types of microarray data mining for biomedical applications:

Clustering

Gene clustering, the process of grouping related genes in the same cluster, is at the foundation of different genomic studies that aim at analyzing the function of genes. Gene clustering methods serve the purpose of interpreting knowledge extracted from microarray datasets in a meaningful way. However, the interpretation of co-expressed genes and coherent patterns depends on the domain knowledge (see for example [23, 25, 44]).

Gene Selection

The selection of significant gene via expression pattern in microarray has brought community challenges, due to small sample size and large number of variables (genes). Given a series of microarray experiments for a specific tissue under different conditions, gene selection serves the purpose of finding the genes most likely differentially expressed under these conditions. In other words, we want to figure out which genes best discriminate among the classes (see for example [20, 67, 90]).

Classification

Microarray sample classification serves the purpose of classifying diseases or predicting outcomes based on gene expression patterns, and then identifying the best treatment. Classification of microarray data is an extremely challenging problem because it usually involves a small number of samples in large dimension (see for example [31, 32, 46, 56]).

One hard problem in microarray data mining is the occurrence of missing values in microarray dataset. This problem may be due to many reasons when certain values in the datasets are not observed in the data collection process. Many microarray data mining algorithms for the downstream analyses cannot be applied to data that include missing values. Many methods for dealing with missing values have been developed so far.

1.2 Background

The data generated in microarray experiments are usually represented by a matrix with genes in rows and different experimental conditions in columns. Unfortunately, these matrices often contain missing values (MVs) due to various reasons. For example, the background and the signal may have similar intensities; the surface of the chip may not be planar; there may be dust on the slides; the probe may not be properly fixed on the chip or washed properly; the hybridization step may not work properly.

These above mentioned imperfections in the experimental steps create suspicious values that are usually thrown away and set as missing [3]. However, many available microarray analysis algorithms require the dataset to be complete without missing value [97], as the underlying statistical methodology is based on balanced data [69].

Obviously, one solution to the missing data is to repeat the experiments, but it is costly and time-consuming. Another one is to remove genes (rows) or experiments (columns) until no missing value exists. By this way, all the observed values in the corresponding row have to be discarded for a gene with only a small number of missing values.

For the subsequent analysis, it is important that the estimates for the missing gene expression values are accurate. Even a small number of badly estimated missing data may lead to misleading results for methods such as hierarchical clustering [25], k-means clustering [84], and principal component analysis [66].

The drawbacks of these simple solutions have stimulated the development of more refined approaches. It has been proven that if the correlations between genes are taken into consideration, then missing value prediction error can be reduced significantly [8, 47, 59, 75, 87]. A detailed review of these sophisticated imputation methods will be exhibited in Chapter 3.

1.3 Statement of the Problem

As we have described in previous section, many methods to deal with missing values have been developed. Currently many approaches have been developed to recover missing values, such as k-nearest neighbour (KNN) [87], Bayesian PCA (BPCA) [59], least squares imputation (LSimpute) [8], local least squares imputation (LLSimpute) [47] and collateral missing value estimation (CMVE) [75].

Troyanskaya *et al.* [87] were the pioneers in dealing with missing values in microarray, by proposing a method called k-nearest neighbour imputation (KNNimpute) in which the missing values are imputed using the weighted mean values of k most similar genes. LLSimpute, LSimpute and CMVE methods can be considered as parameter regression based imputation methods. All of them assume that the relations between predictor gene and target genes are linear, but actually it is impossible to know exactly whether they are linear or not. Although many works have been devoted to the missing value imputation, few studies have been done by employing the property of nonparametric regression. In our work, we will propose a novel nonparametric regression approach which utilizes both linear and non-linear relations between genes.

Nonparametric regression approach (NPRA) only takes gene-wise relationships into consideration. Another problem immediately emerges: how to improve prediction accuracy by using array-wise relationships. Only very few studies have considered array-wise relationships when imputing missing values. Moreover, one drawback of the existing im-

putation methods is their lack of robustness in case of outliers in microarray. In order to deal with outliers in microarray, we further exploit array-wise relationships and employ quantile regression to expect a robust and accurate imputation performance.

Once missing value estimations are done, the next issue is how to assess the performance of different imputation methods. Most imputation methods are evaluated by measures in terms of prediction error between imputed value and true value, such as normalized root mean squared error (NRMSE) [10, 48]. Although NRMSE gives an important measure of performance, it does not fully elucidate the impact of missing values and imputation methods on subsequent analysis of microarray, such as gene clustering, classification and significant gene selection. This has attracted researchers' attention, but only a few papers devoted to the isssue can be found [21, 60, 69]. In this study, further investigation of imputation methods' impact on downstream analysis will be performed.

1.4 Objectives

In Section 1.3, we observed that existing imputation methods have some limitations and the impact of imputation method on the downstream analysis has not been completely investigated. The purpose of this study is as follows:

1. To develop nonparametric regression approach by taking advantage of gene-wise relationships, which suggests that only information of the nearest neighbours should be utilized when imputing a missing entry. Least squares methods and least absolute deviation method have been successfully employed to capture gene relationships. This kind of relationships could also be exploited by virtue of nonparametric regression, which captures both linear and non-linear relationships, and may improve the accuracy of estimates on missing data.

2. To further utilize the array-wise relationships in order to improve prediction accuracy, and construct missing value imputation framework by considering both gene-

and array-wise relationships to achieve maximum accuracy of imputation. The influence of outliers will also be taken into consideration when dealing with missing values.

3. To conduct test on the influence in prediction accuracy of factors, such as methods' parameter, missing rate and pattern, and type of experiment (time series (TS), non-time series (NTS), or mixed (MIX)). More attention should be paid to the factors which affect the performance of imputation method most, whereas little will be focused on the factors to which imputation method is insensitive.

4. To compare our proposed methods with other existing imputation methods, with regard to different datasets, various missing rates and missing patterns. Different datasets consist of time series, non-time series and mixed dataset and missing rate will take value on 5%, 10%, 15%, 20% respectively. However, high missing rate remains beyond the scope of this research. The missing pattern of both missing at random (MAR) and missing not at random (MNAR) will be taken into account in our experimental study.

5. To study the impact of estimation on downstream analysis, such as gene clustering, classification and statistical algorithms for significance analysis of microarrays (SAM), prediction analysis for microarrays (PAM) and microarray analysis of variance (MAANOVA).

The insights from this thesis may help to deal with missing values accurately and efficiently. The proposed solutions to missing value imputation would hopefully benefit the bioinformatics community.

1.5 Organization

This thesis contains 7 chapters. In Chapter 2, the missing value problem will be fully described, in terms of types of microarray, concepts of microarray and the classification

of missing values. It would present a brief summary of the reasons for missing values and argue the need for accurate estimates.

In Chapter 3, literature related to this study will be reviewed. This includes the state-of-the-art work in missing value imputation. Different imputation methods are introduced and assessed with respect to advantages and drawbacks. The topics in the literature review also include various evaluation criteria, both theoretical and experimental.

Chapter 4 presents an approach by exploiting the local relationships between genes. On the basis of KNNimpute, we employ nonparametric regression to capture both linear and non-linear relations between genes. The factors studied include the type of missing pattern, different missing rates, and the number of k nearest neighbour genes. Optimal k is recommended across different types of dataset, which will be subsequently used in the following chapters.

Chapter 5 proposes a novel method by taking global array-wise relationships into consideration. Through a dimension reduction scheme known as principal component analysis, it retrieves some significant array components to represent the whole dataset. In order to reduce the influence of outliers, robust regression is employed for missing value estimation. The choice of the optimal number of significant components was studied, and an evaluation design is recommended in the following chapter. Other factors studied include the influence of the initial estimate, the robustness to noisy data, and computational efficiency.

Chapter 6 outlines the construction of missing value imputation framework by taking into account both gene- and array-wise relationships, and setting up the weight for two estimates which come from utilizing different relationships. A heuristic algorithm for determining the weight is proposed. To ensure the validity of this framework, the impact of missing values and imputation method on gene clustering analysis is also studied.

Chapter 7 summarizes the studies in this thesis, and suggests some directions for future work.

Chapter 2

The Missing Value Problem in Microarray

> "Among the many small problems that have yet to be addressed in microarray analysis, missing data methods stand out in my mind as one of the more pressing." –Gary A. Churchill

With the development of advanced bio-technology, there is an explosive growth in high-throughput genomic and proteomic data such as DNA microarrays. DNA microarrays allow the collection of data about the expression levels of thousands of genes simultaneously in particular cells or tissues, giving a global view of gene expression for the first time [54, 70, 72]. In the past decade, gene expression profile has become a useful biological resource. This allows for a quantitative readout of gene expression on a gene-by-gene basis. One-chip microarrays measure expression of up to tens of thousands of genes, covering most of the human genome.

2.1 Microarray

Microarrays have opened the door of constructing large-scale datasets of molecular information. There are many different types of microarrays (called platforms) in use, but all

have a high density and number of biomolecules fixed onto a well-defined surface. In this section, we will discuss two most commonly used types of microarray and give a brief description of basic aspects of microarray.

2.1.1 Types of microarray

Different technologies for measuring mRNA expression levels are employed for different types of microarray, among which Affymetrix GeneChips and the spotted cDNA (or oligonucleotide) microarrays are the two most commonly used types of microarray technology. Microarray in the type of GeneChip is a silicon chip that can measure the expression levels of thousands of genes simultaneously. This is done by hybridization which is detected using a fluorescent dye and a optical scanner that can record fluorescence-intensities values. The scanners and associated software perform various forms of image analysis to measure and report raw gene expression values. The redundancy in design used in a GeneChip (i.e., a gene is represented by a set of approximately 20 probe pairs) prevents the existence of MVs [10].

Spotted cDNA microarrays are microchips with more than ten thousands of spots where usually each spot corresponds to a unique gene per condition due to cost and design constraints of spotted cDNA microarray experiments [42]. Thus, the loss at a spot usually results in the loss of information for a gene, and then leads to a MV in the gene expression data matrix. There is an exception that double to quadruple spots are assigned to a particular gene. In our work, we concentrate on cDNA microarrays, considering the estimation of MVs in gene expression data.

2.1.2 Basic aspects of microarray

In general, there are five basic aspects of microarrays:

1. Preparing DNA chip using the chosen target DNAs;

2. Generating a hybridization solution containing a mixture of fluorescently labeled cDNAs;

3. Incubating the hybridization mixture containing fluorescently labeled cDNAs with DNA chip;

4. Detecting bound cDNA using laser technology and storing data in a computer;

5. Analyzing data using computational methods.

We are obviously interested in 5, but without some knowledge of 1 to 4, we would be in danger.

2.2 Biological Background

The genome of an organism plays a central role in the control of cellular processes. All organisms on Earth except viruses consist of cells. For instance, yeast has one cell whereas human beings have trillions of cells. All cells have a nucleus and there is a deoxyribonucleic acid (DNA) inside it. With few exceptions, almost every cell in the body of an organism has the same DNA. DNA has coding segments which are called genes, and non-coding segments. Genes code for proteins or (less commonly) other large molecules that do the essential work in every organism.

2.2.1 DNA and gene

DNA is composed of four basic molecules called nucleotides, which are identical except that each contains a different nitrogen base. Each nucleotide consists of phosphate, sugar and one of the four bases: Adenine, Guanine, Cytosine, and Thymine (denoted by A, G, C, and T). The structure of DNA is a double helix, where A forms two hydrogen bonds with T on the opposite strand, while G forms three hydrogen bonds with C on the opposite strand. A gene is a region of DNA that controls a discrete hereditary characteristic, such as birth, growth and so on, usually corresponding to a single mRNA

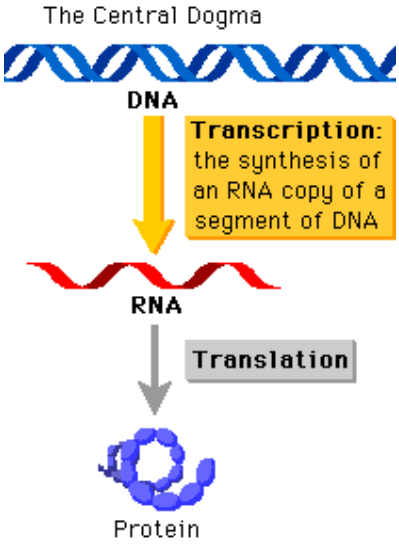

Figure 2.1: The central dogma of molecular biology. Information flows from DNA to RNA by transcription process, and from RNA to protein by translation

carrying the information for constructing a protein. All cells in the same organism have the same genes, but these genes can be expressed differently at different times and under different conditions.

2.2.2 The central dogma of molecular biology

The mechanism by which proteins are produced from their corresponding genes is a two-step process. The first step is the transcription of a gene from DNA into a temporary molecule known as RNA (ribonucleic acid) which is a long chain of DNA as defined in biology dictionary. During the second step, translation, cellular machinery builds a protein using the RNA information as a blueprint. Although there are exceptions to this process, these steps (along with DNA replication) are known as the central dogma of molecular biology.

A segment of DNA is copied into a complementary strand of RNA. The process of transcription is catalyzed by the enzyme called RNA polymerase. Near most of the genes lies a special DNA pattern called promoter, located upstream of the transcription start site, which informs the RNA polymerase where to begin the transcription. This is

achieved with the help of transcriptional factors that recognize the promoter sequence and bind to it.

Messenger RNA (mRNA) is a kind of RNA molecule that transfers the coding information for protein synthesis from chromosomes to ribosomes. Chromosomes are compact spools of DNA and the set of chromosomes within a cell makes up a genome. These chromosomes are duplicated before cells divide, in a process called DNA replication. Ribosomes are the center of protein synthesis. They accept mRNA and use transfer RNA (tRNA) to translate genes into proteins.

Translation occurs after the transcription of DNA to mRNA. The translation of mRNA into protein depends on adaptor molecules that recognize both an amino acid and a triplet of nucleotides. These adaptors consist of a set of small RNA molecules known as tRNA, each about 80 nucleotides in length. The ribosome is a complex of more than 50 different proteins associated with several structural rRNA molecules. rRNA is a machinery for synthesizing proteins by translating mRNA. Each ribosome is a large protein synthesizing machine, on which tRNA molecules position themselves for reading the genetic message encoded in an mRNA molecule.

When a protein is synthesized, a genetic template is limited to 20 amino acids. Many proteins, once synthesized, may undergo posttranslational modification. Judging from the name, this process occurs after translation. In biology dictionary, the process is defined as follows: a number of proteins are synthesized in an inactive form. They can then be activated by another protein, a protease, which cuts the inactive protein at specific sites. This liberates a smaller part of the protein which is now active.

2.3 Standard Form of Microarray

The microarray dataset is considered in the format of a matrix $\mathbf{X} = (x_{ij})$ with m genes (rows) and n array hybridisations or experimental conditions (columns), with $m >> n$, that may contain missing entries. Each row of $\mathbf{X}$ represents the expression levels of a given gene along the n hybridisations and x_{ij} represents the expression level of gene i in array j. Some entry in $\mathbf{X}$ may be missing and this is denoted by an addition matrix $\mathbf{M} = (M_{ij})$ where $M_{ij} = 0$ if the entry is missing and $M_{ij} = 1$, otherwise. A particular gene with MVs to be estimated is called the target gene, whereas the set of genes with available information for estimating the target gene's MVs is the set of candidate genes.

2.4 Missing Values

Missing values create much difficulty in scientific research since most data analysis procedures are not designed for them. There have been several published articles focusing on estimation of missing value for microarray data since 2001, whereas much work has been devoted to similar problems in many other fields with varying degrees of sophistication. The question has been studied in the context of non-response issues in sample surveys and missing data in experiments by Little and Rubin [64]. The rows with missing values can be utilized for further analyses after the imputation of the missing values.

There are many different algorithms for imputation: hot deck imputation and mean imputation [64], regression imputation [80], cluster-based imputation [7], and tree-based imputation [49], maximum likelihood estimation (MLE) [19], and multiple imputations (MI) [41]. Statisticians and other researchers not only have invented numerous methods for handling missing data, but also have invented many forms of missing data. In the next section, we will elaborate on the classification of missing values. Many researchers have been devoted to better understanding and modeling of real-life missing data mechanisms.

As far as microarray is concerned, the presence of missing values constitutes a problem

of crucial importance for downstream data analyses, since many employed methods require complete matrices. The downstream processing methods which include supervised learning algorithms [12, 31] and unsupervised approaches, such as clustering methods with the Hierarchical Clustering (HC) [21], the k-means [34] and the Self-Organizing Map (SOM) [82, 84], have been applied to the analysis of gene expression data as well. Other statistical analysis methods applied to microarray are principal component analysis (PCA) [30], independent component analysis (ICA) [68] and singular value decomposition (SVD) [4].

2.5 Statistical Classification of Missing Values

Missing values are certain values in microarray datasets that are not observed. It occurrs in the phase of data collecting for various reasons, such as administrative error, defective technique, or technology failure. For example, an intended replication may be omitted, a feature of the robotic apparatus may fail, a scanner may have insufficient resolution, or an image may be corrupted [51].

It is beneficial to classify missing values on the basis of the mechanism that produces them. Roughly all of the causes of missing values can be classified by the following classification system, which is based on the relationship between the missing values and data points that have been observed [64].

Missing Completely at Random(MCAR)

> MCAR means that missingness is independent of their own unobserved values and the observed data. That is to say, it arises from chance events that are unrelated to the nature of the investigation. For instance, a spot in microarray is obscured accidentally by a dust particle.

Missing at Random(MAR)

Missingness does not depend on their own unobserved value but does depend on the observed data. This class requires that the cause of the missing data be unrelated to the missing values, but may be related to the observed values. MAR represents a weakening of the assumptions of MCAR.

Missing not at Random(MNAR)

In this class, the missing data mechanism is related to the missing values. Missingness depends on their own unobserved values. It usually occurs when the raw intensity values are zero or small. For example, spots show no fluorescence or have undefined log-intensities because their background-corrected intensities are negative.

Some researchers reported that the proportion of MVs in some microarray dataset is very severe. For example, Brevern *et at.* [21] pointed out that the percentage of gene profiles with at least one MV can be higher than 85%. Since most microarray data analyses only accept complete expression values, the gene expression levels have to be preprocessed in order to impute the missing values before the data analysis. In the following chapter, I will give an overview of the techniques of missing value imputation in microarray. I will also analyse the advantages and disadvantages of each imputation method to facilitate the understanding.

Chapter 3

Literature Review

In the past decade, statisticians and biologists have invented more than a dozen of methods for dealing with missing values in microarray. This chapter overviews the techniques of missing value imputation in microarray which could be categorized based on the level of complexity.

Simple approaches

Simple approaches to deal with missing values mainly include leaving out the genes containing missing values from further analysis although it will usually lose too much useful information, repeating the experiments although this strategy can be expensive and time-consuming, and imputing the missing values by zero or row average although it might lead to high estimation errors [87] due to not taking the gene (or array) correlations into consideration.

Sophisticated approaches

It has been proven that if the correlations between genes are considered, then missing value prediction error can be reduced significantly [8]. Generally, missing value estimation problem has two parts: selection of genes for estimation and design of an imputation rule. Currently many approaches have

Table 3.1: Classification of sophisticated imputation methods

Category	*Imputation methods*	*Time of proposal*
Cluster-based methods	KNNimpute [87]	2001
	SKNN [48]	2004
	WeNNI [45]	2004
	GMCimpute [60]	2004
	Multi-stage approach [98]	2007
Regression-based methods	SVDimpute [87]	2001
	LSimpute [8]	2004
	LLSimpute [47]	2005
	CMVE [75]	2005
	PLSimpute [58]	2004
	SVRimpute [95]	2006
	LADimpute [14]	2006
Bayesian methods	BPCA [59]	2003
	Bayesian gene selection [103]	2003
Iterative methods	IKNNimpute [10]	2007
	ILLSimpute [13]	2006
	IPLSimpute [9]	2006
External biological knowledge incorporated	GOimpute [88]	2006
	POCSimpute [28]	2006
	iMISS [37]	2006
Others	LinCmb [43]	2005
	SEQimpute [92]	2007
	Meta [42]	2007
	FRAA [27]	2006
	IFRAA [26]	2006
	LinImp [69]	2005

been developed to recover missing values. Depending on the gene selection process and characteristic of the imputation rule, sophisticated imputation approaches can be differentiated into several classes, as shown in Table 3.1.

3.1 Classification of Imputation Methods

Since there are voluminous sophisticated research works in dealing with missing values in microarray, it is useful to provide a classification table for better understanding and

comparison. Table 3.1 puts imputation methods into six groups and describes the papers that will be reviewed in the following sections.

3.2 Methods for Dealing with Missing Values in Microarray

Owing to the high number of genes and arrays involved with missing values, we need to use some method to impute the missing values as accurately as possible before continuing subsequent microarray data mining.

3.2.1 Cluster-based imputation methods

k-nearest Neighbor Imputation

Troyanskaya *et al.* [87] were the pioneer in dealing with missing values in microarray data by proposing KNNimpute and SVDimpute. KNNimpute method can be regarded as an improved hot deck imputation method [40] that uses the weighted average values of most similar genes for estimating missing values. Measures for gene similarity in KNNimpute method include Euclidean distance, Pearson correlation and variance minimization. Although both Pearson correlation coefficients and Euclidean distance are likely to be influenced by outliers, they concluded that the latter measure is adequate based on their experiments since log-transforming the data reduces the effect of outliers on gene similarity determination.

Given target gene $\mathbf{x}_t^T$, k-nearest neighbor genes $\mathbf{x}_{s_i}^T (i = 1 \dots k)$ are first taken from matrix $\mathbf{X}$ except any genes that have the same missing position with $\mathbf{x}_t^T$. The distance between target gene $\mathbf{x}_t^T$ and neighboring gene $\mathbf{x}_{s_i}^T$ is defined as:

$$d_{ts_i} = D(\mathbf{x}_t^T, \mathbf{x}_{s_i}^T) = \{\frac{1}{n_{ts_i}} \sum_{j=1}^{n} [x_{tj} - x_{s_i j}]^2 m_{tj} m_{s_i j}\}^{1/2}, \quad i = 1 \dots k, \tag{3.1}$$

where $n_{ts_i} = \sum_{j=1}^{n} m_{tj} m_{s_i j}$ is the number of jointly available values between $\mathbf{x}_t^T$ and $\mathbf{x}_{s_i}^T$, m_{tj} and $m_{s_i j}$ are missingness values for x_{tj} and $x_{s_i j}$ respectively. The weight of gene $x_{s_i}^T$ for estimation is calculated as follows

$$w_{s_i} = \frac{\frac{1}{d_{s_i}}}{\sum_{i=1}^{k} \frac{1}{d_{s_i}}}, \quad i = 1 \ldots k. \tag{3.2}$$

Equations (3.1) and (3.2) show that contribution of each neighboring gene is weighted by the distance of its expression to that of target gene $\mathbf{x}_i^T$.

Currently, there is no absolutely golden rule for the selection of k. A small k will overemphasize a few dominant genes in estimating the missing values, whereas a large k leads to including the genes that have little or even no correlation with target gene. Troyanskaya *et al.* suggest that KNNimpute is relatively insensitive to the exact value of k within the range of 10-20 neighbors [87]. The rationale behind KNNimpute is that those genes closest to the target gene are the most informative, since the missing values in target gene are more likely to behave similarly to that of the neighbor genes. KNNimpute performs well especially when the local correlation is strong. Even though KNNimpute might miss considering negative correlations between genes, which could lead to estimation error [75], it is still the most widely used imputation method due to its simplicity, efficiency and availability. It is the only imputation method implemented in SAM, PAM and MAANOVA [69].

One more issue to clarify is that some authors [51, 60] describe KNNimpute where neighbor genes are not allowed to have any missing values. This might result in a problem, especially in datasets with many missing values, because only a few or no neighbors are free of missing values and the imputation will become poor or impossible in the worst case.

Brás *et al.* [9] proposed a new version of KNNimpute, which takes advantage of array-wise relationships rather than gene-wise relationships. In their method, the gene

expression matrix $\mathbf{X}$ is first transposed before implementing the available KNNimpute software. They call their proposed method KNNarray in the sense that a missing value x_{ij} in $\mathbf{X}$ is estimated by a weighted average of the corresponding ith position of the k similar arrays.

Sequential k-nearest Neighbor Imputation

Kim *et al.* [48] developed a cluster-based imputation method called SKNN whose main characteristic is to utilize previously imputed values for later imputation. SKNN method imputes the missing values sequentially from the gene having least missing entries after sorting the genes according to the missing rate. During each iteration, the gene containing the least number of missing values is chosen as the target gene, and KNNimpute is applied to estimate the missing values in this target gene where only those genes who have no missing values or whose missing values have already been imputed are regarded as candidate genes. Although it uses the imputed values for later imputation, it exhibits practical usefulness in resuming the data originating from microarray experiments which have high missing rate.

Weighted nearest neighbors imputation method

WeNNI includes a measure of spot quality to improve the accuracy of the missing value imputation. WeNNI differentiates itself from other imputation methods in that it adopts continuous spot quality weight, whereas most traditional missing imputation methods consider spots to be of binary value, either missing or present, depending on a cutoff separating poor spots from good spots [45]. However, WeNNI can only outperform KNNimpute and row average. The main contribution of Johansson *et al.* [45] is that they bring the idea of spot quality to the community, which could be generalized in other methods.

GMC Imputation

Ouyang *et al.* [60] introduced an imputation method based on Gaussian mixture

clustering (GMC) and model averaging that has smaller RMSE than KNNimpute and SVDimpute. The assumption for GMCimpute is that microarray data are generated by a Gaussian mixture of some number of components. For each missing value, an estimate is first made from each of the components in the mixture, and then the estimate by the mixture is a linear combination of the component-wise estimates, weighted by the probabilities that the gene belongs to the components. The final estimate by GMCimpute is the average of the estimates by several mixtures.

The main contribution of Ouyang *et al.* [60] is that they examined the bias introduced by imputation to clustering by means of calculating the number of mis-clustered genes. This measures the difference between clustering with true values and that with imputed values, providing another evaluation metric besides root mean squared error. Their paper is one of the few papers that study the impact of missing value on subsequent microarray analysis, such as clustering.

A Multi-stage Approach to Clustering and Imputation

Wong *et al.* [29] described an alternative approach to the clustering of microarray data, leading to an associated imputation method. This method is motivated by Godfrey's work where two-stage clustering has been successfully used in genotype-by-environment analyses with missing data.

3.2.2 Regression-based imputation methods

In this section, a survey of the major regression-based imputation methods is given. There are numerous regression-based imputation methods based on different regression rules, i.e., PCA regression, linear regression, partial least squares regression, etc. A brief description of each method follows.

