


Winter 2013

New microarray image segmentation using Segmentation Based Contours method

Yuan Cheng

Follow this and additional works at: <https://digitalcommons.latech.edu/dissertations>

 Part of the [Applied Mathematics Commons](#), [Applied Statistics Commons](#), [Biomedical Engineering and Bioengineering Commons](#), and the [Mathematics Commons](#)

**NEW MICROARRAY IMAGE SEGMENTATION USING
SEGMENTATION BASED CONTOURS METHOD**

by

Yuan Cheng, B.S., M.S.

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

COLLEGE OF ENGINEERING AND SCIENCE
LOUISIANA TECH UNIVERSITY

February 2013

UMI Number: 3570075

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI 3570075

Published by ProQuest LLC 2013. Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against unauthorized copying under Title 17, United States Code.



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

LOUISIANA TECH UNIVERSITY

THE GRADUATE SCHOOL

OCTOBER 31, 2012

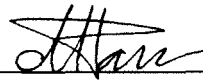
Date

We hereby recommend that the Dissertation prepared under our supervision by
Yuan Cheng, B.S., M.S.

entitled New Microarray Image Segmentation Using Segmentation Based
Contours Method

be accepted in partial fulfillment of the requirements for the Degree of

Ph.D. in Computational Analysis and Modeling



Supervisor of Dissertation Research

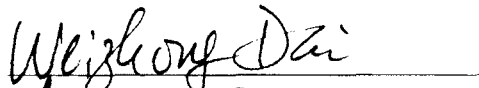
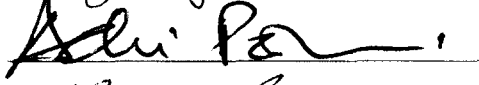




Head of Department

Computational Analysis and Modeling

Department

Recommendation concurred in:

Advisory Committee

Approved:

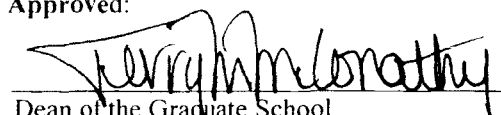


Director of Graduate Studies



Dean of the College

Approved:



Dean of the Graduate School

ABSTRACT

The goal of the research developed in this dissertation is to develop a more accurate segmentation method for Affymetrix microarray images. The Affymetrix microarray biotechnologies have become increasingly important in the biomedical research field. Affymetrix microarray images are widely used in disease diagnostics and disease control. They are capable of monitoring the expression levels of thousands of genes simultaneously. Hence, scientists can get a deep understanding on genomic regulation, interaction and expression by using such tools.

We also introduce a novel Affymetrix microarray image simulation model and how the Affymetrix microarray image is simulated by using this model. This simulation model embraces all realistic biological characteristics and experimental preparation characteristics, which could have different impacts on the quality of microarray image during the real microarray experiment. The most important aspect is that this model could provide the “ground true information,” which allows us to have a deep understanding on different segmentation algorithms performance.

After the simulation, the new proposed segmentation algorithm Segmentation Based Contours (SBC) method is presented as well as the modifications of the Active Contours Without the Edges (ACWE) method. By modifying the ACWE method with higher order finite difference scheme and fast scheme, we establish the new segmentation algorithm Segmentation Based Contours method. In the end, we compare the gene

signal values obtained from the new proposed algorithm Segmentation Based Contours method and the best currently known method. This gene expression signal comparison is more meaningful in gene expression analysis, since it represents the whole gene expression level rather than the small transcripts hybridization abundance level. Different types of experimental comparison results will be presented to show that the new proposed Segmentation Based Contours method is more efficient and accurate.

APPROVAL FOR SCHOLARLY DISSEMINATION

The author grants to the Prescott Memorial Library of Louisiana Tech University the right to reproduce, by appropriate methods, upon request, any or all portions of this Dissertation. It is understood that "proper request" consists of the agreement, on the part of the requesting party, that said reproduction is for his personal use and that subsequent reproduction will not occur without written approval of the author of this Dissertation. Further, any portions of the Dissertation used in books, papers, and other works must be appropriately referenced to this Dissertation.

Finally, the author of this Dissertation reserves the right to publish freely, in the literature, at any time, any or all portions of this Dissertation.

Author Cheryn

Date 10/31/2012

TABLE OF CONTENTS

ABSTRACT.....	iii
LIST OF TABLES.....	viii
LIST OF FIGURES	xi
ACKNOWLEDGEMENTS.....	xiv
CHAPTER ONE INTRODUCTION OF DNA AND DNA MICROARRAY.....	1
1.1 DNA	1
1.2 DNA Transcription and Translation.....	3
1.3 DNA Microarray	5
1.3.1 cDNA Microarray	7
1.3.2 Affymetrix Microarray.....	9
CHAPTER TWO INTRODUCTION OF AFFYMETRIX MICROARRAY IMAGE...	15
2.1 Overview of Affymetrix Image Analysis Methods.....	17
2.2 Affymetrix Microarray Image Analysis Process.....	19
2.3 Affymetrix Microarray Image Analysis Flow in GCOS	25
CHAPTER THREE MICROARRAY IMAGE SIMULATION METHOD	31
3.1 Database for Simulation Model.....	32
3.2 Microarray Simulation Method	35
3.3 Microarray Simulation Process	41
3.4 Microarray Simulation in 4×4 Blocks	46

3.5 Microarray Simulation in One Block	48
CHAPTER FOUR MICROARRAY IMAGE SEGMENTATION METHOD	49
4.1 Active Contours Without the Edges Method.....	49
4.2 Advantages of Active Contours Without the Edges Method	52
4.3 Segmentation Based Contours Method	54
4.3.1 Reduce the Length Constraint	55
4.3.2 Using a Fast Algorithm.....	55
4.3.3 Using a Higher Order Finite Difference Scheme	56
4.4 Apply Segmentation Based Contours Method on Affymetrix Image	57
CHAPTER FIVE EXPERIMENTAL RESULTS	61
CHAPTER SIX CONCLUSIONS.....	118
APPENDIX A SOURCE CODE FOR SEGMENTATION METHOD	120
APPENDIX B SOURCE CODE FOR WRITING IMAGE TO DAT FILE.....	127
APPENDIX C SOURCE CODE FOR WRITING OUTPUT TO CEL FILE.....	130
REFERENCES	133

LIST OF TABLES

Table 2.1:	Pixels matrix for one spot cell	20
Table 2.2:	Part of header information for DAT file.....	27
Table 2.3:	Part of format description for CEL file	29
Table 3.1:	Data sets for simulation model	33
Table 3.2:	Data input format	34
Table 3.3:	List of noise parameters	38
Table 3.4:	List of manufacturing parameters	39
Table 3.5:	List of hybridization parameters	40
Table 3.6:	List of scanning parameters	41
Table 4.1:	Ground truth pixels in one cell.....	59
Table 4.2:	Segmentation results from GCOS.....	59
Table 4.3:	Segmentation results from SBC.....	59
Table 4.4:	Intensity results comparison	60
Table 5.1:	System and MATLAB information.....	63
Table 5.2:	Preliminary comparison for one block simulated image	65
Table 5.3:	Paired t test results for $d1 = Signal_{true} - Signal_{SBC}$	66
Table 5.4:	Paired t test results for $d2 = Signal_{true} - Signal_{GCOS}$	67
Table 5.5:	Two sample t test for $D1$ and $D2$	68
Table 5.6:	Quartiles summary information	70

Table 5.7: Different metrics comparisons for Canine_a one block simulated image	71
Table 5.8: Standard error of performance and R squared	74
Table 5.9: Sample replication size table	79
Table 5.10: Standard error performance comparison for Canine_a	80
Table 5.11: Pearson correlation comparison for Canine_a	81
Table 5.12: Paired t test for the SBC analyzed gene signal value from Canine_a.....	83
Table 5.13: Paired t test for the GCOS analyzed gene signal value from Canine_a.....	85
Table 5.14: Minkowski distance comparison for Canine_a.....	87
Table 5.15: Euclidean distance metric comparison for Canine_a.....	89
Table 5.16: Correlation distance metric comparison results for Canine_a	90
Table 5.17: Chebychev distance metric comparison for Canine_a	92
Table 5.18: Summary comparison of standard error of performance and correlation	94
Table 5.19: Summary comparison of paired t test for SBC for Canine_a	94
Table 5.20: Summary comparison of paired t test for GCOS for Canine_a	95
Table 5.21: Summary comparison of clustering distance metrics for Canine_a	95
Table 5.22: Two sample t test for the SBC averaged analyzed gene signal value	96
Table 5.23: Two sample t test for the GCOS averaged analyzed gene signal value	97
Table 5.24: Two sample t test for D_{11} and D_{12}	98
Table 5.25: Quartiles comparison information for D_{11} and D_{12}	99
Table 5.26: Comparison for averaged analyzed gene signal for Canine_a group.....	101
Table 5.27: Summary comparison for eight groups simulated images.....	108
Table 5.28: Paired t test for the SBC average analyzed gene signal value	109
Table 5.29: Paired t test for the GCOS average analyzed gene signal value	110

Table 5.30: Two sample t test for the SBC average analyzed gene signal value	111
Table 5.31: Two sample t test for the GCOS average analyzed gene signal value.....	111

LIST OF FIGURES

Figure 1.1: Building block of DNA from [1]	2
Figure 1.2: DNA helix structure from [1]	2
Figure 1.3: DNA transcription from [1]	3
Figure 1.4: Central dogma of molecular biology from [6]	4
Figure 1.5: cDNA microarray image	5
Figure 1.6: Affymetrix microarray image	6
Figure 1.7: cDNA microarray experiment from [13]	8
Figure 1.8: Affymetrix microarray chip from [12]	10
Figure 1.9: Probe level design in Affymetrix microarray from [16]	11
Figure 1.10: Affymetrix microarray experiment process from [17]	12
Figure 2.1: Probe set structure from [28]	21
Figure 2.2: Background subtractions from [28]	22
Figure 2.3: GCOS microarray image analysis flow	26
Figure 2.4: DAT file structure shown in MATLAB	27
Figure 2.5: CEL file structure shown in MATLAB	28
Figure 2.6: CDF file structure shown in MATLAB	30
Figure 2.7: CHP file structure shown in MATLAB	30
Figure 3.1: GCOS platform	34
Figure 3.2: Simulation steps	35

Figure 3.3: Microarray simulation process	42
Figure 3.4: Segmented image by SBC	44
Figure 3.5: Original microarray image for Canine_a genome	47
Figure 3.6: Simulated microarray Canine_a image in 4×4 blocks	47
Figure 4.1: Affymetrix microarray image segmented by SBC	58
Figure 5.1: Simulated 16-block image for Canine_a Genome	63
Figure 5.2: Simulated one block image for Canine_a Genome	64
Figure 5.3: Boxplot for $D1$ and $D2$	69
Figure 5.4: Cluster Tree for Canine_a one block simulated image	73
Figure 5.5: Regression plot using SBC signal as independent variable	75
Figure 5.6: Regression plot using GCOS signal as independent variable	76
Figure 5.7: Residual plot I using SBC signal as independent variable	77
Figure 5.8: Residual plot II using GCOS signal as independent variable	77
Figure 5.9: Boxplot for the absolute value of $D11$ and $D12$	99
Figure 5.10: Clustering tree plot for the average analyzed gene signal value	102
Figure 5.11: Regression plot for average SBC signal	104
Figure 5.12: Regression plot for average GCOS signal	104
Figure 5.13: Residual plot from average SBC	105
Figure 5.14: Residual plot from average GCOS	106
Figure 5.15: Clustering tree for Bovine_a	112
Figure 5.16: Clustering tree for Bovine_b	113
Figure 5.17: Clustering tree for Vitis_a	113
Figure 5.18: Clustering tree for Vitis_b	114

Figure 5.19: Clustering tree for Yeast-1	114
Figure 5.20: Clustering tree for Yeast-2.....	115
Figure 5.21: Clustering tree for Canine_a.....	115
Figure 5.22: Clustering tree for Canine_b	116

ACKNOWLEDGEMENTS

The writer of this dissertation, Yuan Cheng, owes great appreciation to many people at Louisiana Tech University. My greatest gratitude goes to my academic advisor, Dr. Mihaela Paun, for her precious guidance, generous encouragement and support to finish my research work. It is my great honor to be her doctoral student. This dissertation could not have been completed without her help and suggestions. I would also like to thank Dr. Weizhong Dai, Dr. Raja Nassar and Dr. Andrei Paun for their warmhearted help. From their courses, I gained a deep understanding of Mathematics, Statistics and Computer Science, which have been applied to my research work. Sincere acknowledgement is further extended to Dr. Bogdan Strimbu for his kind suggestions and guidance as a member of my advisory committee.

To my friends and family, thanks for your company and for your great support of my living and learning in the United States. I would like to thank my father, Wenzhang Cheng, my mother, Jie Ji and my husband, Xiang Li, for their love, understanding and endless encouragement. Finally, I would like to express my appreciation to all my friends.

CHAPTER 1

INTRODUCTION OF DNA AND DNA MICROARRAY

In this chapter, an overview of the DNA microarray on the molecular biology level, aiming at providing the appropriate background for understanding the microarray segmentation problem will be presented.

1.1 DNA

All living cells on earth store their hereditary information in double-stranded molecules of DNA from Molecular Biology of the Cell [1]. These double-stranded molecules of DNA contain four types of monomers, which form the long paired chains based on the complementary rule. A (adenine), T (thymine), C (cytosine), G (guanine) are strung together, encoding the hereditary information. By interpreting this sequence information from a DNA strand, scientists are capable of deciphering the hereditary information contained in cells.

In 1869, Friedrich Miescher first discovered the nucleic acid from his experiment. In 1952, Alfred Hershey and Martha Chase first established that DNA was the molecules carrying the hereditary information for all living cells [2]. In 1953, James D. Watson and Francis Crick first elaborated and presented the DNA double-stranded molecular model [3]. This double helix model brought a significant impact on understanding the DNA transcription and translation process. In [1], the nucleotide was introduced, consisting of

two sections. One part is called the deoxyribose with a phosphate group in Figure 1.1. The other part is called the base, which is either A, G, C or T.

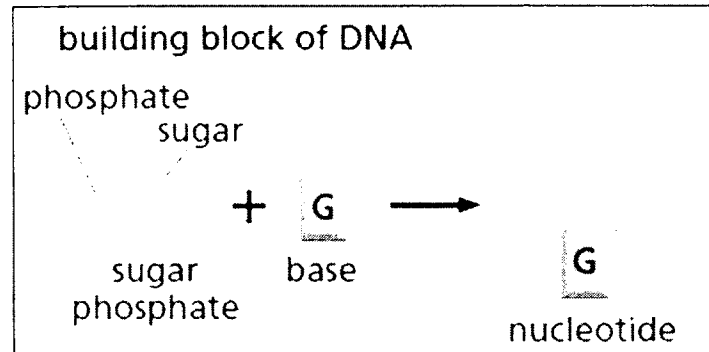


Figure 1.1 Building block of DNA from [1]

Next, several nucleotides are connected together by the phosphate group, which constructs the DNA strand. These two DNA strands are synthesized according to the complementary structures of the bases, where A binds to T, and G binds to C. After this synthesis process, two DNA strands twist on each other to form the double helix shown in Figure 1.2.

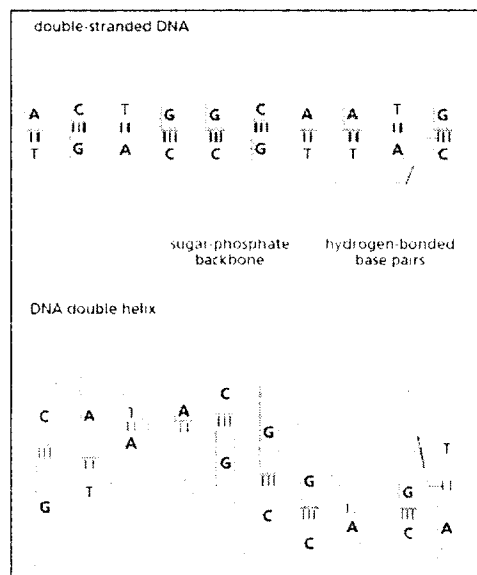


Figure 1.2 DNA helix structure from [1]

1.2 DNA Transcription and Translation

In order to carry the genomic information, the DNA sequence must undergo the process of replication and transcription with the help of RNA (ribonucleic acid) and protein. RNA has the similar intermediary structure with the DNA strand stored in cytoplasm. There are, however, some differences in RNA compared with DNA. In RNA, the backbone is formed by ribose instead of deoxyribose. In addition, those four bases are the same with one exception: where U (uracil) replaces T (thymine) [1, 3]. Thus, in RNA, A is paired with U and C is paired with G.

This process starts from the transcription, as the DNA sequence is treated as the template for RNA synthesis. The genetic information in a specific sequence is transferred into a complementary special sequence of messenger RNA (mRNA) as seen in Figure 1.3. Three bases in RNA transcripts are considered as the genetic code called “codon.” Several of these triplet codons guide the synthesis of polymers of protein, which is the translation process. Thus, from DNA to protein, hereditary information is deciphered.

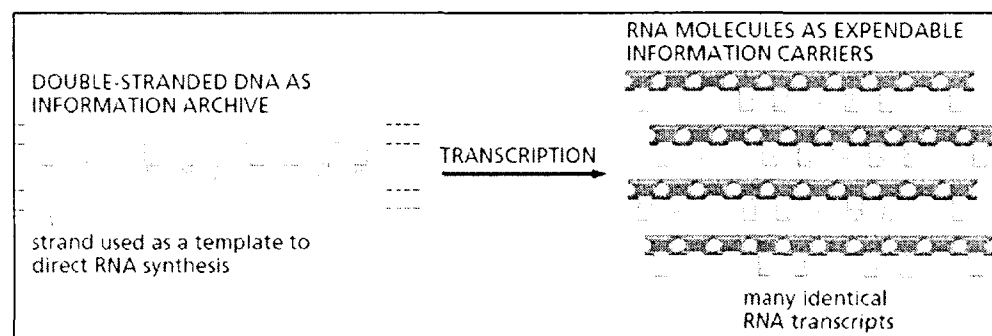


Figure 1.3 DNA transcriptions from [1]

Each genetic code is read out by a small sequence of RNA molecules called the “transfer RNA.” It matches up the genetic code, which guides the order of amino acids to form the protein molecules. There are $4^3=64$ total possible codons. Each mRNA starts

with the beginning codon AUG and ends up with the ending codon UAA, UAG, or UGA. All the other sequences between the starting codon and ending codon are the Open Reading Frame (ORF), which stores all the genetic information from DNA sequence. All of this process is shown in Figure 1.4.

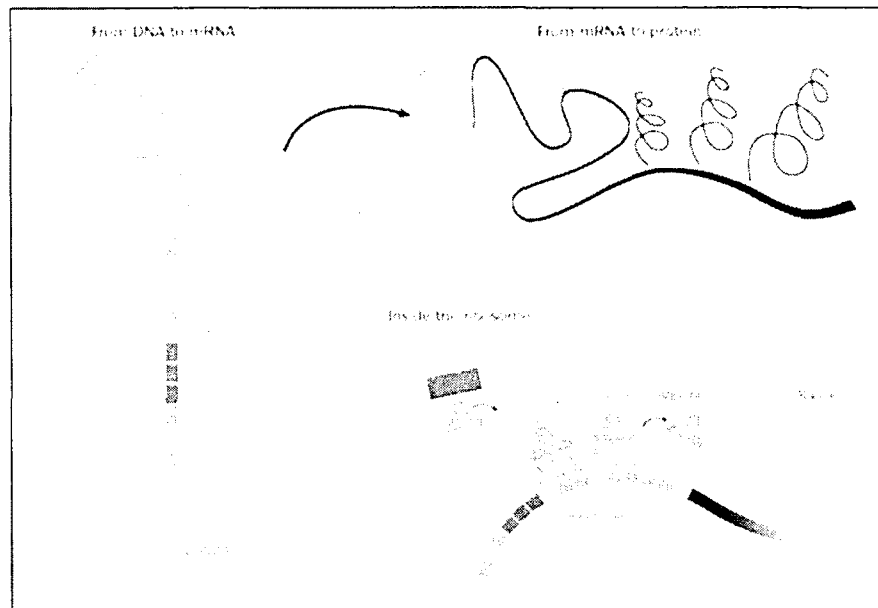


Figure 1.4 Central Dogma of Molecular Biology from [4]

Each DNA sequence experiences three stages: the replication, the transcription and the translation, and genetic information is passed down through this process. The subsequence of DNA that is transferred into protein is called a “gene” [5]. Thus, this process is called the “gene expression.” In the genetics field, gene expression is the most significant and basic foundation for transforming the genotype to the phenotype. Different organism phenotype is caused by controlling the different properties of the gene expression [6]. By using DNA microarray technology, scientists are able to monitor and manage thousands of genes’ expression simultaneously. Therefore, it is an important method allowing us to understand and analyze gene expression efficiently.

1.3 DNA Microarray

DNA microarrays are part of a new class of biotechnologies allow the monitoring of thousands of genes expression levels simultaneously. It is extremely important in the pharmaceutical and clinical field since they can help the scientists get a better understanding on genome regulation and interaction [7]. There are two basic DNA Microarray techniques currently used nowadays: spotted microarray image (Complementary DNA Microarray) shown in Figure 1.5 for cDNA microarray and oligonucleotide microarray image shown in Figure 1.6 for Affymetrix microarray. Among these techniques, the high density oligonucleotide microarray technology provided by Affmetrix GeneChip Company [8] has been widely utilized by thousands of researchers because of its high sensitivity and accuracy [9].