Singular Value Decomposition Imputation

One of the important characteristics of SVDimpute is that it attempts to utilize the

global information in the entire matrix when predicting the missing values, in contrast to the KNNimpute which takes advantage of the local pairwise relations between genes. The basic idea underlying this method is to find the dominant components, which in this case is identical to principle components of the whole gene expression matrix, and then to predict the missing values in target genes by regressing against these dominant components.

If we perform singular value decomposition [96] to the $m \times n$ matrix $\mathbf{X}$, $m > n$, $\mathbf{X}$ will be expressed as the product of three matrices,

$$\mathbf{X}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}\mathbf{V}^T_{n\times n}\,, \tag{3.3}$$

where the $m \times m$ matrix $\mathbf{U}$ and the $n \times n$ matrix $\mathbf{V}$ are orthogonal matrices, and matrix $\mathbf{V}^T$ now contains n eigengenes, and $\mathbf{\Sigma}$ is an $m \times n$ matrix that contains all zeros except for the diagonal $\sigma_{i,i}, i = 1, \cdots, n$. Holter *et al.* [36] concluded that the product of the first two or three columns of $\mathbf{U\Sigma}$ and the corresponding rows of $\mathbf{V}^T$ can capture the fundamental patterns in cell cycle data.

As just mentioned, there are n eigenvalues on the diagonal of matrix $\mathbf{\Sigma}$ corresponding to eigengenes in $\mathbf{V}^T$. In SVDimpute, once these diagonal elements are rank-ordered and the k most significant eigengenes are selected, the missing value is estimated by first regressing gene i against these k eigengenes, and then using the coefficients of the regression to reconstruct x_{ij} from a linear combination of the k eigengenes [87].

SVDimpute requires a complete data matrix without missing values, and therefore missing values need to be given initial estimates by other methods, such as row average, before applying SVDimpute. Then SVDimpute repeatedly performs SVD on the imputed matrix, until the root mean squared error between two consecutive imputed matrices falls below a given threshold, such as 0.01.

SVDimpute is the pioneer method using global information. Further development

based on global information includes the introduction of Bayesian estimation into principal component analysis [59], partial least squares [58], a covariance-based method to rank genes [76] and support vector regression [95]. Thus, SVDimpute performs well when global structure exists in the expression data.

Least Square Imputation

Hellem *et al.* [8] proposed regression-based method LSimpute which is based on the least squares principle and utilizes correlations between genes or arrays. The least squares principle is based on minimizing the sum of squared errors of a regression model. Missing values are imputed as the weighted average of the predicted values from the regression of genes with missing values against each highly correlated gene. The highly correlated genes are selected based on the absolute Pearson correlation, and the weight assigned to each similar gene is as follows,

$$w_i = (\frac{r_{yx_i}^2}{1 - r_{yx_i}^2 + \varepsilon})^2 , \tag{3.4}$$

where ε=10^{-6}, r_{yx_i} is the Pearson correlation coefficient between target gene and neighbour gene.

LSimpute performs best when data have a strong local correlation structure as demonstrated by three example datasets [8]. Although it differentiates from KNNimpute in the sense that they have two different ways to choose the nearest neighbours, it has the same order of computational complexity as KNNimpute.

Local Least Square Imputation

LLSimpute method [47] is also a least squares based imputation method, where a target gene that has missing values is represented as a linear combination of similar genes. However, it should be noticed that LLSimpute and LSimpute use different approaches for imputation, although both are least squares related. LSimpute method explores uni-

variate linear regression, while LLSimpute method employs multivariate linear regression using all k nearest neighbours.

The missing values are imputed on the basis of the least square estimates, determined by the pseudoinverse of the k nearest neighbours expression matrix [47]. When the missing rate is relatively small, the neighbour genes with MVs will be excluded from the least squares system. If the percentage of MVs is high, MVs are initially estimated by the row average. The main contribution of Kim and his colleagues is that they proposed a heuristic algorithm for estimating parameter k which makes LLSimpute look like a non-parametric missing value estimation method, such as BPCA. A drawback of LLSimpute is that when the patterns of gene expression are highly correlated, multi-collinearity is probably the main cause for its poor performance.

Collateral Missing Value Estimation

CMVE employs covariance function to capture both positive and negative correlation between genes, and introduces multiple parallel estimates of missing values. The final imputed value is a weighted average of three estimates for the missing values [75],

$$\chi = \rho \cdot \Phi_1 + \Delta \cdot \Phi_2 + \Lambda \cdot \Phi_3 \,,$$

where $\rho = \Delta = \Lambda = 0.03$ ensures an equal weight to the respective estimates Φ_1, Φ_2 and Φ_3. These estimates are calculated using the k genes with highest covariance to the target gene through least square regression and linear programming methods.

Partial Least Square Imputation

Nguyen *et al.* [58] employed partial least squares to impute the missing values. Since the number of samples is much smaller than the number of genes in the training data, dimension reduction is necessary. PLSimpute first selects linear combinations of genes (referred to as components), showing high covariance with the gene having MVs. The

first linear combination has the highest covariance with the target gene, and subsequent steps seek the strongest mode of candidate and target gene covariation in a direction orthogonal to the previously selected components until a total number of K_p components are selected. The missing values are estimated by regressing the target gene against the K_p PLS gene components. In practice, MVs are first imputed by initial estimates from KNNimpute since X^A_{-j} is required to be complete. Their results showed that PLSimpute provides improvement in accuracy over some ranges of expression values where KNNimpute did not perform well.

SVRimpute

Wang *et al.* [95] proposed a method based on the support vector regression (SVR) to estimate the missing values and use orthogonal input coding scheme to address the issue of multiple missing entries in one row of certain expression profile. Because the SVR can estimate only one missing value in a row at one time, in the case of more than one missing entry in one row, they successfully employed orthogonal input coding scheme. Their idea is novel in the sense that it utilizes multi-missing values in one row of a certain gene expression profile and imputes the missing value into a much higher dimensional space, in order to obtain better performance.

Least Absolute Deviation Imputation

Cao and Poh [14] proposed a novel imputation method employing least absolute deviation estimate, referred to as LADimpute, to impute missing entries in microarray. The proposed LADimpute method also takes into consideration the local similarity structures in addition to employment of least absolute deviation estimate. Once those genes similar to the target gene with missing values are selected based on Euclidean distance or Pearson correlation coefficient metric, all missing values in the target gene can be estimated by the linear combination of the similar genes simultaneously. The experimental results show that LADimpute exhibits its robust performance over some datasets with high missing

rate or noise level.

3.2.3 Bayesian imputation methods

Bayesian PCA imputation

BPCA method consists of three elementary processes that are principal component (PC) regression, Bayesian estimation and expectation maximization (EM)-like repetitive algorithm. In PC regression, missing values G^{miss} in $\mathbf{G}$ are estimated from those genes having no missing value G^{obs} by using the PCA result. Bayesian estimation obtains the posterior distribution of θ and X by Bayes theorem:

$$p(\theta, X|G) \propto p(G, X|\theta)p(\theta)\,, \tag{3.5}$$

where $p(\theta)$ is a prior distribution.

Variational Bayes (VB) algorithm, which is similar to the EM algorithm, is used to execute Bayesian estimation for both model parameter θ and missing values G^{miss}. It obtains the posterior distribution for θ and G^{miss}, $q(\theta)$ and $q(G^{miss})$ respectively. Finally, the missing values in the expression matrix $\mathbf{G}$ are estimated to the expectation with respect to the estimated posterior distribution [59]:

$$\hat{G}^{miss} = \int G^{miss} q(G^{miss}) dG^{miss}\,, \tag{3.6}$$

$$q(G^{miss}) = p(G^{miss}|G^{obs}, \theta_{true})\,. \tag{3.7}$$

BPCA also depends on the number of principal axes (eigenvectors) k, similar to KN-Nimpute. But Oba *et al.* [59] proposed a formula to calculate the value of k, which is $k = n - 1$, where n is the number of samples. A slight drawback of BPCA is that the

convergence rate can be so slow that the computation process may take hours. Moreover, BPCA might not be very accurate if genes have dominant local similarity structures due to the assumption of only a global covariance structure.

Bayesian Gene Selection

Zhou *et al.* [103] proposed a missing value estimation method from the viewpoint of linear or nonlinear regression with Bayesian variable selection. They pointed out that the genes selected by KNNimpute are not necessarily among the best choices for linear prediction of the target gene because the gene selection and missing value estimation are treated as two independent procedures, namely using two different models. On the basis of this motivation, in their study they formulated the gene selection problem as a linear or nonlinear regression with Bayesian variable selection [50, 79], and then devised a Gibbs sampler to solve it. They aimed at the auto-selection of the number of the nearest neighbor genes, and then both linear and nonlinear regression are used for the estimation rule. The procedure for the fast implementation has been developed for the essential steps of the algorithm.

3.2.4 Iterative imputation methods

With similar modifications to KNNimpute, LLSimpute and PLSimpute, some imputation methods implement iterated imputation intending to improve the quality stepwise. The iterative procedure reimputes the missing values using the imputation results from the last iteration for a given number of iterations or until the imputed values converge.

In ILLSimpute, Cai *et al.* [13] found that it is not wise to set a uniform number of similar genes for all target genes, regardless of the quality difference for different target genes. Instead, they proposed to set up a distance ratio threshold to cut off dissimilar genes. Only candidate genes whose distances to target gene lie within the threshold are considered as similar genes. A limitation of the ILLSimpute method is that

its implementation produces inconsistent results in some cases, which result in a large variation among the replicate datasets [89].

With the aim of improving the MVs estimates, Brás *et al.* [9] proposed IPLSimpute by employing an iterative procedure to PLSimpute, and concluded that the use of iteration can refine the MV estimates in the presence of high proportion of missing entries, since it minimises the bias incurred by the initial estimates. Later, they also extended KNNimpute method to IKNNimpute, and studied the effect of missing value estimates on the subsequent analysis, i.e., the differential analysis of gene expression, in addition to using normalized root mean squared errors measure.

3.2.5 External biological knowledge incorporated methods

All of the above imputation methods are solely based on the gene expression datasets themselves and make no use of domain knowledge.

GOimpute [88] is the first algorithm that utilizes external information on functional similarities embedded in the Gene Ontology (GO) databases, in addition to expression data similarity, to facilitate the missing value estimation. GOimpute method is generic in the sense that it describes an imputation framework, with emphasis on the selection of neighbor genes. It outperforms KNNimpute especially in the case of limited number of arrays or at high missing percentage. However, GOimpute failed to improve LLSimpute, due to the limitation on the number and accuracy of the gene functions annotated in gene ontology databases.

Integrative Missing Value Estimation method (iMISS) [37] incorporates information from multiple reference microarray datasets to improve missing value estimation. This approach takes advantage of the conserved gene expression patterns among multiple datasets to enhance the neighbor gene selection. However, it is difficult to find a set of genes that shows consistent expression similarity to the target gene over multiple datasets.

Unlike GOimpute and iMISS which mainly improve neighbor gene selection through external information, POCSimpute [28] exploits the biological phenomenon of synchronization loss in microarray experiments, which is a common phenomenon in cyclic systems. However, the convex sets based on cyclic loss property are only applied well for time series microarray dataset and degenerate for a dataset without cyclic loss phenomenon.

3.2.6 Others

The literature focusing on missing values in microarray and algorithms to recover them have been increasing in the last several years. There are some imputation methods which are not put in the above five categories, e.g., LinCmb [43], FRAA [27], Meta [42], LinImp [69] and SEQimpute [92]. The increasing number of papers suggests the importance of estimating missing values, which has drawn a lot of attention from the community. Among these imputation methods, how do researchers determine the best method to use for imputing missing values? The following section details this further from the perspective of evaluation criteria.

3.3 A Review on Evaluation Criteria

This section summarizes the evaluation criteria for the above algorithms, based on both theoretical and experimental foundations.

3.3.1 Theoretical evaluation

Missing value imputation methods can be evaluated on the following criteria:

1. accuracy,
2. robustness,
3. real impact on subsequent analysis,

4. computational cost.

Accuracy is tested by calculating prediction error between actual values and imputed values after missing values are estimated. It reflects the degree of recovering the missing values. The smaller the prediction error, the better the imputation method. Robustness means whether the imputation method is sensitive to different types of datasets, various missing rates, or different missing patterns.

After imputation, what attracts our interest is the impact on subsequent microarray data mining process. Lastly, we have to mention computational cost, also called computational burden. How much time does it take to finish imputation? For example, iterative methods are normally taking more time to impute missing values than non-iterative methods. The smaller the computation time, the more preferable the method. Currently with the advanced computer hardware, computational cost is not a top priority in terms of evaluation criteria. That is the reason why we put it last in the list of evaluation criteria.

In order to evaluate the performance of imputation methods, a portion of missing values are introduced by knocking out values in the complete dataset in the pattern of MAR or MNAR, introduced in Section 2.5. Hence, we know y_{real} because the missing entries are artificial, and the procedure of producing artificial missing entries will be described in detail in Section 3.3.2. Specifically, accuracy can be calculated in the following ways:

Normalized Root Mean Squared Error(NRMSE)

$$NRMSE = \sqrt{\frac{mean[(y_{guess} - y_{real})^2]}{Var(y_{real})}}, \tag{3.8}$$

where y_{guess} and y_{real} are estimated values and actual values respectively. The mean and the variance are calculated over missing entries in the complete matrix. Note that when the expression matrix is given, $Var(y_{real})$ is given as a constant. Thus according to the definition of NRMSE, a smaller NRMSE

value indicates a better imputation quality.

Most of the existing methods use this measure to evaluate the accuracy of the imputation methods, as in BPCA and LLSimpute, which allows direct comparisons of imputation accuracy across different datasets. Troyanskaya *et al.* [87] normalize the RMSE in a different way, dividing it by the average data value in the complete dataset. Ouyang *et al.* [60] normalize the RMSE by dividing it by the root mean square of all the values in the complete dataset, while Bϕ *et al.* [8] do not normalize, and just take RMSE as evaluating measure.

Robust Normalized Squared Error(RNSE)

$$RNSE = \frac{\sqrt{median[(\mathbf{y_{real}} - \mathbf{y_{guess}})^{\mathbf{2}}]}}{mad(\mathbf{y_{real}})}\,, \tag{3.9}$$

where $\mathbf{y_{real}}$ and $\mathbf{y_{guess}}$ are the vectors of actual and estimated values, respectively and *mad* is defined as

$$mad(\mathbf{y_{real}}) = \frac{median|\mathbf{y_{real}} - median(\mathbf{y_{real}})|}{0.6745}\,.$$

Squared Pearson Correlation Coefficients(R^2)

Besides NRMSE and RNSE, Pearson correlation coefficients are used to evaluate imputation methods. NRMSE provides the accuracy on average, but the accuracy across each array is unlikely to be uniform and this is revealed by Pearson correlation coefficients. Pearson correlation coefficients are calculated between imputed data and complete data for each array, as shown in [48] and [10]. From this evaluation, we could assess how the data structure of each array will be preserved after imputation.

Paired T-test

The performance of two different imputation methods (or the same imputation method with different parameters) can be compared using t-test. The purpose of using t-test is to test whether the prediction errors obtained by method A are significantly smaller on average compared to the prediction errors obtained by method B, assuming that method A leads to more accurate results than method B. Considering the residual from two methods $\varepsilon^A = y_{real} - y^A_{guess}$ and $\varepsilon^B = y_{real} - y^B_{guess}$, where y^A_{guess} and y^B_{guess} are the estimated values using method A and method B respectively, ε^A and ε^B are the prediction errors for two methods. By taking the difference $d_i = |\varepsilon^A_i| - |\varepsilon^B_i|$ for each missing value ($i = 1, 2, \cdots, N$, where N is the number of missing values), we can test whether the average difference in terms of prediction error

$$\bar{d} = \frac{1}{N}\sum_{i=1}^{N} d_i$$

is significantly larger than zero [8].

Wilcoxon Signed Rank Test

This measure is to test the bias on the imputation method, whether it gives a consistent underestimation or overestimation of the true values. Considering the residuals $\varepsilon = y_{real} - y_{guess}$ of a given imputation method, the null hypothesis (H_0) states that negative and positive residuals are equally likely [78].

Wilcoxon signed rank test is a nonparametric alternative to the paired t-test, which is similar to the Fisher sign test. This test assumes that there is information in the magnitudes of the differences between actual (y_{real}) and imputed (y_{guess}) values, as well as the signs. One can get the statistic by means of taking the paired y_{real} and y_{guess}, calculating the differences ε, ranking them

from smallest to largest by absolute value, and then adding all the ranks associated with positive differences. Finally, the p-value associated with this statistic is found from an appropriate table.

3.3.2 Experimental evaluation

To evaluate the performance of imputation methods, we need a complete dataset on which simulation study can be conducted. The original dataset always has missing values. In order to compare and validate imputation methods, a clean dataset without missing values is formed by discarding those genes with missing values from the original dataset. In some cases, when the missing percentage in one particular array is too high, that array is deleted before removing the genes with missing values [47]. After having the complete data, a specific percentage of the entries will be labeled as missing in the complete dataset, pretending that they are missing. Finally, we recover these missing values by applying imputation method. The whole experimental evaluation workflow can be demonstrated in Figure 3.1.

There are various missing data patterns, including missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR), which are illustrated in detail in Section 2.5. MAR pattern applies well in the cases in which missing entries are caused by effects such as irregularities in the spot production, hybridization failure, dust on the chip, etc., and they will have random distribution. Each of the existing imputation methods dealing with missing values in microarray considers missing at random (MAR) pattern, while only a few papers test the performance of imputation method in terms of MNAR pattern [9, 10, 75], since missing not at random is the most complicated situation.

In this situation, the missingness depends on the expression value that are not observed. If the signal is too low, or the image processing software used for spotted cDNA microarrays flags out signals that cannot be distinguished from the background, missing

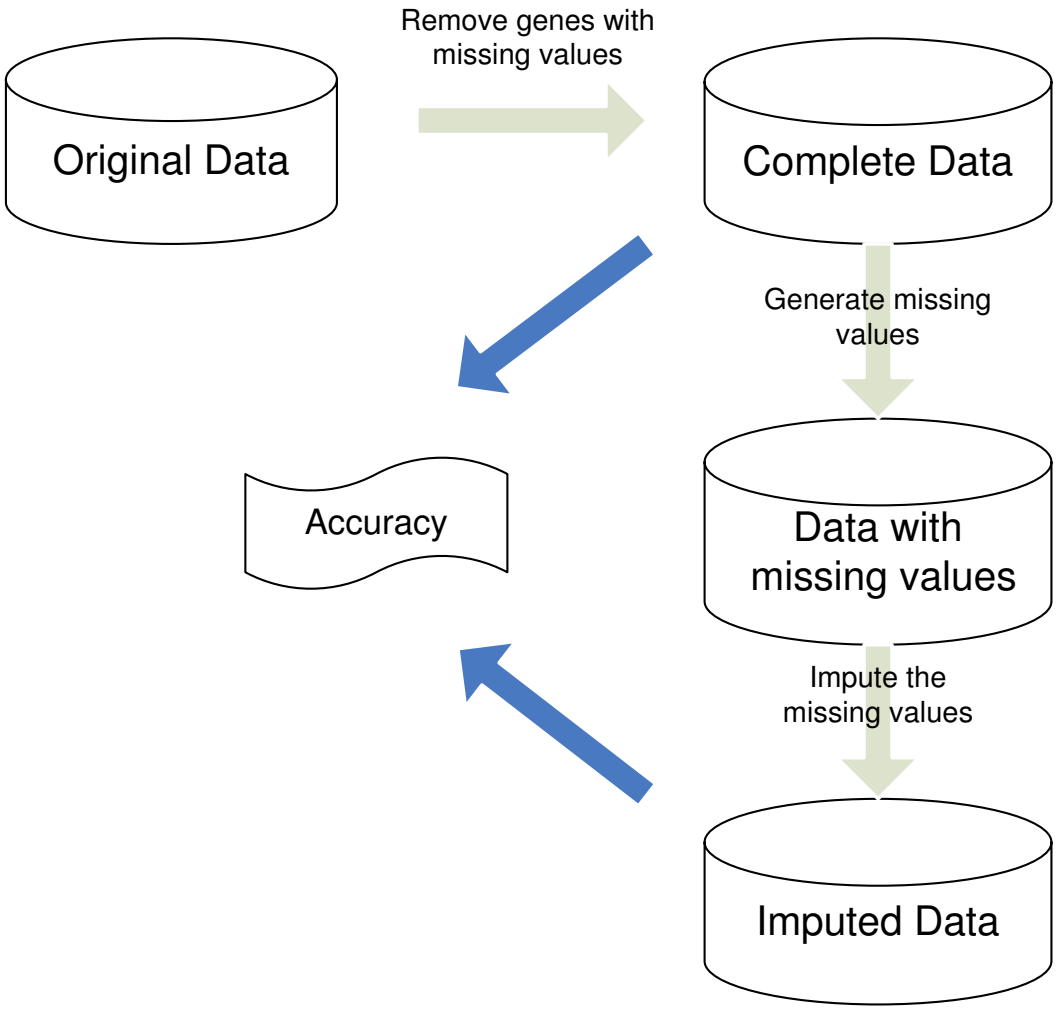

Figure 3.1: The workflow of experimental evaluation on imputation method

entries are not introduced randomly in such cases, but instead missing pattern depends on the signal intensity, leading to MNAR scenario. That is to say, randomly introduced missing values may not truly reflect the nature of actual microarray data missing pattern.

Sehgal *et al.* [75] tried to retain the same distribution of missing values as in original dataset by proposing a delay function. Brás *et al.* [10] considered a mixture pattern of missing at random and not at random which appears to be close to the realistic missing data patterns, i.e., an abnormally high frequency of missing values in some arrays. There are complex statistical issues associated with MNAR, but how to reflect the actual missing pattern in the simulation study still remains open. More methods or procedures need to be proposed and investigated.

Chapter 4

Nonparametric Regression Approach for Imputation Based on Gene-wise Relationships

Chapter 4 focuses on proposing a nonparametric regression approach (NPRA) to address the missing value problem. This method takes into consideration non-parametric regression as well as k-nearest neighbour method. In Section 4.1, very basic knowledge about nonparametric regression is introduced and important attention is given to kernel regression. Section 4.2 presents the rationale behind NPRA and Section 4.3 provides the notations used and describes the algorithm of NPRA in detail, demonstrating the case of both single missing entry and multiple missing entries in a gene. In Section 4.4 and Section 4.5, a comprehensive comparison of NPRA and other existing imputation methods is conducted under different scenarios and then results are analyzed by means of different evaluation metrics. Section 4.6 concludes this chapter with a summary.

4.1 Introduction

Many approaches have been developed to recover missing values, such as least square imputation (LSimpute) [8], local least squares imputation (LLSimpute) [47] and collateral missing value estimation (CMVE) [75]. These three typical regression-based methods exploit the local similarity of structure between genes. All of them are based on the assumption of linear relations between genes. However, the cold stress response experiment clearly shows a non-linear behaviour in gene expression data [71]. Zhou *et al.* [102] also found some genes showing a strong nonlinear property and employed non-linear models to impute missing values by means of Bayesian variable selection [103].

This chapter focuses on the employment of nonparametric regression model to capture both the linear and non-linear relations between genes [55]. Regression analysis is perhaps the most widely used tool in data analysis. We first briefly outline parametric regression and then move quickly to nonparametric regression. The most popular parametric regression model so far is the linear regression model [65] given by

$$y_i = \beta_0 + X_i'\beta + e_i\,, \qquad i = 1, \cdots, n\,, \tag{4.1}$$

where $X_i \in \mathbb{R}^q$, and β is a $q \times 1$ vector of unknown parameters, whereas a more general nonlinear regression model is described as follows

$$y_i = f(X_i, \beta) + e_i\,, \qquad i = 1, \cdots, n\,, \tag{4.2}$$

where $f(\cdot, \cdot)$ has a known functional form with β again being a vector of unknown parameters. For example, Zhou *et al.* [102] posit a model of the form $f(x, \beta) = exp(x'\beta)$. Having specified the functional form of the regression model up to a finite number of unknown parameters, methods such as ordinary least squares or nonlinear least squares could then be used to estimate the unknown parameter vector β in (4.1) or (4.2) respectively.

Since parametric methods require practitioners to specify the exact parametric form of the model prior to estimation, one must confront the case that the model may not be consistent with the data. Nonparametric regression models allow one to analyze data without any prior knowledge about the data. In Section 4.1.1, we will introduce the theory of nonparametric regression.

4.1.1 Nonparametric regression

One of the main aims of nonparametric regression is to reflect an important structure in the data without any assumption about the data. Rather than presume one knows the exact functional form of the model, one assumes the model satisfies some regularity conditions such as smoothness and moment conditions. One important characteristic of nonparametric regression is the need to have more data to achieve the same degree of precision as a parametric regression model.

The general nonparametric regression model is written in the following manner

$$y_i = f(X_i') + e_i = f(x_{i1}, x_{i2}, \ldots, x_{iq}) + e_i\,, \quad i = 1, \ldots, n\,, \tag{4.3}$$

where $f(\cdot)$ is a smooth and continuous but unspecified function and e_i is the normally distributed error with mean 0 and variance σ^2. Moreover, the objective of non-parametric regression is to estimate the regression function $f(\cdot)$ directly rather than to estimate parameters. In simple words, one assumes that y_i has relation with X_i'. The relation might be linear or non-linear, but we do not know it exactly.

An important special case of the general model is nonparametric simple regression, where there is only one predictor

$$y_i = f(x_i) + e_i\,, \quad i = 1, \ldots, n\,. \tag{4.4}$$

If $f(\cdot)$ is a smooth function, then one can estimate $f(\cdot)$ with $\hat{f}(\cdot)$ nonparametrically using kernel methods [24].

4.1.2 Kernel estimator

In (4.4), there is single regressor and kernel regression can be employed to estimate function $f(\cdot)$. Very similar to parametric regression, a weighted sum of the y_i is used to obtain the fitted values as the following,

$$\hat{y}_{i(ker)} = \sum_{j=1}^{n} w_{ij(ker)} y_j \,, \tag{4.5}$$

and

$$w_{ij(ker)} = \frac{K[(x_i - x_j)/h]}{\sum_{j=1}^{n} K[(x_i - x_j)/h]} \,, \tag{4.6}$$

where $K(u)$ is called kernel function and $h > 0$ is called bandwidth.

Note that equation (4.5) can be thought of as calculating $\hat{f}(x_i)$ for $i = 1, \ldots, n$. Each of the n data points is assigned a distinct weight $w_{ij(ker)}, j = 1, \ldots, n$, for any point of x_i. $K(u)$, the kernel function, is a decreasing function of $\|u\|$. A close examination of the equation (4.6) lends some insight into the weighting scenario, that is, the points closest to x_i receive more weight than those more remote from x_i. The kernel function should be symmetric and the bandwidth, h, is a smoothing parameter, which determines how fast the weights decrease as the distance from x_i increases. The rate at which the weights decrease relative to the locations of the $x_i's$, in turn, controls the smoothness of the resulting estimate of f.

There are many commonly used forms of the kernel function, such as the Gaussian kernel and uniform kernel. Gaussian kernel is a widely used kernel function of the following form:

$$K(u) = (2\pi)^{-1/2} e^{-u^2/2} \,. \tag{4.7}$$

It has been pointed out that it is not the choice of kernel function but the choice of bandwidth, that is critical to the performance of the nonparametric regression. Small values of h would make the resulting fit undersmoothed or overfit and possess high variance, whereas large values of h would lead to oversmoothed, or underfit with high bias.

Currently, there is no generally accepted method for choosing the bandwidth h for the estimation of the unknown regression function $f(\cdot)$ by kernel estimator given by (4.6). Methods that are available consist of subjective choice and automatic methods such as plug-in, cross-validation, and modified AIC procedure [52].

4.2 Basic Idea of Nonparametric Regression Approach

The rationale behind nonparametric regression approach (NPRA) is to take into consideration gene-wise information. It is almost impossible to know exactly the relations between genes. Under such circumstances, the use of nonparametric regression model is explored, since it detects either linear or non-linear relationship among genes. It is beneficial to use non-parametric regression model [15] when the relationship between predictor gene and target gene is unknown. In our previous work, we only chose the nearest neighbour gene for target gene to impute missing values. In this study, we will extend the previous work to consider k-nearest neighbour genes in non-parametric regression.

We have seen that almost all of the imputation algorithms make use of the information between genes for missing value imputation. Gene-wise information can be classified as local or global. Local means that only a subset of nearest genes are taken into consideration, such as KNNimpute [87] or LSimpute [8], whereas global means that all genes in the dataset are considered, such as SVDimpute [87] or BPCA [59].

Local information is preferred over global information, since only a small number of genes are in a cluster and are strongly correlated, which may be a consequence of the

characteristic of gene co-expression network. Agrawal *et al.* [2] and Noort *et al.* [91] pointed out that the gene co-expression network is a small world and scale free network. In such a network, there could be many nodes but the network is sparse. That leads to the result that many nodes have few connections and only a small number of nodes have many interactions.

NPRA mainly exploits the local similarity structure among genes, and it begins by selecting a set of genes closest to the gene with missing values. The number of the selected nearest genes appears to be an issue, because selecting too many genes will bring deviation to imputation process, while one may miss some important genes if the number is too little. It is also a relevant problem for KNNimpute, LSimpute and LLSimpute to choose the number of nearest neighbours k, and all have given suggestions from their own perspectives.

4.3 Nonparametric Regression Approach for Imputation

In microarray data mining, relations are usually captured in a gene-wise fashion, since genes that have close biological function would express very similarly. This is based on the fact that there is a connection between the function of a gene and its expression pattern. Genes involved in specific cellular processes are co-regulated and co-expressed. But only a small number of genes would co-express similarly and using a small set of genes for missing value imputation has led to good performance [10, 8, 47].

4.3.1 Notation

Recall Section 2.3 in Chapter 2, a microarray dataset is denoted by an $m \times n$ matrix $\mathbf{X} = (x_{ij})$, where m is the number of genes (rows) and n is the number of arrays.

$$\mathbf{X} = \begin{bmatrix} \mathbf{g}_1^T \\ \vdots \\ \mathbf{g}_m^T \end{bmatrix} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix} \in \mathbb{R}^{m \times n}$$

In the matrix $\mathbf{X}$, a row $\mathbf{g}_i^T \in \mathbb{R}^{1 \times n}$ represents expression of the i^{th} gene in n experiments (where the superscript T denotes transpose).

Some entry in $\mathbf{X}$ may be missing and is denoted by an addition matrix $\mathbf{M} = (M_{ij})$

$$\mathbf{M} = \begin{pmatrix} M_{11} & M_{12} & \cdots & M_{1n} \\ M_{21} & M_{22} & \cdots & M_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ M_{m1} & M_{m2} & \cdots & M_{mn} \end{pmatrix} \in \mathbb{R}^{m \times n},$$

where $M_{ij} = 0$ if the entry is missing and $M_{ij} = 1$, otherwise.

4.3.2 Single missing entry in a gene

Without loss of generality, we suppose that target gene vector $\mathbf{g}_t^T$ has a missing value at the first array, denoted by α. For the k selected nearest genes, let $\mathbf{g}_{s_i}$ be the i^{th} selected gene with its first element γ_{s_i}, where s_i denotes the index for representing k selected nearest genes for $i = 1, \cdots, k$. The selected genes have complete values without missing

entries. Then we get the following matrix,

$$\begin{bmatrix} \mathbf{g}_t^T \\ \mathbf{g}_{s_1}^T \\ \mathbf{g}_{s_2}^T \\ \vdots \\ \mathbf{g}_{s_k}^T \end{bmatrix} = \begin{pmatrix} \alpha & x_{t,2} & x_{t,3} & \cdots & x_{t,n} \\ \gamma_{s_1} & x_{s_1,2} & x_{s_1,3} & \cdots & x_{s_1,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_{s_k} & x_{s_k,2} & x_{s_k,3} & \cdots & x_{s_k,n} \end{pmatrix} = \begin{pmatrix} \alpha & \mathbf{y}_t^T \\ \gamma_{s_1} & \mathbf{x}_{s_1}^T \\ \vdots & \vdots \\ \gamma_{s_k} & \mathbf{x}_{s_k}^T \end{pmatrix} \in \mathbb{R}^{(k+1)\times n},$$

where $\mathbf{y}_t = [x_{t,2}, x_{t,3}, \cdots, x_{t,n}]^T$ is a subvector of $\mathbf{g}_t$ excluding the missing value α.

NPRA is summarized as three parts:

1. selection of k similar genes,
2. nonparametric regression with the k similar genes,
3. weighted average of the k estimates.

STEP 1: Selection of k similar genes

NPRA first selects the k-nearest genes from $\mathbf{X}$, where k is a pre-determined number. We use L_2-norm (Euclidean distance) as a distance measure to select the k nearest genes for imputing the missing value α as follows,

$$d_{sin}(\mathbf{g}_t, \mathbf{g}_{s_i}) = \{\frac{1}{n-1}\sum_{j=2}^{n}[x_{t,j} - x_{s_i,j}]^2\}^{1/2}. \tag{4.8}$$

STEP 2: Nonparametric regression with the k similar genes

Based on the k nearest genes, we consider $\mathbf{y}_t$ as the response variable and $\mathbf{x}_{s_i}$ as predictor variable to conduct nonparametric simple regression,

$$\mathbf{y}_t = f_{s_i}(\mathbf{x}_{s_i}). \tag{4.9}$$

For example, we assume that there is a relation between $\mathbf{g}_t^T$ and $\mathbf{g}_{s_1}^T$, and estimate the unknown function $f_{s_1}(\cdot)$ by $\hat{f}_{s_1}(\cdot)$ on the basis of (4.5) and (4.6), and then get an estimate of α which is $\hat{f}_{s_1}(\gamma_{s_1})$. Similarly for $\mathbf{g}_{s_i}^T, i = 2, \cdots, k$, we conduct nonparametric simple regression and estimate unknown function $\hat{f}_{s_i}(\cdot)$.

STEP 3: Weighted average of the k estimates

Finally a weighted average of $\hat{f}_{s_i}(\gamma_{s_i}), i = 1, \cdots, k$ is the final estimate of α as the following,

$$w(s_i) = \frac{1}{d_{sin}(\mathbf{g}_t, \mathbf{g}_{s_i})} / \sum_{i=1}^{k} \frac{1}{d_{sin}(\mathbf{g}_t, \mathbf{g}_{s_i})}\,, \quad i = 1 \ldots k\,. \tag{4.10}$$

and

$$\alpha = \sum_{i=1}^{k} w(s_i) \cdot \hat{f}_{s_i}(\gamma_{s_i})\,. \tag{4.11}$$

4.3.3 Multiple missing entries in a gene

In this section, we discuss the case in which there are more than one missing value in a gene vector. In this case, to recover the total of n' missing values in any location of the target gene $\mathbf{g}_t$, first the k similar gene vectors for $\mathbf{g}_t$,

$$\mathbf{g}_{s_i}^T \in \mathbb{R}^{1 \times n}$$

are found.

For simplicity, suppose that the target gene has two missing values in the first and second position, denoted by α_1 and α_2. For the k selected nearest genes, they must have

observed values in the first and second position as follows,

$$\begin{bmatrix} \mathbf{g}_t^T \\ \mathbf{g}_{s_1}^T \\ \mathbf{g}_{s_2}^T \\ \vdots \\ \mathbf{g}_{s_k}^T \end{bmatrix} = \begin{pmatrix} \alpha_1 & \alpha_2 & x_{t,3} & \cdots & x_{t,n} \\ \gamma_{s_1,1} & \gamma_{s_1,2} & x_{s_1,3} & \cdots & x_{s_1,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_{s_k,1} & \gamma_{s_k,2} & x_{s_k,3} & \cdots & x_{s_k,n} \end{pmatrix} = \begin{pmatrix} \alpha_1 & \alpha_2 & \mathbf{y}_t^T \\ \gamma_{s_1,1} & \gamma_{s_1,2} & \mathbf{x}_{s_1}^T \\ \vdots & \vdots & \vdots \\ \gamma_{s_k,1} & \gamma_{s_k,2} & \mathbf{x}_{s_k}^T \end{pmatrix},$$

where $\mathbf{y}_t = [x_{t,3}, \cdots, x_{t,n}]^T$ is a subvector of $\mathbf{g}_t$ excluding the missing values α_1 and α_2 and $\gamma_{s_i,1}$ and $\gamma_{s_i,2}, i = 1, \cdots, k$ must be observable.

The Euclidean distance between $\mathbf{g}_t$ and $\mathbf{g}_{s_i}$ can be calculated by

$$d_{mul}(\mathbf{g}_t, \mathbf{g}_{s_i}) = \{\frac{1}{n_{s_i}} \sum_{j=3}^{n} [x_{t,j} - x_{s_i,j}]^2 M_{s_ij}\}^{1/2}, \qquad i = 1 \ldots k\,, \tag{4.12}$$

where $n_{s_i} = \sum_{j=3}^{n} M_{s_ij}$ is the number of available values in $\mathbf{x}_{s_i}^T$, and M_{s_ij} is missingness value for x_{s_ij}.

After selecting k similar genes g_{s_i}, we regress $\mathbf{y}_t$ against each of $\mathbf{x}_{s_i}$ as follows,

$$\mathbf{y}_t^* = f_{s_i}^*(\mathbf{x}_{s_i}^*)\,, \tag{4.13}$$

where $\mathbf{y}_t^*$ and $\mathbf{x}_{s_i}^*$ are the simultaneously observable elements in the vectors $\mathbf{y}_t$ and $\mathbf{x}_{s_i}$. If there are also some missing entries in $\mathbf{x}_{s_i}$ and the number of missing entries is quite large, the nonparametric regression analysis would be affected. In this specific case, we impute $\mathbf{x}_{s_i}$ first by a simple method, such as row average, and then (4.13) will degenerate to (4.9).

Finally, similar to the case of single missing entry, a weighted average of $\hat{f}_{s_i}(\gamma_{s_i}), i =$

$1, \cdots, k$, is the final estimate of α as the following,

$$w(s_i) = \frac{1}{d_{mul}(\mathbf{g}_t, \mathbf{g}_{s_i})} / \sum_{i=1}^{k} \frac{1}{d_{mul}(\mathbf{g}_t, \mathbf{g}_{s_i})}\,, \quad i = 1 \ldots k\,. \tag{4.14}$$

and

$$\alpha_1 = \sum_{i=1}^{k} w(s_i) \cdot \hat{f}^*_{s_i}(\gamma_{s_i,1})\,, \qquad \alpha_2 = \sum_{i=1}^{k} w(s_i) \cdot \hat{f}^*_{s_i}(\gamma_{s_i,2})\,. \tag{4.15}$$

4.4 Evaluation

4.4.1 Dataset

In order to assess our proposed method, four datasets were selected from publicly available microarray data. The datasets were classified into three types: time-series (TS), non-time series(NTS) or mixed (MIX) experiments, as shown in Table 4.1.

Table 4.1: Overview of datasets

	GASCH	LISTERIA	CALCINEURIN	BC
Type of microarray	Time series	Time series	Mixed	Non-time series
Original dataset	6,152 × 173	16,838×39	6,102×24	3,226×22
Complete dataset	2,239 × 159	6,850×39	4,379×24	3,226×22
Overall missing rate (%)	3	7.2	3.2	0

Prior to analysis, data were log-transformed (base 2) before applying the imputation methods where datasets were not downloaded in log_2 scale.

GASCH [62]

The first dataset is from a study of response to environmental changes in yeast [62]. It contains 6,152 genes and 173 experiments that have time-series

of specific treatments. After removing columns having more than 10% missing values and then selecting genes that don't have missing value, we got a complete matrix of 2,239 genes and 159 experiments.

LISTERIA [22]

The second dataset is from a study about infection time series by Listeria monocytogenes. It has 16,838 genes and 39 time series in the original dataset. After removing all genes with missing values, resulting in a $6,850 \times 39$ data matrix.

CALCINEURIN [33]

The third dataset is from a study of gene expression regulated by the calcineurin /Crzlp-signalling pathway in yeast. It has 6,102 genes and 24 experiments in the original dataset. After removing all genes with missing vaules, a $4,379 \times 24$ data matrix is obtained.

Breast Cancer [35]

Hedenfalk *et al.* [35] monitored the global expression patterns of 7 breast cancer samples with BRCA1 mutation, 8 with BRCA2 mutation, and 7 sporadic cases with neither mutations using cDNA microarrays. There are 6,512 cDNA used which represent 5361 unique genes. The dataset is formed by selecting 3,226 genes for analysis and available publicly, which has multiple experiments with no time course relationships.

4.4.2 Missing data setup

We knocked out values in the complete dataset, pretending that they are missing and compared imputed values with the actual value to evaluate the accuracy of each imputation method. We employed two strategies to produce missing entries. Multiple missing

values were allowed in a gene at different experiments. One is to randomly remove a specific percentage of entries (5%, 10%, 15%, and 20%) from the complete dataset, and the other one is to assign MVs to the elements in the m rows of the complete dataset ($m \times n$) by randomly sampling m rows from the original dataset, which is a $p \times n$ matrix, and using their missing positions. The latter used the missing value distribution estimated from the original data, and the missing rate in this case is denoted by r^*. Brás *et al.* [10] stated that the missing entries produced by the second strategy is similar to the realistic missing pattern in original dataset.

In summary, we applied five missing rates (namely 5%, 10%, 15%, 20% and r^*) to each dataset, which consists of both MAR and MNAR pattern, resulting in datasets allowing for a comprehensive comparison of different imputation methods. Missing value generation was repeated 20 times for each dataset and each missing rate, yielding a total of 400 different datasets with missing values for comparison.

4.4.3 Performance measurements

We employed the normalized root mean squared error (NRMSE) as main metric by which we assessed the performance of proposed NPRA.

$$NRMSE = \sqrt{\frac{mean[(y_{guess} - y_{real})^2]}{Var(y_{real})}},$$

where y_{guess} and y_{real} are estimated values and actual values respectively. The mean and the variance were calculated over missing entries in the complete matrix. Moreover, we also evaluated the imputation accuracy of the methods using squared Pearson correlation coefficients between the complete and imputed data within each column.

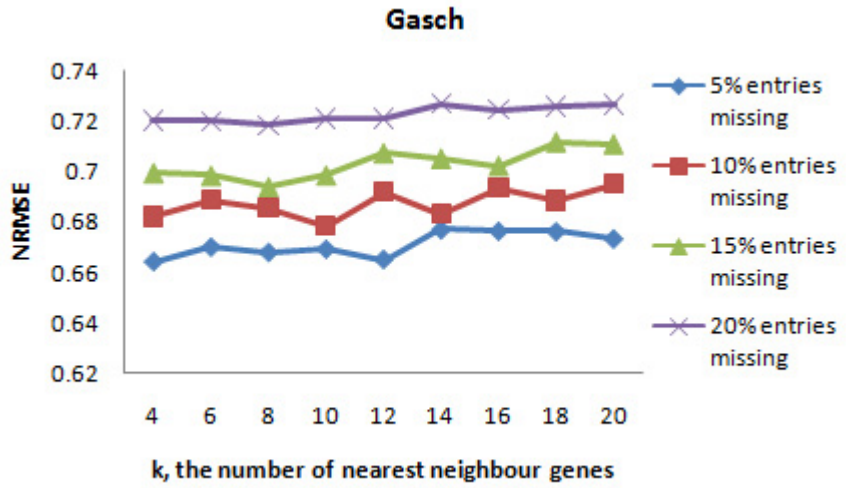

Figure 4.1: NRMSE over a number of nearest neighbours used for NPRA for different missing percentages on gasch data

4.5 Results and Discussion

In this section, we assessed the accuracy of each method under different conditions, such as type of data (time series, mixed, and non-time series), missing rate (5%, 10%, 15%, and 20%), and missing pattern (MAR and MNAR).

4.5.1 Choosing k in NPRA

For NPRA imputation, the number of nearest neighbour genes, k, must be selected. This parameter k, is dependent on the data type, but has no theoretical way to be appropriately determined. Arnone and Davidson [5] believe genes are influenced on average by no more than eight to ten other genes. It was found in [87] that KNNimpute is relatively insensitive to the exact value of k within the range of 10-20 neighbours and the best estimation results were observed in this range, based on a yeast cDNA array data. Also, $k = 10$ is suggested in LSimpute, whereas Sehgal *et al.* [75] believe k in the range of 10-15 is the most appropriate. For each dataset and percentage of missing data, we examined the performance of NPRA using $k = 4$ to 20 nearest neighbour genes for prediction.

For time series data, Figures 4.1 and 4.2 show that k in the range of 4-12 gives superior

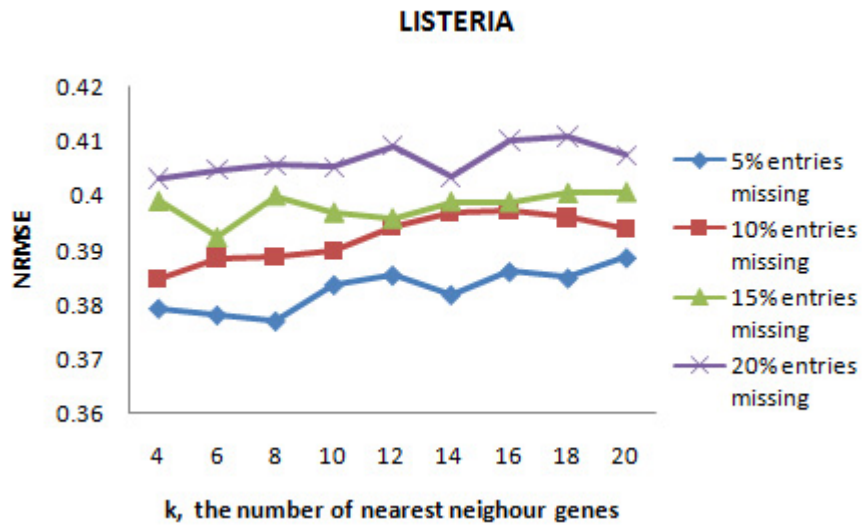

Figure 4.2: NRMSE over a number of nearest neighbours used for NPRA for different missing percentages on listeria data

accuracy for low missing percentage. When the missing rate increases, this optimal range will narrow. The curves in Figure 4.1 seem smoother than in Figure 4.2. Then we studied the influence of k on prediction ability by performing paired t-test (see Section 3.3.1) over the absolute residuals between real and actual values using different k on Listeria data. The comparative results were shown in Tables 4.2 and 4.3. Each table exhibits the results for two different missing percentages, separated by the diagonal. Each entry in Tables 4.2 and 4.3 corresponds to the p-value for the comparison with different values of k.

The p-value indicates whether the estimates by NPRA with different k are statistically equivalent or not. That is to say, the paired t-test is aimed at how the parameter value (k) affects NPRA's prediction ability. Results from Tables 4.2 and 4.3 suggest that k values of 4 and 6 almost give statistically equivalent performance for Listeria data, although difference in NRMSE between various values of k in the range of 4-12 is small.

In the case of mixed data, Figure 4.3 shows how the Calcineurin data responds to the changing value of k. The range k = 4-14 gives statistically equivalent prediction error for lower missing percentage (p-value>0.1865), however, this optimum range narrows to k = 8-12 for higher missing rates (p-value>0.2646). In the non-time series data of BC, there

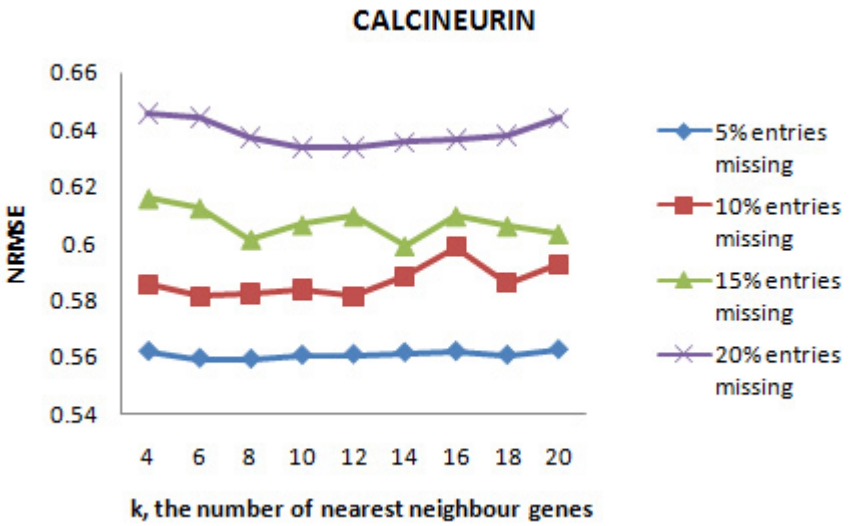

Figure 4.3: NRMSE over a number of nearest neighbours used for NPRA for different missing percentages on calcineurin data

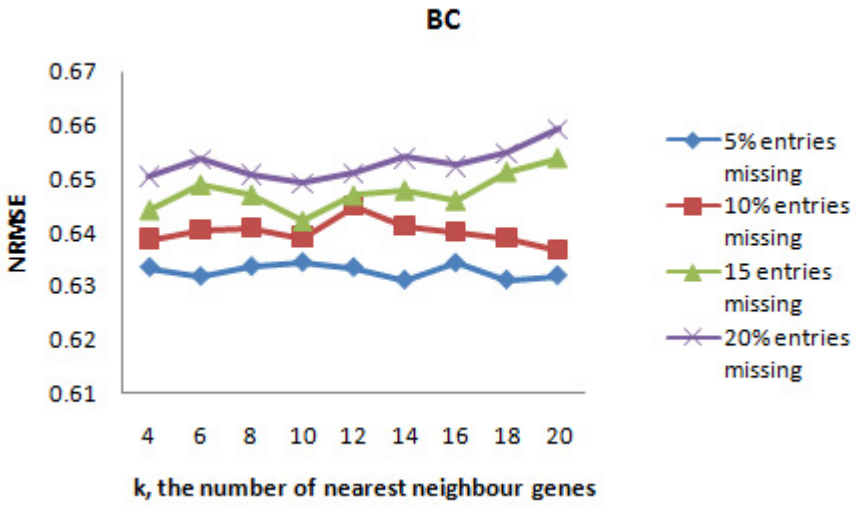

Figure 4.4: NRMSE over a number of nearest neighbours used for NPRA for different missing percentages on breast cancer data

Table 4.2: Probability level (p-value) of T-test based on residuals of 5% entries missing (above diagonal) and 10% entries missing (below diagonal) for listeria data, using different k

	$k=4$	$k=6$	$k=8$	$k=10$
$k=4$		1.71E-01	1.10E-01	3.09E-06
$k=6$	1.83E-01		3.22E-03	8.50E-05
$k=8$	2.99E-01	7.48E-03		6.14E-05
$k=10$	6.14E-06	2.33E-07	6.09E-06	

Table 4.3: Probability level (p-value) of T-test based on residuals of 15% entries missing (above diagonal) and 20% entries missing (below diagonal) for listeria data, using different k

	$k=4$	$k=6$	$k=8$	$k=10$
$k=4$		6.19E-01	5.10E-03	8.51E-03
$k=6$	1.53E-01		1.12E-03	1.23E-07
$k=8$	4.40E-01	1.52E-02		3.62E-06
$k=10$	7.22E-01	8.24E-05	5.57E-04	

is evidence of a better estimation performance when k falls in the range of 4-20 for lower missing percentage, as shown in Figure 4.4. When the missing percentage increases, the optimum range shrinks to k = 8-12 (p-value>0.2716).

In this section, we evaluated whether choosing the correct value for k has impact on the predictor error. The choice of k depends on different datasets, missing percentage, and types of microarray data. A high value of k results in the reduction of prediction accuracy due to incorporation of farthest genes, whereras a low value of k may ignore the similar genes, also leading to erroneous results. From the results of our study based on Figures 4.2-4.4, we suggest that $k = 8$ for NPRA be used for the subsequent studies.

4.5.2 Comparative studies with KNNimpute, LSimpute and LLSimpute

After selecting the optimum parameter values for NPRA, we compared the performance of NPRA, KNN, LSimpute and LLSimpute, where simulation studies were performed for

missing values from 5% to 20%.

Parameter setting for each method

Here, $k = 8$ was selected for NPRA following previous study, while $k = 10$ was used for the other three methods (KNNimpute, LSimpute and LLSimpute), since it has been strongly recommended by respective author [87, 8] except for LLSimpute. It has been pointed out that considering a sufficient number of genes is helpful in LLSimpute, so that the optimal value of k for LLSimpute can be large and up to hundreds of genes [47, 101]. However, LLSimpute shows a high peak when k is close to the number of samples, so we do not have to test the performance when k is close to the number of samples. Later on, we will test it in the case of large k.