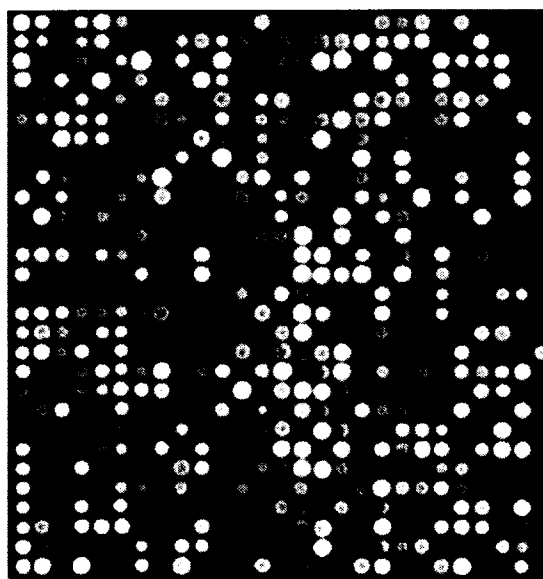


Figure 1.5 cDNA microarray image

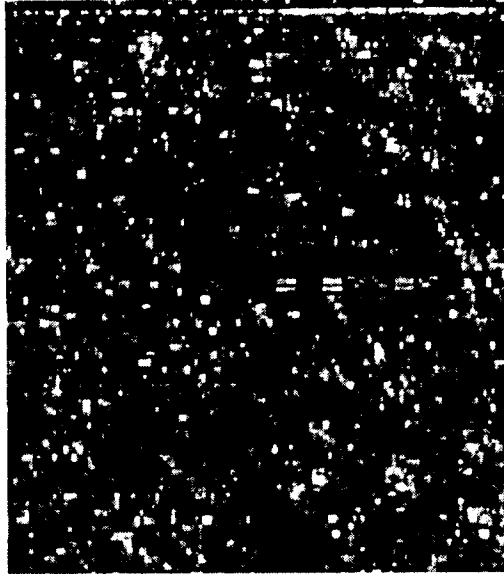


Figure 1.6 Affymetrix microarray image

The microarray technique is originated from the Southern Blotting technology. The Southern Blotting technique is mainly used in molecular biology to detect a specific sequence of DNA in DNA samples. This technique has two important characteristics; one is the transportation of the DNA fragments, and the other one is the probe hybridization of the DNA fragments. In Southern Blotting, DNA strands are first cut into smaller fragments by using restriction endonucleases. Next, these tiny DNA fragments are separated by size by gel electrophoresis method. After classification and separation, the DNA fragments are transferred to a sheet of nitrocellulose or nylon membrane. This membrane is exposed to a single DNA hybridization probe with specific sequence. In addition, this DNA sequence is labeled in order to be easily detected. After hybridization, extra DNA fragments will be washed off, and hybridization fragments will be visualized on film. In this way, the specific DNA sequence is detected.

Though the Southern Blotting is very effective for detecting the DNA special sequence, it is not a convenient method. The main disadvantage for the Southern Blot

method is that it is rather time consuming and labor-intensive. Thus, microarray technology is innovative, because it can manipulate and manage thousands of genes at the same time. In 1995, the first DNA microarray was proposed for gene expression analysis [10].

A common microarray experiment contains the following six steps:

- Experiment preparation. Two samples are selected as the treated sample and the untreated sample. For example, one sample is from a normal tissue, and the other sample is from a tumor tissue.
- Interest Nucleic acid separation and purification. For example, the RNA sequence for expression analysis or the DNA sequence for the comparisons.
- Reverse transcription is performed to obtain the labeled sequence. For example, the mRNA is reverse transcribed to cDNA. Also, a label is added in this process through molecular combination.
- The cDNA sequence is mixed and hybridized in the solution. Next, the mix is denatured and spotted on a microarray, which could be a gene chip or a glass microarray.
- The microarray is scanned by a special laser scanner, which can detect the label quantitatively and qualitatively.
- Microarray image and raw data is generated after the scan process is performed.

1.3.1 cDNA Microarray

The cDNA microarray isolates the RNA sequence from both the control sample (normal sample) and the experiment sample (diseased sample). Next, it operates the reverse transcription process, which allows it to convert the RNA sequences of interest

into cDNAs. After the reverse transcription, the cDNAs will be further labeled with fluorescent probes, Cy3 for control sample and Cy5 for experiment sample. The Cy3 is in a green channel with 530nm wavelength, and Cy5 is in a red channel with 630nm wavelength [11]. When finishing the labeling process, cDNA microarray is scanned both at the ~540nm and ~630nm for each channel correspondently. Two 16-bit monochromatic images are generated after scanning, which are Red and Green images. In these two images, each spot represents a specific gene [12, 13].

Normally, a cDNA experiment [13, 14] consists of the steps illustrated in Figure 1.7.

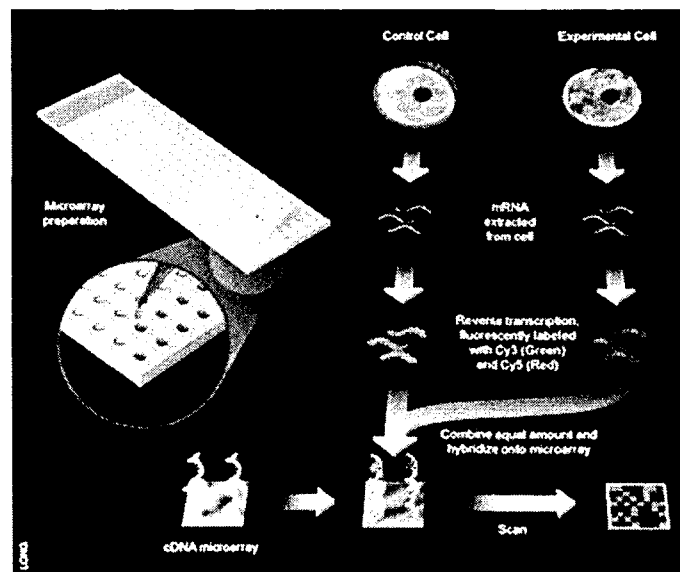


Figure 1.7 cDNA microarray experiment from [13]

- In the experiment preparation step, the normal sample and the experiment sample are selected.
- In the isolation step, the RNA sequences of interests are extracted and purified.

- In the reverse transcription step, the RNA sequences are reversely transcribed into cDNA sequences.
- In the hybridization and label step, the cDNA is labeled with a fluorescent dye. Next, the labeled cDNA sequence is hybridized. After full hybridization, extract DNA sequences will be washed away if they were not hybridized at all.
- In the scanning step, the microarray will be scanned in the two channels.
- In the data extraction step, intensity data of each spot will be extracted for the subsequent analysis.

1.3.2 Affymetrix Microarray

The Affymetrix microarray technique (Figure 1.8) is originated from late 1980s by Stephen Fodor together with other scientists. Fodor at all introduced the semiconductor technique for biological setting in microarray fabrication process. This process helped to construct a system to measure more and more various mRNA sequences in one sample. In addition, Affymetrix microarray introduced small oligonucleotide sequences (probes), containing 25-nucleotides located variously in their sequence composition. This is an impressive characteristic compared to the cDNA microarray, which uses single and long probes to detect the transcript of interests because small probes could bring a better discrimination between similar related transcripts over long oligonucleotides, especially when mRNAs are highly abundant. Hence, we mainly focused our research interests on Affymetrix microarray image analysis.



Figure 1.8 Affymetrix microarray chip from [12]

Probe sets are designed for each mRNA sequences [15]. Each gene normally consists of 11 to 20 different probes, which corresponds to a single transcript at different locations. For Affymetrix microarray, it usually has tens of thousands of different probe sets. This feature makes the Affymetrix microarray more desirable than cDNA microarray, since it could allow scientists to monitor and manipulate such amounts of genes at the same time.

Another significant characteristic is that Affymetrix introduces the Perfect Match (PM) and the Mismatch (MM) in a pair into the probes as shown in Figure 1.9. In other words, each probe pair consists of two probes, PM and MM. These two probes are exactly the same, except for the one base in the middle. For example, PM has 25-nucleotides, which are perfectly hybridized to the mRNA sequences; whereas, MM has the same 25-nucleotides, but there is only one base in the middle of the 25 bases that is different from what the PM has. Each PM should be uniquely different from each other.

In this case, false signals transcription caused from similar complete sequences were completely eliminated, and MM was used to help scientists to learn and control the unspecific signal and background signal.

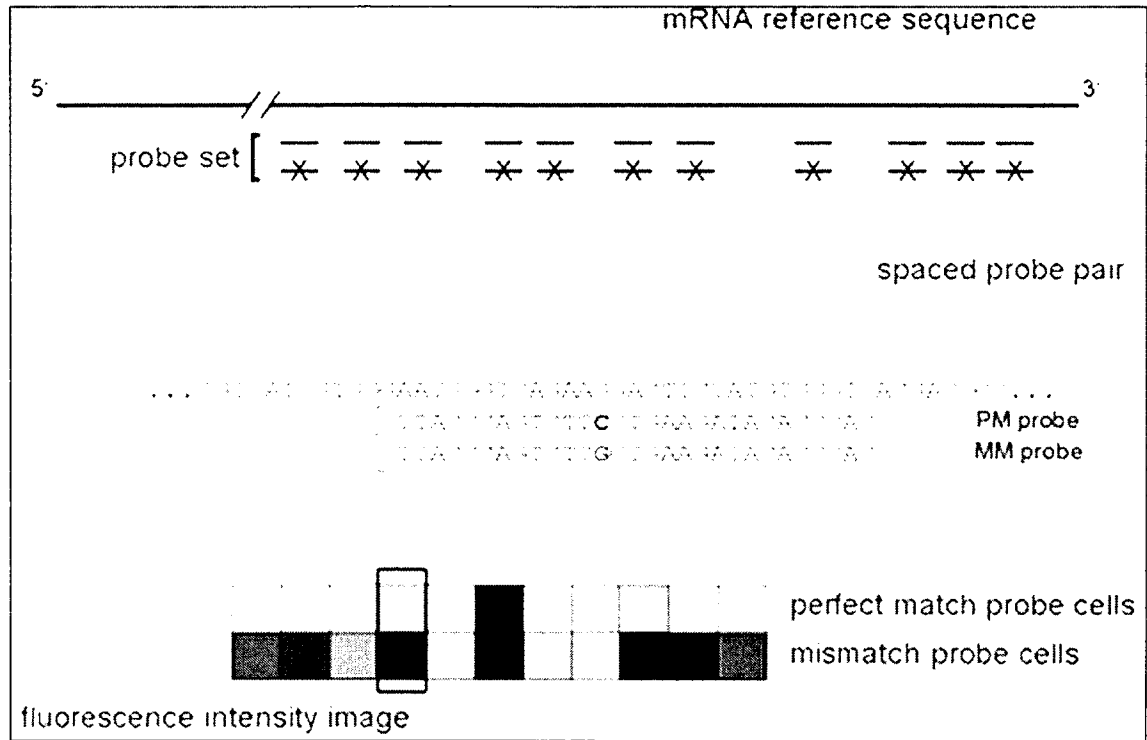


Figure 1.9 Probe level design in Affymetrix microarray from [16]

Normally, an Affymetrix experiment contains the following steps. This whole process shown in Figure 1.10 usually requires two and a half days:

- First, the sample of interest is selected.
- Next, the RNA sequences are isolated. The RNA quality is monitored and checked. After checking the quality of RNA sequences, good quality RNA sequences are labeled. These mRNAs experience the reverse transcription to cDNA, which is labeled by *In Vitro* Transcription (IVT).

- Next, this mixture is injected into the microarray platform. Hybridization is performed on the gene microarray platform under specific temperature and hours.
- After complete hybridization, the chip is scanned by a special laser, generating the Affymetrix microarray image in 16-bit gray level.
- Finally, the intensity of each pixel on the chip is recorded according to the emission of the fluorescent dye.

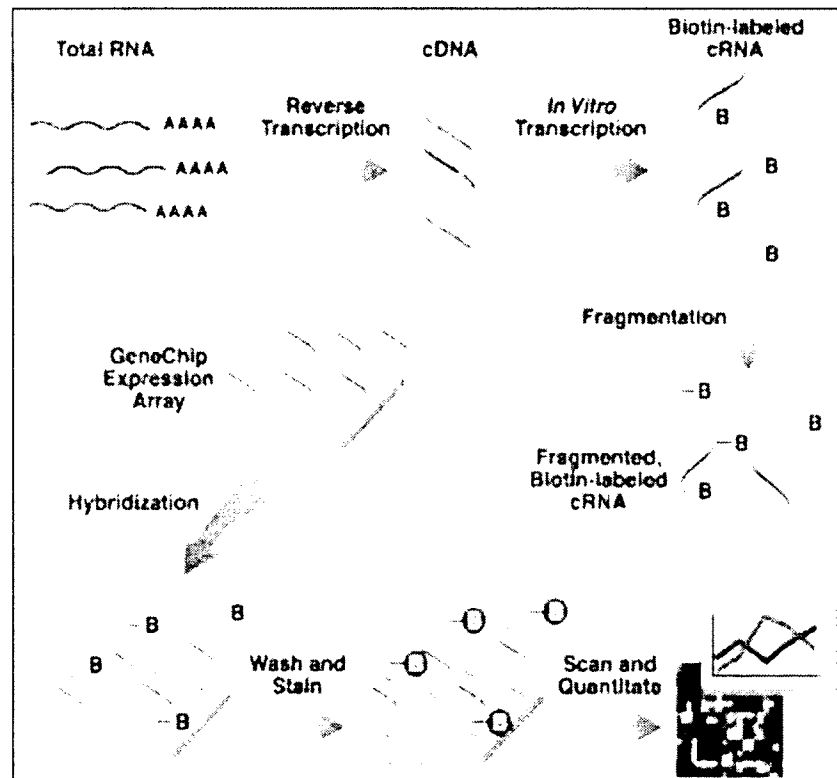


Figure 1.10 Affymetrix microarray experiment process from [17]

There are several microarray types developed by Affymetrix. They are different in many aspects, such as different emphasis on genes, exons or genome wide transcriptions and different use of mismatch probes on different number of probes.

The standard expression array is the most common array used in the public research area, which could be canine, rat, human, fly, yeast, bacteria and plant. Such probe arrays are available as public resources at UniGene, GenBank, dbEST and so on. The microarray data for this dissertation mainly came from this standard expression array database.

The exon array could provide the gene expression information based on the exon level. From this point of view, the splicing patterns could be clearly monitored and learned. It is known that not all the DNA sequence may be translated into a protein. After generating the mRNA, there is an important step that removes the non-coding sections in mRNA. These non-coding sequences are referred as to "introns." The rest of the exons are constructed together in different ways resulting in various genes. This whole process is called "splicing." It plays a significant role in the human genome system, because different splicing and construction of the exons will contribute to completely different proteins.

The gene array contains more up-to-date genome annotations for human and mouse. Hence, it is more accurate compared to the standard array. It is usually little smaller than the standard array since it does not carry any mismatch probes but the 5-micron feature. This array is the next generation of standard arrays. It begins to include a large amount of perfect match probes for each gene and to drop all the mismatched probes. Another impressive characteristic this array has is that it removes the 3'-bias end of each transcript. Instead, it uses 26 different probes to cover the whole transcript. Removing this 3'-bias end will provide more accurate gene information when alternative splicing happens in 3'-end and so on.

The tiling array only covers several organisms such as human, mouse and yeast. The tiling array uses 25-mer probes which are evenly located every 35 bases with around 10 bases as the gap between each probe. It only uses the evenly located probes on the non-repetitive part of the genome sequence rather than using the probes which corresponds to the relevant gene expression sequences. This type of array is widely used in transcript elements mapping and protein binding identification.

CHAPTER 2

INTRODUCTION OF AFFYMETRIX MICROARRAY IMAGE

Over the last decade, the microarray biotechnologies have become increasingly important in the biomedical research field, since they are capable of monitoring the expression levels of thousands of genes simultaneously. This quality of the technology that allows researchers to access such a large number of genes simultaneously while the traditional methods are limited in the number of genes that can be researched at one time, sparked the interests of scientists in researching and improving their understanding of genomic regulation and gene interaction. The DNA microarray technology has provided the scientific community with a tool to be used in understanding the basic aspects of life development and especially in exploring genetic causes and anomalies occurring in the human body.

The microarray applications currently are very wide; one of the first applications of microarrays was genome sequencing analysis using hybridization, tissue microarrays used in the study of cancer, including the molecular profiling of tumor specimens and of the applications determining gene copy number. Drug discovery is one of the largest aspects. The microarray's capabilities make them a perfect candidate for various stages of drug discovery, validation and clinical studies. Other applications of microarrays are in DNA computing, bioinformatics, and data mining, where the microarrays are required

tools for solving computational problems, analyzing huge amounts of data with similar characteristics, by using diverse analytical methods: Bayesian methods, neural networks, clustering, multivariate statistical analysis, and information retrieval [18].

Last, but not the least, and the direction where our interest lies was the gene expression analysis, with the goal of gene discovery and the possibility of using these results in monitoring and detecting the changes in gene expression from different cells. There are already chips with arrays of many types of genes such as human or species like rat, mouse and *Escherichia coli* and more. The Affymetrix Company manufactures chips for analysis of DNA microarrays, chips that scientifically match significant parts of human and non-human genomes.

The method developed in this dissertation aimed to provide a better segmentation method compared to the ones currently used. We expected that the improvement could lead the way to a quantitative feature of the DNA arrays. Such results would impact directly the many fields that use DNA arrays; the most important impact will be in a better prediction of genes that activate different diseases. We looked to provide a stepping stone towards quantitative results from DNA array experiments (at the moment we receive rather qualitative signals of the gene-disease relationships from such experiments). With the advancement of the hardware in digital photography and the processing/manipulation of cells, we fully expected the images obtained after the DNA arrays experiments to reach much higher resolutions and have significantly lower noise in the signals and, thus, the proposed algorithm to lead to dramatic improvements as opposed to the currently used Affymetrix segmentation method. This new method will lead, in turn, to quantitative results which would have a significant impact in shedding

light on the cellular processes. This segmentation of a picture is one of the three important steps in microarray image processing, together with spot gridding and information extraction. It directly affects the accuracy of gene expression analysis in the data mining process that follows [19, 20, 21,22].

2.1 Overview of Affymetrix Microarray Image Analysis Methods

In a microarray experiment, the image analysis could be viewed as one of the most crucial steps of processing, which could have a large impact on the subsequent data analysis, such as clustering or identification of different gene expression levels. During the microarray experiment process, usually two samples of (a healthy sample versus diseased sample) microarrays are hybridized with complementary DNA labeled with usually two different fluorescent dyes, Cy3 and Cy5. Next, the hybridized microarrays are processed by a microarray scanner to visualize the red and green fluorescence. In other words, the hybridized microarrays are imaged at each spot. In this way, a raw 16-bit TIFF image is obtained. The fluorescence intensity of each spot represents the hybridized level of the sample. Therefore, analyzing the microarray image is one of the most important steps in a microarray experiment. The microarray image analysis can be described as a three step process [11].

The addressing or gridding step is performed to find the exact location of each spot and to assign the coordinates to each spot. The purpose is to define the spot region based on the microarray image layout information. After gridding on the microarray image, each spot is assigned with a geometric location, which is a square or a rectangle. The center of each spot and the region between the center and the boundary are used to detect the object curve within the square. However, in real microarray experiments, the

misalignment usually happens. For example, the microarray chip may not be arranged exactly in the center during the scanning process. Or the sub-array chip may be shifted subtly during the hybridization process. All these issues will be considered and handled in our image analysis process [23].

The segmentation process was the main concern in our research. In the data acquisition process, the segmentation of spots is the one of the most challenging tasks and has a significant impact on the gene expression analysis process that follows. The task is to identify the pixels either as foreground (within the printed spot) or as background (beyond the printed spot). In this sense, the image segmentation is a process that divides an image into two mutually exclusive regions: foreground and background. The key point at this stage is to get the exact shape of the foreground pixels. This exactness does not usually happen in the previously used segmentation methods in the literature. In this way, the foreground and the background regions are classified and the fluorescence intensity for the spot is calculated according to this classification.

However, the microarray images are hard to segment since they have highly varying image contrast different from experiment to experiment and also contain a high level of background noise and image artifacts. The segmentation step is further complicated by the non-uniform shape and surface intensity distribution in the experiment pictures.

The intensity extraction follows next. The value of each pixel represents the expression level of hybridization for that specific DNA sequence. Hence, the next step in processing DNA arrays is to calculate foreground fluorescence intensity, background intensity for each spot based on the results from the segmentation. In addition, at this step

some other calculations are performed such as the possibility of random hybridization, noise and quality measures. Many methods use the mean or the median pixel value as the whole value of the foreground spot mask. Additionally, these methods make use of the statistical tests to measure the background intensities relative to the foreground intensities, [11, 24]. Therefore, the result produced in the segmentation step is extremely important in the subsequent image analysis process.

In recent years, several methods have been developed to segment microarray spots and have been incorporated into commercial microarray image analysis software packages [25, 26].

The last step of the process is the intensity to gene expression signal value step. The intensity only represents the abundance of hybridization for target interested sequence in each spot, not for each gene. The last step is to summarize the intensity values into the signal value, which represent the expression level for each gene.

2.2 Affymetrix Microarray Image Analysis Process

In Affymetrix, all microarray image analysis is accomplished in Gene Chip Operating Software (GCOS) produced by Affymetrix Company. It provides a set of comprehensive analysis tools for data management and control in the processing of microarrays. The software summarizes all the probe intensity values and combines them into gene signal values after image gridding process. Besides these characteristics, this software enables data analysis to be customized, automated and integrated with various laboratory systems.

First, segmentation and intensity extraction are performed by the built-in GCOS. In Affymetrix microarray image, each probe spot cell contains $n \times n$ pixels depending on

the experiment design. After identifying the position of each probe, GCOS omits the outer boundary pixels. Only the inner pixels are included and considered to be within the foreground area. The method chooses the 75th percentile of the rest of the pixels in the square to represent the intensity for each probe. Table 2.1 is the pixels matrix of one spot in microarray image.

The outer highlighted pixels are dropped off in Table 2.1. The 75th percentile of remaining inner pixels is recorded as the intensity value for the spot. The reason why GCOS omits the outer pixels is that it is believed that such pixels are not reliable and may carry some noise and errors, for they may be located by the misalignment in the scanning process, or they may be influenced by the neighboring probes which have large amount of emission. These intensity values are recorded into the CEL file.

Table 2.1 Pixels matrix for one cell

256	166	413	301	309	473
294	256	166	234	204	286
166	204	166	256	196	174
196	369	279	458	219	264
181	166	241	286	451	376
234	219	249	376	166	219

In Affymetrix, GCOS chooses the 75th percentile of the interior pixels of each probe cell as the intensity for each probe. Research from Harry Zuzan [23] shows that with the increasing of the pixel values, the variance would become unstable when choosing the 75th percentile as the probe intensity. In addition, this method is not robust enough when dealing with different qualities of cells. Hence, in this dissertation, we introduced an intensity extraction algorithm named as “Segmentation Based Contours” method, which is a modified version of the ACWE method [27]. The ACWE model will

detect a curve which is constrained in a specified image without any gradient calculation but minimizing the energy based function. Thus, the ACWE presented by Tony F. Chan and Luminita A. Vese [27] has more advantages in finding objects within a microarray image in which boundaries are not defined by gradient. We will present more details for this ACWE method and its modified method SBC.