Evaluation in terms of NRMSE and RNSE

For each data type, Tables 4.4-4.7 show the methods' NRMSE and RNSE for different percentages. Our proposed method NPRA shows good performance on all four datasets across different missing rates. For various percentages (5, 10, 15, and 20%) of missing entries, NPRA performs the best consistently except that in the case of Gasch data, NPRA is second best.

Table 4.4: Methods' prediction errors on listeria data over different missing rates

Method	5%		10%		15%		20%	
	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE
KNNimpute	0.4753	0.3740	0.4758	0.3744	0.4819	0.3819	0.4826	0.3838
LSimpute	0.4284	0.3268	0.4296	0.3294	0.4321	0.3320	0.4351	0.3325
LLSimpute	0.4301	0.2946	0.4445	0.3146	0.4587	0.3197	0.4665	0.3316
NPRA	0.3815	0.2821	0.3881	0.2972	0.3976	0.2992	0.3990	0.3065

As missing percentage increases, NRMSE and RNSE increase correspondingly for all methods. For mixed and non-time series data, LLSimpute is less robust to increasing missing rate than the other three methods, which is shown in Tables 4.6 and 4.7. However, for time series data, the robustness gives similar performance for all methods although

Table 4.5: Methods' prediction errors on gasch data over different missing rates

Method	5%		10%		15%		20%	
	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE
KNNimpute	0.9647	0.6911	0.9788	0.6814	0.9816	0.6969	0.9817	0.6911
LSimpute	0.7891	0.6366	0.7931	0.6249	0.8019	0.6241	0.7998	0.6282
LLSimpute	0.6421	0.5431	0.6594	0.5367	0.6771	0.5531	0.6838	0.5593
NPRA	0.6736	0.5753	0.6873	0.5676	0.7048	0.5729	0.7206	0.5859

Table 4.6: Methods' prediction errors on calcineurin data over different missing rates

Method	5%		10%		15%		20%	
	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE
KNNimpute	0.9179	0.6507	0.9180	0.6486	0.9189	0.6498	0.9401	0.6539
LSimpute	0.6248	0.5262	0.6365	0.5494	0.6629	0.5520	0.6693	0.5543
LLSimpute	0.5848	0.4859	0.6278	0.5314	0.6968	0.5609	0.7736	0.5980
NPRA	0.5406	0.4833	0.5794	0.5012	0.6085	0.5213	0.6379	0.5259

they show significant differences with respect to NRMSE and RNSE.

Evaluation in terms of bias analysis

We assessed the imputation methods further in terms of bias by means of Wilcoxon signed ranks test, reviewed in Section 3.3.1. The Wilcoxon Signed Ranks test does not require the assumption that the population is normally distributed, but does assume that the population probability distribution is symmetric. Wilcoxon signed ranks test is designed to test a hypothesis about the median of a population distribution. It often involves the use of matched pairs, for example, actual and imputed data, in which case it tests for a median difference of zero.

For time series data, KNNimpute and LSimpute generate more biased estimates than LLSimpute and NPRA, whereas for Calcineurin data, the results show that all methods are unbiased except for the case of 20% missing percentage. In the case of Breast Cancer data, the other three methods are less biased than KNNimpute which means that negative and positive residuals are not equally likely for KNNimpute.

Table 4.7: Methods' prediction errors on breast cancer data over different missing rates

Method	5%		10%		15%		20%	
	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE
KNNimpute	0.9660	0.6793	0.9612	0.6765	0.9644	0.6810	0.9630	0.6752
LSimpute	0.6535	0.4653	0.6575	0.4564	0.6568	0.4690	0.6653	0.4542
LLSimpute	0.8950	0.6041	0.9155	0.6151	0.9912	0.6426	1.0761	0.6755
NPRA	0.6344	0.4419	0.6363	0.4412	0.6406	0.4537	0.6524	0.4439

In conclusion, in terms of bias analysis, NPRA and LLSimpute are much more preferred than KNNimpute and LSimpute, although they also generate biased estimate occasionally when missing rate is high.

Evaluation in terms of squared correlation coefficients

Pearson correlation coefficients between the complete and imputed data within each column reflects how well the imputation method recovers the structure of the dataset. The higher the squared correlation coefficients is, the better the relationship between complete data and imputed data in a column. It is another measure to assess the imputation method besides NRMSE and RNSE.

Figures 4.5 and 4.6 depict the squared correlation coefficients between the actual and estimated values in each array of Listeria data across different missing percentages. For low missing rate, the curves show the methods' capability to retain similarity structure within each array. As missing rate increases, the performance of all imputation methods decreases more or less. KNNimpute presents the worst capacity for retaining the structure in each array by significant drop of squared correlation coefficients in two different missing percentages (5% and 20%).

For the other time series data gasch, LLSimpute and NPRA show the best performance, as shown in Figures 4.7 and 4.8. LSimpute depicts a very large fluctuation among different arrays, whereas KNNimpute shows a rather robust performance for different arrays. NPRA shows less accuracy than LLSimpute in terms of NRMSE and RNSE in

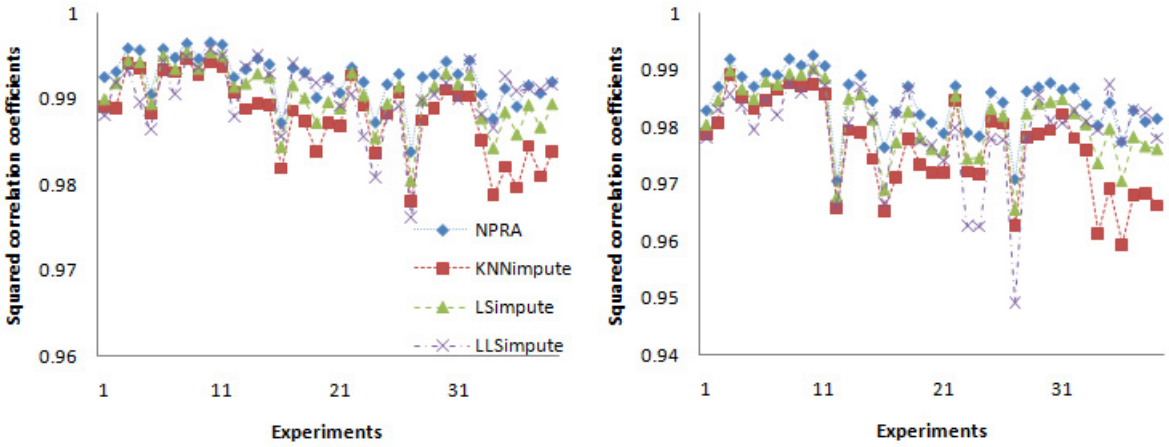

Figure 4.5: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for listeria with 5% (left) and 10% (right) artificial missing values

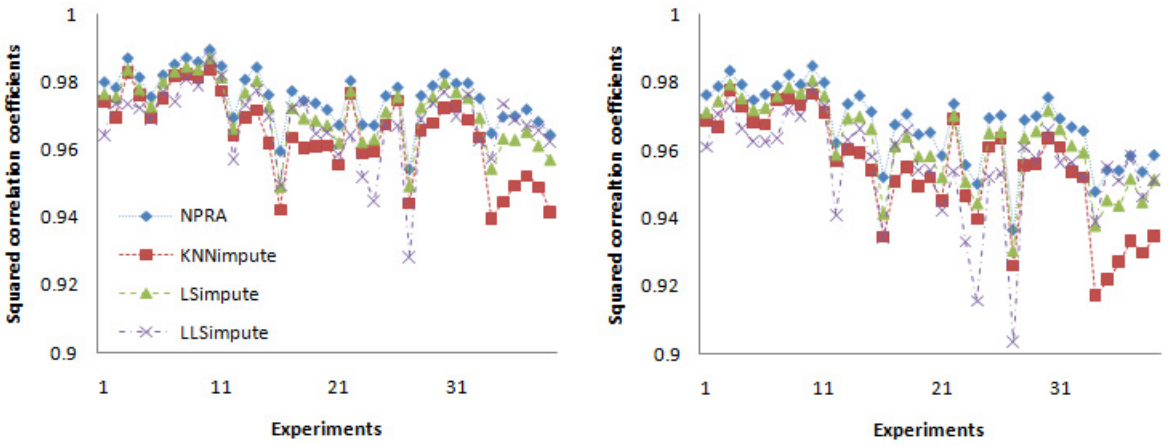

Figure 4.6: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for listeria with 15% (left) and 20% (right) artificial missing values

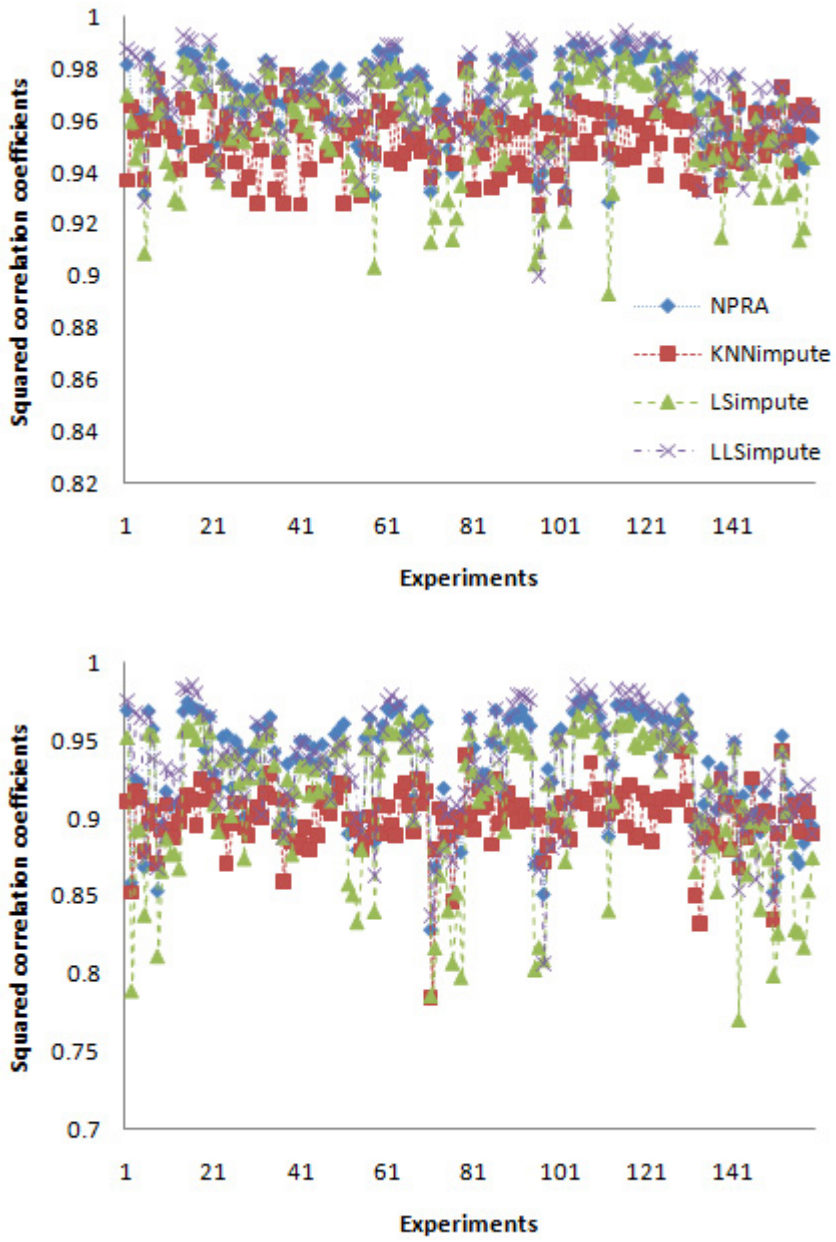

Figure 4.7: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for gasch with 5% (above) and 10% (bottom) artificial missing values

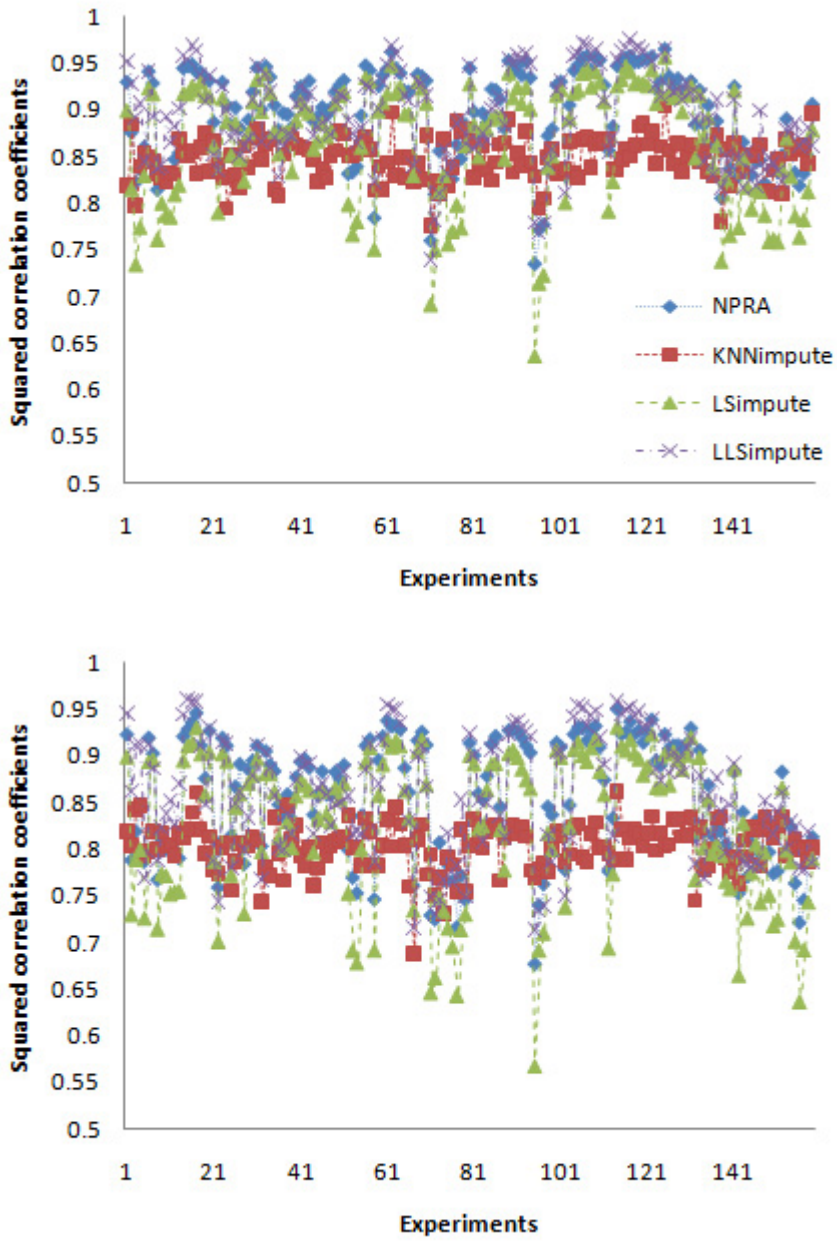

Figure 4.8: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for gasch with 15% (above) and 20% (bottom) artificial missing values

Table 4.5, but there is not much difference with respect to the ability of retaining the structure in each array.

LLSimpute is easily affected by data similarity structure. Thus, as the data similarity structure is weaker, such as for mixed and non-time series data, LLSimpute exhibits very poor ability to retain the data structure, showing a high variance among different arrays, especially for high missing rate in Figures 4.10 and 4.12. Although NPRA and LLSimpute are the two best imputation methods for mixed data in Table 4.6, NPRA shows a more consistent performance than LLSimpute across the different missing percentages. Therefore, it is desirable to assess imputation methods by means of different metrics.

In conclusion, regarding the results discussed so far from Figures 4.5 to 4.12, KNNimpute is not preferable for missing value estimation across all datasets. NPRA is the best option for time series and mixed data, regardless of the missing percentage, while LLSimpute is also applicable in low missing rate data. In the case of non-time series data, NPRA and LSimpute show very similar favorable results in Figures 4.11 and 4.12, but NPRA is preferred in terms of NRMSE. Squared correlation coefficients are aimed at offering another evaluation metric other than NRMSE, since NRMSE provides the overall assessment between imputed and complete data, while squared correlation coefficients describe the relationship between imputed and complete data in each array which seems more detailed. These two metrics are complementary and suggested in the evaluation of missing value imputation methods.

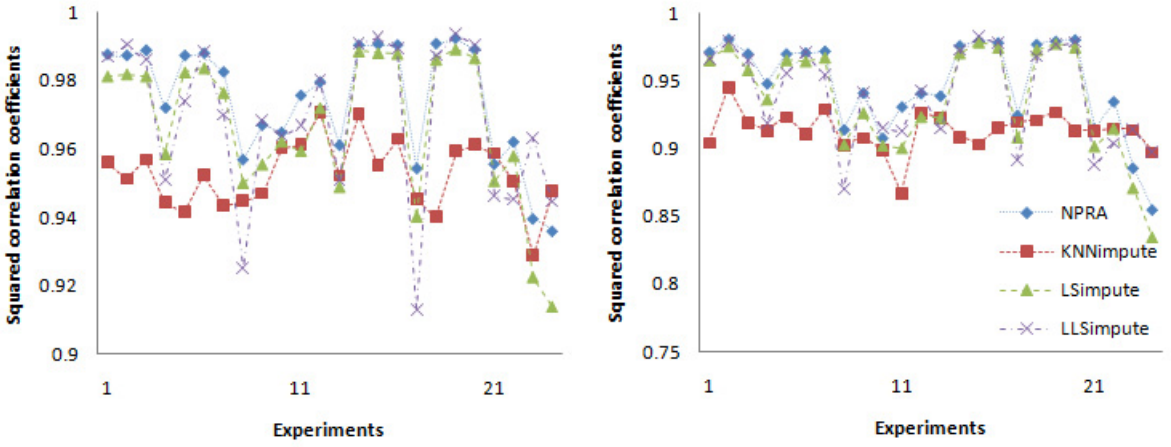

Figure 4.9: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for calcineurin with 5% (left) and 10% (right) artificial missing values

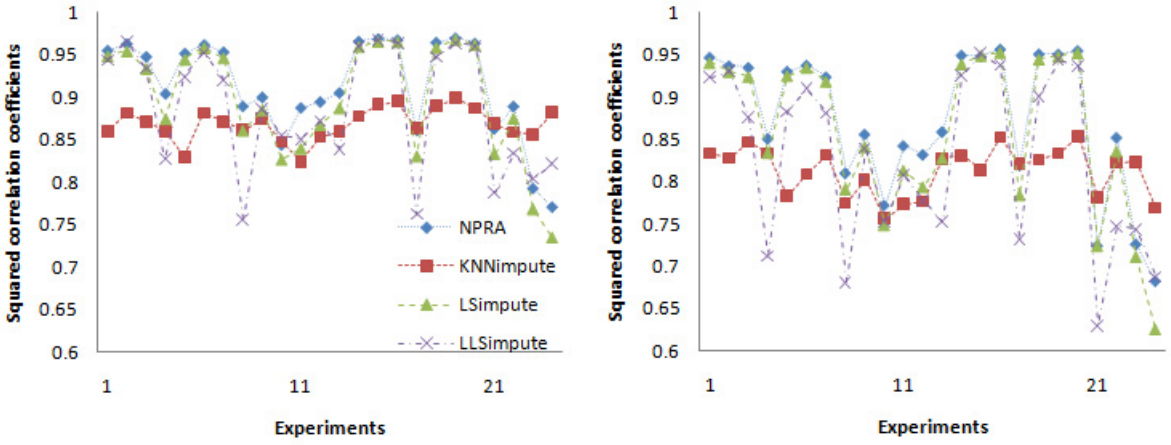

Figure 4.10: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for calcineurin with 15% (left) and 20% (right) artificial missing values

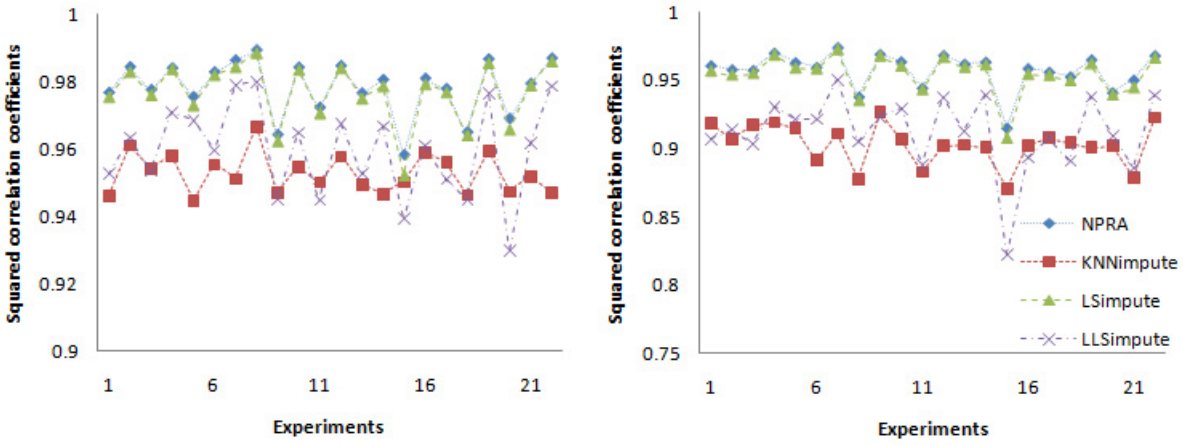

Figure 4.11: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for breast cancer with 5% (left) and 10% (right) artificial missing values

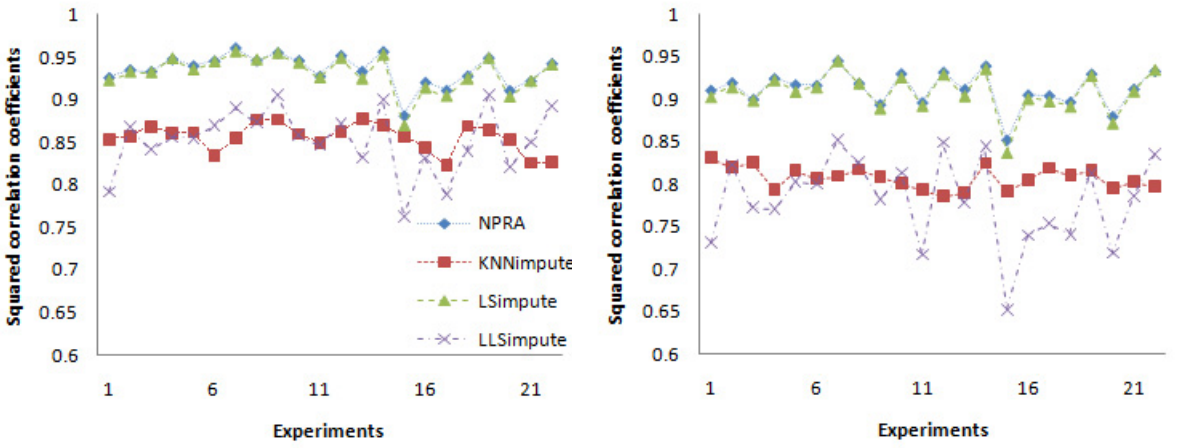

Figure 4.12: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data for breast cancer with 15% (left) and 20% (right) artificial missing values

4.5.3 Comparative studies on a realistic model of the missingness

There are various missing data mechanisms, including missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). We have just reviewed the performance of comparative imputation methods on MAR pattern. Here it is of interest to see how NPRA performs when the MAR assumption is violated. We are aimed at studying the influence of missing data pattern in the accuracy of imputation methods.

Three datasets Gasch, Listeria and Calcineurin were used for comparative study, and Breast Cancer data was excluded since there is no missing value in the original data. We employed the strategy introduced in Section 4.4.2 to generate a realistic model of missingness. The artificial missing rate as shown in Table 4.8, is very close to the real one in the original data (Table 4.1). We have simulated 20 independent studies for each dataset.

Table 4.8: Methods' prediction errors on MNAR pattern over three datasets

Data	Gasch		Listeria		Calcineurin	
Missing percentage (r^*)	3.21%		7.29%		3.28%	
	NRMSE	RNSE	NRMSE	RNSE	NRMSE	RNSE
KNNimpute	0.9751	0.6996	0.4882	0.3768	0.9382	0.6759
LSimpute	0.8016	0.6395	0.4507	0.3336	0.7098	0.5455
LLSimpute	0.6563	0.5360	0.4973	0.3406	0.8376	0.5881
NPRA	0.6916	0.5739	0.4216	0.3079	0.6392	0.5126

Overall, the missing value estimation accuracy is affected by the type of missing entries (MAR or MNAR), even if the missing percentage is the same by comparing the results in Table 4.8 and Tables 4.4-4.6. All imputation methods generate higher prediction error (Table 4.8) and lower squared correlation coefficients between actual and imputed value in each array (Figure 4.13) when applied to datasets with a more realistic missing pattern, even though the missing rate is similar to or less than that of datasets with randomly

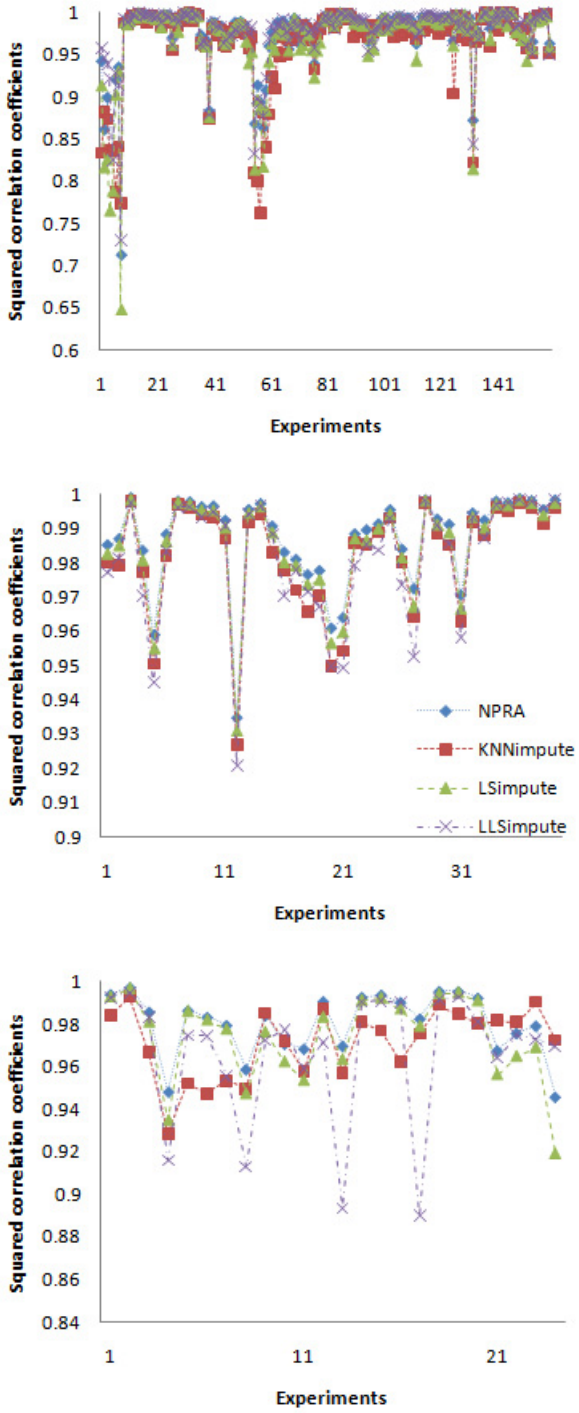

Figure 4.13: Comparison of the performance of KNNimpute, LSimpute, LLSimpute and NPRA by the squared correlation coefficients for each column between the complete and imputed data on MNAR pattern over three datasets: Gasch(top), Listeria(middle) and Calcineurin(bottom)

generated missing entries. Specifically, in some arrays, squared correlation coefficients decay very fast to as low as 0.65 because of the unequally distributed missing entries, however the squared correlation coefficients in Figure 4.7 (top) is above 0.9 for each array in the case of 5% randomly missing values.