Next, intensity values for each spot are combined transformed into the gene expression signal value. The Affymetrix GCOS software uses the MAS5 algorithm to calculate the signals from intensities [28, 29, 30]. The Genechip array designed by Affymetrix Company is the probe level design array. A Genechip array contains many probe cells, where each probe cell is related with a specified target sequence probe. Probe spots are tiled into probe pairs with a Perfect Match (PM) and a Mismatch (MM). There is only one base in the middle changed in MM sequence, where it does not follow the complement rule. All related PM and MM together consist of a probe pair, related to a whole expressed gene transcript shown in Figure 2.1.

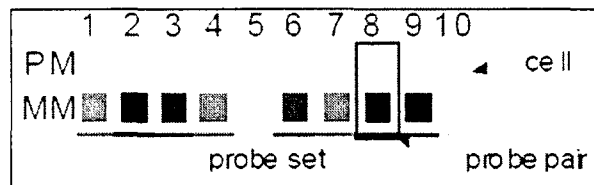


Figure 2.1 Probe set from [28]

Before calculating the signals, the MAS5 conducts global background subtraction and noise correction based on the raw intensities in CEL file. This background adjustment noise correction could even out the background errors caused by different cell locations. First the whole chip is divided into 16 rectangular zones as shown in Figure 2.2. Next, the

distance d_k is computed between the chip coordinate (x, y) and the center of each sub zone. Next, the weighting factor W_k is obtained based on d_k .

$$W_k(x, y) = (d_k^2(x, y) + \text{smooth})^{-1}, \text{smooth} = 100. \quad (2.1)$$

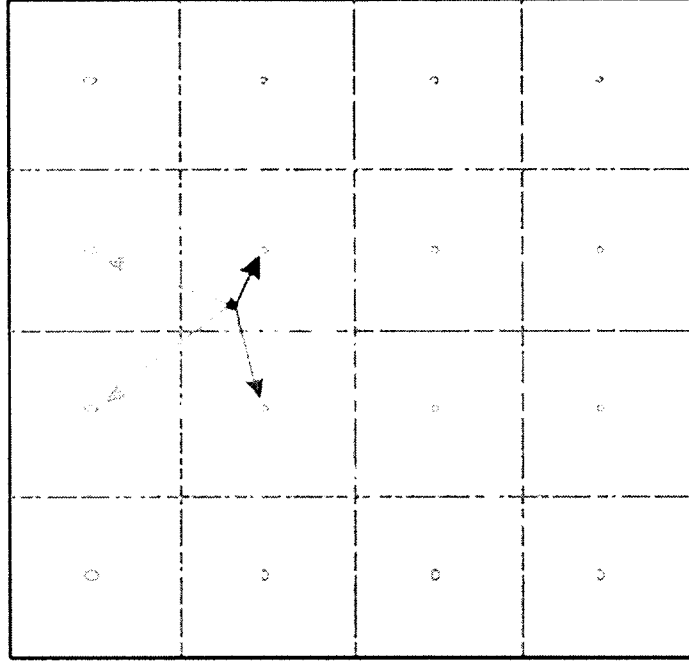


Figure 2.2 Background subtractions from [28]

Based on such distances d_k and W_k together and a constant b , a weighted sum is obtained, which is used for each probe cell (x, y) .

$$b(x, y) = \frac{1}{\sum W_k(x, y)} \sum b Z_k W_k(x, y). \quad (2.2)$$

Now, MAS5 computes the adjusted intensity value by shifting the original intensity value down based on the local background b . This b is considered to be the noise correction. For noise correction, local noise factor n is obtained based on the standard deviation in each sub zone.

$$n(x, y) = \frac{1}{\sum W_k(x, y)} \sum n Z_k W_k(x, y). \quad (2.3)$$

Next, an initial threshold and a floor are specified such that no adjusted intensity value is below that threshold. The adjusted intensity is calculated from subtracting this local background.

$$A(x, y) = \max(I'(x, y) - b(x, y), \text{NoiseFrac} * n(x, y)),$$

$$\text{where } I'(x, y) = \max(I(x, y), 0.5), \text{NoiseFrac} = 0.5. \quad (2.4)$$

After adjusting the background for each cell, the MAS5 algorithm uses the new intensities to calculate the signal for each probe as follows:

1. An ideal mismatch value is calculated and subtracted to adjust the PM intensity. T_{bi} is the one step biweight algorithm. In an Affymetrix microarray, the reason why it introduces the MM probe is that it comprises the background noise and cross hybridization, which will bring impact on the PM probe. Hence, the ideal possible MM value should be less than PM value. However, in some cases, the MM value is larger than PM value. This result indicates that this MM value is a physical impossible measurement. It cannot be used to calculate the signal value. Instead, an adjusted value should be estimated based on the whole gene probe set level. MAS5 uses the one step biweight algorithm to calculate this specific background estimation SB_i .

$$SB_i = T_{bi}(\log PM_{ij} - \log MM_{ij}), j = 1, 2, \dots, n_i. \quad (2.5)$$

The one-step biweight algorithm begins by calculating the median M for a data set with n values. In the signal measurement, this data set consists of the $\log(PM - IM)$ probe values of a probe set. Next, we calculate the absolute distance for each data point from the median, and calculate S , the median of the absolute distances from M . The median absolute deviation, MAD, is an initial measure of spread.

For each data point i , a uniform measure of distance from the center is given.

$$u_i = \frac{x_i - M}{C \times S + \varepsilon}. \quad (2.6)$$

Next, calculate the weight by the bi-square function.

$$w(u) = \begin{cases} (1 - u^2)^2, & |u| \leq 1, \\ 0, & |u| > 1. \end{cases} \quad (2.7)$$

Finally, the corrected values can be calculated by the one-step w -estimate.

$$T_{bi}(x) = \frac{\sum w(u_i)x_i}{\sum w(u_i)}. \quad (2.8)$$

If the background estimate SB_i is large, the related values in the probe set are reliable. This SB_i is capable of constructing the ideal adjusted mismatch IM if necessary. If SB_i is small, more of PM values are used to calculate the ideal adjusted mismatch IM . These different cases which determine the ideal adjusted mismatch IM are described as follows:

$$IM_{i,j} = \begin{cases} MM_{i,j}, & \text{when } MM_{i,j} < PM_{i,j}, \\ \frac{PM_{i,j}}{2^{SB_i}}, & \text{when } MM_{i,j} \geq PM_{i,j} \text{ and } SB_i > 0.03, \\ \frac{PM_{i,j}}{\left(\frac{0.03}{1 + \frac{0.03 - SB_i}{10}} \right)}, & \text{when } MM_{i,j} \geq PM_{i,j} \text{ and } SB_i \leq 0.03. \end{cases} \quad (2.9)$$

When MM value is less than PM value, this MM provides a reliable estimation for the probe background. When MM value is not less than PM value, this MM value is not reliable, but still provides some relevant information for the probe. If SB_i is less than or equal to 0.03, the MM value provides the least information estimation.

2. The adjusted PM intensities are log-transformed to stabilize the variance.

Given the adjusted ideal mismatch MM, probe value (PI) is calculated with the numerical stability.

$$V_{i,j} = \max (PM_{i,j} - IM_{i,j}, D), \text{ where } D = 2^{-20}. \quad (2.10)$$

Next, log-transformation is performed on probe value for each probe cell.

$$PV_{i,j} = \log(V_{i,j}), j = 1, 2, \dots, n_i. \quad (2.11)$$

Absolute expression value for each probe set is obtained by performing the one step biweight estimate algorithm.

$$SignalLogValue = T_{bi}(PV_{i,1}, \dots, PV_{i,n}). \quad (2.12)$$

3. The biweight measurement is used to calculate the robust mean of the input values. Signal is output as the anti-log of the Signal Log Value. Finally, the reported signal for each probe set is obtained.

$$ReportedSignal = nf \times sf \times 2^{SignalLogValue}. \quad (2.13)$$

2.3 Affymetrix Microarray Image Analysis Flow in GCOS

After finishing the microarray experiment, the most crucial step is to extract most reliable data information from the microarray image, obtaining the intensity value for each probe on the chip. The probe intensity is the foundation of the whole microarray image analysis because all the subsequent data analysis is based on the probe intensity value, calculating gene expression signals and so on. Thus, how to achieve more accurate probe intensity values was our main research interest. For Affymetrix microarray image, all such analysis was accomplished in GCOS. Figure 2.3 is the GCOS microarray image analysis flow chart. The gene chip was scanned after microarray experiment. The raw image information was stored in DAT file and we used GCOS to open this DAT file. Alignment gridding was automatically performed and intensity values were written in CEL file. When the intensity values were obtained, GCOS implemented the MAS5 algorithm to analyze the CEL file and related CDF file to calculate the gene signal value

for each probe set. This gene signal value was stored in CHP file and TXT file. One was in special format in CHP file. The other one was in text format in TXT file.

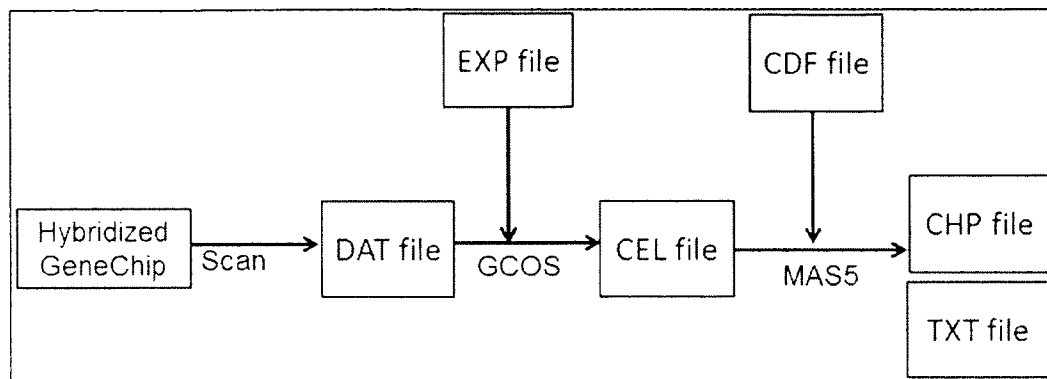


Figure 2.3 GCOS microarray image analysis flow

The DAT file shown in Figure 2.4 contains the data information of raw 16-bit (TIFF) optical image followed by the relevant header information shown in Table 2.2. It also includes that array chip layout information and experiment information, etc. The CEL file shown in Figure 2.5 contains the information for each probe cell. It includes the layout coordinates of each cell, the intensity value for each cell, the number of pixels included for each cell and the standard deviation of each cell, etc. It is written in a special format shown in Table 2.3.

```

>> I=affyread('...')
: =
      Name: 'canine_a.DAT'
      DataPath: 'E:\chengyuan\Affy\MAS Algorithm'
      LibPath: 'E:\chengyuan\Affy\MAS Algorithm'
      FullPathName: 'E:\chengyuan\Affy\MAS Algorithm\canine_a.DAT'
      ChipType: 'Canine'
      NumPixelsPerRow: 5621
      NumRows: 5621
      MinData: 12
      MaxData: 65534
      PixelSize: 1
      CellMargin: 1
      ScanSpeed: 30
      ScanDate: '12-Feb-2004 11:49:34'
      ScannerID: '81102810 M10 '
      UpperLeftX: 107
      UpperLeftY: 144
      UpperRightX: 5457
      UpperRightY: 236
      LowerLeftX: 202
      LowerLeftY: 5352
      LowerRightX: 5352
      LowerRightY: 5454
      ServerName: ''
      Image: {5621x5621 uint16}

```

Figure 2.4 DAT file structure shown in MATLAB

Table 2.2 Part of header information for DAT file

Index	Description	Type
1	Type of file, must be 0xFC.	BYTE
2	Number of pixels per line.	WORD
3	Number of lines in the image.	WORD
4	The total number of data points (pixels) in the image.	DWORD
5	Minimum pixel value in the image.	DWORD
6	Maximum pixel value in the image.	DWORD
7	Mean pixel value.	double
8	Standard deviation of the pixel values	double
9	Number of pixels per row (padded with spaces), preceded with "CLS=."	char[9]
10	Number of rows in the image (padded with spaces), preceded with "RWS=."	char[9]

```

>> I=affyread('canine_a.CEL')

I =

        Name: 'canine_a.CEL'
      DataPath: 'F:\chengyuan\Affy\MAS Algorithm\
      LibPath: 'F:\chengyuan\Affy\MAS Algorithm\
FullPathName: 'F:\chengyuan\Affy\MAS Algorithm\canine_a.CEL'
      ChipType: 'Canine'
        Date: '04-Oct-2010 22:50:43'
FileVersion: 4
      Algorithm: 'Percentile'
      AlgParams: (1x312 char)
NumAlgParams: 16
      CellMargin: 2
          Rows: 732
          Cols: 732
      NumMasked: 0
NumOutliers: 56114
      NumProbes: 535524
UpperLeftX: 307
UpperLeftY: 184
UpperRightX: 5457
UpperRightY: 286
LowerLeftX: 202
LowerLeftY: 5352
LowerRightX: 5352
LowerRightY: 5454
ProbeColumnNames: (5x1 cell)
          Probes: (535524x5 single)

```

Figure 2.5 CEL file structure shown in Matlab

Table 2.3 Part of format description for CEL file

TAG	Description
Version	The version number. Always set to 3.
TAG	Description
Cols	The number of columns in the array (of cells).
Rows	The number of rows in the array (of cells).
TotalX	Same as Cols.
TotalY	Same as Rows.
OffsetX	Not used, always 0.
OffsetY	Not used, always 0.
GridCornerUL	XY coordinates of the upper left grid corner in pixel coordinates.
GridCornerUR	XY coordinates of the upper right grid corner in pixel coordinates.
GridCornerLR	XY coordinates of the lower right grid corner in pixel coordinates.
GridCornerLL	XY coordinates of the lower left grid corner in pixel coordinates.
Axis-invertX	Not used, always 0.
Axis-invertY	Not used, always 0.
swapXY	Not used, always 0.

The CDF file shown in Figure 2.6 contains the information for each probe set gene. It includes the number of probe sets, the name of each gene probe set, the number of probe pairs of PM and MM, and the coordinates for each probe pairs, etc.

The CHP file shown in Figure 2.7 contains the experiment results created from CEL and CDF files. It includes the gene expression value for each probe set and includes the pixel resolution, etc.

```

>> I=affyread('Canine_a.cdf')

I =

        Name: 'Canine_a.cdf'
      ChipType: 'Canine_a'
       LibPath: 'F:\chengyuan\Affy\Win7_xp_share\ACWE'
 FullPathName: 'F:\chengyuan\Affy\Win7_xp_share\ACWE\Canine_a.cdf'
         Date: '29-Sep-2008 16:01:02'
         Rows: 732
         Cols: 732
 NumProbeSets: 23913
 NumCCProbeSets: 9
 ProbeSetColumnNames: {6x1 cell}
       ProbeSets: [23922x1 struct]

```

Figure 2.6 CDF file structure shown in MATLAB

```

>> I=affyread('Canine_a.CHP')

I =

        Name: 'canine_a.CHP'
   DataPath: 'F:\chengyuan\Affy\MAS_Algorithm'
    LibPath: 'F:\chengyuan\Affy\MAS_Algorithm'
 FullPathName: 'F:\chengyuan\Affy\MAS_Algorithm\canine_a.CHP'
      ChipType: 'Canine'
     AssayType: 'Expression'
         Date: '04-Oct-2010 22:51:50'
      CellFile: 'canine_a.CEL'
      Algorithm: 'ExpressionStat'
    AlgVersion: '5.1'
   NumAlgParams: 16
     AlgParams: {1x200 char}
 NumChipSummary: 6
   ChipSummary: {1x159 char}
BackgroundColors: {1x1 struct}
         Rows: 732
         Cols: 732
 NumProbeSets: 23913
 NumCCProbeSets: 0
       ProbeSets: [23913x1 struct]

```

Figure 2.7 CHP file structure shown in MATLAB

The TXT file is the text format of CHP file, which contains the same information as the CHP file. The EXP file is the text file, which contains the experiment details, such as the experiment date, time, name, scanning machine and pixel size, etc.

CHAPTER 3

MICROARRAY IMAGE SIMULATION METHOD

The Affymetrix GeneChip microarrays have become a crucial component of gene expression and genotype research for many laboratories. Data analysis remains a major challenge for the effective use of GeneChip data. There is high interest in analyzing the microarray data and improving in the existing analyzing methods.

We will compare our proposed segmentation method SBC with the method currently used by the Affymetrix. The Affymetrix GeneChip Operating Software (GCOS) is an operating system that controls Affymetrix instruments, acquires data, and executes gene expression analysis. In addition, GCOS contains an embedded database that manages both experiment information and data. The comparison will not be made in respect to the segmentation time, the Affymetrix method is by far much faster than our method, and it will concentrate on performance, in the number of genes that can be detected. We planned to run an experiment to detect active expressed genes in different organisms. Since the experiment involved detecting genes' expressions, even a small improvement in the detection rate could be crucial in determining a gene of interest.

However, due to the lack of the "ground true information," it was difficult to evaluate different intensity extraction algorithms. Therefore, we utilized an advanced microarray simulation model [31], which played a significant role in validating different

kinds of segmentation analysis algorithms. It contained all the experiment and manufacture steps for producing one microarray image in practice. It also embraced biological realistic characteristics, which could affect the microarray image quality significantly. The most important thing is that this model could provide the “ground true information” to help us have a deep understanding on how the algorithm performs. The simulation program can be downloaded from [32]. In order to simulate data carrying real genetic information, we used the analyzed intensity of real Affymetrix microarray images as data input obtained from GCOS for the simulation model.

In this chapter, we will present the data input used in the simulation model and the description of this simulation method.

3.1 Database for Simulation Model

The original Affymetrix microarray images can be downloaded from [33]. All the data on this website are the sample test data provided by Affymetrix Company as a free test online source. Some of those data are not available, Canine2.0, Chicken, Citrus, Cotton, Dros Test Yease, Focus-Ecoli, HG-U133, MG-U74, Mouse 430, etc. In our research, we utilized eight different high resolution microarray images from [33] as the data source for simulation model. Table 3.1 presents the data used in our research.

Table 3.1 Data sets for simulation model

Bovine_a:	A replicate probe array file for the Bovine Genome Array.
Bovine_b:	A replicate probe array file for the Bovine Genome Array.
Canine_a:	A replicate probe array file for the Canine Genome Array.
Canine_b:	A replicate probe array file for the Canine Genome Array.
Vitis_a:	A replicate probe array file for the Vitis Vinifera (grape) Genome Array.
Vitis_b:	A replicate probe array file for the Vitis Vinifera (grape) Genome Array.
Yeast_a:	A replicate probe array file for the Yeast (YG-S98) S98 Genome Array.
Yeast_b:	A replicate probe array file for the Yeast (YG-S98) S98 Genome Array.

Bovine_a and Bovine_b are all selected from Bovine genome. This Bovine genome array can be used to understand over 23,000 Bovine transcripts. This array is an idea microarray chip for scientists to study cattle. Researchers are able to monitor genetic mechanisms, which regulate different kinds of traits such as disease resistance, meat and dairy production, stress tolerance and so on.

Canine_a and Canine_b were selected from the Canis families' genome, which is an important model organism used for human disease study in the biomedical field. This Canine array enables researchers to interrogate 18,000 Canine genomes mRNA/EST transcripts and 20,000 non-redundant predicted genes simultaneously.

Vitis_a and Vitis_b were selected from the Vitis genome. This Vitis genome array is the first array to provide comprehensive analysis for *V. vinifera* genome and is provided as free sample test data from the Affymetrix database. There are sixteen pairs for each oligonucleotide probe set to measure the specific sequence of target genes. All sequence of target genes were selected from GenBank, dbEST, and RefSeq online gene sequence research databases.

Yeast-1 and Yeast-2 were selected from the Yeast genome. This genome array contains probe sets to detect transcripts from two most important species, which are

Saccharomyces cerevisiae and *Schizosaccharomyces pombe*. It provided the comprehensive coverage for these two species, including around 5,744 probe sets for 5,800 genes in *S. cerevisiae* and 5,021 probe sets for all 5,031 genes in *S. pombe*.

In Affymetrix, all these array files were provided in DAT file format. The researcher needed to download GCOS to open this DAT file format. Next, the raw microarray image will be shown in GCOS platform. The GCOS automatically allocated the grid alignment information from DAT file. The intensity value for each spot cell was analyzed and stored in the CEL file. When the intensity value was obtained, the researcher was capable at calculating the gene signal value by using the build in MAS5 algorithm, recorded in CHP file. The GCOS platform is shown in Figure 3.1.

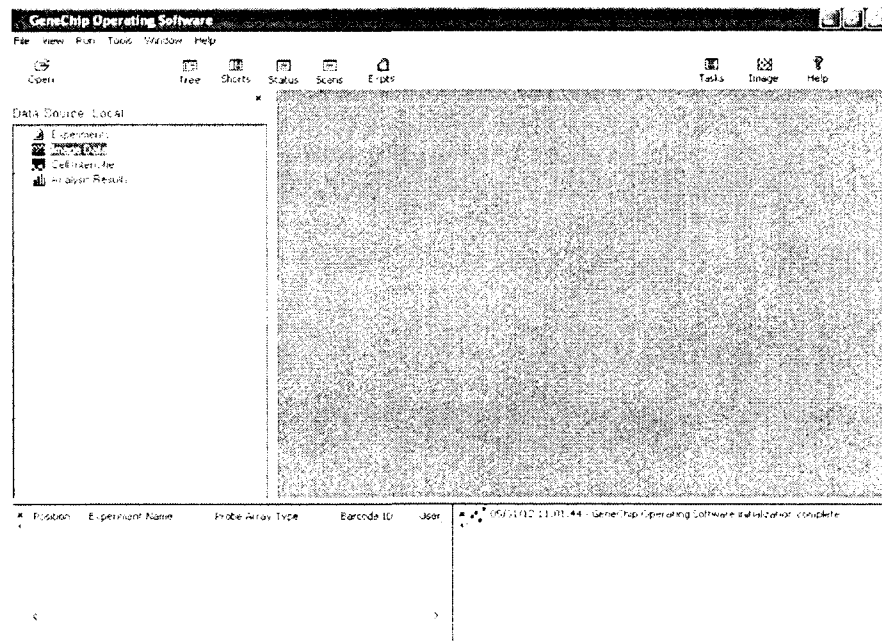


Figure 3.1 GCOS platform

3.2 Microarray Simulation Method

In order to compare the segmentation results from the two methods SBC and Affymetrix GCOS, we introduced an advanced Affymetrix microarray image simulation model. This simulation model played a significant role in validating different kinds of segmentation analysis algorithms, since it contains all the manufacture steps for microarray image. This model embraced biological realistic characteristics which could affect the microarray image quality significantly. Most important thing is that this model could provide the “ground true information” which led us have a deep understanding on how the segmentation algorithms perform.