KNNimpute is less capable to deal with the MNAR pattern in terms of NRMSE and RNSE, and shows a poor ability to reconstruct the data structure in terms of squared correlation coefficients. LLSimpute is not robust over the three datasets by being an unfavorable method twice and a favorable method only once in terms of accuracy. In addition, its less robustness is specially shown from the bottom one in Figure 4.13, where squared correlation coefficients of LLSimpute fluctuate up and down more sharply over a couple of experiments than other imputation methods.

In conclusion, the proposed NPRA exhibits its strong competitiveness as follows. Firstly, it performs most favorably in terms of accuracy, giving the lowest prediction error on Listeria and Calcineurin data and the second lowest on Gasch data. Secondly, it preserves the structure of data better than the other three imputation methods almost for all columns of time series datasets (top and middle in Figure 4.13). Especially for some arrays with high missing values, NPRA always performs the best over the other methods by having higher squared correlation coefficients. For mixed Calcineurin data, although in 5 out of 24 experiments, it does not give the highest squared correlation coefficients, overall it is still the most favorable in terms of both accuracy and squared correlation coefficients.

4.6 Summary

In this chapter, we proposed a nonparametric regression approach (NPRA) for imputation by taking advantage of gene-wise relationships, and made some comparative studies with KNNimpute, LSimpute and LLSimpute, demonstrating the influence of some factors,

such as the value of parameter k, type of experiment (TS, MIX and NTS), missing data pattern (MAR and MNAR), and missing percentage, in the accuracy of NPRA. The results show that missing rate and missing data pattern, and the type of experiment affect the data similarity structure, hence influencing the methods' performance. As missing rate increases, the methods' prediction ability decreases. Moreover, the prediction ability on a missing pattern close to that of original data is weaker than on a random missing pattern.

All the evaluation metrics used to assess the methods' performance gave consistent results. KNNimpute is unfavorable because it is less capable of taking advantage of data similarity structures, generating more biased estimates, higher prediction error, and showing a poor ability to recover the data structure. Novel NPRA method based on non-parametric regression combined with k-nearest neighbour, consistently exhibits superior accuracy compared to the other three imputation methods over four experimental datasets. Our study also shows that NPRA is a more robust and accurate missing value estimation method on MNAR pattern than LSimpute and LLSimpute.

NPRA makes full use of the gene-wise relationships for imputation. Relationships between genes are the type of relationships widely used as the basis for imputation. Even though NPRA shows good performance in our experiments, it also has some shortcomings.

Firstly, when the number of arrays is too small, it is not suitable, since non-parametric regression requires a higher number of samples to estimate the function than parametric regression in which the function is already known and just the parameters need to be estimated. Secondly, if we want to impute values for a gene with many missing values, for instance if a gene has 60% missing values, we do not recommend NPRA, since few arrays remain to determine how this gene is correlated with other genes in the dataset, thus most likely resulting in less accurate estimates. Thirdly, the Euclidean distance used by NPRA is sensitive to outlier values in microarray, although log-transforming the data

significantly reduces their influence in nearest neighbour gene determination.

Under such circumstances, utilizing how the arrays are correlated appears to be a good solution, since typically the expression profile for each array is available for thousands of genes. Relationships between arrays have already been explored in some work [9, 28, 8], though not used as widely as gene-wise relationships. Moreover, most of the imputation methods lack robustness in case outlying values are present in the microarray data. In the next chapter, a more robust missing value imputation method based on array-wise relationships will be proposed.

Chapter 5

Robust Principal Component Analysis Approach for Imputation Based on Array-wise Relationships

In previous chapter, you have seen how it works for imputation to use the gene-wise relationships due to the existence of some degree of association between genes. The focus of this chapter is to impute missing values by making use of array-wise relationships. The proposed RPCA imputation method employs robust regression to estimate the missing values by using the principal component arrays. In Section 5.2, we very briefly describe the precise relation between singular value decomposition analysis and principal component analysis. Further, we investigate the sensitivity of RPCA to initial estimates, and study the existence of an optimal number of principal components K_{pc} for RPCA in Section 5.3. Section 5.4 demonstrates our RPCA algorithm in detail and Section 5.5 exhibits a series of comparative studies with BPCA and LLSimpute.

5.1 Introduction

Gene-wise relationships have been successfully utilized for missing value imputation. However, information between arrays can provide useful information for imputation as well. For temporal gene expression data in which a given biological process is followed in time, array hybridisations from adjacent time points are highly correlated, hence we may expect some relations between arrays. Moreover, this kind of array relationships may also occur when biologically similar samples are used in different microarray experiments.

Chapter 5 intends to propose robust estimation for MVs with principal component analysis method. The main motivation was to take advantage of the correlation between arrays and reduce the effect of outliers in estimation of missing values. In our previous work, we have proposed LADimpute [14] which performs quantile regression between genes. Inspired by the idea of LADimpute, we are going to perform robust regression between target array and principal component arrays. This is aimed at capturing the global correlation structure between arrays, whereas Chapter 4 is dedicated to exploit the local correlation structure among genes.

5.1.1 Related work

There are some references which have brought important relevance to our idea. Bϕ *et al.* [8] present a method called LSimpute$_{array}$, to impute missing values from array correlations using a multiple regression model. Brás and Menezes [9] also explored the array-based relationships by applying KNNimpute, LLSimpute and PLSimpute in the array space where PLSimpute offers an advantage in terms of regression parameters' identifiability and increases the power of the estimation, since the number of genes is larger than the number of arrays.

Alter *et al.* [4] have visualized eigenarrays resulting from singular value decomposition analysis of cell-cycle data, inferring that an eigenarray that represents a state across all

genes. Gan *et al.* [28] employ this property to capture the global array-wise variation, in contrast to SVDimpute which captures the gene-wise correlation. This could be considered as an extension of SVDimpute from another perspective. For each eigenarray, there is a corresponding eigenvalue which indicates the relative significance of the eigenarray in terms of the fraction of the overall expression data. The decorrelation of the eigenarrays suggests the possibility that some of the eigenarrays represent independent cellular states [28].

Before illustrating proposed RPCA imputation method, we will interpret the PCA in the context of microarray. This interpretation and the accompanying nomenclature will serve as a foundation for understanding the RPCA method described later.

5.2 Principal Component Analysis

Singular value decomposition (SVD) and principal component analysis (PCA) are common techniques for analysis of multivariate data, especially for gene expression data [4, 53]. There is a direct relation between PCA and SVD in the case where principal components are calculated from the covariance matrix [93].

5.2.1 Mathematical definition of SVD

Recalling Section 2.3, microarray data is denoted by $m \times n$ matrix $\mathbf{X} = (x_{ij})$, which has rank r. Typically, m is the number of genes (rows) and n is the number of arrays (columns), where $m \gg n$ and thus $r \leq n$.

$$\mathbf{X} = \begin{bmatrix} \mathbf{g}_1^T \\ \vdots \\ \mathbf{g}_m^T \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 \cdots \mathbf{a}_n \end{bmatrix} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix} \in \mathbb{R}^{m \times n}$$

In the matrix $\mathbf{X}$, a row $\mathbf{g}_i^T \in \mathbb{R}^{1\times n}$ represents expression of the i^{th} gene in n experiments (where the superscript T denotes transpose). Alternatively, the m-dimensional vector $\mathbf{a}_j$ is referred to as the expression profile of the j^{th} array.

The equation for singular value decomposition of $\mathbf{X}$ is as follows:

$$\mathbf{X}_{m\times n} = \mathbf{U}_{m\times n}\mathbf{S}_{n\times n}\mathbf{V}^T_{n\times n}\,, \tag{5.1}$$

where the $m \times n$ matrix $\mathbf{U}$ and the $n \times n$ matrix $\mathbf{V}$ are orthogonal matrices, and $\mathbf{S}$ is an $n \times n$ diagonal matrix [93]. The columns of $\mathbf{U}$ are called the left singular vectors, $\mathbf{u}_k$, and form an orthonormal basis for the array expression profile, while the rows of $\mathbf{V}^T$ contain the elements of the right singular vectors,$\mathbf{v}_k$, and form an orthonormal basis for the gene expression profile. Thus, $\mathbf{u}_k$ is referred as eigenarrays and $\mathbf{v}_k$ is called eigengenes.

$\mathbf{S}$ is an $n \times n$ matrix that contains all zeros except for the diagonal $s_{i,i}, i = 1, \cdots, n$ which are called singular values, indicating the relative significance of the i^{th} eigengene or eigenarray in terms of the fraction of the overall expression they captured. Thus, $\mathbf{S} = diag(s_1, \cdots, s_n)$. Conventionally, the singular vectors are sorted by high-to-low singular values, with the highest singular value s_1 in the upper left position. Furthermore, $s_i > 0$ for $1 \leq i \leq r$, and $s_i = 0$ for $(r + 1) \leq i \leq n$. It is noticeable that in any real gene expression analysis application, $r = n$ due to the presence of noise, although the last singular values may be very close to 0.

5.2.2 Relation between PCA and SVD

Principal component analysis is a statistical technique for determining the key variables for simplifying complex datasets. If principal components are calculated from the covariance matrix [93], there is mathematical relation between SVD and PCA. We refer to an eigengene or eigenarray in (5.1) generically as a component in PCA, or, by analogy with SVD, as a singular vector.

A PCA analysis of DNA microarray data can consider the genes as variables or the experiments as variables or both [66]. When genes are variables, the right singular vectors $\mathbf{v}_k$ are the same as principal components of $\mathbf{g}_i$, which indicate the features of genes. When experiments are the variables, the left singular vectors $\mathbf{u}_k$ are the same as the principal components of $\mathbf{a}_j$, which indicate the features of the experimental conditions, showing whether the experiments are measuring different gene expression states or measuring similar states created by different mechanisms. In PCA for gene expression data, only K_{pc} most significant principal components are used [66] and the other $n - K_{pc}$ principal components are treated as noise and the signal-to-noise ratio (SNR) is described by the following equation:

$$p = \frac{\sum_{i=1}^{K_{pc}} s_i^2}{\sum_{i=K_{pc}+1}^{n} s_i^2} . \tag{5.2}$$

5.3 Quantile Regression with K_{pc} Principal Components

In this section, we introduce the imputation method that takes advantage of principal component analysis as well as robust regression. Rather than select k nearest genes for performing PCA, as in [101], we conduct PCA on the whole gene expression matrix to capture global correlation structure among arrays across all genes for imputation.

5.3.1 Initial values for PCA

Computations involved with PCA, as shown in (5.1), requires a complete dataset $\mathbf{X}$. However, this matrix contains missing entries so estimates from row average were used as the initial estimates to fill in $\mathbf{X}$. The sensitivity of RPCA method to the initial estimate will be studied in Section 5.5.2.

5.3.2 Robust regression

It is known that linear regression has been dominated by least squares (LS) estimate, mostly because of its solid theoretical foundation and ease of implementation. The assumption for LS estimate is that the model has normally distributed errors. However, linear least-squares estimates can behave badly when the error distribution is not normal, particularly when the errors are heavy-tailed. One solution is to remove outliers from the least-squares fit, while another approach, referred to as robust regression, is to employ a fitting criterion that is not as vulnerable as least squares to unusual data.

The most common general method of robust regression is M-estimation, introduced by Huber (1964) [38]. Consider the linear model

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \varepsilon_i \\ &= X_i'\beta + \varepsilon_i \,, \end{aligned}$$

for the i^{th} of n observations. The fitted model is

$$\begin{aligned} y_i &= b_0 + b_1 x_{i1} + b_2 x_{i2} + \cdots + b_k x_{ik} + e_i \\ &= X_i'\mathbf{b} + e_i \,. \end{aligned}$$

The general M-estimator minimizes the objective function [39]

$$\sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(y_i - X_i'\mathbf{b}) \,, \tag{5.3}$$

where the function ρ gives the contribution of each residual to the objective function.

The main purpose of robust regression is to provide resistant (robust) results in the presence of outliers. In order to achieve this stability, robust regression limits the influence of outliers, such as by assigning $\rho(e_i) = |e_i|$, rather than $\rho(e_i) = e_i^2$ for least-squares estimation. Currently, there are several robust regression methods, such as Tukey's bi-weight

M-estimator and quantile regression [63]. For RPCA, we employ quantile regression using 50^{th} percentile for simplicity. However, a further study focusing on the selection of a better robust method is desirable.

5.3.3 Single missing entry in an array

The array which has missing values is called target array. For simplicity, in this section we assume that the target array $\mathbf{a}_1$ only has one missing value in the first position, denoted by M, and the available values in target array $\mathbf{a}_1$ are denoted by $\mathbf{y}^T = (x_{21}, \cdots, x_{m1})$,

$$\mathbf{X} = \Big[\mathbf{a}_1, \cdots, \mathbf{a}_n\Big] = \begin{pmatrix} M & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix} \in \mathbb{R}^{m\times n}.$$

Based on (5.1), we construct array principal components. According to the relations between SVD and PCA, as mentioned previously, we obtain the array principal components as follows,

$$\begin{aligned} \mathbf{PC} &= \begin{pmatrix} PC_{11} & PC_{12} & \cdots & PC_{1n} \\ PC_{21} & PC_{22} & \cdots & PC_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ PC_{m1} & PC_{m2} & \cdots & PC_{mn} \end{pmatrix} \\ &= \begin{pmatrix} PC_{11} & PC_{12} & \cdots & PC_{1n} \\ \mathbf{PC}_{-1,1} & \mathbf{PC}_{-1,2} & \cdots & \mathbf{PC}_{-1,n} \end{pmatrix} \in \mathbb{R}^{m\times n}, \end{aligned}$$

where the vectors in principal components without the first elements, are denoted by $\{\mathbf{PC}_{-1,1}, \mathbf{PC}_{-1,2}, \cdots, \mathbf{PC}_{-1,n}\}$ and the first element of each principal component is denoted by $PC_{1,j}, j = 1, \cdots, n$.

After performing principal component analysis, K_{pc} principal components are selected to estimate MVs in target array by a regression model with the target array as the response variable and K_{pc} array principal components as predictors. The corresponding regression model is

$$\mathbf{y} = \sum_{i=1}^{K_{pc}} \mathbf{PC}_{-1,i}\beta_i + \varepsilon\,. \tag{5.4}$$

Then $50th$ percentile quantile regression is used to fit the regression model in a robust manner. Robust regression usually provides an alternative analysis to least square regression. The assumption for LS estimate is that the model has normally distributed errors. When this fundamental assumption is violated, the need for robust estimate emerges.

The $50th$ percentile quantile regression is considered conceptually the simplest one, though not necessarily the best, since it does not require a tuning mechanism like most of the other robust regression procedures. This kind of quantile regression estimates β_i by minimizing the sum of absolute values of the residuals as follows,

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=2}^{m} |x_{i1} - \sum_{j=1}^{K_{pc}} PC_{i,j}\beta_j|\,. \tag{5.5}$$

Then, the missing value in $\mathbf{a}_1$ can be estimated by

$$M = \sum_{i=1}^{K_{pc}} PC_{1,i}\hat{\beta}_i = PC_{1,1}\hat{\beta}_1 + PC_{1,2}\hat{\beta}_2 + \cdots + PC_{1,K_{pc}}\hat{\beta}_{K_{pc}}\,.$$

5.3.4 Multiple missing entries in an array

Without loss of generality, in this section we assume that the target array $\mathbf{a}_1$ has two missing values in the first and the last position, denoted by M_1 and M_2 as follows,

$$\mathbf{X} = \Big[\mathbf{a}_1, \cdots, \mathbf{a}_n\Big] = \begin{pmatrix} M_1 & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m-1,1} & x_{m-1,2} & \cdots & x_{m-1,n} \\ M_2 & x_{m2} & \cdots & x_{mn} \end{pmatrix} \in \mathbb{R}^{m\times n},$$

where the observable values in target array $\mathbf{a}_1$ are denoted by $\mathbf{y}^T = (x_{21}, \cdots, x_{m-1,1})$.

Note that after performing principal component analysis on $\mathbf{X}$, we have

$$\{\mathbf{PC}_1, \mathbf{PC}_2, \cdots, \mathbf{PC}_n\} = \begin{pmatrix} PC_{11} & PC_{12} & \cdots & PC_{1n} \\ PC_{21} & PC_{22} & \cdots & PC_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ PC_{m-1,1} & PC_{m-1,2} & \cdots & PC_{m-1,n} \\ PC_{m1} & PC_{m2} & \cdots & PC_{mn} \end{pmatrix},$$

where from the second to last second element of each principal component are denoted by vectors $\mathbf{PC}_{2:m-1,j}, j = 1, \cdots, n$.

The known elements of $\mathbf{a}_1$ can be represented by

$$\mathbf{y} = \sum_{i=1}^{K_{pc}} \mathbf{PC}_{2:m-1,i}\beta_i + \varepsilon\,, \tag{5.6}$$

where K_{pc} is the number of selected principal components, and β_i are the coefficients of

the linear combination, found from $50th$ percentile quantile regression, as shown in (5.7).

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=2}^{m-1} |x_{i1} - \sum_{j=1}^{K_{pc}} PC_{i,j}\beta_j| \,. \tag{5.7}$$

Accordingly, the missing values in $\mathbf{a}_1$ can be estimated by

$$M_1 = \sum_{i=1}^{K_{pc}} PC_{1,i}\hat{\beta}_i = PC_{1,1}\hat{\beta}_1 + PC_{1,2}\hat{\beta}_2 + \cdots + PC_{1,K_{pc}}\hat{\beta}_{K_{pc}} \,,$$

and

$$M_2 = \sum_{i=1}^{K_{pc}} PC_{m,i}\hat{\beta}_i = PC_{m,1}\hat{\beta}_1 + PC_{m,2}\hat{\beta}_2 + \cdots + PC_{m,K_{pc}}\hat{\beta}_{K_{pc}} \,,$$

where M_1 and M_2 are the missing values in the target array.

5.4 RPCA Algorithm

The previous section discussed both single and multiple missing entries in a target array. RPCA makes use of K_{pc} principal components to build a regression model and utilizes an expectation maximization method to arrive at the final estimate. Given a gene expression matrix $\mathbf{X}$ with N missing entries, the outline for RPCA algorithm is given below:

Step 0

Initialisation: replace the MVs in $\mathbf{X}$ by the estimate given by row average, obtaining $\mathbf{X}^{complete(0)}$.

Step 1

Using $\mathbf{X}^{complete(0)}$, perform principal component analysis and select K_{pc} array components as predictor variables, then regress each target array against K_{pc} array components, in order to obtain a vector of estimates, $\mathbf{y}^{(1)}_{guess}$, and update the complete gene expression matrix into $\mathbf{X}^{complete(1)}$.

Step 2

For each l^{th} cycle ($l = 2, \ldots$):

(a) Using $\mathbf{X}^{complete(l-1)}$, re-estimate the missing entries, by performing PCA and conducting robust regression for each target array against new K_{pc} array components.

(b) Obtain a vector of new estimates $\mathbf{y}_{guess}^{(l)}$, and update the complete expression matrix into $\mathbf{X}^{complete(l)}$.

(c) Calculate $\delta^{(l)} = \sum_{i=1}^{N} (\mathbf{y}_{guess}^{(l)}[i] - \mathbf{y}_{guess}^{(l-1)}[i])^2$, the sum of squared differences between the MV estimates.

Step 3

If $\delta^{(l)} < \tau$, stop. Otherwise return to Step 2 and iterate until convergence.

The convergence criterion was set to $\tau = 10^{-10}$. If it could not be reached, we set the maximal number of iterations to be 50.

5.5 Results and Discussion

In this section, the accuracy of RPCA imputation was evaluated for each of the missing rates $5\%, 10\%, 15\%$, and 20% on pattern of missing at random. We include the same four datasets from Chapter 4 in this study.

5.5.1 Effect of K_{pc} on RPCA

The use of RPCA imputation requires the selection of the number of principal components. RPCA imputation is repeated for a sequence of values for the parameter K_{pc} for each dataset, in order to provide general guidelines for the selection of K_{pc} in practice. The results show that in many cases there is no particular choice of K_{pc} which gives consistent accuracy, since the choice of K_{pc} is much dependent on each dataset. Thus, implementing the evaluation design to find K_{pc} is preferable.

We examined different values of K_{pc} for each dataset, since the number of arrays is different. Note that the maximal K_{pc} is equal to the number of arrays, and depending on the singular values in (5.2), only a small number of principal components are most significant. As shown in Tables 5.1-5.4, the optimal K_{pc} for each data set is totally different.

Time series data Table 5.1 presents that for time series data Listeria, NRMSE has the lowest value when $K_{pc} = 8$, although it has a robust performance on the range between 6 and 9. The accuracy deteriorates rapidly outside this range and NRMSE is unacceptably high. For the other time series data Gasch, we test the value of K_{pc} from 8 up to 100, because Gasch has 159 columns (arrays) in total. In general, there is evidence of a better accuracy from Table 5.2 when $K_{pc} = 16$ across different missing rates.

Table 5.1: NRMSE of different numbers of principal components used for RPCA on listeria data with different missing percentages

K_{pc}	2	4	6	7	8	9	10	12	15	20
5%	0.3888	0.3402	0.3049	0.2886	0.2859	0.2999	0.3133	0.3261	0.3576	0.4462
10%	0.3761	0.3320	0.2976	0.2927	0.2882	0.3039	0.3085	0.3252	0.3856	0.4468
15%	0.3778	0.3424	0.3011	0.2989	0.2903	0.3301	0.3118	0.3329	0.3865	0.4546
20%	0.3769	0.3395	0.3060	0.2986	0.2925	0.3099	0.3313	0.3391	0.3907	0.4605

Table 5.2: NRMSE of different numbers of principal components used for RPCA on gasch data with different missing percentages

K_{pc}	8	10	16	18	20	30	50	60	80	100
5%	0.5211	0.4991	0.4694	0.4689	0.4893	0.5252	0.6016	0.6472	0.7758	0.8959
10%	0.5212	0.4954	0.4854	0.4946	0.5061	0.5403	0.6129	0.6854	0.8004	0.8832
15%	0.5286	0.5054	0.4894	0.4856	0.5265	0.5606	0.6235	0.6933	0.8007	0.9004
20%	0.5268	0.5170	0.4911	0.5172	0.5398	0.5563	0.6714	0.7582	0.8505	0.9243

Mixed data Not only does the optimal K_{pc} depend on the specific dataset, but it is also affected by different missing percentages in the same dataset. Table 5.3 gives a good illustration about that. In the case of 5% missing entries, the optimal range for K_{pc} is

from 8 to 10, which shows the lowest NRMSE. As the number of missing entries increases, the optimal range changes to between 6 and 9 for 20% missing percentage. Thus, it is desirable to implement some evaluation design to choose the best K_{pc}.

Table 5.3: NRMSE of different numbers of principal components used for RPCA on calcineurin data with different missing percentages

K_{pc}	2	4	6	7	8	9	10	11	12	15
5%	0.5206	0.4197	0.3798	0.3729	0.3576	0.3543	0.3540	0.3605	0.3791	0.4794
10%	0.5337	0.4293	0.3900	0.3960	0.3838	0.3776	0.3773	0.4108	0.4212	0.4727
15%	0.5330	0.4375	0.4130	0.3935	0.4000	0.3968	0.4011	0.4236	0.4511	0.5112
20%	0.5331	0.4326	0.4167	0.4182	0.4155	0.4119	0.4349	0.4349	0.4983	0.5544

Non-time series data We also found, not surprisingly, that some knowledge about the characteristics of data is beneficial in choosing the number of principal components. The number of components that results from PCA is sometimes associated with the number of underlying biological processes that give rise to the patterns in the data [93].

The application of PCA allows us to summarize the ways in which gene responses vary under different conditions, and simplify the analysis. Thus, it is of interest to describe biological meaning to the significant eigenarrays. For Breast Cancer example data [35], it is known to contain three types of samples, BRCA1, BRCA2 mutation, and sporadic. Making full use of biological information proves to be very helpful when choosing the optimal K_{pc}, as shown in Table 5.4, where $K_{pc} = 3$ performs best overall, in terms of NRMSE accuracy. Table 5.4 also exhibits that with more than 6 principal components, the prediction accuracy is quite low and it is far from satisfactory when K_{pc} is larger than 12.

To summarize, the performance of RPCA imputation depends on the value of K_{pc}. By varying the value of K_{pc}, we studied its influence in NRMSE accuracy. The results show that the optimal K_{pc} for RPCA imputation differs among different datasets and depends on missing rate as well. Thus, we would not suggest the optimal value of K_{pc} for practical

Table 5.4: NRMSE of different numbers of principal components used for RPCA on breast cancer data with different missing percentages

K_{pc}	2	3	4	5	6	8	10	12	15	20
5%	0.6072	0.5965	0.6042	0.6022	0.6231	0.6769	0.7516	0.8794	1.0873	1.2232
10%	0.6236	0.6123	0.6223	0.6309	0.6570	0.6965	0.7782	0.8911	1.1499	1.1116
15%	0.6273	0.6246	0.6380	0.6611	0.6437	0.7431	0.8077	0.9673	1.1779	1.0149
20%	0.6372	0.6273	0.6505	0.6491	0.6749	0.7507	0.8580	0.9735	1.1720	1.1720

application, although BPCA [59] recommends an optimal parameter which is equal to the number of arrays minus 1. Sometimes BPCA incurs a very heavy computational burden and loses its competitiveness (as shown in Table 5.8), due to its repetitive algorithm.