The simulation model contained six main steps which are slide manufacturing, data input, biological noise, slide hybridization, slide scanning and image reading (Figure 3.2). By operating the model parameters, the simulation process will provide three different qualities images, which are high, normal and bad quality.

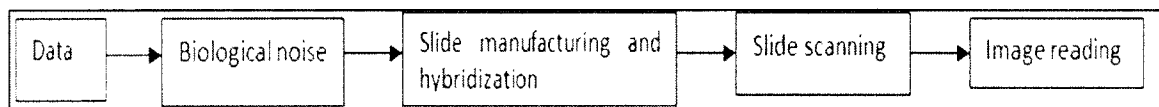


Figure 3.2 Simulation steps

Data input for the Affymetrix microarray image is the intensity for each cell in the microarray chip. In addition to the intensity data, the cell location, probe name and location and their identifiers should also be specified in a proper format by using a file input module. The requirement for the input data format is described in Table 3.2. In our research, we used the analyzed intensity data obtained from real Affymetrix microarray image as the data input for the simulation model.

Hence, the simulated microarray image also carried the real genome information rather than randomly generated data.

Table 3.2 Data input format

Data	Intensity data for each cell
Time	Time instants
Genes	Probe names
Spot	Probe locations
Name	Dataset name
Type	cDNA or oligonucleotide or ratios
scale	Input data scale

- Input variable: Data. This variable contained the intensity matrix of each cell in microarray chip. Each column corresponds to one sample of the microarray. Next, all conditions of the data were saved as a data matrix.
- Input variable: Time. This was a vector variable. This vector contains the time instants for different microarray experiments. Time scale can vary. The total length of this vector should equal to the number of rows in the Data matrix.
- Input variable: Name. This is a string variable. It indicates the name of the dataset saved.
- Input variable: Info.genes. This is an array variable. It stores the name of the genes/probes. Each name corresponded to each gene/probe
- Input variable: Info.spots. This is a matrix variable. Coordinates x and y were given for each cell which indicates the location. The info.spots has a matrix form of $[x,y]$, where x and y were both column vectors.

- Input variable: Info.*. This is an optional variable if the user needs to define more information for the cell.
- Input variable: Type. This is a string variable. There are three optional strings, ratios, expression and intensity. Ratios represent the gene expression type. Expression represents the cDNA microarray type. Intensity represents the Affymetrix microarray type.
- Input variable: Scale. This is a string variable. There are two options which are linear and log. These two options indicated that if the input data is in log scale or linear scale.

Biological errors were related to the experiment preparation process [34, 35]. These intrinsic errors were presented no matter what sort of measurements is used. As to the measurement noise errors, they were more related to the measurement technology used in the experiment [36]. After such errors were taken into account, there are also some other studied error sources added in this section [37, 38, 39, 40, 41].

This was the most important step in this whole simulation process since this step introduced many realistic biological statistical error and noise model for the simulated data. Furthermore, the specified noise and error model parameters to obtain different qualities of microarray image such as high quality, median quality and low quality. Default model parameters were set as follows: for Simple noise model, SNR noise model, Dror noise model, Hartemink noise model, Hierarchical error model, Rocke noise model, and Hein noise model. Table 3.3 shows the error noise parameters for these models.

Table 3.3 List of noise parameters

Kernel	Kernel used to model the population effect.
Copies	Number of times the population effect is applied.
Error model	Error model to be used; each error model has its own parameters
Simple noise model	(0.01, 0.001)
SNR noise model	(0, 10)
Dror noise model	(1, 0.01, 0, 36, 13, 0.76, 0, 0.21)
Hartemink noise model	(0.2, 0.01, 1)
Hierarchical error model	(0.012, 0.010, 0.085, 0.094, 0.011)
Rocke noise model	(5, 0.1, 1, 1)
Hein noise model	(0.341, 0.335, 0, 50, 0.5, 1, 0.5, 10)

The slide manufacturing option was the data extraction process from slides. This step was also an important step. In this step, the user specified the microarray chip layout such as how many sub-arrays and how many probes on the chip and so on. It included also several error models which may be caused by the spotting and printing process. These may be the variation of the cell position and size, or the print tip and the spot shape deformations and so on. Table 3.4 contains the parameters used in this step.

Table 3.4 List of manufacturing parameters

	Good	Normal	Bad	Affymetrix
Stype	cdna	cdna	cdna	oligo
Sspot	circle	gaussian	gaussian	
Spix	12	12	12	10
Smovprob	0.01	0.1	0.5	0.1
Smov	0	1	2	1
S_{μ}	5	5	5	4
S_{σ^2}	0.001	0.01	0.1	0.01
P	0	1	1	
Pp	0.0	0.5	0.9	
Ph	0	3	3	
Pw	0	2	2	
Pb	0	1	2	
Cprob	0	0.1	0.25	
Cnum	0	4	8	
Ccut	0	3	6	
B	[4,2]	[4,2]	[4,2]	[1,1]
Bspace	50	50	50	
Bcurve	0	1	2	
Bmaxc	0	3	10	

The slide hybridization was for the spot shape simulation. There were several models included in this step to control the spot shape. Since there was no single model that can simulate all the types of microarray images, several models such as Gaussian and polynomial hyperbolic models were implemented [42]. Researchers may set up parameters in different error models to control the spot shape. The final spot shape will be influenced by the multiplicative Gaussian noise with the researcher-specified parameter values in error noise models. In addition, this step allowed users to choose which type of microarray was related, two-channel or one-channel. For Affymetrix oligonucleotide microarray, the rectangular spot influenced by the Gaussian noise model was used. Like the previous steps, the user could also specify the parameters in each error models. Table 3.5 contains the parameters used in this step.

Table 3.5 List of hybridization parameters

	Good	Normal	Bad	Affymetrix
H_{σ^2}	0.001	0.01	0.1	0.01
Herrors	1	1	1	1
Hbgnoise	10	30	50	20
Hbgvar	0.001	0.01	0.03	
Hbggrad	1	1	1	1
Hnoscratch	0	1	3	0
HSlength	0	0.3	0.9	
HSwidth	0	3	5	
Hnoair	0	1	3	
$H_{\sigma air}$	0	15	30	
H_{σ}	1	10	20	
Hbleed	0	2	10	
Hbleedsize	0	5	10	
Hbleeddist	0	0.4	0.4	

After the hybridization process, the microarray image was read by an optical scanning. Although the advanced scanners are usually of high quality, they could still generate misalignment errors. For example, the slide was not perfectly scanned straight in the chip. This misalignment happens very often in Affymetrix microarray image cases. In addition, there may be some saturate measurement values generated due to the finite dynamic range of the scanner.

After the scanning step, the microarray image was generated automatically. The user could also define if the built in segmentation algorithm is used to analyze the simulated image straightly. Table 3.6 contains the parameters used in this step.

Table 3.6 List of scanning parameters

	Good	Normal	Bad
Rpower	1	10	20
Rb	16	16	10
Req	0	0	0
Rth	7	5	3
RRch	2	2	2
RGch	1	1	1
Rerrors	0	1	1
Rangle	0	0.1	1
Rmm	0	0	1

3.3 Microarray Simulation Process

When evaluating the performance of the segmentation algorithms, it is hard to discover the exact gene expression levels or pixel level intensities. Hence, we utilized the simulation module to obtain different qualities of microarray images. The “ground true information” for each simulated image was pre-assigned before analysis. In order to obtain a comprehensive understanding of different algorithms’ performance, we analyzed the simulated image based on the gene expression level in order to observe the expression sensitivity of different algorithms will be directly analyzed. In our research, we mainly focused on high quality resolution simulation since nowadays every high density oligonucleotide microarray is produced in an advanced experimental environment. Bubble, scratch and other layout disadvantages are rarely generated.

Therefore, we could control the gene signal values and intensity values to implement the simulation for each microarray image group. All the simulated images for each quality microarray group will share the same signal and intensity inputs. Next, these images were analyzed separately with both SBC and GCOS to generate two sets of intensity values and used the built in GCOS gene signal of the MAS5 algorithm to

compute the corresponding signal values from those two sets of intensity values. In this way, we could compare the gene expression results between these two methods. This feature was more desirable than just to validate the intensity accuracy because the gene expression level has a more influential meaning in practice than the intensities have.

When the simulation step was completed, a simulated microarray image TIFF file was generated and written into a specific DAT file. Next, the DAT file and TIFF file were imported into both SBC and GCOS in order to obtain two sets of intensity and signal values. Therefore, not only was the intensity accuracy evaluated, but also the signal accuracy was validated using both SBC and GCOS methods in the evaluation process. This whole process is illustrated in Figure 3.3.

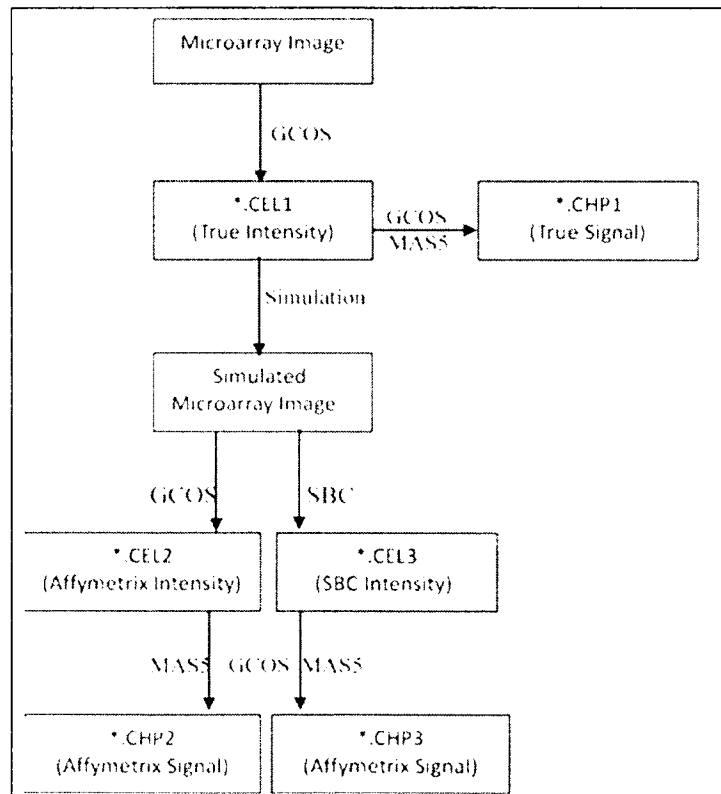


Figure 3.3 Microarray simulation process

1. Write simulated microarray image into DAT file.

In what follows, we defined each step of the process, after obtaining the simulated microarray images, the Affymetrix GCOS will analyze the images. However, the Affymetrix GCOS software cannot analyze the microarray image type file (JPG, TIFF and so on) directly. In order to allow the Affymetrix GCOS to analyze the simulated image, we needed to rewrite the simulated microarray image into the *.DAT file. The *.DAT file contains the 16-bit grey level image pixels data, header information, layout information and so on. It is a special data format designed by Affymetrix. The first section in DAT file is the header information stored in DAT file. The pixel data matrix is stored as 16-bit unsigned integer value at byte 512 following the header. The new simulated microarray image has everything as the original image except has for the image pixels data. Thus, we extracted the new simulated pixels data and wrote them into the original DAT file and remained any other layout information the same. In this case, we got a new DAT file corresponded to the new simulated microarray image.

2. Analyze the new DAT file by GCOS.

When the simulated image was written into the specific DAT file format, the new DAT file was imported into GCOS. GCOS automatically analyzed and generated the CEL2 file and CHP2 file, which presented the intensity and expression information for that specific DAT file. In such a way, one set of intensity values and signal values from GCOS was obtained.

3. Analyze the new simulated image by SBC.

SBC is the segmentation based contour algorithm implemented in JAVA platform. The input of SBC is a batch file, which contains both image and grid location information

together. New simulated image and its corresponding grid information are written into this batch file and imported into SBC. The intensity results were saved as the output in a TXT file, which contains four columns. They were the numbers of pixels assigned as the foreground, foreground/background mean, cell intensity and cell background median. In addition, the SBC generated a segmented image shown in Figure 3.4 with the boundary colored in white for each cell.

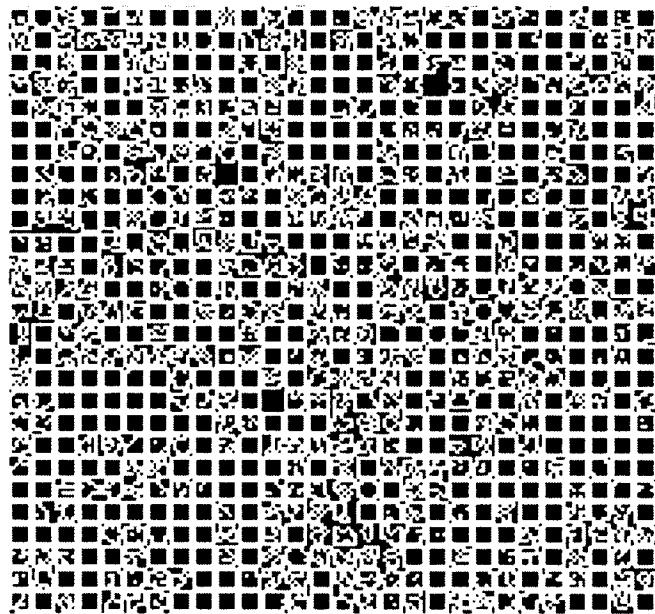


Figure 3.4 Segmented image by SBC

4. The SBC intensity output is written into CEL file to get the corresponding gene expression signal values.

SBC algorithm will generate intensity output in TXT file after image segmentation. Since our objective was to detect the signal expression sensitivity for each segmentation algorithm, we needed to calculate the gene expression values by MAS5 based on the intensity values obtained from SBC. However, the GCOS requires special CEL file format as the input to MAS5 algorithm. In another words, we needed to write

the SBC intensity TXT file output into this specified CEL file. The CEL file format was an ASCII text file, which was divided into many sections begin with a section name defined by a line. It contained the information for each probe cell. It included the layout coordinates of each cell, the intensity value for each cell, the number of pixels included for each cell and the standard deviation of each cell, etc.

Therefore, CEL3 was generated from the intensity values obtained from SBC. Next, we used MAS5 to analyze these intensity values stored in CEL3 file to get the corresponding gene expression signal values stored in CHP3 file.

We started with the picture that we want to segment and using the segmentation method, embedded into the GCOS, we created a CELL1 file, which stores the results of the intensity calculations on the pixel values of the DAT file. Using the data acquisition module in the GCOS, we extracted the gene values, the so called “true-values” in CHP1 file. On the CELL1 file, we applied the algorithm, and we generated a new picture. This picture was segmented using the segmentation module in the GCOS and generated CELL2 file. It was segmented using the SBC method and generated CELL3 file. To eliminate any bias in our analysis, we processed both CELL2 and CELL3 files with the data acquisition module inside the GCOS and obtained gene values in CHP2 file from Affymetrix and gene values in CHP3 file from SBC. These two sets of gene signal values were compared with the “true values” and statistical analysis performed to determine which segmentation method performs better.

Hence, after modifying the original simulation algorithm, we were able to generate Affymetrix type of images by using the obtained intensity value, the so-called “true values.” Previous studies in the literature did not have such information; therefore,

the statistical analysis that could be done was rather limited. After applying our modification to the algorithm, we were able to generate any number of pictures we needed for an experiment. To these pictures we could apply any segmentation method in existence, and we could analyze which segmentation method offered values closer to the “true values.” This simulation method also enables us to identify weak or strong points within each segmentation method. In our research, we only compared the performance of two methods: the SBC and the GCOS methods since GCOS was the currently known method used in Affymetrix microarray analysis.

3.4 Microarray Simulation in 4×4 Blocks

Usually, a normal image is a 16-bit grey intensity image in a special DAT file format. The DAT file contains a 512 byte header following by the raw image data. A normal image, for instance, contains 4733×4733 grid of pixels which contains all the cells as well as the small, vacant border area. Therefore, the total size for one image would be $2 \times 4733^2 + 512 = 44803090$ bytes which is approximately 45M. Sometime, the size of an image can be more than 50M up to 100M. Additionally, during the simulation process, a large amount of grid layout information and noise error statistical models are implemented. However, due to the limits of the computer RAM and MATLAB memory space, our first approach we could not simulate a whole Affymetrix microarray image at once from the original simulation model. Thus, we did an iteration to simulate the microarray image piece by piece, and integrate all the pieces together into a whole image.

First, we divided the whole image probe cells area into 4×4 pieces from upper left to lower right. Each piece contained 200×200 cells at most. After simulating the image piece by piece, we omitted the border area in the simulated image, and we

extracted the simulated cells area substituting them into the original image cells area. In this case, we made sure that the probe chip layout format and border area was exactly the same with the original microarray image but with different probe cells' pixels. Figure 3.5 is the original microarray image for Canine_a genome. Figure 3.6 is the simulated microarray Canine_a image in 4×4 blocks.

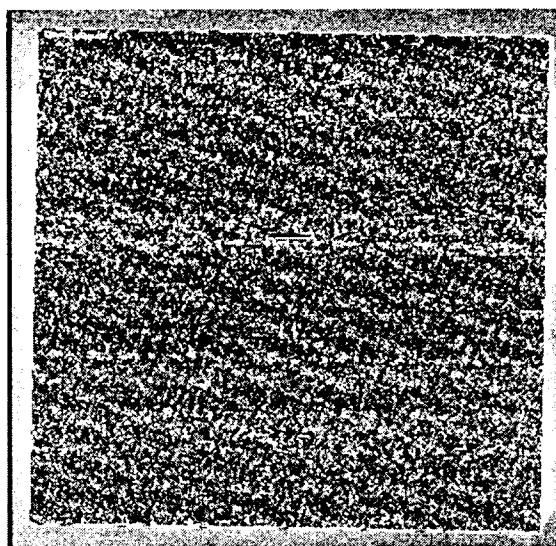


Figure 3.5 Original microarray image for Canine_a genome

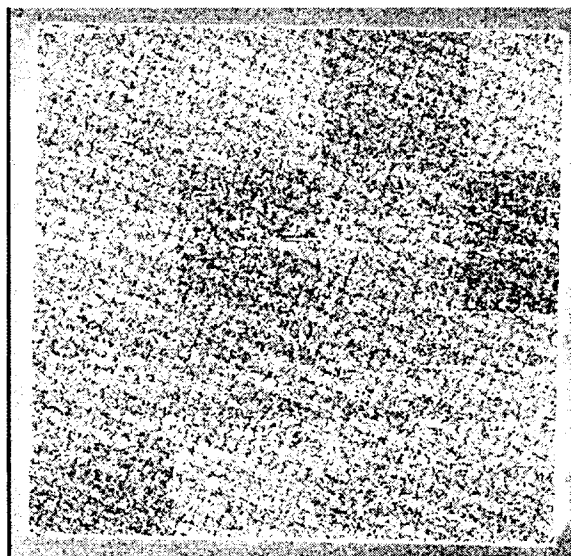


Figure 3.6 Simulated microarray Canine_a image in 4×4 blocks

The 4×4 blocks simulated microarray image has captured some properties of the microarray experiment. However, there are some population block effects caused by the piece wise simulation. This block effect motivated us to seek the possibility of one block simulation.

3.5 Microarray Simulation in One Block

In the first stage of this research, due to the limit memory of CPU and MATLAB, it was not possible to simulate the Affymetrix microarray image at one time. Hence, we divided the original real microarray image into 4×4 blocks. Next, we implemented simulation on each sub area iteratively. In this way, the Affymetrix microarray image could be generated in computer but with obvious an population block effect which will never happen in real microarray experiment image.

Therefore, we implemented the simulation model on a high performance computer allowing simulate Affymetrix microarray image at one time. This guarantees that the simulated Affymetrix microarray image was fully capturing the real biological characteristics without any block effect.

CHAPTER 4

MICROARRAY IMAGE SEGMENTATION METHOD

In this chapter, we will present the Active Contours Without the Edges (ACWE) segmentation method.

4.1 Active Contours Without the Edges Method

The Active Contours Without the Edges (ACWE) is a method introduced by Tony F. Chan [27], as a model for active contours to detect objects in a given image. The technique used is based on curve evolution and the method can detect objects that have very smooth boundaries. The method uses an algorithm that employs finite difference partial differential equation. This ACWE method has been used in literature in multiple studies. Mark Moelich and Tony F. Chanin [43] developed a tracking algorithm based on the ACWE, segmentation algorithm that is able to handle changes that result from deformations in the object that is tracked. Ahmad Almhdie in [44] presented a method based on ACWE algorithm as a segmentation method used for mouse brain MRI images, and Nassir Salman in [45] introduced an image segmentation algorithm based on ACWE used to extract individual components from a medical image. We can continue with other examples, Olivier Rousseau in [46] used ACWE for heart segmentation. Hence, we have already established that the ACWE method is one of the frequently used segmentation methods.