Moreover, the optimal value of K_{pc} may be different from array to array since the RPCA imputation procedure is executed for each target array. Choosing an optimal value for each target array requires intensive computation. It is recommendable to implement evaluation design to find optimal K_{pc} and take advantage of the external information about arrays.

5.5.2 Sensitivity of RPCA to initial values

As demonstrated in Section 5.4, RPCA imputation method requires a complete expression matrix for applying principal component analysis. So estimates from row average were used as initial estimates for the RPCA algorithm. In this section, the sensitivity of RPCA imputation to the initial estimate was assessed. To see how RPCA would perform with poor initial estimates, RPCA imputation was run with initial estimates from row average and KNNimpute respectively for Listeria, Calcineurin and Breast Cancer datasets with 5% missing values.

From Table 5.5, we find that RPCA prediction error using row average as initial estimates is quite similar to that using KNNimpute. Thus, the performance of RPCA imputation is not affected by initial estimates, in the sense that it is not necessary to give accurate initial estimates for RPCA. Just a simple method for initial estimate, such

Table 5.5: Sensitivity of RPCA imputation method to initial estimates. Given are NRMSE and RNSE of RPCA with initial estimates from row average and KNNimpute respectively over different datasets with 5% missing rate.

Initial estimate	Row average		KNNimpute	
	NRMSE	RNSE	NRMSE	RNSE
Listeria	0.285247	0.195489	0.285248	0.195499
Calcineurin	0.374854	0.319405	0.375857	0.319358
Breast cancer	0.623974	0.428017	0.623973	0.428019

as row average, is good enough.

5.5.3 Comparative study with BPCA and LLSimpute

In this subsection, we compared RPCA imputation method with BPCA and LLSimpute to see whether RPCA represents an improvement over previously proposed methods, since BPCA and LLSimpute are the most recognized methods by the community. We conducted comparative studies from several perspectives, such as accuracy, variation, and computation time, in order to get a comprehensive understanding of the performance of the proposed RPCA imputation method.

Evaluation in terms of accuracy

The performance of RPCA imputation method is evaluated in Figures 5.1-5.2 for all types of datasets (Time series, Mixed, and Non-time Series). In each of the datasets and each level of missing rate, 5 simulations are conducted with independent selections of artificial missing values. For RPCA, the number of principal components was chosen to be the one that exhibited the best performance in each test. For LLSimpute, Kim *et al.* [47] showed it performed well for a large value of k, say over 100. Thus we used 100 for k in LLSimpute to test its performance. Overall, we can see that for various percentages of missing entries, RPCA showed the least NRMSE consistently for time-series and non-time series datasets.

Figure 5.1 also shows that the performances of RPCA and BPCA are better than

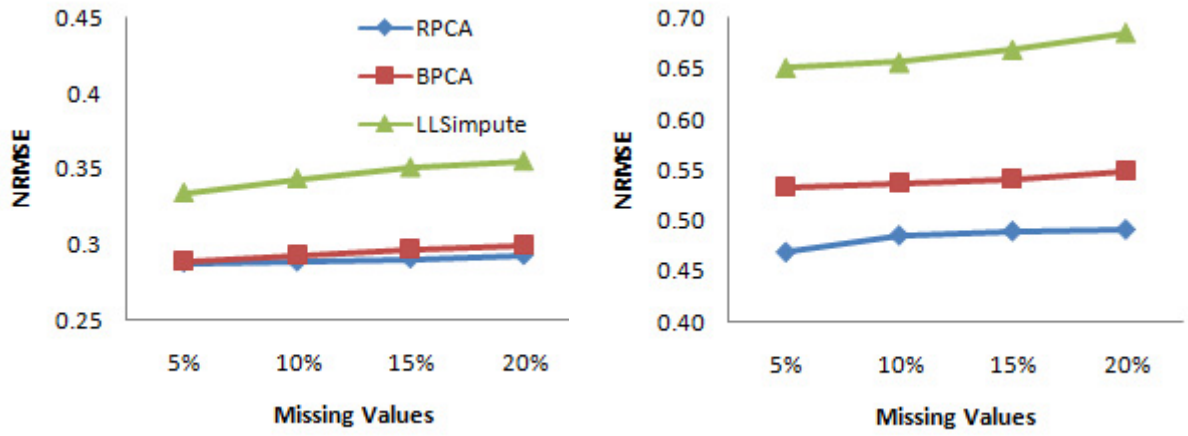

Figure 5.1: Comparison of the NRMSEs against percentage of missing entries for three methods (LLSimpute, BPCA and RPCA) on Listeria (left) and Gasch (right) data

that of LLSimpute over a whole range of missing rates. The range of NRMSE by RPCA imputation, for example, was 0.4694 to 0.4911 in comparison with 0.6511 to 0.6856 of the LLSimpute method in large Gasch data. The poor performance of LLSimpute is probably because of multi-collinearity of the selected k genes. In Listeria data, RPCA and BPCA show similar results and the best performance, although RPCA shows a slightly better performance for high missing rate. This may be due to the strong global similarity structure in time series data, which is explored by RPCA and BPCA, although they adopt different strategies. RPCA takes advantage of global correlation structure between arrays, whereas BPCA makes use of that between genes.

A further illustration of the improved accuracy of RPCA is given for Breast Cancer data in Figure 5.2. We found that RPCA has the least NRMSE, while BPCA performs worst for non-time series data. BPCA method is strongly influenced by data similarity, and in non-time series data, the proportion of low similarity genes is higher than in time series data, which can result in the bad performance of BPCA. Although RPCA's prediction ability decreases for increasing levels of missing rate, it outperforms BPCA and LLSimpute, since it takes robust regression into consideration. Very interestingly, we found some outliers in the Breast Cancer raw data set after careful observation. This

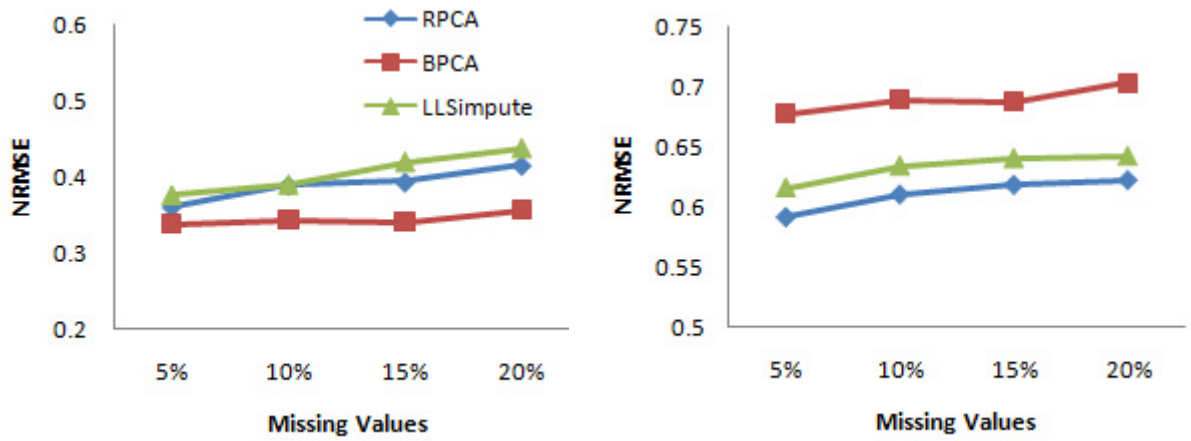

Figure 5.2: Comparison of the NRMSEs against percentage of missing entries for three methods (LLSimpute, BPCA and RPCA) on Calcineurin (left) and Breast Cancer (right) data

fact also suggests that employing robust regression is advisable, since it reduces the effect of outliers in estimation of missing values. RPCA imputation benefits a lot from quantile regression when dealing with outliers.

For mixed data, Figure 5.2 clearly shows that BPCA and RPCA are more favourable than LLSimpute, although BPCA is better than RPCA, in terms of NRMSE accuracy. As the missing rate increases, NRMSE increases for all methods and the performances become worse.

In conclusion, RPCA shows very high competitiveness in terms of NRMSE accuracy for time series and non-time series data, whereas BPCA gives best performance for mixed data. Moreover, RPCA does very well in imputing missing values in datasets with outliers, such as cancer microarray data. Although $\log_2$ transformation of the gene expression values may reduce the influence of outliers, it is still advisable to employ robust regression.

Imputation methods' performance on noisy data

Even on the high density oligonucleotide arrays such as Affymetrix GeneChip oligonucleotide (Affy) arrays, a significant percentage of probes could be blemished. Hence, in

Table 5.6: The proportion of total variance explained by the first and second components for different datasets

Data	$\mathbf{PC}_1$	$\mathbf{PC}_2$
Listeria	81.56%	5.46%
Gasch	53.16%	9.10%
Calcineurin	64.71%	11.43%
BC	62.42%	6.38%

gene expression values, a high percentage of them may be noisy [100]. When microarray data are noisy, principal component analysis can still detect the underlying patterns. This ability of the PCA distinguishes itself among the commonly used gene expression analysis methods. The square of each singular value s_i in (5.1) is proportional to the variance explained by each principal component. The relative variance of each principal component is often calculated by

$$p_k = \frac{s_k^2}{\sum_i s_i^2} \,. \tag{5.8}$$

Table 5.6 provides the relative variance explained by the first and second components. We decided to take Listeria data as an example, since the first component in Listeria has the greatest relative variance among four datasets, which indicates the highest correlation structure between arrays and suggests that Listeria has the lowest noisy level in some sense. In the following, we evaluate the performance of RPCA with respect to noisy data, and exhibit how the results of each imputation method respond to various noise levels.

In order to demonstrate the performances of RPCA and BPCA on noisy data, I prepared the datasets based on Listeria data by adding random noise of various levels, with normal distribution. After building matrices of random numbers with the normal distribution of mean $\mu = 0$ and various standard deviations ($\sigma = 0.01, 0.05, 0.1, 0.15, 0.2$ and 0.25), each noise matrix was added to the Listeria data with 5% artificial missing rate in order to build the six noisy datasets.

Figure 5.3 shows that the performance of RPCA was less sensitive to the noise level

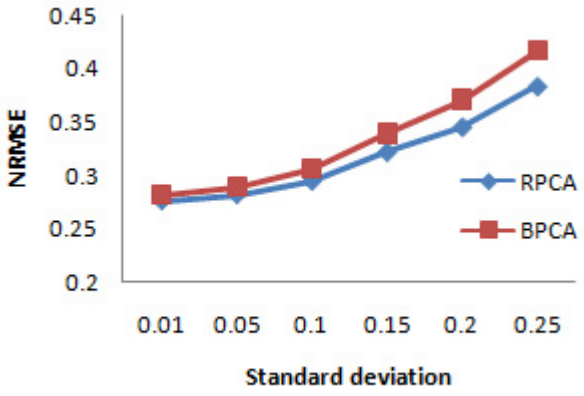

Figure 5.3: Comparison of the NRMSEs with respect to noise levels. We added artificial noise with normal distribution of mean $\mu = 0$ and various standard deviations ($\sigma = 0.01, 0.05, 0.1, 0.15, 0.2$ and 0.25) to Listeria dataset

than BPCA. Although both of them have higher prediction error as the data becomes more noisy, RPCA method outperforms BPCA consistently for various noise levels. This fact can be attributed to the robust regression employed in RPCA.

Variation of imputation methods' performance

To assess the variance in prediction error of each method, we conducted all imputation methods 10 times for each percentage of missing entries. Table 5.7 gives the average and variance of NRMSE from 10 runs of simulation in breast cancer example data. In addition to having the smallest NRMSE for each missing rate, RPCA consistently outperforms BPCA, in terms of variance. Although there is no significant difference between RPCA and LLSimpute, regarding variance, the prediction accuracy of RPCA surpasses that of LLSimpute.

Computational Efficiency

As mentioned previously in Section 3.3.1, computational cost is also an evaluation metric, though better estimation accuracy always has a greater priority than speed. However, we cannot avoid the trade-off between the computational burden and the accuracy, when the time cost of the imputation algorithm appears to be unacceptable for some datasets.

Table 5.7: Variation of prediction error for breast cancer data over a range of missing rates. Given are the averages and variances of NRMSE for RPCA, BPCA and LLSimpute method.

	RPCA		BPCA		LLSimpute	
	Mean	Variance	Mean	Variance	Mean	Variance
5%	0.6018	1.54E-04	0.6761	2.43E-04	0.6216	2.78E-04
10%	0.6190	5.17E-05	0.6862	3.76E-04	0.6338	8.07E-07
15%	0.6163	1.01E-05	0.6848	9.31E-05	0.6374	4.09E-05
20%	0.6281	2.27E-05	0.6928	3.91E-04	0.6422	3.55E-07

Table 5.8 exhibits the running time of the three imputation methods in Gasch example dataset for the 5% missing rate. It is not surprising that BPCA has the worst performance, with running time up to dozens of hours. Although that is not always the case for BPCA, it is advisable to balance the computational efficiency and accuracy when imputing missing values. One possible reason for BPCA's bad performance is that its convergence is slow due to its EM-like repetitive algorithm. The other reason might be that the number of arrays in Gasch dataset is quite large, up to 159. Although BPCA claims better estimation performance when the number of the samples is large [59], computation efficiency is still an issue worth paying attention to.

The difference between RPCA and BPCA is that RPCA imputation method effectively controls the computational burden. Applying maximum number of iterations (50) affords the advantage of reducing the computational time even when δ cannot reach $\tau = 10^{-10}$. Very interestingly, we found that larger number of iterations does not necessarily ensure more accurate estimates for RPCA in some cases. For instance, we tried to

Table 5.8: The running times were calculated for Gasch dataset with 5% missing rate. (Intel Pentium 4 CPU 2.80GHz with 1GB RAM was used).

Imputation method	Implementation	Running time
RPCA	R	224s
LLSimpute	Matlab	356s
BPCA	Matlab	40h

increase the number of iterations from 50 to 500 (results not shown), but the NRMSE does not become smaller. In addition, RPCA seems less computationally intensive than LLSimpute from Table 5.8. However, a further study would be desirable for a systematic comparison.

5.6 Summary

In this chapter, we introduced the robust principal component analysis imputation method, referred to as RPCA. In addition to taking advantage of array-wise relationships, RPCA employs an iterative procedure for imputation. The iterative use of imputed data in RPCA allows a refinement of the predictions and improves the missing value estimates. We assessed the prediction performance of RPCA and compared it with BPCA and LLSimpute over different types of datasets with different missing rates.

In general, RPCA showed a competitive performance for time series and non-time series data. Furthermore, we evaluates its performance on noisy data, concluding that it has a more robust prediction accuracy than BPCA. RPCA also reduced the effect of outliers in missing value estimation. This is due to the fact that RPCA utilizes quantile regression, and PCA can capture the underlying array expression patterns even when the data are noisy.

From the datasets investigated, we found that the performance of RPCA depends on the number of selected principal components K_{pc}, whereas it is not affected by the initial estimates. Therefore, heuristic approaches for deciding on the significant components are desirable. Moreover, so far imputation methods have been evaluated by prediction accuracy. It is of more interest to study how the imputation method affects downstream analysis of microarray data. The next chapter details this further.

Chapter 6

Missing Value Imputation Framework and Impact on Subsequent Analysis

The purpose of this chapter is twofold. Firstly, a missing value imputation framework is constructed by presenting the CMBimpute method, which takes advantage of both gene-wise and array-wise relationships. In Section 6.2, heuristic methods for determining the parameters are proposed. Secondly, we study the impact of missing value imputation on downstream data analyses. Considering the fact that missing value imputation is a pre-processing step for microarray data mining, among which one of the most important applications is gene clustering, we focus on the effect of imputation on clustering analysis, since gene clustering is capable of discovering coherent patterns of gene expression. Section 6.3 employs k-means clustering method to achieve this goal. Section 6.4 presents a rigorous analysis on five imputation methods and Section 6.5 summarizes the whole chapter.

6.1 Introduction

In previous chapters, imputation quality has been measured by normalized root mean squared errors (NRMSE). In addition to NRMSE, it is of more interest to evaluate missing value imputation method in terms of the impact on subsequent biological analysis, such as gene clustering [21, 89], sample classification [1, 17, 57] and significant gene selection [74, 77]. Since the imputated values themselves are not interesting, whether or not the imputed data can affect the subsequent applications is the major concern.

6.1.1 Related work

Classification

It has been suggested that one may use the classification accuracy [94, 100] to measure the quality of estimated values in the case of disease subtype recognition and disease treatment classification, in addition to NRMSE. Wang *et al.* studied the impact of imputation on disease classification and observed that while replacing MVs with zero performs relatively poor, the advanced imputation methods, i.e. KNNimpute, LLSimpute and BPCA, have little difference in affecting the classification performance of the SVM or KNN classifiers [94]. Shi *et al.* [100] also studied the influence of imputation in classification accuracy, and found that imputed data by BPCA and iLLS could reach the same classification accuracy as that achieved on the complete data.

Differentially Expressed Gene Detection

Jörnsten *et al.* [43] examined the effect of imputation on the significance analysis of differential expression, and concluded that good imputation alleviated the impact of missing values. From the simulations, they discovered that missing values affect the detection of differentially expressed genes, and that more sophisticated imputation methods, such as BPCA and LinCmb are much better than row average and KNNimpute.

Scheel *et al.* [69] also studied the influence of missing value imputation on detection of differentially expressed genes from microarray data. The results showed that KNNimpute can lead to a greater loss of differentially expressed genes than LinImp method, and that imputing values that are missing not at random has a more severe effect on the final analysis than imputing values occurring completely at random.

Brás *et al.* [10] presented a modification of KNNimpute, which was called IKNNimpute, and performed the impact of its estimates on significance analysis for differential expression. IKNNimpute has a smaller detrimental effect on the detection of differentially expressed genes, since the iterative procedure allows refining the missing value estimates.

Clustering

As for the impact on clustering, only two studies have been found to investigate the influence of missing value imputation methods so far. Brevern *et al.* studied the impact on hierarchical clustering and showed that even a small number of missing values may significantly bias the hierarchical clustering, and that the effects of missing values are related to the chosen clustering methods as well [21]. Their main finding was that KNNimpute improves the stability of hierarchical clustering compared to the simple replacement by zero.

Tuikkala *et al.* [89] further investigated the effect of missing values on the partitional clustering algorithm and studied whether more advanced imputation methods, like LLSimpute, SVRimpute and BPCA, could provide better results than KNNimpute. Their results showed that imputation always gave better results than ignoring missing values or replacing them with row average. They also found that, among advanced imputation methods, BPCA provides accurate results especially when the missing percentage is lower than 5%.

With known gene clustering information, one may use cluster accuracy to measure

imputation quality, just as classification accuracy does. Since only a few studies focus on the impact of missing values on gene clustering, this chapter will mainly study the impact of missing value imputation methods on gene clustering and demonstrate that using clustering accuracy is indeed a good measurement.

6.2 Missing Value Imputation Framework

We have introduced two imputation methods in previous chapters, the NPRA and RPCA. NPRA mainly captures the local gene-wise relations, whereas RPCA exploits the global array-wise relations. Since NPRA and RPCA take different perspectives into consideration when imputing missing values, the deviations from the true values will not be completely correlated. Therefore, we propose to combine these two methods together by taking a weighted average of the estimates from NPRA and RPCA. In other words, $y_{CMBimpute}$, the estimate value by combined method, referred to as CMBimpute, is described as

$$y_{CMBimpute} = \mu \cdot y_{NPRA} + (1 - \mu) \cdot y_{RPCA}\,, \quad \mu \in [0, 1]\,, \tag{6.1}$$

where y_{NPRA} and y_{RPCA} are estimated values from NPRA and RPCA imputation method respectively.

The motivation of CMBimpute is that NPRA and RPCA take specific approach to data imputation, and a particular type of systematic errors is unavoidable. By combining the two estimates, we borrow strength from each of the methods, and therefore some of the errors may disappear. If the data have strong local correlation structure, CMBimpute puts more weight on NPRA, while if the data are dominated by a global structure or if there are many missing values, CMBimpute puts more weight on RPCA. We expect through selecting appropriate μ, the performance of CMBimpute will be at least as good as the best estimate from two component methods, i.e. NPRA and RPCA.

A couple of works bear important relevance to our idea. Bϕ *et al.* [8] combine gene-

based and array-based imputation estimates, using an adaptive procedure to determine the weight of the two estimates. Gan *et al.* [28] incorporate gene-based and array-based imputation estimates, and prior biological knowledge in corresponding convex sets, and then use a convergence-guaranteed iterative procedure to obtain the final solution. Jörnsten *et al.* [43] describe a method called LinCmb whose estimates are the convex combinations of the estimates by several other imputation methods. CMBimpute is dedicated to extend and generalize this kind of combination in this work.

6.2.1 How to determine K_{pc}

As mentioned before, the value for K_{pc} in RPCA imputation method is an important parameter to choose for obtaining high performance, just like the reduced rank in SVDimpute [87]. Gan *et al.* [28] found in their experiments that 20% of the eigenarrays for set C_v in POCSimpute is a good choice.

Several heuristic approaches to decide the significant components have been proposed. One typical approach is to ignore the components beyond where the cumulative relative variance or eigenvalue becomes larger than a certain threshold [93]. The other approach proposed by Everitt and Dunn [73] is to compare the relative variance of each component to $0.7/n$ (n is the number of arrays), where only those components larger than $0.7/n$ are selected. We extended Everitt's idea in our study and make use of the estimates from NPRA, and then the following heuristic algorithm for estimating parameter K_{pc} is used.

Recall the definition of normalized root mean squared error in (3.8),

$$NRMSE = \sqrt{\frac{mean[(y_{guess} - y_{real})^2]}{Var(y_{real})}},$$

where y_{guess} and y_{real} are estimated values and actual values respectively. In Chapter 5, we chose the optimal K_{pc} by minimizing NRMSE, since the missing entries are artificial and we know their actual values. But when real missing data come, we do not know the

actual values of missing entries. On the other hand, through applying NPRA imputation method, we can get the estimates for missing values from gene-wise relations' point of view. With the help of estimates from NPRA, we obtain a modification of $NRMSE$ of the following form,

$$NRMSE^* = \sqrt{\frac{mean[(y_{K_{pc}} - y_{NPRA})^2]}{Var(y_{NPRA})}},$$

where $y_{K_{pc}}$ and y_{NPRA} are estimated values by RPCA and NPRA respectively.

Let us denote the number of components whose relative variances are larger than $0.7/n$ by K_{max}. Repeating the estimations using several K_{pc}-values from 2 to K_{max}, a K_{pc}-value that produces the best estimation ability can be found by minimizing $NRMSE^*$ as follows,

$$K_{pc}^* = \arg \min_{K_{pc}=2,\cdots,K_{max}} \sqrt{\frac{mean[(y_{K_{pc}} - y_{NPRA})^2]}{Var(y_{NPRA})}}. \tag{6.2}$$

This procedure gives a number of array components that show good performance in (6.2). The motivation for this procedure is that the K_{pc}-value that shows the best performance by comparing the estimates from RPCA with that from NPRA can be close to an optimal K_{pc}-value, because we do not know the actual values of missing entries in real datasets.

6.2.2 Heuristic method to determine μ

To determine the value of μ in (6.1), some known values in the dataset are initially re-estimated, and the residuals of the NPRA and RPCA estimates are determined. Considering the vectors of deviations between the re-estimated values and known values

$$\mathbf{e}_1 = \mathbf{y_{NPRA}} - \mathbf{y_{known}} \quad \text{and} \quad \mathbf{e}_2 = \mathbf{y_{RPCA}} - \mathbf{y_{known}},$$

where $\mathbf{y_{known}}$ is the vector of known values and $\mathbf{y_{NPRA}}$ and $\mathbf{y_{RPCA}}$ are vectors of estimated values for re-estimated data from NPRA and RPCA imputation method respectively, we have

$$\begin{aligned} \mathbf{e} &= \mathbf{y_{CMBimpute}} - \mathbf{y_{known}} \\ &= \mu \cdot \mathbf{y_{NPRA}} + (1-\mu) \cdot \mathbf{y_{RPCA}} - \mathbf{y_{known}} \\ &= \mu \mathbf{e}_1 + (1-\mu)\mathbf{e}_2 \,. \end{aligned} \tag{6.3}$$

The best weighted coefficient, μ^* is to minimize the sum of squared errors for the re-estimated data,

$$\mu^* = \arg\min_{\mu} \sum_i (e[i])^2 = \arg\min_{\mu} \sum_i (\mathbf{y_{CMBimpute}}[i] - \mathbf{y_{known}}[i])^2 \,. \tag{6.4}$$

Notice that $\mathbf{e}^* = \mu^* \mathbf{e}_1 + (1-\mu^*)\mathbf{e}_2, \quad \mu^* \in [0,1]$. Therefore,

$$\sum_i (e^*[i])^2 \leq \min(\sum_i (e_1[i])^2, \sum_i (e_2[i])^2) \,. \tag{6.5}$$

Under the assumption that the joint distribution of $\mathbf{e}_1$ and $\mathbf{e}_2$ is the same for the re-estimated data as for the missing values, (6.5) guarantees that the mean squared error of the CMBimpute will be smaller or equal to either of its two component methods.

Recall the missing percentage in the original data in Table 4.1. CMBimpute first uses NPRA to estimate the missing values in original data, and then generates the same percentage of fake missing entries as in original data, among the originally non-missing entries whose real values are known. The NPRA and RPCA methods are used to estimate the fake missing entries. Since the real values of fake missing entries are known, (6.4) can be employed to determine μ^*.

The above procedure is repeated 10 times to determine the value of μ^* and finally let

it be the mean of these 10 values from the iterations, denoted by $\bar{\mu}^*$. The CMBimpute is defined as

$$\mathbf{y_{CMBimpute}} = \bar{\mu}^* \cdot \mathbf{y_{NPRA}} + (1 - \bar{\mu}^*) \cdot \mathbf{y_{RPCA}} \,. \tag{6.6}$$

6.3 Impact of Missing Value Imputation Method on Clustering

Several clustering algorithms have been proposed to identify co-expressed genes, such as Hierarchical Clustering (HC)[6], k-means clustering [34] and Self-organizing Map (SOM) [82]. Among these techniques, k-means clustering is widely used to group genes sharing similar expression levels under different experimental conditions by biologists [25], since it is very straightforward to interpret the results. The other advantage of k-means clustering is that it is very fast to implement, which is particularly important because of the large number of clusterings performed.