In ACWE, an image u_0 is defined by two regions which are inside and outside the objects within u_0 . The inside region is denoted by u_0^i and the outside region is denoted by u_0^o . Hence, a fitting function is defined, where C is the active curve, and C_1 and C_2 are the constants depending on C .

$$F_1(C) + F_2(C) = \int_{inside(C)} |u_0(x, y) - c_1|^2 dx dy + \int_{outside(C)} |u_0(x, y) - c_2|^2 dx dy. \quad (4.1)$$

Hence, the object's boundary C_0 is the minimal C such that the fitting function is

$$\inf_C \{F_1(C) + F_2(C)\} \approx 0 \approx F_1(C_0) + F_2(C_0). \quad (4.2)$$

After the fitting function is defined, the energy function is presented, where $\lambda_1 > 0, \lambda_2 > 0, v \geq 0, u \geq 0$ are fixed values.

$$F(c_1, c_2, C) = \mu \cdot Length(C) + v \cdot Area(inside(C)) + \lambda_1 \cdot \int_{inside(C)} |u_0(x, y) - c_1|^2 dx dy + \lambda_2 \cdot \int_{inside(C)} |u_0(x, y) - c_2|^2 dx dy. \quad (4.3)$$

For the following calculation, λ_1 and λ_2 are set to be 1 and $v = 0$. Thus, the approximation value u of u_0^o can be defined in

$$u = \begin{cases} average(u_0) inside C, \\ average(u_0) outside C. \end{cases} \quad (4.4)$$

In the level set method, C is represented by the Lipschitz function, $C \subset \Omega$ and $\phi : \Omega \rightarrow R$

$$\begin{cases} C = \partial\omega = \{(x, y) \in \Omega : \phi(x, y) = 0\}, \\ inside(C) = \omega = \{(x, y) \in \Omega : \phi(x, y) > 0\}, \\ inside(C) = \omega = \{(x, y) \in \Omega : \phi(x, y) < 0\}. \end{cases} \quad (4.5)$$

After substitute ϕ for C in above level set, Heaviside function H and Dirac function δ are introduced as follows:

$$H(z) = \begin{cases} 1, & \text{if } z \geq 0, \\ 0, & \text{if } z \leq 0. \end{cases} \quad (4.6)$$

$$\delta_0(z) = \frac{d}{dz} H(z). \quad (4.7)$$

Next, the energy function $F(c_1, c_2, C)$ can be rewritten as follows:

$$\text{Length}\{\phi = 0\} = \int_{\Omega} |\nabla H(\phi(x, y))| dx dy = \int_{\Omega} |(\phi(x, y))| \cdot \delta_0(\phi(x, y)) dx dy. \quad (4.8)$$

$$\int_{\phi > 0} |u_0(x, y) - c_1|^2 dx dy = \int_{\Omega} |u_0(x, y) - c_1|^2 \cdot H(\phi(x, y)) dx dy. \quad (4.9)$$

$$\int_{\phi < 0} |u_0(x, y) - c_1|^2 dx dy = \int_{\Omega} |u_0(x, y) - c_2|^2 \cdot (1 - H(\phi(x, y))) dx dy. \quad (4.10)$$

$$\begin{aligned} F(c_1, c_2, C) = & \mu \cdot \int_{\Omega} |\nabla \phi(x, y)| \cdot \delta(\phi(x, y)) dx dy + v \cdot \int_{\Omega} H(\phi(x, y)) dx dy + \\ & \lambda_1 \cdot \int_{\Omega} |u_0(x, y) - c_1|^2 \cdot H(\phi(x, y)) dx dy + \\ & \lambda_2 \cdot \int_{\Omega} |u_0(x, y) - c_2|^2 \cdot (1 - H(\phi(x, y))) dx dy \end{aligned} \quad (4.11)$$

Finally, the approximation u is obtained.

$$u(x, y) = c_1 \cdot H(\phi(x, y)) + c_2 \cdot (1 - H(\phi(x, y))), (x, y) \in \bar{\Omega}. \quad (4.12)$$

Finite difference technique is implemented to solve the partial differential equation problem.

$$\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^n}{\Delta t} = \delta_h(\phi_{i,j}^n) \left[\begin{array}{l} \frac{\mu}{h^2} \Delta_x^x \cdot \left(\frac{\Delta_+^x \phi_{i,j}^{n+1}}{\sqrt{\frac{(\Delta_+^x \phi_{i,j}^n)^2}{(h)^2} + (\phi_{i,j+1}^n - \phi_{i,j-1}^n)^2 / (2h)^2}} \right) \\ + \frac{\mu}{h^2} \Delta_x^y \cdot \left(\frac{\Delta_+^y \phi_{i,j}^{n+1}}{\sqrt{\frac{(\Delta_+^y \phi_{i,j}^n)^2}{(h)^2} + (\phi_{i+1,j}^n - \phi_{i-1,j}^n)^2 / (2h)^2}} \right) \\ -v - \lambda_1 (u_{0,i,j} - c_1(\phi^n))^2 + \lambda_2 (u_{0,i,j} - c_2(\phi^n))^2 \end{array} \right] \quad (4.13)$$

The whole algorithm is introduced as follows:

1. An evolving curve is initialized. Initialize ϕ^0 by $\phi_0, n=0$.
2. Compute the average energy inside and outside the active curve.

$$c_1(\phi) = \frac{\int_{\Omega} u_0(x,y)H(\phi(x,y))dxdy}{\int_{\Omega} H(\phi(x,y))dxdy}. \quad (4.14)$$

$$c_2(\phi) = \frac{\int_{\Omega} u_0(x,y)(1-H(\phi(x,y)))dxdy}{\int_{\Omega} (1-H(\phi(x,y)))dxdy}. \quad (4.15)$$

3. Detect the exact curve by solving the PDE in ϕ , where $\phi(0, x, y) = \phi_0(x, y)$ in Ω , $\frac{\delta_{\varepsilon}(\phi)}{|\nabla\phi|} \frac{\partial\phi}{\partial\vec{n}} = 0$ on $\partial\Omega$, \vec{n} is the exterior normal vector to the boundary $\partial\Omega$, and $\frac{\partial\phi}{\partial\vec{n}}$ is the normal derivative of ϕ at the boundary.

$$\frac{\partial\phi}{\partial t} = \delta_{\varepsilon}(\phi) \left[\mu \operatorname{div} \left(\frac{\nabla\phi}{|\nabla\phi|} \right) - \nu - \lambda_1(u_0 - c_1)^2 + \lambda_2(u_0 - c_2)^2 \right] = 0, (0, \infty) \times \Omega. \quad (4.16)$$

4. Reinitialize ϕ based on the signed distance function to the curve.
5. Check if the solution is stationary or not. If not, $n = n + 1$ and repeat from step two.

4.2 Advantages of Active Contours Without the Edges Method

From [27], it has been established that the Active Contours Without the Edges method does not determine the curve from being initialized to being stopped at the boundary by using the edge function. The edge detection function technique is the traditional segmentation tool in image processing, machine vision and computer vision, especially in feature detection and feature extraction. All of these aim at identifying the points where the image brightness changes sharply or discontinuity. Normally, several discontinuities are considered during the partition process such as the discontinuities in depth, discontinuities in surface orientation, and changes in material properties and

variations in scene illumination. However, most of these edge function detection techniques are not capable of capturing the spatial relationship information of the pixels. The main disadvantages of these types of methods are sensitivity to noise and inaccurate segmentation over the boundary [47]. Hence, in the ACWE model, it does not rely on the edge detection function, which solves the problem of evolving curve over the defined boundary.

In the ACWE model, the initial curve can be set anywhere within the boundary. In traditional active contours method, however, the initial curve must be defined close to the objects. In this case, ACWE is more robust when applying on the microarray image segmentation process. In our research, the object within the defined boundary was randomly located, which was the property of the microarray experiment.

In the ACWE method, it can detect the boundary of the object even in noisy image, but in traditional segmentation method, the image needs to be smoothed at the beginning if it is in a noisy condition.

The traditional active contours method is to detect the object different from the background. After initializing a curve within an image, the internal and external forces will drive the evolving curve to the contours of the object, which implies that the ACWE method is more convenient to segment a various conditions of images. The original snake's model used in active contours method is presented for a curve C , the external energy is defined as the energy out of the curve C .

$$E_{ext}(C) = \int_0^1 \left| \nabla u_0 (C(S)) \right|^2 ds. \quad (4.17)$$

The internal energy is defined as the energy inside of the curve.

$$E_{int}(C) = \alpha \int_0^1 |C'(S)|^2 ds + \beta \int_0^1 |C''(S)|^2 ds. \quad (4.18)$$

Total energy to be minimized is defined as follows:

$$E(C) = \lambda E_{ext}(C) + E_{int}(C) \quad (4.19)$$

Hence, in this original snake's model, the boundary of the objects may not be detected correctly when the initial curve is defined far from the boundary of the object. Given this disadvantage, the ACWE method introduces a new balloon force into the snake's model. The modified snake's model of the minimum energy is defined by the sum of these three terms $\alpha \int_0^1 |C'(S)|^2 ds + \beta \int_0^1 |C''(S)|^2 ds$, $\lambda \int_0^1 \left| \nabla u_0 (C((S))) \right|^2$ and $v \iint_{\Omega} dx dy$.

This modification reduces the sensitivity of the initial curve and the noise level of the target image. In addition, by setting $\beta=0$, $w(0) = 1$ and $\lim_{x \rightarrow \infty} w(x) = 0$, the total energy function is simplified as follows:

$$E(C) = \lambda E_{ext}(C) + E_{int}(C) = \alpha \int_0^1 |C'(S)|^2 ds + \lambda \int_0^1 w \left| \nabla u_0 (C((S))) \right|^2 ds. \quad (4.20)$$

Therefore, the introduction of the balloon force will be adjusted. By applying these modifications, there is no limit on the initial curve location and the condition of the target image. This implies that the ACWE method has more advantages compared to the traditional edge detection methods.

4.3 Segmentation Based Contours Method

As we mentioned previously, the ACWE method can partition an image into several parts, but when applying it to microarray, it needs to partition the image exactly into two parts: intensity and background. We will use Chan-Vese (C-V) method [23] and make some improvements on the initial conditions to compute the exact boundary of the

probe cell in Affymetrix GeneChip microarray. The proposed modified ACWE method is called as the Segmentation Based Contours (SBC) method.

4.3.1 Reduce the Length Constraint.

Since the Affymetrix types of microarray image is very large, the segmentation is very time consuming. Hence, we tested the possibility of a term reduction in the Chan-Vese model. We still set $\delta_\varepsilon(\phi) = 1$ for simplicity. Some equations could be simplified.

$$\frac{\partial \phi}{\partial t} = \delta_\varepsilon(\phi) [\lambda_1(u_0 - c_1)^2 + \lambda_2(u_0 - c_2)^2] = 0. \quad (4.21)$$

$$\begin{aligned} \frac{\phi_{i,j}^{n+1} - \phi_{i,j}^n}{\Delta t} &= \left[-\lambda (u_{0,i,j} - c_1(\phi^n))^2 + \lambda_2 (u_{0,i,j} - c_2(\phi^n))^2 \right] \\ &= 2(c_1(\phi^n) - c_2(\phi^n))(u_{0,i,j} - (c_1(\phi^n) + c_2(\phi^n))/2). \end{aligned} \quad (4.22)$$

4.3.2 Using a Fast Algorithm.

The fast algorithm it directly calculates the difference of the both inside and outside curve energy functions when detecting a pixel changed from inside curve to the outside curve or opposite. When the changing pixel is detected, the whole region will be updated. Next, each pixel will be swept and the iteration will terminate until the energy remains stationary.

The fast algorithm is introduced as follows:

1. Set up an initial curve which could partition the image into two parts. $\phi = 1$ and $\phi = -1$ are represented for these two parts.

2. For a pixel x related to the intensity y .

If $\phi(x) = 1$, then

$$\Delta F_{12} = (y - c_2)^2 \frac{n}{n+1} - (y - c_1)^2 \frac{m}{m-1}. \quad (4.23)$$

If $\Delta F_{12} < 0$, then

$$\phi = -1. \quad (4.24)$$

If $\phi = -1$, then

$$\Delta F_{21} = (y - c_1)^2 \frac{m}{m+1} - (y - c_2)^2 \frac{n}{n-1}. \quad (4.25)$$

If $\Delta F_{21} < 0$, then

$$\phi = 1. \quad (4.26)$$

Where c_1 is the average of the pixels within the objects; c_2 is the average of the pixels beyond the objects; m is the number of the pixels within the objects; and n is the number of the pixels beyond the objects.

3. Check if the energy function is stationary or not. If not, go back to step two and iteration.

4.3.3 Using a Higher Order Finite Difference Scheme.

By using a higher order finite difference scheme, the segmentation results will be more accurate.

In ACWE, the central finite difference formula is $\frac{\phi_{i+1,j}^n - \phi_{i-1,j}^n}{2h}$ and $\frac{\phi_{i,j+1}^n - \phi_{i,j-1}^n}{2h}$.

Then, we rewrite the central finite difference formula into $\frac{8(\phi_{i+1,j}^n - \phi_{i-1,j}^n) - (\phi_{i+2,j}^n - \phi_{i-2,j}^n)}{12h}$

and $\frac{8(\phi_{i,j+1}^n - \phi_{i,j-1}^n) - (\phi_{i,j+2}^n - \phi_{i,j-2}^n)}{12h}$. In addition, we also improve the forward and

backward finite difference formula, from $\frac{\phi_{i+1,j}^n - \phi_{i,j}^n}{h}$ to $\frac{-\phi_{i+2,j}^n + 4\phi_{i+1,j}^n - 3\phi_{i,j}^n}{2h}$, from $\frac{\phi_{i,j+1}^n - \phi_{i,j}^n}{h}$

to $\frac{-\phi_{i,j+2}^n + 4\phi_{i,j+1}^n - 3\phi_{i,j}^n}{2h}$, from $\frac{\phi_{i,j}^n - \phi_{i-1,j}^n}{h}$ to $\frac{\phi_{i-2,j}^n - 4\phi_{i-1,j}^n + 3\phi_{i,j}^n}{2h}$, and from $\frac{\phi_{i,j}^n - \phi_{i,j-1}^n}{h}$ to

$\frac{\phi_{i,j-2}^n - 4\phi_{i,j-1}^n + 3\phi_{i,j}^n}{2h}$.

4.4 Apply Segmentation Based Contours Method on Affymetrix Image

Before we applied the Segmentation Based Contours (SBC) method on Affymetrix microarray image, there were some adjustments that had to be implemented in SBC.

- First, we conducted the SBC on each cell at a time. Usually, a normal microarray image consists millions of cells. If we just use the SBC to analyze the whole image at once, the segmentation and the extracted information for each cell will not be accurate.
- Second, in order to locate each cell precisely, we introduced the grid calculation for each cell. This grid calculation would detect and correct the misalignment. After the grid process, each cell was assigned an x and y coordinates.
- Third, we decreased the iteration times when finding the evolving curve which decreased the segmentation time dramatically.
- Finally, we adjusted the μ to detect dim cells.

After the adjustment, applying ACWE on microarray images was as follows:

1. Download the Affymetric microarray image library file from Affymetrix website: http://www.Affymetrix.com/support/technical/sample_data/demo_data.affx.
2. After installing the library file, DAT and CEL files related to the experiment were extracted from GCOS.
3. We calculated the grid coordinates for each cell based on the DAT and CEL files. An image type file (16-bit TIFF) of the microarray was extracted from the DAT file.

4. Each cell was segmented iteratively by using the image type file and the grid file. The 75th percentile of the pixels inside the curve were calculated and saved as the intensity for that cell. Next, a TXT file containing the intensity information was generated. In the TXT file, there were four columns. The first column saves the number of pixels included inside the curve. The second column and the fourth column saved the background median and mean. The third column saved the intensity for each cell. Figure 4.1 presents the segmented Affymetrix microarray image with the boundary of each probe cell colored in white.

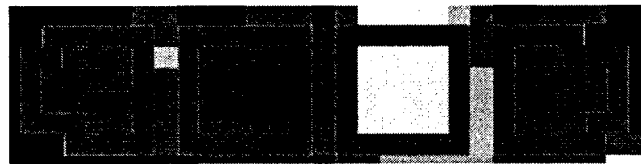


Figure 4.1 Affymetrix microarray image segmented by SBC

In Figure 4.1, we present a zoomed area of a segmented microarray image. More detailed segmentation information is introduced in the following tables with one cell as an example. Table 4.1 illustrates the “ground truth information” for each pixel in one cell. Table 4.2 and Table 4.3 are shown as the segmentation results separately by GCOS and SBC. In Table 4.2, GCOS sets the outer pixels in highlighted area as the boundary for each cell. In Table 4.3, we set the detected boundary pixel value to be 60000. From these three tables, the GCOS was seen as more unstable to be influenced by the different noise level. It just cuts off the cell boundary pixels and calculates the 75th percentile for the rest of pixels. As to SBC, we integrate the level set method to find the exact boundary for each specific cell. This level set method is more robust and accurate than GCOS for the segmentation accuracy.

Table 4.1 Ground truth pixels in one cell

271	376	143	196	98	159	91
56	286	121	83	174	128	234
54	83	76	76	76	83	234
55	354	196	143	76	76	174
57	98	76	76	98	301	76
51	286	76	143	136	76	106
49	91	294	76	76	211	76

Table 4.2 Segmentation results from GCOS

271	376	143	196	98	159	91
56	286	121	83	174	128	234
54	83	76	76	76	83	234
55	354	196	143	76	76	174
57	98	76	76	98	301	76
51	286	76	143	136	76	106
49	91	294	76	76	211	76

Table 4.3 Segmentation results from SBC

60000	60000	60000	60000	60000	60000	60000
60000	60000	60000	60000	60000	60000	60000
60000	60000	76	76	76	83	60000
60000	60000	60000	143	76	76	60000
60000	60000	76	76	98	60000	76
60000	60000	76	143	136	76	106
60000	60000	60000	76	76	60000	76
60000	60000	60000	60000	60000	60000	60000

In Table 4.4, we introduced intensity comparison among "ground truth intensity," SBC intensity and GCOS intensity. It is shown that our intensity value obtained from SBC is more approaching with the true intensity value. More comprehensive comparison method and results will be illustrated in segmentation results section.

Table 4.4 Intensity results comparison

Original Cell intensity	ACWE intensity	GCOS intensity
60	83	143

CHAPTER 5

EXPERIMENTAL RESULTS

In this chapter, we will show how we performed gene expression value comparison between the results obtained from the SBC segmentation algorithm and the GCOS segmentation algorithm. The GCOS algorithm utilizes the 75th percentile pixel value as the intensity value for each spot. It shows some weakness when analyzing different qualities of spot cells. However, SBC algorithm segments each spot adaptively by modifying the C-V model from three perspectives as we described in the previous chapter. After solving the partial differential equations in C-V model, SBC achieves the boundary for each spot cell in microarray image. Next, each spot is partitioned into two areas, intensity and background. When applying the SBC segmentation process, the parameters are set as $\lambda_1 = \lambda_2 = 1, v = 0, h = 1$ and $\Delta t = 0.1$. In addition, smaller μ, ϕ_0 are chosen according to the spot cell size in oligonucleotide microarray image. Once all these parameter are set up in SBC program, it is not necessary to adjust them during the segmentation.

After the simulation on real microarray data from sample data website [33], two sets of intensity and expression values were obtained for each simulated image after applying the SBC and the GCOS algorithms. Statistical comparison analysis was conducted on these two sets of gene expression values. In our research, we mainly

focused on the gene expression accuracy analysis, which was more related to the gene expression levels. This gene expression accuracy analysis has more significant influence in microarray image analysis.

At the beginning of our research, due to the limited memory of CPU and MATLAB, it was not possible to simulate the Affymetrix microarray image at one time. Hence, we divided the original real microarray image into 4×4 blocks. Next, we implemented the simulation on each sub area iteratively. In this way, the Affymetrix microarray image could be generated in a computer but with the obvious population block effect, which will never happen in real microarray experiment image. However, since the simulation model we developed showed great improvements over known methods, we investigated the comparison results for these 16-block simulated images and obtained some encouraging results showing that the analyzed gene signal values obtained from SBC is more accurate and stable compared to the analyzed gene signal values obtained from GCOS. Therefore, we implemented the simulation model on a high performance computer allowing simulate Affymetrix microarray images one at a time. In this case, the simulated Affymetrix microarray image was fully capturing the real biological characteristics without any block effect. Table 5.1 shows the system information and MATLAB information.

Table 5.1 System and MATLAB information

Windows System:	Windows 7 Professional
Version:	2009
Processor:	Intel Core i5-2500K
System Type:	64-bit Operating System
Total Physical Memory:	8GB
Installed Memory:	8GB
Installed Matlab:	Matlab 7.11.0

Figure 5.1 presents the simulated 16-block image for Canine_a genome. Figure 5.2 presents the simulated one block image for Canine_a genome.