Recent comparative studies have also demonstrated that partitional clustering methods often produce more meaningful solutions than hierarchical clustering methods [61]. Therefore, we chose k-means algorithm to demonstrate the impact of missing values imputation method on gene clustering.

6.3.1 k-means clustering

The k-means algorithm is one of the most popular partitional clustering algorithms applied to gene expression data [61]. k-means clustering is different from hierarchical clustering in that the number of clusters, k, needs to be pre-defined. Whereas hierarchical clustering takes the raw data and processes it into clusters without any knowledge of how many clusters there should be, k-means clustering needs to be given an idea of how the data should eventually be partitioned. The goal is to divide the genes into k clusters such that some metric relative to the centroids of the clusters is minimized.

The algorithm is composed of the following steps:

1. Selecting the initial k centroids either randomly or using a given rule.

2. Assigning each gene to the cluster whose centroid is closest to its expression profile.

3. Recalculating the positions of the k centroids when all genes have been assigned.

Repeat Steps 2 and 3 until the centroids no longer move or until a given number of steps have been performed. The k-means algorithm is significantly sensitive to the initial randomly selected cluster centroids, and this effect can be reduced by running multiple times.

In our study, we used R package (cluster) downloadable from the comprehensive R archive network (CRAN) [85]. The package "cluster" provides a more robust version of k-means, called PAM, which is more robust because it minimizes a sum of dissimilarities instead of a sum of squared euclidean distances. While the standard k-means algorithm does not include any rules for the selection of the initial centroids, PAM can look for a good initial set of centroids first [81].

6.3.2 Missing value generation

From the complete data without missing values, we introduced artificial missing entries both equally and unequally distributed. Firstly, these missing entries are generated randomly for each of the missing rates $5\%, 10\%, 15\%$, and 20%. Secondly, distribution of artificial missing positions is estimated from that of missing values in original data, by randomly sampling m rows from the original dataset, which is a $p \times n$ matrix, and using their missing positions to assign MVs to the elements in the m rows of the complete dataset, which is a $m \times n$ matrix. In this way, recall Section 4.4.2, the missing percentage is denoted by r^*. We expect to get a similar missing pattern to the original data, which we hope will be more realistic than missing at random (MAR) pattern.

6.3.3 The performance measurement

After generating missing entries in complete data, we applied several imputation methods to the missing value datasets, and then performs k-means clustering on the imputed data. In order to assess the impact of missing data, we also performed clustering on complete data. The clustering results after imputation were compared to those obtained on complete data using Mean Normalized Hamming Distance (MNHD).

Specifically, the k-means clustering results obtained from complete data are regarded as benchmark clustering, where the i^{th} cluster is denoted by C_i^{ben}, $i = \{1, \cdots, k^{clu}\}$, and the C_i^{ben} clusters are associated with their corresponding gene list L_i^{ben}. Here we replace k with k^{clu}, denoting the number of clusters, in order to avoid confusion, since in NPRA imputation method there is already a parameter called k.

The imputation methods are performed after generating missing values in complete data. The clustering results obtained from the imputed data are considered as imputed clustering. The C_j^{impute} clusters are associated with their corresponding gene list L_j^{impute}, $j = \{1, \cdots, k^{clu}\}$.

To compare clustering results after imputation with those on complete data, we searched for each C_i^{ben} cluster the nearest $C_{j[i]}^{impute}$ cluster from C_j^{impute} clusters. For each cluster C_i^{ben}, the cluster, $C_{j[i]}^{impute}$, which has the maximum number of genes found in L_i^{ben}, is considered as the perfect match as follows

$$j[i] = \arg \max_{j=\{1,\cdots,k^{clu}\}} \sum_{g \in L_i^{ben}} I\{g \in L_j^{impute}\},$$

where $I\{g \in L_j^{impute}\}$ equals one if the gene belongs to the cluster C_j^{impute}, and zero otherwise.

The distance between C_i^{ben} and $C_{j[i]}^{impute}$ was calculated as the normalised Hamming

distance:

$$d(C_i^{ben}, C_{j[i]}^{impute}) = \frac{1}{n(L_i^{ben}) + n(L_{j[i]}^{impute})}\Big(\sum_{g \in L_i^{ben}} I\{g \notin L_{j[i]}^{impute}\} + \sum_{g \in L_{j[i]}^{impute}} I\{g \notin L_i^{ben}\}\Big)\,, \tag{6.7}$$

where $n(L_i^{ben})$ and $n(L_{j[i]}^{impute})$ are the number of genes in clusters C_i^{ben} and $C_{j[i]}^{impute}$, respectively [83].

The Mean Normalized Hamming Distance (MNHD) was obtained as follows,

$$D = \frac{1}{k^{clu}} \sum_{i=1}^{k^{clu}} d(C_i^{ben}, C_{j[i]}^{impute})\,, \tag{6.8}$$

where D is zero if two clusterings are identical and one if the clusterings are totally different.

6.3.4 The complete workflow

To demonstrate the impact of missing values on gene clustering, we simulated missing values in the complete dataset. For each of the missing rates ($5\%, 10\%, 15\%, 20\%$, and r^*), after generating artificial missing entries, KNNimpute, LSimpute, LLSimpute, BPCA and CMBimpute were applied separately on the same simulated dataset to estimate the missing values.

After imputing the missing values in the simulated dataset, on both the complete and imputed data, partitional clustering analysis was done by k-means algorithm, whose clusters were recorded and compared by (6.8). To reduce the influence of the number of clusters, k^{clu}, in the MNHD, we repeated clustering analysis on the same data using different k^{clu}, ranging from 2 to 11. Consequently, the clustering distance is the average MNHD over ten entities.

To summarize, by regarding the complete dataset as a dataset of 0% missing values,

we have six missing rates, each associated with 10 different k^{clu}, and five imputation methods.

6.4 Experimental Results

6.4.1 Dataset description

We have two microarray datasets, Listeria dataset [22] and Breast Cancer data [35], in this study. The Listeria dataset is from a study about infection time series by Listeria monocytogenes. It contains 39 time series in total. Each sample in this dataset contains 6850 gene expression values after data pre-processing.

The Breast Cancer dataset is a non-time series dataset. Hedenfalk *et al.* [35] monitored the global expression patterns of 7 breast cancer samples with BRCA1 mutation, 8 with BRCA2 mutation, and 7 sporadic cases with neither mutations using cDNA microarrays. There are 6512 cDNA used which represent 5361 unique genes. This dataset is formed by selecting 3226 genes for analysis and available publicly, which has multiple experiments with no time course relationships.

6.4.2 Comparative study in terms of clustering accuracy

In this subsection, we analyze and compare CMBimpute with other methods by measuring how well the clustering of the complete dataset was preserved when performing clustering on imputed data with different missing rates.

In order to alleviate the influence of k^{clu} in MNHD, Figures 6.1 and 6.2 plot the average MNHD over k^{clu} ranging from 2 to 11. From these two figures, we can see that when the missing rate is low, CMBimpute outperforms all other methods in terms of clustering distance. As missing rate increases, this advantage of CMBimpute weakens gradually. Overall, the general tendencies are that (1) KNNimpute performed the worst

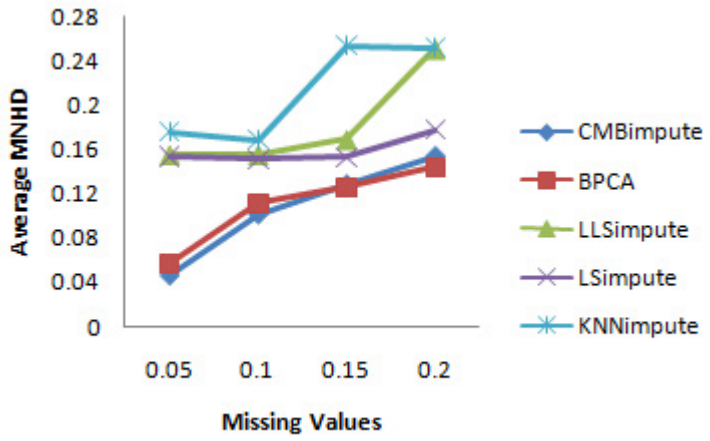

Figure 6.1: Comparison of average MNHD over different k^{clu} ranging from 2 to 11 in Listeria data with various percentages of missing values.

over two datasets, (2) in Listeria data, CMBimpute and BPCA performed equally well, and the best among the five methods, whereas LLSimpute and LSimpute performed in the middle, and (3) in Breast Cancer data, CMBimpute performed the best, especially for low missing rate, although the differences with LSimpute, LLSimpute and BPCA became negligible in the case of 20% missing rate.

From Figures 6.1 and 6.2, it is clear that the missing values had a noticeable impact on the k-means clustering results, and that imputation can reduce the influence of missing values in clustering analysis. This emphasizes why accurate imputation method is desirable for missing values. When missing rate is low, CMBimpute showed its consistent performance over two datasets, in terms of the less disagreement between the original and the recovered clustering. For high missing rate, CMBimpute is also comparable to other methods, though not the best, with its ranking in top 2.

The surprising observation is that the difference is not very remarkable in the case of high missing percentage for Breast Cancer data (Figure 6.2) between LLSimpute, BPCA and CMBimpute, although their NRMSE are 0.6459, 0.7112, and 0.6113 respectively. One possible reason is that k-means clustering is also affected by other factors, such as the number of clusters and initialization, other than missing values. We conclude that

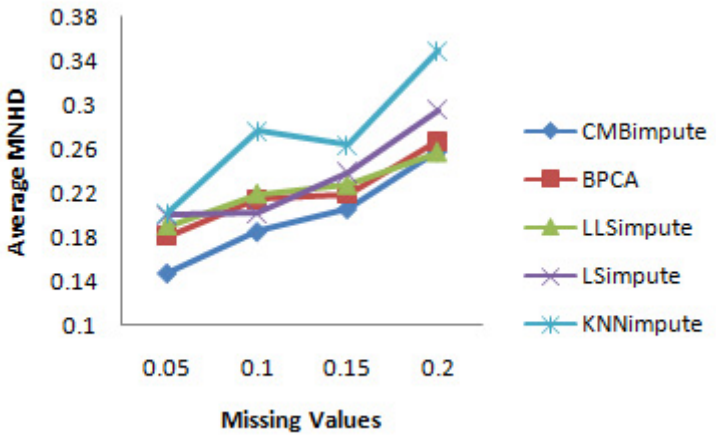

Figure 6.2: Comparison of average MNHD over different k^{clu} ranging from 2 to 11 in Breast Cancer data with various percentages of missing values.

missing values is a very important factor which influences the results of k-means clustering analysis, stressing the importance of more advanced imputation methods.

In order to have a comprehensive understanding, we also present the box plots of MNHD for different numbers of clusters, k^{clu}, for all five imputation methods over the two datasets. Figure 6.3 demonstrates box plots in the case of 5% missing entries with results confirming the superior performance of CMBimpute across the range of the number of clusters. It is important that the imputation methods be able to exhibit lower average MNHD, as shown in Figures 6.1 and 6.2.

Moreover, the disparity of MNHD for different k^{clu} is of great concern. As seen from Figure 6.3, the results for CMBimpute are competitive, if not better than other methods. CMBimpute has the smallest disparity for both datasets in the case of 5% missing values, while LSimpute and KNNimpute show the largest disparity in the two respective datasets. However, the advantage of CMBimpute becomes weakened as missing rate increases, which can be found in the box plots for other missing rates (shown in the appendix).

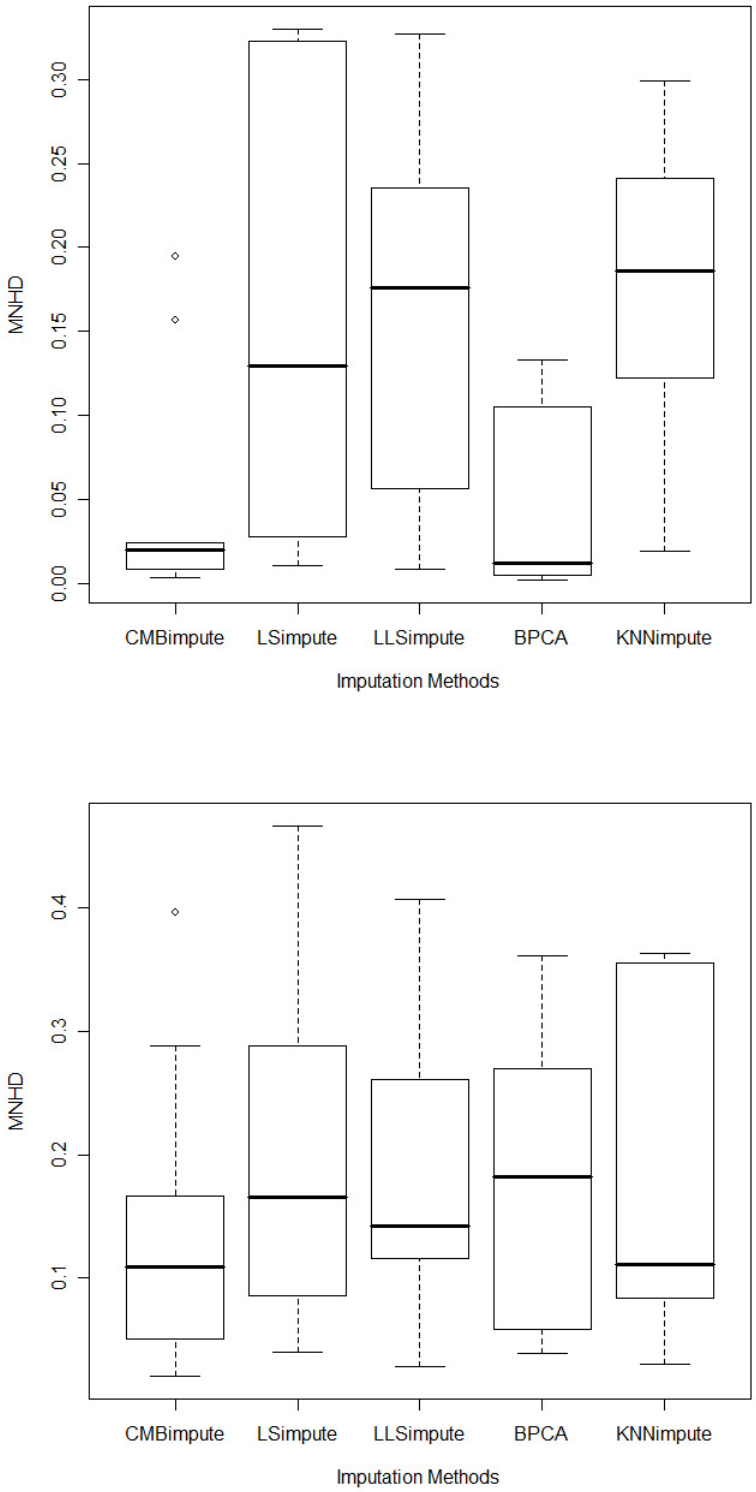

Figure 6.3: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Listeria (top) and Breast Cancer (bottom) data with 5% missing rate.

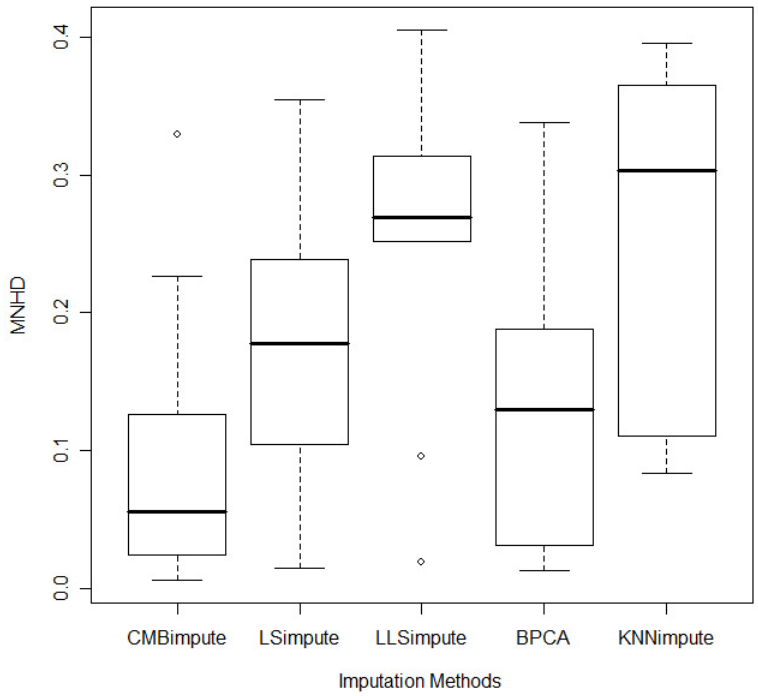

Figure 6.4: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Listeria data on missing not at random pattern.

We also studied the effect of missing mechanism by generating non-random missing value distribution, as introduced in Section 6.3.2. In such a more realistic missing pattern, Figure 6.4 shows the box plots of MNHD for different k^{clu} in Listeria data. We found that CMBimpute and BPCA perform best among all the imputation methods. This is due to the fact that CMBimpute and BPCA allow using the information of the genes having MVs more efficiently, and the iterative procedure allows refining the MV estimates. Specifically, CMBimpute has lower median MNHD across different k^{clu} than BPCA. A possible reason is that CMBimpute takes advantage of both gene-wise and array-wise relationships, whereas BPCA only considers gene-wise relationships.

We also found that imputing values that are missing not at random had a more serious effect on clustering analysis than imputing values that are missing at random. This can be attributed to the fact that the pattern of missing not at random is more complex

than that of missing at random. In reality, the pattern of missing data in a microarray dataset is a mixture of missing at random and missing not at random. Although the number of the clusters, k^{clu}, has influence in clustering analysis when we compared the clustering results from complete data with those obtained on imputed data in Figure 6.4, the disparity of the results across k^{clu} is lower when CMBimpute method is used. It suggests that CMBimpute is more robust and accurate than the widely used imputation methods.

6.5 Summary

The purpose of imputation is not to improve clustering quality, but rather to lessen the influence of missing values in clustering analysis. For more clustering algorithms, one may refer to [44]. This chapter has proposed CMBimpute by constructing a missing value imputation framework, which takes advantage of both gene-wise and array-wise relationships. The impact of missing values on gene clustering was particularly highlighted, which stimulates the need for accurate imputation method.

The performance of CMBimpute was compared with other advanced imputation methods, in terms of the ability of recovering the clusters on complete data. The experimental results show that CMBimpute is better than other comparative missing value imputation techniques, namely KNNimpute, LSimpute, LLSimpute and BPCA, for low missing percentage, and it shows comparable, if not better results, in high missing percentage cases. In addition, CMBimpute is more accurate and robust in MNAR pattern. Therefore, it is justified that clustering accuracy is also a measure to assess imputation methods.

Chapter 7

Conclusion and Future Work

The main purpose of this thesis was to deal with missing values in microarray. Numerous subsequent microarray data analyses have posed challenges for imputing missing values accurately in data pre-processing step. This chapter concludes the study by presenting a summary of research findings, as well as discussing the implications and limitations of this research. Furthermore, we suggest some possibilities for future work.

7.1 Conclusion

In the first part of this thesis (Chapter 4), we developed a novel non-parametric regression approach (NPRA) for imputing missing values. This novel imputation method takes advantage of relationships between genes. Traditional LSimpute and LLSimpute methods only utilize the linear relations, whereas our proposed NPRA captures both linear and non-linear relations between genes by virtue of non-parametric regression. Moreover, the number of similar genes, k, is recommended in our study. In summary, NPRA serves the purpose of capturing local gene-wise relationships.

We apply NPRA to different types of datasets (time series, mixed and non-time series) with two patterns, i.e., missing at random and missing not at random. Using the NRMSE and R^2 between true and imputed MVs as measures of performance, the comparative

studies with KNNimpute, LSimpute and LLSimpute suggest the strong competitiveness of NPRA.

We also explain the influence of some factors in the prediction accuracy of imputation methods for microarray. The results show that the rate and pattern of missing data and the type of microarray experiment affect the data similarity structure, therefore influencing the methods' performance. The methods' prediction ability decreases with increasing missing rate. In addition, prediction ability is weaker in the presence of a missing pattern that is close to that of the original data, than in the presence of a random missing pattern.

In the second part of this thesis (Chapter 5), we make use of array-wise relationship to improve imputation accuracy. We employ the dimension reduction technique, principal component analysis (PCA), to capture array components. One drawback of the existing imputation methods is their lack of robustness to outliers in the data. In order to deal with the outliers in microarray, we bring robust regression into our algorithm RPCA, which significantly reduces the influence of outliers in imputation accuracy. To summarize, RPCA exploits the global array-wise relationships, which are not widely considered in other methods.

We also study the sensitivity of RPCA to initial estimates and the effect of K_{pc}, the number of significant components, on RPCA. Furthermore, we compare RPCA imputation method with LLSimpute and BPCA, which are the two state-of-the-art methods for imputing missing values in microarray. The results show that RPCA outperforms LLSimpute and BPCA for time series and non-time series data, whereas BPCA has the best performance for mixed data.

In addition, we evaluate each method's performance on noisy data by adding artificial noise to the complete data. The results suggest that RPCA is less sensitive to noise than BPCA. This may be due to the fact that PCA can capture the underlying array expression

patterns even in the presence of noise. We also investigate computation efficiency, besides imputation accuracy. This finding shows that we have to consider the trade-off between computational burden and imputation accuracy, although imputation accuracy is the top evaluation metric when assessing imputation methods.

In the third part of this thesis (Chapter 6), we first construct the framework of missing value imputation, which utilizes both gene-wise and array-wise relationships. In this framework, CMBimpute is a weighted average on NPRA and RPCA. By combining the estimates from NPRA and RPCA respectively, we borrow strength from each of the methods and avoid particular type of systematic errors. In addition, we propose a heuristic algorithm to determine the optimal weight μ. To summarize, CMBimpute combines the exploitation of local gene correlations and global array correlations in a given dataset with the selection of optimal μ.

Finally, while most of the imputation methods have been evaluated only in terms of the error between imputed value and true value, e.g., NRMSE, we investigate the influence of missing values and imputation method in gene clustering analysis. The impact of missing values on subsequent analysis has attracted many researchers' interest. Missing value imputation is essential to minimize the adverse effect of missing values on microarray data mining.

Our experimental results suggest that missing values have severe influence on gene clustering, where accurate imputation method is desirable for reducing the biased clusters. CMBimpute performs the best when applied to low percentage missing data, compared to other state-of-the-art imputation methods. Moreover, for high percentage missing data, CMBimpute shows comparable performance, if not better than, to other imputation methods. In MNAR pattern, CMBimpute is more accurate and robust than the other imputation methods. This may be attributed to two reasons: CMBimpute takes into account both gene-wise and array-wise relationships, and CMBimpute allows the re-use

of imputed values through iterative procedures to refine missing value estimates.

The novel algorithms presented in this thesis are of considerable importance. They are more capable of exploring existing gene-wise and array-wise relationship. They also show excellent abilities to impute missing values and lessen the impact on the subsequent microarray data mining.

7.2 Future Work

Each of the accomplishments above demonstrates that dealing with missing values in microarray datasets is a challenging issue. Missing value is not a minor problem, thus it should not be neglected in microarray analysis. From the research undertaken, many questions and possible extensions have emerged. These are outlined as follows:

1. In Chapter 4, NPRA uses Euclidean distance to select k similar genes. However, Euclidean distance may not be very accurate to determine the value of k and reflect the co-expressed genes, especially in the case of high missing percentage. One possible way of overcoming this problem is to look from a mathematical perspective, i.e., using Monte Carlo simulation [77]. The other way is to incorporate some available biological information. The use of available information will somehow contribute to the selection of similar genes, such as acetylation information [99].

2. In Chapter 5, RPCA employs quantile regression using the 50^{th} percentile as a robust measure to alleviate the influence of outliers when imputing missing values. A further study on selecting the better robust method is desirable.

3. Missing value imputation methods are generally proposed as a separate part from microarray data mining. In other words, we first impute missing values, and then perform data analysis, so missing value imputation is considered as a pre-processing step. Obviously, the important part is subsequent data analysis. Thus, to incorporate the part of imputing missing values in microarray data mining may be a

future direction, in order to improve the final result, e.g., gene selection and tumor classification.

4. Each dataset has its own underlying characteristics. How to exploit this kind of information could be a future direction when dealing with missing values. The underlying complexity of the expression data should guide the researcher in determining the best method to use for imputing missing values [11].

5. To keep the study focused, we did not test our methods on non-microarray datasets in this thesis. It is likely that our algorithms could sparkle some insights to estimate missing values in datasets other than gene expression data. It will be interesting to make extensions in other domains and study their performance.

Bibliography

[1] E. Acuna and C. Rodriguez. The treatment of missing values and its effect in the classifier accuracy. In *Classification, Clustering and Data Mining Applications*, pages 639–648. Berlin, Heidelberg Springer-Verlag, 2004.

[2] Himanshu Agrawal. Extreme self-organization in networks constructed from gene expression data. *Phys. Rev. Lett.*, 89(26):268702, Dec 2002.

[3] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, and X. Yu. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000.

[4] O. Alter, P.O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. In *Proc. Natl Acad. Sci. USA*, volume 97, pages 10101–10106, 2000.

[5] Mi Arnone and Eh Davidson. The hardwiring of development: Organization and function of genomic regulatory systems. *Development*, 124:1851–1864, 1997.

[6] Everitt B. *Cluster Analysis*. Heinemann Educ, 1974.

[7] Raj B., Singh R., and Stern RM. Inference of missing spectrographic features for robust speech recognition. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1491–1494, 1998.

[8] H. T. Bϕ, B. Dysvik, and I. Jonassen. LSimpute: Accurate estimation of missing

values in microarray data with least squares methods. *Nucleic Acids Research*, 32:e34, 2004.

[9] Lígia P. Brás and José C. Menezes. Dealing with gene expression missing data. *Systems Biology*, 3:105–119, May 2006.

[10] Lígia P. Brás and José C. Menezes. Improving cluster-based missing value estimation of DNA microarray data. *Biomolecular Engineering*, 2:273–282, June 2007.

[11] Guy N. Brock, John R. Shaffer, Richard E. Blakesley, Meredith J. Lotz, and George C. Tseng. Which missing value imputation method to use in expression profiles: A comparative study and two selection schemes. *BMC Bioinformatics*, 9:12+, January 2008.

[12] M. Brown, W. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. Furey, M. Jr, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. In *Proc. Natl. Acad. Sci. USA*, volume 97, pages 262–267, 2000.

[13] Zhipeng Cai, Maysam Heydari, and Guohui Lin. Iterated local least squares microarray missing value imputation. *J. Bioinformatics and Computational Biology*, 4(5):935–958, 2006.

[14] Yi Cao and Kim Leng Poh. An accurate and robust missing value estimation for microarray data: Least absolute deviation imputation. In *Proceedings of 5th International Conference on Machine Learning and Applications (ICMLA'06)*, pages 157–161, 2006.

[15] Yi Cao, Kim Leng Poh, and Wen Juan Cui. A non-parametric regression approach for missing value imputation in microarray. In *Intelligent Information Systems XVI: Proceedings of the International IIS'08 Conference*, pages 25–34, 2008.

[16] W.C. Chang, C.W. Li, and B.S. Chen. Quantitative inference of dynamic regulatory pathways via microarray data. *BMC Bioinformatics*, 6:44, 2005.

[17] J.H. Cho, D. Lee, J.H. Park, and I.B. Lee. New gene selection method for classification of cancer subtypes considering within-class variation. *FEBS Lett.*, 551:3–7, 2003.

[18] S. Chu, J. DeRisi, M. Eisen, J. Mulholland, D. Botstein, P.O. Brown, and Erskowitz I. The transcriptional program of sporulation in budding yeast. *Science*, 278:680–686, 1998.

[19] Rubin D. *Multiple Imputation for Nonresponse in Surveys*. Siley, New York, 1987.

[20] Storey J. D. and R. Tibshirani. Statistical significance for genome wide studies. *PNAS*, 100(16):9440–9445, 2003.

[21] Alexandre G de Brevern, Serge Hazout, and Alain Malpertuy. Influence of microarrays experiments missing values on the stability of gene groups by hierarchical clustering. *BMC Bioinformatics*, 5:114, 2004.

[22] Baldwin D.N., Vanchinathan V., Brown P.O., and Theriot J.A. A gene-expression program reflecting the innate immune response of cultured intestinal epithelial cells to infection by Listeria monocytogenes. *Genome Biology*, 4(1), 2002.

[23] Ben A. Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3-4):281–297, 1999.

[24] Sam Efromovich. *Nonparametric curve estimation: Methods, theory and applications*. Springer, New York, 1999.

[25] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci USA*, 95(25):14863–14868, December 1998.

[26] S. Friedland, A. Niknejad, M. Kaveh, and H. Zare. An algorithm for missing value estimation for DNA microarray data. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing(ICASSP'06)*, 14-19 May 2006.

[27] Shmuel Friedland, Amir Niknejad, and Laura Chihara. A simultaneous reconstruction of missing data in DNA microarrays. *Linear Alg. Appl*, 416:8–28, 2006.

[28] Xiangchao Gan, Alan Wee-Chung Liew, and Hong Yan. Microarray missing data imputation based on a set theoretic framework and biological knowledge. *Nucleic Acids Res.*, 34(5):1608–1619, 2006.

[29] A.J.R. GODFREY, G.R. WOOD, S. GANESALINGAM, M. A. NICHOLS, and C.G. QIAO. Two-stage clustering in genotype-by-environment analyses with missing data. *The Journal of Agricultural Science*, 139:67–77, 2002.

[30] G.H. Golub and C.F. Van Loan. *Matrix computation*. Johns Hopkins University press,Baltimore,MD, 1996.

[31] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, October 1999.

[32] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[33] Yoshimoto H., Saltman K., Gasch A. P., Li H.X., Ogawa N., David B., BROWN Patrick O., and CYERT Martha S. Genome-wide analysis of gene expression regulated by the Calcineurin/Crzlp signaling pathway in Saccharomyces cerevisiae. *The Journal of Biological Chemistry*, 277(34):31079–31088, 2002.

[34] J.A. Hartigan and M.A. Wong. A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[35] I. Hedenfalk, D. Duggan, Y. Chen, M. Radmacher, M. Bittner, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, O. Kallioniemi, B. Wilfond, A. Borg, and J. Trent. Gene-expression profiles in hereditary breast cancer. *The New England Journal of Medicine*, 344:539–548, 2001.

[36] N. Holter, M. Mitra, A. Maritan, M. Cieplak, J. Banavar, and N. Fedoroff. Fundamental patterns underlying gene expression profiles: Simplicity from complexity. In *Proc. Natl Acad. Sci. USA*, volume 97, pages 8409–8414, 2000.

[37] Jianjun Hu, Haifeng Li, Michael Waterman, and Xianghong Zhou. Integrative missing value estimation for microarray data. *BMC Bioinformatics*, 7:449, October 2006.

[38] P. J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35:73–101, 1964.

[39] P. J. Huber. Robust regression: Asymptotics, conjectures and Monte Carlo. *Ann. Stat.*, 1:799–821, 1973.

[40] Chen J. and Shao J. Jackknife variance estimation for nearest neighbour imputation. *Journal of the American Statistical Association*, 96:260–269, 2001.

[41] Schafer J. *Analysis of Incomplete Multivariate Data*. Chapman & Hall Inc., 1997.

[42] Rebecka Jörnsten, Ming Ouyang, and Hui-Yu Wang. A meta-data based method for DNA microarray imputation. *BMC Bioinformatics*, 8:109, March 2007.

[43] Rebecka Jörnsten, Hui-Yu Wang, William J. Welsh, and Ming Ouyang. DNA microarray data imputation and significance analysis of differential expression. *Bioinformatics*, 21:4155–4161(7), 15 November 2005.

[44] Daxin Jiang, Chun Tang, and Aidong Zhang. Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1370–1386, 2004.

[45] Peter Johansson and Jari Hakkinen. Improving missing value imputation of microarray data by using spot quality weights. *BMC Bioinformatics*, 7:306, June 2006.

[46] Javed Khan, Jun S. Wei, Markus Ringnér, Lao H. Saal, Marc Ladanyi, Frank Westermann, Frank Berthold, Manfred Schwab, Cristina R. Antonescu, Carsten Peterson, and Paul S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6):673–679, June 2001.

[47] H. Kim, Gene H. Golub, and Haesun Park. Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics*, 21:187–198(12), 2005.

[48] Ki-Yeol Kim, Byoung-Jin Kim, and Gwan-Su Yi. Reuse of imputed data in microarray analysis increases imputation efficiency. *BMC Bioinformatics*, 5:160, October 2004.

[49] Breiman L., Friedman J.H., Olshen R.A., and Stone C.J. *Classification and Regression Trees*. Chapman & Hall Inc., 1984.

[50] K. E. Lee, N. Sha, E. R. Dougherty, M. Vannucci, and B. K. Mallick. Gene selection: a Bayesian variable selection approach. *Bioinformatics*, 19(1):90–97, January 2003.

[51] Mei-Ling Ting Lee. *Analysis of Microarray Gene Expression Data*, chapter 7, pages 85–92. Springer US, 2004.

[52] Qi Li and Jeffrey Scott Racine. *Nonparametric econometrics: Theory and practice*. Princeton University Press, Princeton and Oxford, 2007.

[53] Li Liu, Douglas M. Hawkins, Sujoy Ghosh, and S. Stanley Young. Robust singular value decomposition analysis of microarray data. *Proc Natl Acad Sci USA*, 100(23):13167–13172, Nov 2003.

[54] D.J. Lockhart and E.A. Winzeler. Genomics, gene expression and DNA arrays. *Nature*, 405:827–836, 2000.

[55] Harvey M. and Arthur C. *Fitting models to biological data using linear and nonlinear regression*. Oxford University Press, 2004.

[56] G. Natsoulis, L. El Ghaoui, G. R. Lanckriet, A. M. Tolley, F. Leroy, S. Dunlea, B. P. Eynon, C. I. Pearson, S. Tugendreich, and K. Jarnagin. Classification of a large microarray data set: algorithm comparison and analysis of drug signatures. *Genome Res*, 15(5):724–736, May 2005.

[57] D. Nguyen and D. Rocke. Multi-class cancer classification via partial least sqaures with gene expression profiles. *Bioinformatics*, 18(9):1216–1226, 2002.

[58] Danh V. Nguyen, Naisyin Wang, and Raymond J. Carroll. Evaluation of missing value estimation for microarray data. *Journal of Data Science*, 2(4):347–370, 2004.

[59] S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara, and S. Ishii. A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, 19:2088–2096, 2003.

[60] M. Ouyang, W. J. Welsh, and P. Georgopoulos. Gaussian mixture clustering and imputation of microarray data. *Bioinformatics*, 20(6):917–923, April 2004.

[61] D'haeseleer P. How does gene expression clustering work? *Nature Biotechnology*, 23:1499–1501, 2005.

[62] Gasch A. P., Spellman P. T., Kao C. M., Carmel-Harel O., Eisen M. B., Storz G., Botstein D., and Brown P. O. Genomic expression programs in the response of

yeast cells to environmental changes. *Molecular Biology of the Cell*, 11:4241–4257, 2000.

[63] Koenker R. and Hallock K. Quantile regression. *Journal of Economic Perspectives*, 15:143–156, 2001.

[64] Little R. and Rubin D. *Statistical analysis with missing data*. Wiley, New York, 1987.

[65] C. Radhakrishna Rao. *Linear statistical inference and its applications*. Wiley, New York, 1973.

[66] S. Raychaudhuri, J.M. Stuart, and R. Altman. Principal components analysis to summarize microarray experiments: Application to sporulation time series. In *Biocomputing 2000: Proceedings of the Pacific Symposium*, pages 452–463, 2000.

[67] Dudoit S, Yang YH, Callow MJ, and Speed TP. Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Stat Sinica*, 12:111–139, 2002.

[68] Lee S-I and Batzogolou S. Application of independent component analysis to microarrays. *Genome Biology*, 4:R76, 2003.

[69] I. Scheel, M. Aldrin, I.K. Glad, R. Srum, H. Lyng, and A. Frigessi. The influence of missing value imputation on detection of differentially expressed genes from microarray data. *Bioinformatics*, 21:4272–4279, 2005.

[70] M. Schena, D. Shalon, R.W. Davis, and P.O. Brown. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270:467–470, 1995.

[71] M. Scholz, F. Kaplan, C.L. Guy, J. Kopka, and J. Selbig. Non-linear PCA: a missing data approach. *Bioinformatics*, 21(20):3887–3895, 2005.

[72] A. Schulze and J. Downward. Navigating gene expression using microarrays - a technology review. *Nat. Cell Biol.*, 3:E190–E195, 2001.

[73] Everitt S.E. and Dunn G. *Applied Multivariate Data Analysis*. London:Arnold, 2001.

[74] Muhammad Shoaib B. Sehgal, Iqbal Gondal, and Laurence Dooley. Missing value imputation framework for microarray significant gene selection and class prediction. In *Data Mining for Biomedical Applications*, pages 131–142, 2006.

[75] Muhammad Shoaib B. Sehgal, Iqbal Gondal, and Laurence S. Dooley. Collateral missing value imputation: A new robust missing value estimation algorithm for microarray data. *Bioinformatics*, 21(10):2417–2423, 2005.

[76] Muhammad Shoaib B. Sehgal, Iqbal Gondal, and Laurence S. Dooley. Missing values imputation for cDNA microarray data using ranked covariance vectors. *Int. J. Hybrid Intell. Syst.*, 2(4):295–312, 2005.

[77] Muhammad Shoaib B S. Sehgal, Iqbal Gondal, Laurence S S. Dooley, and Ross Coppel. Ameliorative missing value imputation for robust biological knowledge inference. *Journal of biomedical informatics*, December 2007.

[78] S. Siegel and N.J. Castellan. *Nonparametric Statistics for Behavioral Sciences*. McGraw-Hill, New York, 1988.

[79] M. Smith and R. Kohn. Nonparametric regression using Bayesian variable selection. *Journal of Econometrics*, 75(2):317–343, December 1997.

[80] Kuzin SS. Data imputation based on regression models with variations of entropy. In *Conference of European Statistics*, pages 18–20, 2000.

[81] A. Struyf, M. Hubert, and P.J. Rousseeuw. Integrating robust clustering techniques in S-plus. *Computational Statistics and Data Analysis*, 26:17–37, 1997.

[82] Kohonen T. *Self-Organizing Maps*. Springer Verlag, 2001.

[83] Lange T., Roth V., Braun M.L., and Buhmann J.M. Stability-based validation of clustering solutions. *Neural computation*, 16:1299–1323, 2004.

[84] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub. Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc Natl Acad Sci USA*, 96(6):2907–2912, Mar 1999.

[85] R Development Core Team. R: A language and environment for statistical computing, 2007. ISBN 3-900051-07-0.

[86] J.P. Townsend. Resolution of large and small differences in gene expression using models for the Bayesian analysis of gene expression levels and spotted DNA microarrays. *BMC Bioinformatics*, 5:54, 2004.

[87] O. Troyanskaya, M. Cantor, G. Sherlock, Brown P., T. Hastie, R. Tibshirani, D. Botstein, and R.B. Altman. Missing value estimation methods for DNA microarray. *Bioinformatics*, 17:520–525(6), 2001.

[88] J. Tuikkala, L. Elo, O. S. Nevalainen, and T. Aittokallio. Improving missing value estimation in microarray data with gene ontology. *Bioinformatics*, 22(5):566–572, March 2006.

[89] Johannes Tuikkala, Laura L. Elo, Olli S. Nevalainen, and Tero Aittokallio. Missing value imputation improves clustering and interpretation of gene expression microarray data. *BMC Bioinformatics*, 9:202, April 2008.

[90] V. G. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *Proc Natl Acad Sci U S A*, 98(9):5116–5121, April 2001.

[91] Vera van Noort, Berend Snel, and Martijn Huynen. The yeast coexpression network has a small-world, scale-free architecture and can be explained by a simple model. *EMBO Reports*, 5(3):280–284, March 2004.

[92] S. Verboven, K. V. Branden, and P. Goos. Sequential imputation for missing values. *Comput Biol Chem*, 31(5-6):320–327, October 2007.

[93] Michael E. Wall, Andreas Rechtsteiner, and Luis M. Rocha. *Singular Value Decomposition and Principal Component Analysis*, chapter 5, pages 91–109. Kluwel, Norwell, MA, Mar 2003.

[94] Dong Wang, Yingli Lv, Zheng Guo, Xia Li, Yanhui Li, Jing Zhu, Da Yang, Jianzhen Xu, Chenguang Wang, Shaoqi Rao, and Baofeng Yang. Effects of replacing the unreliable cDNA microarray measurements on the disease classification based on gene expression profiles and functional modules. *Bioinformatics*, 22(23):2883–2889, 2006.

[95] Xian Wang, Ao Li, Zhaohui Jiang, and Huanqing Feng. Missing value estimation for DNA microarray gene expression data by support vector regression imputation and orthogonal coding scheme. *BMC Bioinformatics*, 7:32, 2006.

[96] D.S. Watkins. *Fundamentals of Matrix Computations*. Wiley, New York, 1991.

[97] E.C. Wit and J.D. McClure. *Statistics for Microarrays:Design, Analysis and Inference*. John Wiley & Sons Ltd, Chichester, UK, 2004.

[98] Dorothy S. Wong, Frederick K. Wong, and Graham R. Wood. A multi-stage approach to clustering and imputation of gene expression profiles. *Bioinformatics*, 23(8):998–1005, April 2007.

[99] Qian Xiang, Xianhua Dai, Yangyang Deng, Caisheng He, Jiang Wang, Jihua Feng, and Zhiming Dai. Missing value imputation for microarray gene expression data using histone acetylation information. *BMC Bioinformatics*, 9:252, May 2008.

[100] Shi Y., Cai Z., and Lin G. *Classification accuracy based microarray missing values imputation*. New Jersey: Wiley-Interscience, 2007.

[101] D. Yoon, E. K. Lee, and T. Park. Robust imputation method for missing values in microarray data. *BMC Bioinformatics*, 8 Suppl 2, 2007.

[102] X. Zhou, X. Wang, and E. R. Dougherty. Construction of genomic networks using mutual-information clustering and reversible-jump markov-chain-monte-carlo predictor design. *Signal Processing*, 83(4):745–761.

[103] X. Zhou, X. Wang, and E. R. Dougherty. Missing-value estimation using linear and non-linear regression with Bayesian gene selection. *Bioinformatics*, 19(17):2302–2307, November 2003.

Appendix A

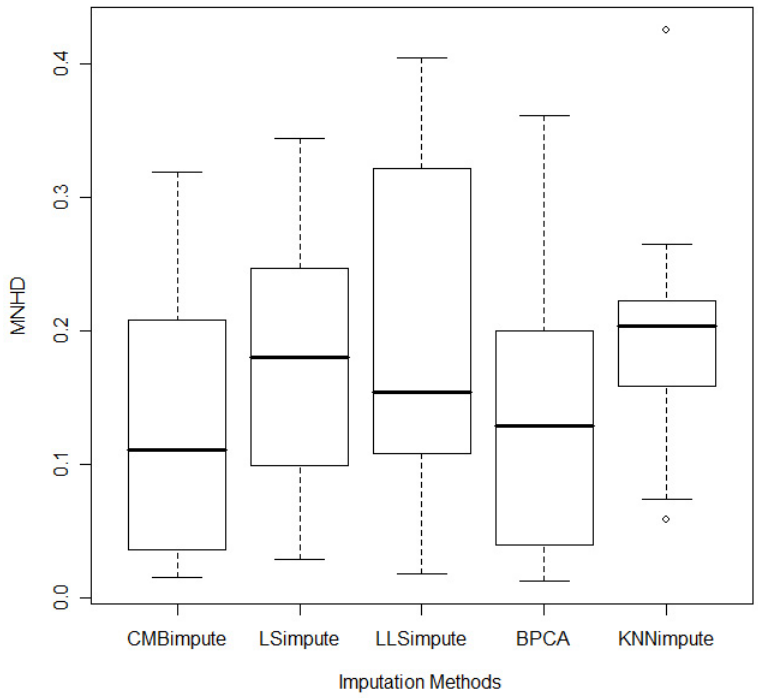

Figure A.1: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Listeria data with 10% missing rate.

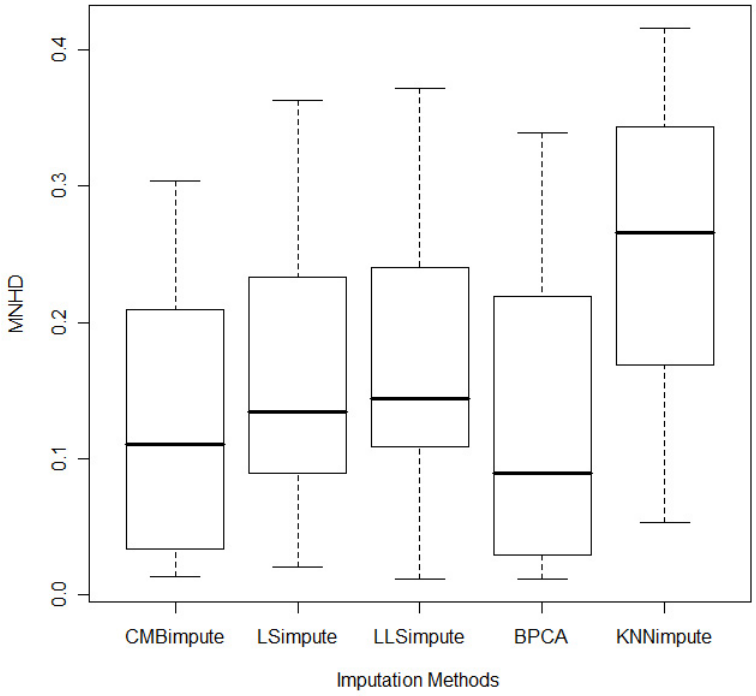

Figure A.2: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Listeria data with 15% missing rate.

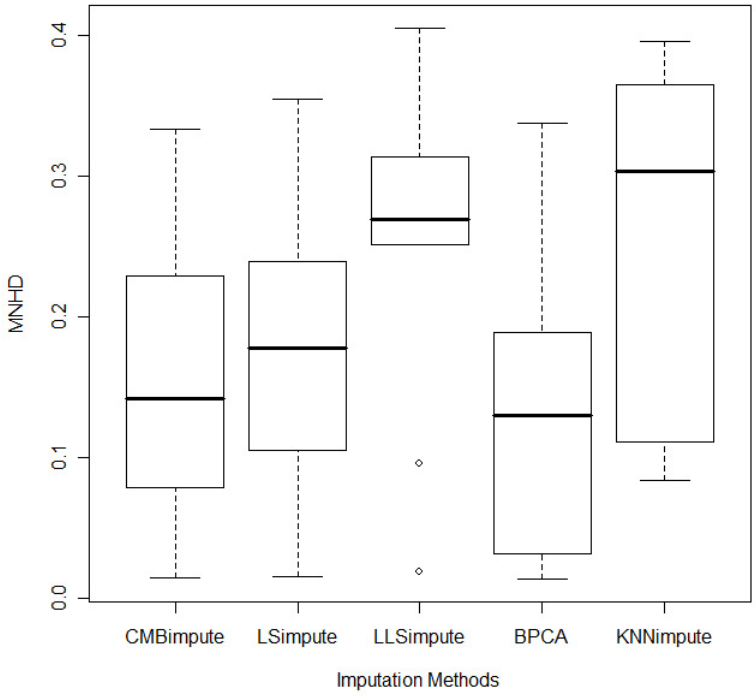

Figure A.3: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Listeria data with 20% missing rate.

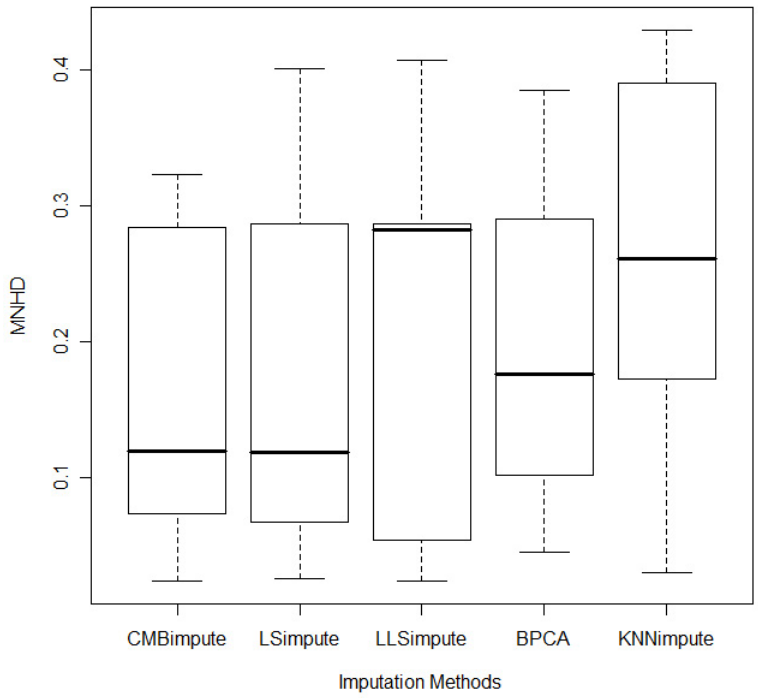

Figure A.4: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Breast Cancer data with 10% missing rate.

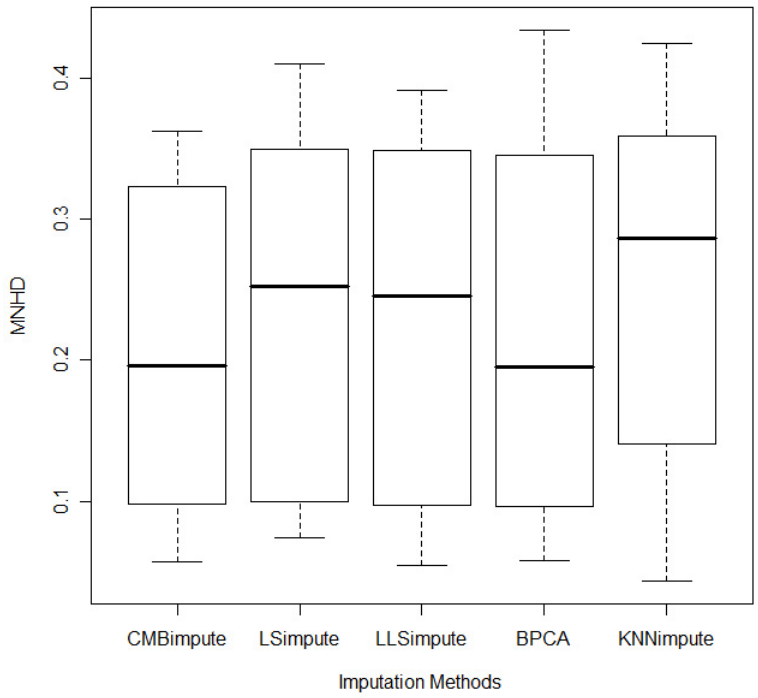

Figure A.5: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Breast Cancer data with 15% missing rate.

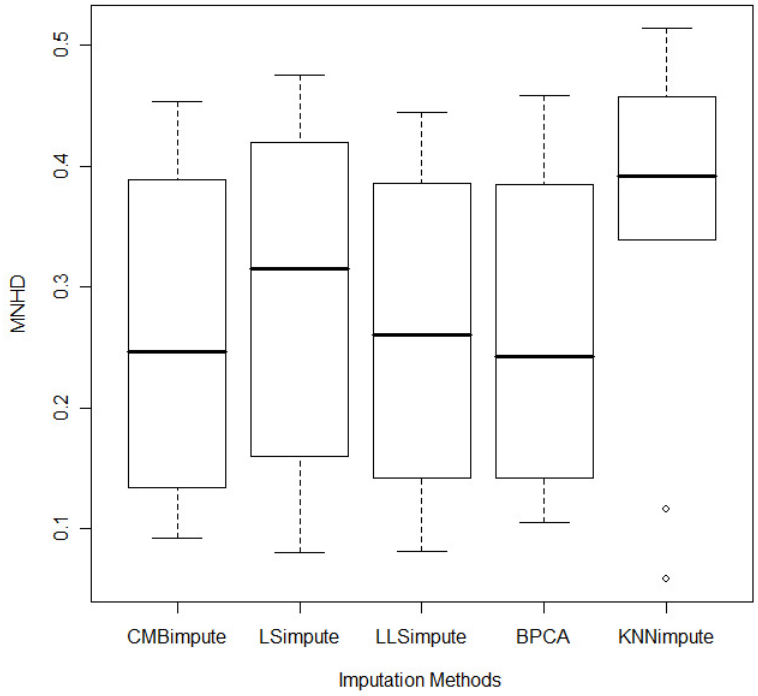

Figure A.6: Box plots of MNHD for different k^{clu} ranging from 2 to 11 in Breast Cancer data with 20% missing rate.