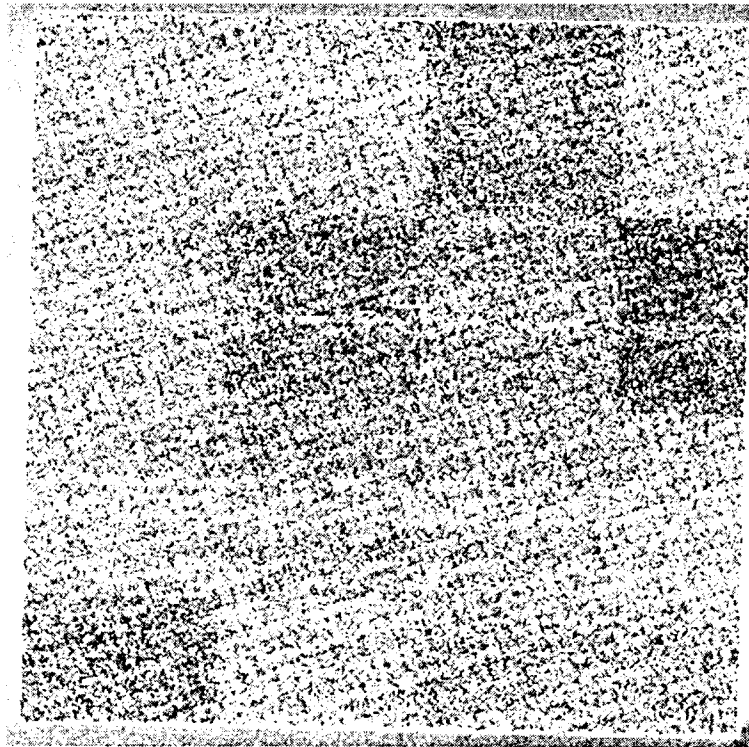


Figure 5.1 Simulated 16-block image for Canine_a Genome

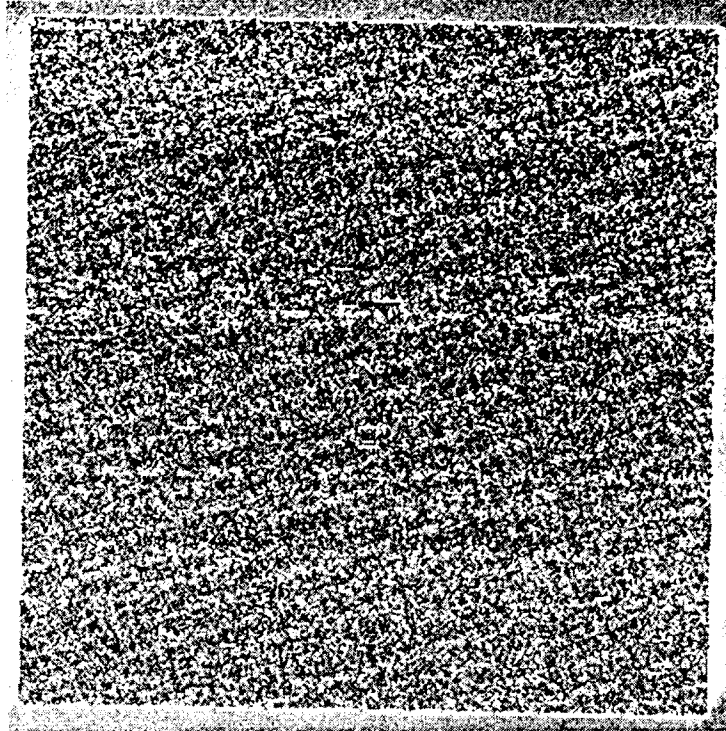


Figure 5.2 Simulated one block image for Canine_a Genome

We started with the one block simulated Affymetrix image generated from the sample of 23,912 genes in Canine_a genome. The sum of squares error metric was investigated for this simulated image, which is considered to be the most useful measures of dispersion. By calculating the sum of squares error $\sum(\text{observed data} - \text{true data})^2$ between the true gene signal value and the analyzed gene signal value, we could get a general understanding on how far the analyzed gene signal value is away from the true gene signal value. Table 5.2 shows the comparison result of sum of squares error. From Table 5.2, the SBC analyzed gene signal values have less sum of squares error compared to the GCOS analyzed gene signal values. This result may indicate that the gene signal value obtained from SBC has a better capability to approach the true gene signal value compared to the gene signal value obtained from GCOS. In addition, for each pair, if the

absolute difference between the SBC analyzed signal value and the true signal value is smaller compared to the difference between the GCOS analyzed signal value and the true signal value, value one will be assigned for SBC and value zero will be assigned for GCOS and vice versa. Next, we summed up all the numbers assigned to SBC and assigned to GCOS separately. The corresponding rate value was obtained through dividing this summation by the total number of genes. Hence, from this rate measurement of value, the percentage number of analyzed gene values for each method which approaches more to the true gene values for each method SBC and GCOS is investigated. From Table 5.2, it can be seen that SBC has higher percentage rate value 0.67 compared to GCOS which has 0.33. This higher percentage rate indicates that there are 67% of gene signal values that show that the analyzed gene signal values obtained from SBC more nearly approaching the true gene signal values compared to the analyzed gene signal values obtained from GCOS.

Table 5.2 Preliminary comparison for one block simulated image

Signal	SBC	GCOS
Sum of squares error	9860.32	33188.65
Rate	0.67	0.33

In order to investigate whether the analyzed gene signal value is significantly different from the true gene signal value, paired t test was performed separately on $d1$ and $d2$.

$$d1 = Signal_{true} - Signal_{SBC}. \quad (5.1)$$

$$d2 = Signal_{true} - Signal_{GCOS}. \quad (5.2)$$

In this paired t test, the null hypothesis and alternative hypothesis are presented as follows:

$$H_0: \mu_{d1} = 0 \text{ and } H_1: \mu_{d1} \neq 0.$$

$$H_0: \mu_{d2} = 0 \text{ and } H_1: \mu_{d2} \neq 0.$$

We hypothesized that the mean difference between the true gene signal value and the SBC gene signal value equals zero, and the mean difference between the true gene signal value and the GCOS gene signal value equals zeros at the given significance levels $\alpha = 0.05$, $\alpha = 0.01$ and $\alpha = 0.001$. The alternative hypothesis was that the mean difference between the true gene signal value and the SBC gene signal value does not equals zero and the mean difference between the true gene signal value and the GCOS gene signal value does not equals zeros at the given significance levels $\alpha = 0.05$, $\alpha = 0.01$ and $\alpha = 0.001$. We chose three different significance levels as they represent different criteria used to reject the null hypothesis. The lower the significance level, the more the data must diverge from the null hypothesis to be significant. We used these three significance levels for all the rest of the hypothesis tests performed in our research. Table 5.3 and Table 5.4 show the paired t test results for $d1$ and $d2$.

Table 5.3 Paired t test results for $d1 = Signal_{true} - Signal_{SBC}$

Significant level	Decision for P-value = 0.0015
$\alpha = 0.05$	Reject
$\alpha = 0.01$	Reject
$\alpha = 0.001$	Accept

Table 5.4 Paired t test results for $d2 = Signal_{true} - Signal_{GCOS}$

Significant level	Decision for P-value < 0.000001
$\alpha = 0.05$	Reject
$\alpha = 0.01$	Reject
$\alpha = 0.001$	Reject

Table 5.3 shows that the null hypothesis $H_0: \mu_{d1} = 0$ failed to be rejected at 0.001 significance level and were rejected at the other two significance levels. Hence, at the 0.001 significance level, we can conclude that the mean difference between the true gene signal value and the SBC gene signal value is zero. However, in Table 5.4, the null hypothesis $H_0: \mu_{d2} = 0$ was rejected at all significant levels. Hence, the mean difference between the true gene signal value and the GCOS gene signal value was significantly not zero. Therefore, when we selected a more conservative significance level 0.001 for the lowest probability of not being true, we can conclude that the mean of the SBC gene signal value was closer to the mean of the true gene signal value as compared to the mean of the GCOS gene signal value. The gene signal value obtained from SBC more nearly approaches the true gene signal value on average.

In order to evaluate and compare such differences in magnitude, we set:

$$D1 = |Signal_{true} - Signal_{SBC}|. \quad (5.3)$$

$$D2 = |Signal_{true} - Signal_{GCOS}|. \quad (5.4)$$

We compared the difference between the analyzed gene signal value and the true gene signal value after taking the absolute value sign to get better understanding on “how far” but not on “in which direction.” Our main interest was to decrease the difference between the true gene signal value and the analyzed gene signal value. Hence, two sample left tail t test was performed on the mean of $D1$ and the mean of $D2$ at the given

significant levels $\alpha = 0.05$, $\alpha = 0.01$ and $\alpha = 0.001$. The null hypothesis and alternative hypothesis were presented as follows:

$$H_0: \mu_{D1} = \mu_{D2}.$$

$$H_1: \mu_{D1} < \mu_{D2}.$$

This left tail t test could provide more information assuming the fact that the mean difference between true gene signal value and the SBC gene signal value was smaller than the mean difference between the true gene signal value and the GCOS gene signal value. The two sample t test results are shown in Table 5.5.

Table 5.5 Two sample t test for $D1$ and $D2$

Significant level	Decision for P-value < 0.000001
$\alpha = 0.05$	Reject
$\alpha = 0.01$	Reject
$\alpha = 0.001$	Reject

From Table 5.5, the null hypothesis $H_0: \mu_{D1} = \mu_{D2}$ was rejected at all significant levels, which indicated that we may accept the alternative hypothesis $H_1: \mu_{D1} < \mu_{D2}$. Thus, we could conclude that there was a significant difference between the mean of $D1$ and the mean of $D2$ and the mean of $D1$ was significantly less than the mean of $D2$. Hence, from this two sample t test, the mean difference between true gene signal value and SBC gene signal value are shown significantly smaller than the mean difference between true gene signal value and GCOS gene signal value. The gene signal values obtained from SBC were closer to the true gene signal values as compared to the gene signal values obtained from GCOS.

In addition to the paired t test and two sample t test, we also drew the boxplot for $D1$ and $D2$ as shown in Figure 5.3 to check if the SBC method provided gene signal

values that were closer to the true gene signal values as compared to the gene signal values provided by GCOS. The commonly used quartiles for $D1$ and $D2$ are described in Table 5.6.

The boxplot showed that the interquartile range and the total range of $D1$ are smaller than those of $D2$. In addition, the mean of $D1$ is smaller than the mean of $D2$, which was consistent with the previous two sample t test result. Hence, from Figure 5.3 and Table 5.6, we may conclude that the absolute difference between true gene signal value and SBC gene signal value is smaller compared to the absolute difference between true gene signal value and GCOS gene signal value. Therefore, there was less dissimilarity between true gene signal value and SBC gene signal value. SBC segmentation method is providing more accurate gene signal values.

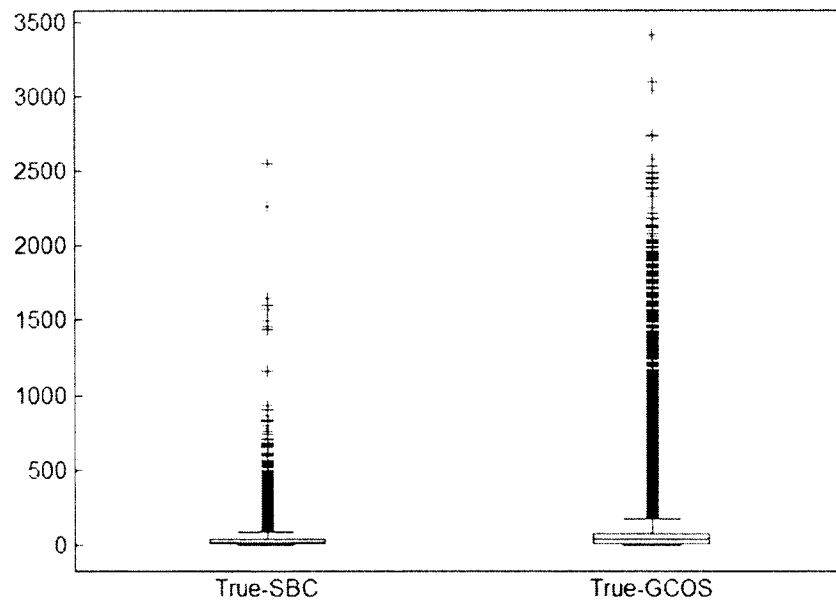


Figure 5.3 Boxplot for $D1$ and $D2$

Table 5.6 Quartiles summary information

Quartiles	0.025	0.25	0.5	0.75	0.975	1
$D1: Signal_{true} - Signal_{SBC} $	0.45	5.96	17.41	37.47	108.61	2545.95
$D2: Signal_{true} - Signal_{GCOS} $	0.74	8.62	32.27	71.38	553.84	3411.45

After this hypothesis test analysis, we continued the analysis by introducing more advanced statistical comparison techniques such as clustering analysis to investigate the dissimilarity between the true gene signal value and the analyzed gene signal value obtained from SBC and GCOS. Clustering analysis is the most commonly used method in microarray image data analysis process [48, 49, 50, 51]. There are two important options when performing the clustering analysis. One is the choice of the distance used to evaluate the dissimilarity between different groups of data. In this option, Sum of absolute difference $\sum_i^n |d_i|$ (Manhattan distance), Person correlation distance $1 - \frac{Z_{score}(observed\ data) * Z_{score}(true\ data)}{n}$ and Euclidean distance $\sqrt{\sum_i^n d_i^2}$ are the most important metrics in performing clustering analysis. The other one is the choice of the clustering technique on how to classify different groups of data based on the distance measurements obtained from the first option.

Table 5.7 lists all the distance metrics used for calculating the distance between the true gene signal value and the corresponding analyzed gene signal value obtained from SBC and GCOS. Table 5.7 indicates that the analyzed gene signal value obtained from SBC has smaller dissimilarity with true gene signal value as compared to the analyzed gene signal value obtained from GCOS. In addition, the analyzed gene signal value obtained from SBC has higher correlation with the true gene signal value as compared to the analyzed gene signal value obtained from GCOS. Hence, we may

conclude that SBC gene analyzed signal values are closer to the true gene signal values than are the GCOS analyzed gene signal values.

Table 5.7 Different metrics comparisons for Canine_a one block simulated image

Measurements	SBC Signal	GCOS Signal
Sum of absolute difference (Manhattan distance)	702602.7	1986828
Standard error of performance	63.76	214.62
Pearson correlation	0.99	0.98
Euclidean distance	9860.317	33188.65
Chebychev distance	2545.951	3411.448
Correlation distance	0.01015285	0.01485263
City block distance	702602.7	1986828

For Canine_a simulated microarray image, several commonly used difference metrics were investigated in Table 5.7. The SBC analyzed gene signal value had the less dissimilarity with true gene signal value compared to GCOS analyzed gene signal value. Hence, we conducted the cluster analysis. This could provide more information that if the SBC analyzed gene signal value and true gene signal value could be classified in the same group based on this small dissimilarity.

Cluster analysis is the most widely used statistical technique in gene expression analysis. In cluster analysis, a set of objects are assigned into different groups, where the objects in the same group share the least dissimilarity to each other than to those in different group. There are many different types of cluster algorithms used in different fields. The hierarchical clustering has many applications, and it is the most widely used method in bioinformatics gene expression data analysis [51, 52]. It merges different samples based on the pairwise distance similarity measurements to form same group until all different objects are evaluated and contained in on single group. The result of a

hierarchical clustering is a clustering complete tree plot with gene expression patterns as leaves and the root as the convergence point of all groups.

In our research, we chose Unweight Pair Group Method [53] with Arithmetic Mean as the bottom up agglomerative hierarchical clustering algorithm to measure the dissimilarity among different objects. This agglomerative hierarchical clustering algorithm is established to be the most commonly used cluster technique in bioinformatics gene expression data mining analysis. The bottom up scheme starts from the individual patterns and combines similar group together, ending up with the root based on the difference metrics selected as the measurement. In our analysis, we chose the Euclidean difference metric as the dissimilarity measurement for the clustering algorithm as classifying different groups of gene signal values.

Figure 5.4 shows the dendrogram cluster tree plot after implementing the hierarchical clustering algorithm on those three sets of gene signal values based on the Euclidean distance metric. The three sets of data were automatically divided into two different groups by average linkage clustering criterion. The SBC analyzed gene signal values and the true gene signal values were classified in one group. The GCOS analyzed gene signal values were classified in another different group. Therefore, the SBC analyzed gene signal value had significant less dissimilarity to the true gene signal values compared to the GCOS analyzed gene signal values. The SBC analyzed gene signal value more nearly approaches the true gene signal value compared to the GCOS analyzed gene signal value.

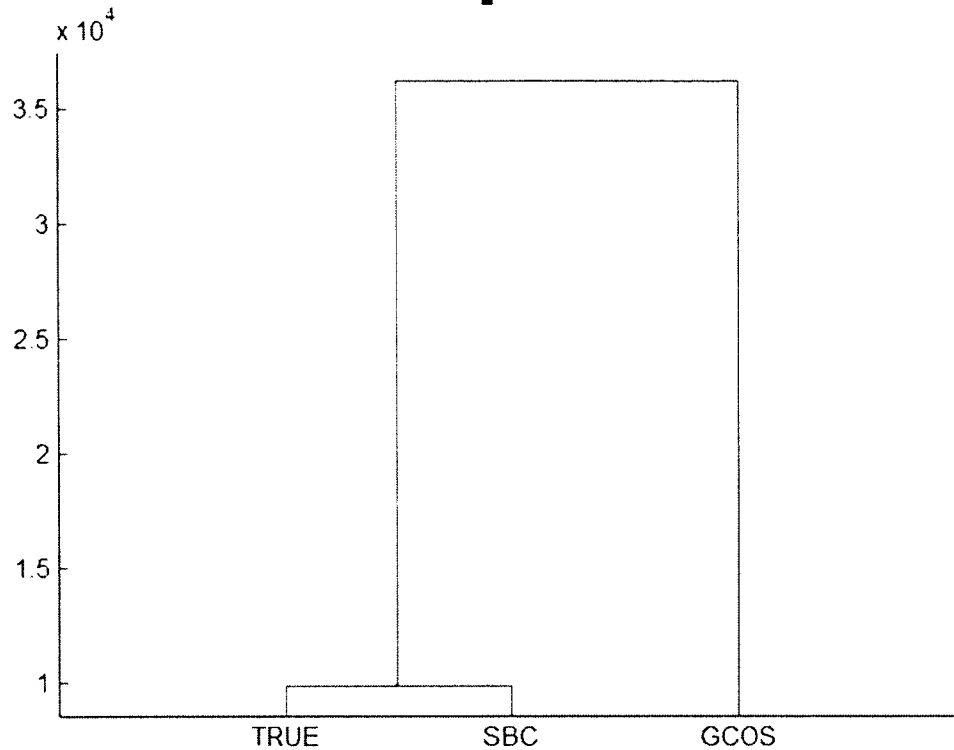


Figure 5.4 Cluster Tree for Canine_a one block simulated image

From the paired t test, two sample t test and clustering analysis, we may conclude that the SBC analyzed gene signal value had less dissimilarity with the true gene signal value compared to the GCOS analyzed gene signal value. Hence, we desired to investigate if the SBC analyzed gene signal value had a better capability to predict the true gene signal value compared to the GCOS analyzed gene signal value. Inverse regression was performed to provide more information on this prediction analysis.

Calibration problem in regression was studied and defined as the inverse regression in [54, 55]. It aimed to build the relationship on the observed data from a known observation of the dependent variable to predict a corresponding explanatory variable. This can be known as the reverse process of common regression. In our research, we aimed at evaluating predicted true gene signal values from observed

analyzed gene signal values. Hence, the calibration inverse regression model was constructed, where true gene signal value was the dependent variable and analyzed gene signal value was the independent variable. In our research, we built two regression models, one had the SBC gene signal value as the independent variable, and the other one had the GCOS gene signal value as the independent variable.

Standard error of performance $\sqrt{\frac{\sum_i^n d_i^2}{n}}$ and R squared are shown in Table 5.8 for these two models. where d_i is the difference between the observed data and the true data. The standard error of performance was a measurement to estimate what type of difference was likely to be between the reference value and the prediction value when the inverse regression model was used introduced by Trevor Hastie [56] and A.M.C. Davies [54].

Table 5.8 Standard error of performance and R squared

	True signal with SBC signal	True signal with GCOS signal
Standard error of performance	49.0563	59.2638
R squared	0.9798	0.9705

From Table 5.8, the smaller standard error of performance and higher R squared are observed, when using SBC analyzed gene signal value as the independent variable. Hence, when we used the SBC analyzed gene signal value to predict the reference true gene signal value, it was likely to have more accurate prediction than we used the GCOS analyzed gene signal value. The SBC gene signal value had better linear relationship with the true gene signal value.

Additionally, we investigated the linear relationship between the analyzed gene signal value and the true gene signal value in regression model. This regression model

provided us more visual information on how well the SBC analyzed gene signal value and the GCOS analyzed gene signal value were correlated with the true gene signal value and their prediction performance. Figure 5.5 and Figure 5.6 present the inverse regression plots for the SBC analyzed gene signal value and the GCOS analyzed gene signal value.

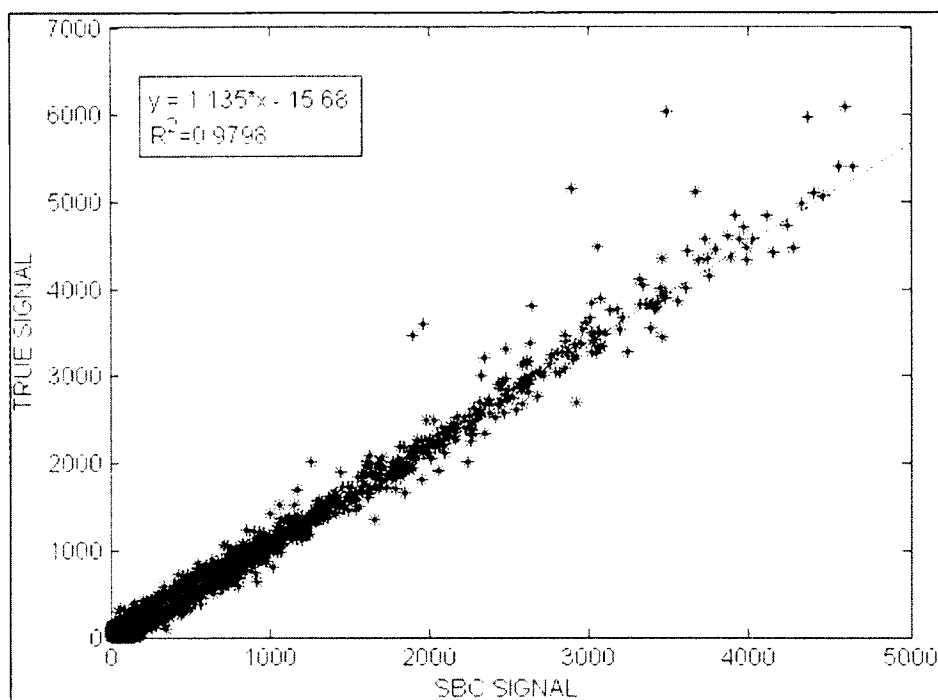


Figure 5.5 Regression plot using SBC signal as independent variable

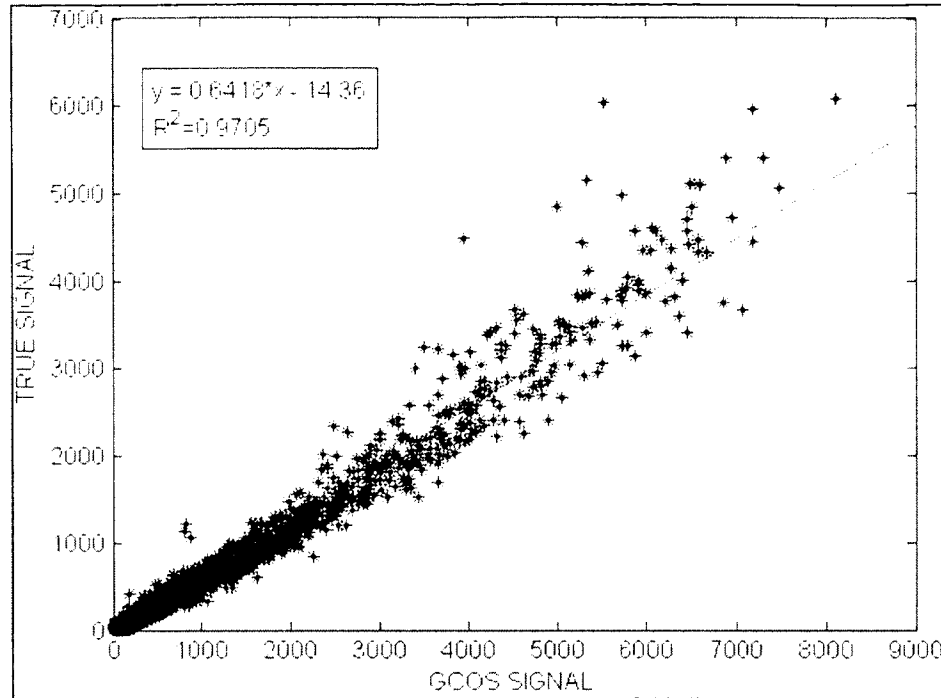


Figure 5.6 Regression plot using GCOS signal as independent variable

Setting the true gene signal value as the dependent variable, we built two inverse regression models; one had the SBC analyzed gene signal value as the independent variable and the other one had the GCOS analyzed gene signal value as the independent variable in the inverse regression model. By comparing these two inverse regression models, it can be shown in Figures 5.5 and 5.6, that SBC signal had a better linear fit correlation with the true signal compared to GCOS signal.

Figures 5.7 and 5.8 present the residual plot for these two inverse regression models to show the model of fit capability. Residuals in Figure 5.7 from SBC signal model were more compressed and evenly distributed around zero horizontal line. However, residuals in Figure 5.8 from GCOS signal model were more scattered and exhibiting a trend, slowly increasing with the increasing number of the predicted value.

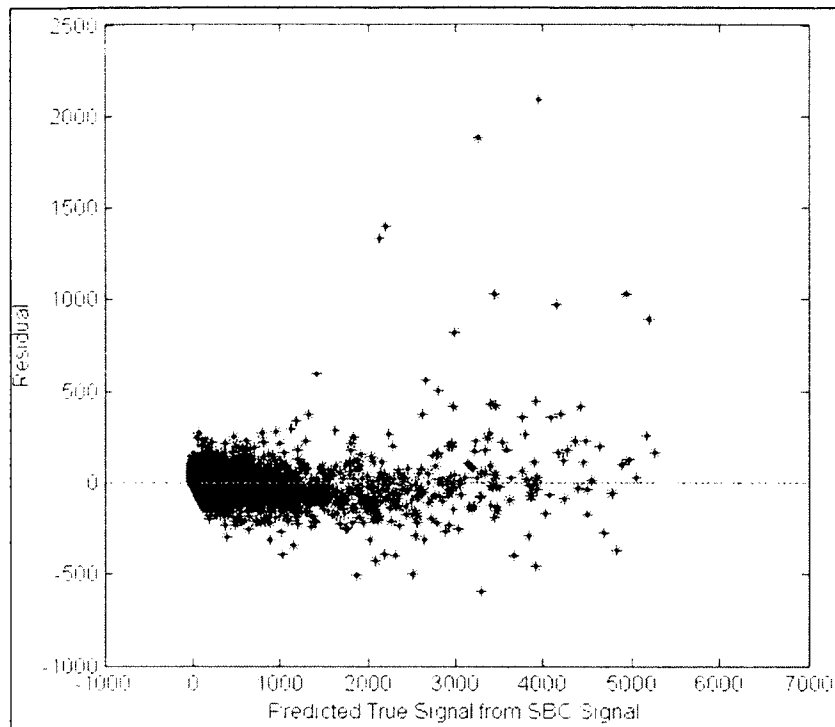


Figure 5.7 Residual plot I using SBC signal as independent variable

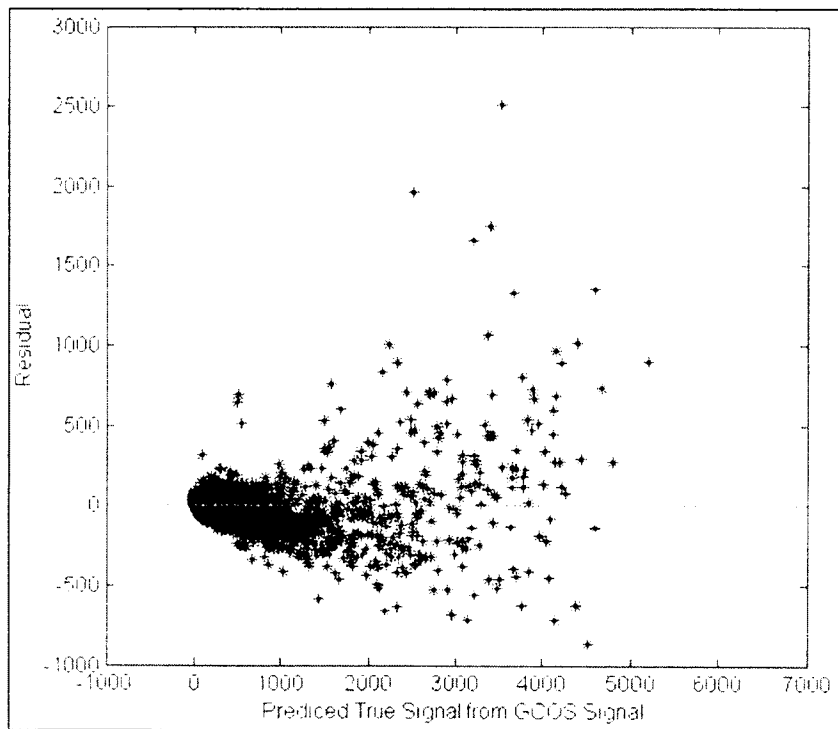


Figure 5.8 Residual plot II using GCOS signal as independent variable

The true gene signal, the SBC analyzed gene signal and the GCOS analyzed gene signal were all measured for the same gene on the expression level in the same microarray experiment. Therefore, straight linear regression relationship was expected between the true gene signal and the analyzed gene signal. By performing the inverse regression analysis, we may concluded that the SBC analyzed gene signal value was more stable to predict the true gene signal value with less standard error of performance and higher R squared. In addition, the SBC analyzed gene signal value had a better linear fit with the true gene signal value. From the above comparison results, we achieved as standard error of performance, R squared and residual plot, SBC analyzed gene signal values were more accurate in estimating and predicting the true gene signal value as compared to the GCOS analyzed gene signal values.

Since one simulated microarray image cannot provide the unbiased comparison analysis results, the unbiased sampling replication number needed to be evaluated based on the appropriate sampling error [57]. A commonly used technique would be to increase the simulation replications, which will approach all the characteristics of the population. Hence, for different microarray images, the standard deviations and average mean μ should be related to the sample replication determination. The sample size determination formula $n = \left(\frac{t \cdot CV}{SE}\right)^2$ and $CV = \frac{\mu}{s}$ was introduced in our research, where at significant level $\alpha = 0.05$. Hence, $t_{0.05}$ is 1.96 and μ and s are determined based on the given population characteristics. Table 5.9 shows the sample replication size table for different sampling error rate.

Table 5.9 Sample replication size table

Array	μ	s	SE = 1%	SE = 0.75%	SE = 0.65%	SE = 0.5%	SE = 0.1%
Canine_a	278	640	20	36	48	81	2029
Bovine_a	486	1690	46	82	110	185	4637
Canine_b	319	722	20	35	47	79	1968
Bovine_b	425	1475	46	82	110	185	4634
Vitis_a	604	1072	12	22	29	48	1211
Vitis_b	633	1113	12	21	28	48	1189
Yeast-1	368	964	26	47	62	105	2629
Yeast-2	378	1001	27	48	64	108	2696

Considering the simulation time and segmentation time for each simulated microarray image, we chose 50 as the replication size, where SE = 0.75% for six out of eight microarray images and sampling error between 0.75% and 1% for the rest two microarray images. Therefore, 50 simulation replications for each microarray image guarantee that SE is below 1%.

Table 5.10 presents the standard error of performance of 50 simulation replication images for Canine_a group. If the standard error of performance for SBC analyzed gene signal value was less than the standard error of performance for GCOS, a value one was assigned on the last column in Table 5.10. If value one was assigned, this indicated that the SBC analyzed gene signal value had a better linear relationship with the true gene signal value as compared to the GCOS analyzed gene signal value. The SBC analyzed gene signal value more nearly approaches to the true gene signal value.

Table 5.10 Standard error performance comparison for Canine_a

Simulation	Standard error of performance (SBC)	Standard error of performance (GCOS)	SBC is better
1	63.76	214.62	1
2	66.12	58.50	0
3	85.78	80.59	0
4	65.41	58.17	0
5	84.84	216.55	1
6	70.29	214.63	1
7	69.29	220.17	1
8	72.93	68.10	0
9	58.30	48.67	0
10	62.26	215.55	1
11	59.22	216.76	1
12	50.16	209.02	1
13	63.59	54.96	0
14	62.35	53.97	0
15	68.72	213.49	1
16	68.91	214.06	1
17	65.04	215.02	1
18	64.28	216.55	1
19	53.71	45.64	0
20	57.65	50.31	0
21	59.95	215.94	1
22	56.27	205.96	1
23	48.56	207.83	1
24	52.54	43.01	0
25	73.99	67.18	0
26	67.41	220.61	1
27	54.21	42.61	0
28	58.99	50.72	0
29	58.56	218.83	1
30	52.67	215.05	1
31	173.00	165.74	0
32	57.30	226.46	1
33	69.81	213.49	1
34	70.73	214.06	1
35	160.85	154.05	0
36	156.12	147.65	0
37	174.09	167.20	0
38	56.08	214.62	1
39	51.54	208.03	1

Table 5.10 Contd.

40	147.66	139.61	0
41	151.15	142.26	0
42	126.72	116.17	0
43	80.21	221.91	1
44	70.38	226.46	1
45	68.83	213.70	1
46	51.85	208.81	1
47	74.08	226.83	1
48	60.81	213.51	1
49	58.16	214.63	1
50	50.58	220.17	1

Table 5.11 presents the Pearson correlation of 50 simulation replication images for Canine_a group. If the Pearson correlation for SBC analyzed gene signal value was less than the Pearson correlation for GCOS, a value one was assigned on the last column in Table 5.9. If value one was assigned, this indicated that the SBC analyzed gene signal value was more correlated to the true gene signal value as compared to the GCOS analyzed gene signal value.

Table 5.11 Pearson correlation comparison for Canine_a

Simulation	Pearson Correlation for SBC	Pearson Correlation for GCOS	SBC is better
1	0.9898	0.9851	1
2	0.9915	0.9951	0
3	0.9916	0.9950	0
4	0.9915	0.9946	0
5	0.9906	0.9841	1
6	0.9905	0.9859	1
7	0.9916	0.9860	1
8	0.9911	0.9946	0
9	0.9916	0.9952	0
10	0.9905	0.9843	1
11	0.9910	0.9856	1
12	0.9920	0.9844	1
13	0.9915	0.9953	0

Table 5.11 Contd.

14	0.9911	0.9946	0
15	0.9914	0.9849	1
16	0.9922	0.9844	1
17	0.9914	0.9863	1
18	0.9917	0.9841	1
19	0.9912	0.9944	0
20	0.9918	0.9947	0
21	0.9918	0.9834	1
22	0.9910	0.9849	1
23	0.9910	0.9855	1
24	0.9915	0.9948	0
25	0.9914	0.9953	0
26	0.9913	0.9859	1
27	0.9896	0.9941	0
28	0.9919	0.9950	0
29	0.9916	0.9849	1
30	0.9918	0.9852	1
31	0.9720	0.9890	0
32	0.9913	0.9854	1
33	0.9918	0.9849	1
34	0.9915	0.9844	1
35	0.9741	0.9890	0
36	0.9740	0.9891	0
37	0.9724	0.9888	0
38	0.9910	0.9851	1
39	0.9920	0.9869	1
40	0.9756	0.9898	0
41	0.9731	0.9877	0
42	0.9768	0.9900	0
43	0.9910	0.9849	1
44	0.9918	0.9854	1
45	0.9911	0.9844	1
46	0.9899	0.9875	1
47	0.9912	0.9847	1
48	0.9909	0.9850	1
49	0.9920	0.9859	1
50	0.9917	0.9860	1

Table 5.12 presents the paired t test for Canine_a group of 50 simulated images performed on $d1 = Signal_{true} - Signal_{SBC}$ with the null hypothesis and alternative hypothesis listed below.

$$H_0: \mu_{d1} = 0 \text{ and } H_1: \mu_{d1} \neq 0.$$

P values are shown for each image on the last column in Table 5.12. The decision results are also shown at the given significance levels, 0.05, 0.01 and 0.001.

When the null hypothesis failed to be rejected, that meant there was not enough evidence to show that the mean difference between the true gene signal value, and the SBC analyzed gene signal value was significantly not zero. Hence, there was no significant difference between the mean of the SBC analyzed gene signal value and the mean of the true gene signal value. When the null hypothesis was rejected, that meant the mean difference between the true gene signal value and the SBC analyzed gene signal value was significantly not zero. Hence, there was significant difference between the mean of the SBC analyzed gene signal value and the mean of the true gene signal value.

Table 5.12 Paired t test for the SBC analyzed gene signal value from Canine_a

Simulation	Decision for $\alpha = 0.05$	Decision for $\alpha = 0.01$	Decision for $\alpha = 0.001$	P-values
1	Reject	Reject	Accept	0.00151394
2	Reject	Accept	Accept	0.01726763
3	Reject	Reject	Reject	< 0.0000001
4	Accept	Accept	Accept	0.30516950
5	Reject	Reject	Reject	< 0.0000001
6	Reject	Reject	Reject	0.00000002
7	Reject	Reject	Reject	< 0.0000001
8	Reject	Reject	Reject	< 0.0000001
9	Reject	Reject	Reject	< 0.0000001
10	Reject	Reject	Accept	0.00493396
11	Reject	Reject	Reject	< 0.0000001
12	Reject	Reject	Reject	< 0.0000001

Table 5.12 Contd.

13	Accept	Accept	Accept	0.62280550
14	Accept	Accept	Accept	0.05122368
15	Reject	Reject	Reject	< 0.0000001
16	Reject	Reject	Reject	< 0.0000001
17	Accept	Accept	Accept	0.22036240
18	Accept	Accept	Accept	0.07014492
19	Reject	Reject	Reject	< 0.0000001
20	Reject	Reject	Reject	< 0.0000001
21	Reject	Reject	Reject	0.00072543
22	Reject	Reject	Reject	< 0.0000001
23	Reject	Reject	Reject	< 0.0000001
24	Reject	Reject	Reject	< 0.0000001
25	Reject	Reject	Reject	< 0.0000001
26	Reject	Reject	Reject	0.00017473
27	Reject	Reject	Reject	< 0.0000001
28	Reject	Reject	Reject	0.00003934
29	Reject	Reject	Reject	0.00000273
30	Reject	Reject	Reject	< 0.0000001
31	Reject	Reject	Reject	< 0.0000001
32	Reject	Reject	Reject	< 0.0000001
33	Reject	Reject	Reject	< 0.0000001
34	Reject	Reject	Reject	< 0.0000001
35	Reject	Reject	Reject	< 0.0000001
36	Reject	Reject	Reject	< 0.0000001
37	Reject	Reject	Reject	< 0.0000001
38	Reject	Reject	Reject	< 0.0000001
39	Reject	Reject	Reject	< 0.0000001
40	Reject	Reject	Reject	< 0.0000001
41	Reject	Reject	Reject	< 0.0000001
42	Reject	Reject	Reject	< 0.0000001
43	Reject	Reject	Reject	< 0.0000001
44	Reject	Reject	Reject	< 0.0000001
45	Reject	Reject	Reject	0.00000003
46	Reject	Reject	Reject	< 0.0000001
47	Reject	Reject	Reject	< 0.0000001
48	Reject	Reject	Accept	0.00416029
49	Reject	Reject	Reject	< 0.0000001
50	Reject	Reject	Reject	< 0.0000001

Table 5.13 presents the paired t test for Canine_a group of 50 simulated images performed on $d2 = Signal_{true} - Signal_{GCOS}$ with the null hypothesis and alternative hypothesis listed below.

$$H_0: \mu_{d2} = 0 \text{ and } H_1: \mu_{d2} \neq 0.$$

P values are shown for each image on the last column in Table 5.13. The decision results are also shown at the given significance levels, 0.05, 0.01 and 0.001.

When the null hypothesis failed to be rejected, that meant there was not enough evidence to show that the mean difference between the true gene signal value, and the GCOS analyzed gene signal value was significantly not zero. Hence, there was no significant difference between the mean of the GCOS analyzed gene signal value and the mean of the true gene signal value. When the null hypothesis was rejected, that meant the mean difference between the true gene signal value and the GCOS analyzed gene signal value was significantly not zero. Hence, there was significant difference between the mean of the GCOS analyzed gene signal value and the mean of the true gene signal value.

Table 5.13 Paired t test for the GCOS analyzed gene signal value from Canine_a

Simulation	Decision for $\alpha = 0.05$	Decision for $\alpha = 0.01$	Decision for $\alpha = 0.001$	P-values
1	Reject	Reject	Reject	< 0.0000001
2	Reject	Reject	Reject	< 0.0000001
3	Reject	Reject	Reject	< 0.0000001
4	Reject	Reject	Reject	< 0.0000001
5	Reject	Reject	Reject	< 0.0000001
6	Reject	Reject	Reject	< 0.0000001
7	Reject	Reject	Reject	< 0.0000001
8	Reject	Reject	Reject	< 0.0000001
9	Accept	Accept	Accept	0.17197370
10	Reject	Reject	Reject	< 0.0000001
11	Reject	Reject	Reject	< 0.0000001

Table 5.13 Contd.

12	Reject	Reject	Reject	< 0.0000001
13	Reject	Reject	Reject	< 0.0000001
14	Reject	Reject	Reject	0.00000795
15	Reject	Reject	Reject	< 0.0000001
16	Reject	Reject	Reject	< 0.0000001
17	Reject	Reject	Reject	< 0.0000001
18	Reject	Reject	Reject	< 0.0000001
19	Reject	Reject	Reject	< 0.0000001
20	Accept	Accept	Accept	0.86454040
21	Reject	Reject	Reject	< 0.0000001
22	Reject	Reject	Reject	< 0.0000001
23	Reject	Reject	Reject	< 0.0000001
24	Reject	Reject	Reject	< 0.0000001
25	Reject	Reject	Reject	< 0.0000001
26	Reject	Reject	Reject	< 0.0000001
27	Reject	Reject	Reject	< 0.0000001
28	Accept	Accept	Accept	0.21564290
29	Reject	Reject	Reject	< 0.0000001
30	Reject	Reject	Reject	< 0.0000001
31	Reject	Reject	Reject	< 0.0000001
32	Reject	Reject	Reject	< 0.0000001
33	Reject	Reject	Reject	< 0.0000001
34	Reject	Reject	Reject	< 0.0000001
35	Reject	Reject	Reject	< 0.0000001
36	Reject	Reject	Reject	< 0.0000001
37	Reject	Reject	Reject	< 0.0000001
38	Reject	Reject	Reject	< 0.0000001
39	Reject	Reject	Reject	< 0.0000001
40	Reject	Reject	Reject	< 0.0000001
41	Reject	Reject	Reject	< 0.0000001
42	Reject	Reject	Reject	< 0.0000001
43	Reject	Reject	Reject	< 0.0000001
44	Reject	Reject	Reject	< 0.0000001
45	Reject	Reject	Reject	< 0.0000001
46	Reject	Reject	Reject	< 0.0000001
47	Reject	Reject	Reject	< 0.0000001
48	Reject	Reject	Reject	< 0.0000001
49	Reject	Reject	Reject	< 0.0000001
50	Reject	Reject	Reject	< 0.0000001

Table 5.14 presents the Minkowski distance metric comparison for 50 simulation replication images of Canine_a group. This Minkowski distance is a generalization of both the Euclidean distance and the Manhattan distance. If the Minkowski distance for SBC analyzed gene signal value was less than the Minkowski distance for GCOS, a value one was assigned on the last column in Table 5.14. If value one was assigned, this indicated that the SBC analyzed gene signal value had less dissimilarity with the true gene signal value as compared to the GCOS analyzed gene signal value.

Table 5.14 Minkowski distance comparison for Canine_a

Simulation	Minkowski Distance for SBC	Minkowski Distance for GCOS	SBC is better
1	4211.80	11319.20	1
2	3879.50	3620.41	0
3	5030.95	4884.74	0
4	3909.66	3724.42	0
5	5215.08	11588.73	1
6	4371.60	11329.68	1
7	4022.55	11661.54	1
8	4290.29	4301.41	1
9	3352.41	2998.08	0
10	3854.42	11495.30	1
11	3608.77	11470.49	1
12	2817.01	11138.84	1
13	3694.73	3451.64	0
14	3707.36	3485.22	0
15	4047.60	11290.95	1
16	3932.16	11426.56	1
17	3793.59	11403.90	1
18	3802.05	11588.73	1
19	3152.21	2995.43	0
20	3347.84	3216.32	0
21	3428.99	11545.44	1
22	3297.30	10996.41	1
23	2661.52	11043.57	1
24	3090.47	2693.76	0
25	4321.79	4093.69	0

Table 5.14 Contd.

26	3974.47	11728.18	1
27	3811.73	3071.12	0
28	3364.70	3144.85	0
29	3419.27	11619.84	1
30	3066.89	11487.53	1
31	9919.52	9685.08	0
32	3423.76	12048.97	1
33	4027.10	11290.95	1
34	4162.12	11426.56	1
35	9095.63	9003.37	0
36	8857.26	8602.88	0
37	9871.26	9686.59	0
38	3413.36	11319.20	1
39	2854.86	11025.45	1
40	8341.22	8165.39	0
41	8545.57	8320.00	0
42	7091.48	6878.61	0
43	4803.63	11759.76	1
44	4095.02	12048.97	1
45	4025.67	11385.35	1
46	3346.63	11067.18	1
47	4393.55	12067.78	1
48	3596.24	11328.69	1
49	3281.92	11329.68	1
50	2810.19	11661.54	1

Table 5.15 presents the Euclidean distance metric comparison for 50 simulation replication images of Canine_a group. If the Euclidean distance for SBC analyzed gene signal value was less than the Euclidean distance for GCOS, a value one was assigned on the last column in Table 5.15. If value one was assigned, this indicated that the SBC analyzed gene signal value had less dissimilarity with the true gene signal value as compared to the GCOS gene signal value.

Table 5.15 Euclidean distance metric comparison for Canine_a

Simulation	Euclidean distance for SBC	Euclidean distance for GCOS	SBC is better
1	9860.32	33188.65	1
2	10224.29	9046.84	0
3	13265.08	12461.79	0
4	10114.46	8994.62	0
5	13120.15	33487.29	1
6	10869.69	33189.58	1
7	10714.73	34046.66	1
8	11277.45	10530.10	0
9	9014.94	7526.40	0
10	9627.11	33332.33	1
11	9156.99	33519.05	1
12	7757.08	32322.46	1
13	9832.91	8498.66	0
14	9642.11	8345.75	0
15	10626.21	33014.10	1
16	10656.60	33102.52	1
17	10058.29	33250.46	1
18	9940.32	33487.29	1
19	8305.48	7057.04	0
20	8914.60	7780.56	0
21	9270.79	33392.16	1
22	8701.43	31848.69	1
23	7509.78	32138.96	1
24	8124.77	6651.55	0
25	11441.62	10388.90	0
26	10424.26	34115.29	1
27	8382.38	6588.64	0
28	9121.46	7843.03	0
29	9056.30	33839.99	1
30	8145.14	33255.69	1
31	26752.48	25629.96	0
32	8860.87	35019.78	1
33	10795.52	33014.10	1
34	10937.98	33102.52	1
35	24873.14	23821.47	0
36	24141.60	22831.65	0
37	26921.69	25855.56	0
38	8672.50	33188.65	1
39	7969.38	32169.83	1

Table 5.15 Contd.

40	22833.23	21588.95	0
41	23374.32	21999.50	0
42	19595.76	17963.76	0
43	12403.41	34316.12	1
44	10883.20	35019.78	1
45	10644.51	33045.81	1
46	8017.38	32290.09	1
47	11456.02	35075.94	1
48	9403.27	33016.23	1
49	8993.18	33189.58	1
50	7821.88	34046.66	1

Table 5.16 presents the correlation distance metric comparison for 50 simulation replication images of Canine_a group. If the correlation distance for SBC analyzed gene signal value was less than the correlation distance for GCOS, a value one was assigned on the last column in Table 5.16. If value one was assigned, this indicated that the SBC analyzed gene signal value had less dissimilarity with the true gene signal value as compared to the GCOS gene signal value.

Table 5.16 Correlation distance metric comparison results for Canine_a

Simulation	Correlation distance for SBC	Correlation distance for GCOS	SBC is better
1	702602.70	1986828.00	1
2	724797.10	541573.70	0
3	826781.00	659337.30	0
4	725176.00	537168.40	0
5	817402.00	1987510.00	1
6	740425.00	1983415.00	1
7	745560.20	2026214.00	1
8	767690.30	585245.90	0
9	693226.20	501522.60	0
10	699760.50	1978458.00	1
11	689894.90	1994482.00	1
12	663211.50	1923072.00	1
13	722084.10	528692.90	0

Table 5.16 Contd.

14	704024.10	519750.60	0
15	740293.80	1960800.00	1
16	748187.50	1971284.00	1
17	728175.00	1984374.00	1
18	724231.50	1987510.00	1
19	672622.70	482240.80	0
20	692837.70	503588.20	0
21	703038.70	1982577.00	1
22	683651.10	1899540.00	1
23	677806.10	1914651.00	1
24	662901.80	478672.20	0
25	765293.50	588251.50	0
26	731935.30	2014574.00	1
27	665706.30	477797.10	0
28	695688.50	508171.50	0
29	691885.90	2012252.00	1
30	662983.50	1972600.00	1
31	1489280.00	1256894.00	0
32	689086.50	2068795.00	1
33	747323.80	1960800.00	1
34	747538.30	1971284.00	1
35	1442715.00	1187092.00	0
36	1411510.00	1154407.00	0
37	1501049.00	1273509.00	0
38	675912.20	1986828.00	1
39	670624.70	1924983.00	1
40	1371311.00	1107114.00	0
41	1392729.00	1126679.00	0
42	1289697.00	971279.50	0
43	795094.70	2042043.00	1
44	756395.00	2068795.00	1
45	746655.20	1967977.00	1
46	672355.80	1919913.00	1
47	760242.10	2083066.00	1
48	699960.30	1969807.00	1
49	692344.90	1983415.00	1
50	661885.10	2026214.00	1

Table 5.17 presents the Chebychev distance metric comparison for 50 simulation replication images of Canine_a group. If the Chebychev distance for SBC analyzed gene signal value was less than the Chebychev distance for GCOS, a value one was assigned on the last column in Table 5.17. If value one was assigned, this indicated that the SBC analyzed gene signal value had less dissimilarity with the true gene signal value as compared to the GCOS gene signal value.

Table 5.17 Chebychev distance metric comparison for Canine_a

Simulation	Chebychev distance for SBC	Chebychev distance for GCOS	SBC is better
1	4211.80	11319.20	1
2	3879.50	3620.41	0
3	5030.95	4884.74	0
4	3909.66	3724.42	0
5	5215.08	11588.73	1
6	4371.60	11329.68	1
7	4022.55	11661.54	1
8	4290.29	4301.41	1
9	3352.41	2998.08	0
10	3854.42	11495.30	1
11	3608.77	11470.49	1
12	2817.01	11138.84	1
13	3694.73	3451.64	0
14	3707.36	3485.22	0
15	4047.60	11290.95	1
16	3932.16	11426.56	1
17	3793.59	11403.90	1
18	3802.05	11588.73	1
19	3152.21	2995.43	0
20	3347.84	3216.32	0
21	3428.99	11545.44	1
22	3297.30	10996.41	1
23	2661.52	11043.57	1
24	3090.47	2693.76	0
25	4321.79	4093.69	0
26	3974.47	11728.18	1
27	3811.73	3071.12	0

Table 5.17 Contd.

28	3364.70	3144.85	0
29	3419.27	11619.84	1
30	3066.89	11487.53	1
31	9919.52	9685.08	0
32	3423.76	12048.97	1
33	4027.10	11290.95	1
34	4162.12	11426.56	1
35	9095.63	9003.37	0
36	8857.26	8602.88	0
37	9871.26	9686.59	0
38	3413.36	11319.20	1
39	2854.86	11025.45	1
40	8341.22	8165.39	0
41	8545.57	8320.00	0
42	7091.48	6878.61	0
43	4803.63	11759.76	1
44	4095.02	12048.97	1
45	4025.67	11385.35	1
46	3346.63	11067.18	1
47	4393.55	12067.78	1
48	3596.24	11328.69	1
49	3281.92	11329.68	1
50	2810.19	11661.54	1

In Table 5.18, we present the summary comparison of standard error of performance and correlation coefficients for all the 50 simulated microarray images of Canine_a genome. Table 5.18 illustrates that for each of these two metrics, the correlation and standard error of performance, there are 60% of the cases that SBC was a better segmentation method as compared to GCOS.

Table 5.18 Summary comparison of standard error of performance and correlation

Canine_a	No. of Images show SBC is better	Ratio
Correlation	30 out of 50	60%
Standard error of performance	30 out of 50	60%

In Table 5.19 and Table 5.20, we present the summary comparison of the paired t test for all the 50 simulated microarray images of Canine_a genome. Table 5.19 shows the summary comparison of paired t test for SBC. Table 5.20 shows that summary comparison of paired t test for GCOS. At each significance level, there were more images showing that we had not enough evidence to reject the null hypothesis in Table 5.19 as compared in Table 5.20. Hence, the gene signal value obtained from SBC had a higher percentage of the cases more approach to the true gene signal value. The SBC provided more accurate gene signal values as compared to GCOS.

Table 5.19 Summary comparison of paired t test for SBC for Canine_a

Canine_a	No. of Images show Accept for SBC	Ratio
Paired t test at 0.05 level	five out of 50	10%
Paired t test at 0.01 level	six out of 50	12%
Paired t test at 0.001 level	nine out of 50	18%

Table 5.20 Summary comparison of paired t test for GCOS for Canine_a

Canine_a	No. of Images show Accept for GCOS	Ratio
Paired t test at 0.05 level	three out of 50	6%
Paired t test at 0.01 level	three out of 50	6%
Paired t test at 0.001 level	three out of 50	6%

In Table 5.21, we present the summary comparison of the clustering distance metrics for all the 50 simulated microarray images of Canine_a genome. In each of these distance metrics, the gene signal value obtained from SBC had a higher percentage of the cases than it had less dissimilarity with the true gene signal value. Hence, SBC provided more accurate analyzed gene signal as compared to GCOS.

Table 5.21 Summary comparison of clustering distance metrics for Canine_a

Canine_a	No. of Images show SBC is better	Ratio
Minkowski distance	31 out of 50	62%
Euclidean distance	30 out of 50	60%
Correlation distance	30 out of 50	60%
Chebychev distance	31 out of 50	62%

To validate this improvement in a more global level, we propose the average analyzed gene signal value for both SBC and GCOS $AverageSignal_{SBC}$ and $AverageSignal_{GCOS}$. We started with the average analyzed gene values of 50 simulated microarray images from Canine_a genome. For each gene, we computed the

$AverageSignal_{SBC}$ and $AverageSignal_{GCOS}$. Then, we had two sets of analyzed average signal value with one set from SBC and the other set from GCOS. Similar statistical analysis was performed on these two sets and the set of true gene signal value. T test, standard error performance, correlation coefficient and clustering distance were investigated for this average analyzed gene values.

$$AverageSignal_{SBC} = \frac{\sum_{i=1}^{50} Signal_{SBC}}{50}. \quad (5.5)$$

$$AverageSignal_{GCOS} = \frac{\sum_{i=1}^{50} Signal_{GCOS}}{50}. \quad (5.6)$$

Two sample t test with two tails was performed between the mean of $AverageSignal_{SBC}$ and the true gene signal value with the null hypothesis and alternative hypothesis are shown below.

$$H_0: \mu_{AverageSignal_{SBC}} = \mu_{TrueGeneSignal}$$

$$H_1: \mu_{AverageSignal_{SBC}} \neq \mu_{TrueGeneSignal}$$

Table 5.22 shows that we had not enough evidence to reject the null hypothesis. Hence, there was no significant difference between the mean of the SBC average analyzed gene signal value and the mean of the true gene signal value.

Table 5.22 Two sample t test for the SBC averaged analyzed gene signal value

Significant level	Decision for P-value = 0.4267
$\alpha = 0.05$	Accept
$\alpha = 0.01$	Accept
$\alpha = 0.001$	Accept

Next, we also performed the two sample t test between the mean of $AverageSignal_{GCOS}$ and the set of true gene signal value with the null hypothesis and alternative hypothesis are shown below.

$$H_0: \mu_{AverageSignalGCOS} = \mu_{TrueGeneSignal}$$

$$H_1: \mu_{AverageSignalGCOS} \neq \mu_{TrueGeneSignal}$$

In Table 5.23, the null hypothesis was rejected at all levels. Therefore, there was significant difference between the mean of the GCOS average analyzed gene signal value and the mean of the true gene signal value. We may conclude that the SBC average analyzed gene signal value more approached the true gene signal value on average when compared to the GCOS average analyzed gene signal value.

Table 5.23 Two sample t test for the GCOS averaged analyzed gene signal value

Significant level	Decision for P-value < 0.00001
$\alpha = 0.05$	Reject
$\alpha = 0.01$	Reject
$\alpha = 0.001$	Reject

In addition to the two sample t test performed between the average analyzed gene signal value and true gene signal value, we also want to evaluate the absolute value of difference $D11$ and $D12$ in more detail.

$$D11 = |AverageSignal_{SBC} - TrueGeneSignal|. \quad (5.7)$$

$$D12 = |AverageSignal_{GCOS} - TrueGeneSignal|. \quad (5.8)$$

We compared these two differences after taking the absolute value sign in order to understand on “how far” but not on “in which direction.” Our main interest was to decrease the difference between the true gene signal value and the average analyzed gene signal value. Hence, two sample t test with left tails was performed on $D11$ and $D12$ with the null hypothesis was that there was no significant difference between the mean of $D11$ and the mean of $D12$ based on the given the significance levels, 0.05, 0.01 and 0.001.

$$H_0: \mu_{D11} = \mu_{D12}.$$

$$H_1: \mu_{D11} < \mu_{D12}.$$

This left tail test could provide us more information to know if the mean difference between true gene signal value and the average SBC analyzed gene signal value is smaller than the mean difference between the true gene signal value and the average GCOS analyzed gene signal value.

From Table 5.24, the null hypothesis H_0 was rejected at all significance levels, which indicated that we might accept the alternative hypothesis H_1 for $\mu_{D11} < \mu_{D12}$. Thus, there was significant difference between the mean of $D11$ and the mean of $D12$, and the mean of $D11$ was significantly less than the mean of $D12$. Hence, from this two sample t test, the difference between the true gene signal value and the average SBC gene signal value was shown significantly smaller than the difference between the true gene signal value and the average GCOS gene signal value averagely. The gene signal value obtained from SBC more nearly approached the true gene signal values compared to the gene signal values obtained from GCOS on average. Besides the two sample t test, we also presented the boxplot for $D11$ and $D12$ shown in Figure 5.9 to provide more information in visual. Therefore, we could have a better understanding on the comparison between $D11$ and $D12$. The commonly used quartiles for $D11$ and $D12$ were described in Table 5.25.

Table 5.24 Two sample t test for $D11$ and $D12$

Significant level	Decision for P-value < 0.00001
$\alpha = 0.05$	Reject
$\alpha = 0.01$	Reject
$\alpha = 0.001$	Reject

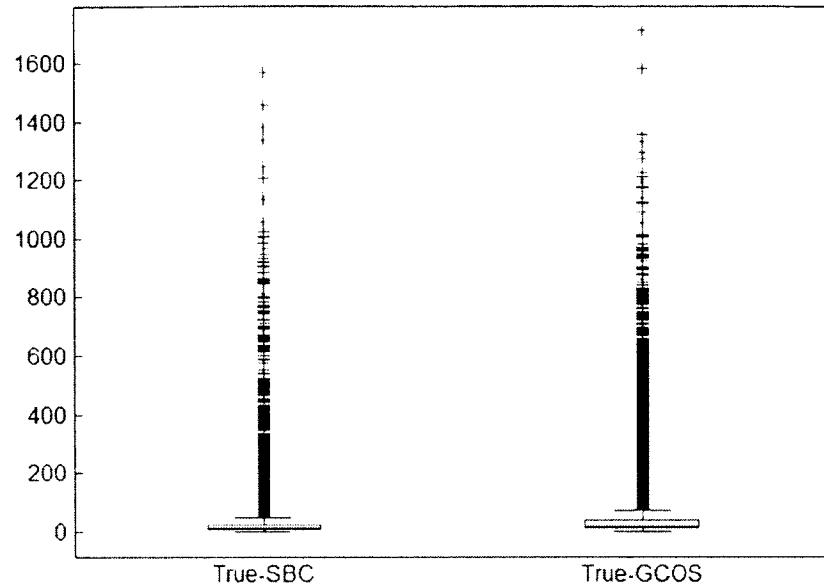


Figure 5.9 Boxplot for the absolute value of $D11$ and $D12$

Table 5.25 Quartiles comparison information for $D11$ and $D12$

Quartiles	0.025	0.25	0.5	0.75	0.975	1
$D11$ for $ \text{true-SBC} $	1.18	9.21	15.51	23.60	111.53	1566.67
$D12$ for $ \text{true-GCOS} $	3.26	11.96	19.54	35.84	275.78	1710.31

The boxplot showed that the interquartile range and the total range of $D11$ were both smaller than those of $D12$. In addition, the mean of $D11$ was smaller than the mean of $D12$, which was consistent with the previous two sample t test result. Hence, from Figure 5.9 and Table 5.25, we may conclude that the absolute difference between the true gene signal value and the average SBC gene signal value was smaller compared to the absolute difference between the true gene signal value and the average GCOS gene signal value. Therefore, there was less dissimilarity between the true gene signal value and the average SBC gene signal value. SBC segmentation method provided more accurate gene signal values as compared to GCOS.

After this hypothesis test analysis, we continued our research by performing the similar comparison as what we did for one simulated microarray image, such as clustering analysis and inverse regression analysis to investigate the dissimilarity and prediction capability between the true gene signal value and the average analyzed gene signal value obtained from SBC and GCOS.

There are two important options when performing the clustering analysis. One is the choice of the distance used to evaluate the dissimilarity between different groups of data. In this option, Sum of absolute difference $\sum_i^n |d_i|$ (Manhattan distance), Person correlation distance $1 - \frac{Z_{score}(observed\ data) * Z_{score}(true\ data)}{n}$ and Euclidean distance $\sqrt{\sum_i^n d_i^2}$ are the most important metrics used in clustering analysis. In addition, we also included the Minkowski distance, which was the generalization form of Euclidean distance. The other one was the choice of the clustering technique on how to classify different groups of data based on the distance measurements obtained from the first option.

Table 5.26 lists all the distance metrics we calculated between the true gene signal value and the average analyzed gene signal value obtained from SBC ($AverageSignal_{SBC}$) and between the true gene signal value and average analyzed gene signal value obtained from GCOS ($AverageSignal_{GCOS}$). Table 5.26 indicates that the average analyzed gene signal value obtained from SBC had smaller dissimilarity with the true gene signal value compared to the average analyzed gene signal value obtained from GCOS. In addition, the average analyzed gene signal value obtained from SBC had higher correlation with the true gene signal value compared to the analyzed gene signal

value obtained from GCOS. Hence, we may conclude that SBC provided more accurate gene signal value closer to the true gene signal value on average as compared to GCOS.

Table 5.26 Comparison for averaged analyzed gene signal for Canine_a group

Metrics	Average SBC analyzed gene signal value	Average GCOS analyzed gene signal value
Standard error of performance	64.8206	101.7257
Euclidean distance	10023.75	15730.69
Correlation	0.9975975	0.9923104
Manhattan distance	610965.9	1054203
Minkowski distance	3973.867	5278.628

For Canine_a simulated microarray image, several commonly used difference metrics were investigated in Table 5.26. The SBC average analyzed gene signal value had the less dissimilarity with true gene signal value compared to the GCOS average analyzed gene signal value. Hence, we wished conduct the cluster analysis which could allow us to have a better understanding that if the SBC average analyzed gene signal value and true gene signal value could be classified in the same group based on this smaller dissimilarity.

Similarly, we chose Unweight Pair Group Method with Arithmetic Mean as bottom up agglomerative hierarchical clustering algorithm to measure the dissimilarity among different objects. This agglomerative hierarchical clustering algorithm is established to be the most commonly used cluster technique in bioinformatics gene expression data mining analysis. The bottom up scheme starts from the individual patterns and combines similar group together, ending up to the root based on the difference metrics selected as the measurement. Euclidean difference metric was chosen

as the dissimilarity measurement for the clustering algorithm as classifying different groups of gene signal values.

Figure 5.10 shows the dendrogram cluster tree plot after implementing the hierarchical clustering algorithm on $AverageSignal_{SBC}$, $AverageSignal_{GCOS}$ and true gene signal value, based on the Euclidean difference metric. The three sets of data were automatically divided into two different groups by average linkage clustering criterion. The SBC average analyzed gene signal values and true gene signal values were classified in one group. The GCOS average analyzed gene signal values were classified in another different group. Therefore, the SBC average analyzed gene signal values had significant less dissimilarity with the true gene signal values compared to the GCOS average analyzed gene signal values. The SBC average analyzed gene signal value more nearly approached the true gene signal value when compared to the GCOS average analyzed gene signal value.

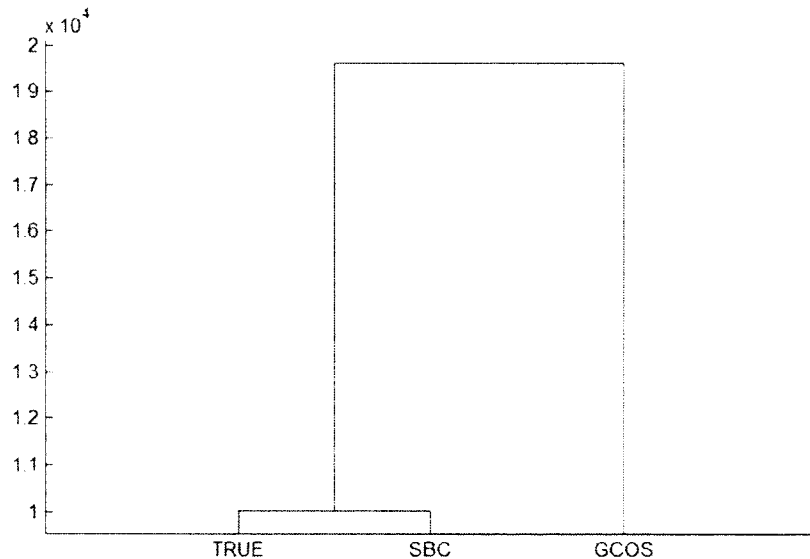


Figure 5.10 Clustering tree plot for the average analyzed gene signal value

From two sample t test and clustering analysis, we concluded that the SBC average analyzed gene signal value had less dissimilarity with the true gene signal value compared to the GCOS average analyzed gene signal value. Hence, we investigated if the SBC average analyzed gene signal value had a better capability to predict the true gene signal value when compared to the GCOS average analyzed gene signal value. Inverse regression was performed to provide more information on this prediction analysis. The inverse regression aimed to build the relationship on the observed data from a known observation of the dependent variable to predict a corresponding explanatory variable. In our research, we aimed at evaluating predicted true gene signal values from observed average analyzed gene signal values. Hence, the calibration inverse regression model was constructed, where true gene signal value was the dependent variable and the average analyzed gene signal value was the independent variable.

Setting the true gene signal value as the dependent variable, we built two inverse regression models, with one having the SBC average analyzed gene signal value ($AverageSignal_{SBC}$) as the independent variable and the other one having the GCOS average analyzed gene signal value ($AverageSignal_{GCOS}$) as the independent variable. By comparing these two inverse regression models, it can be shown in Figure 5.11 and Figure 5.12 that the SBC average gene signal value had a better linear fit correlation with the true signal compared to the GCOS average gene signal value. This indicated that the SBC average analyzed gene signal value was more nearly approaching to the true gene signal value as compared to the GCOS average analyzed gene signal value.

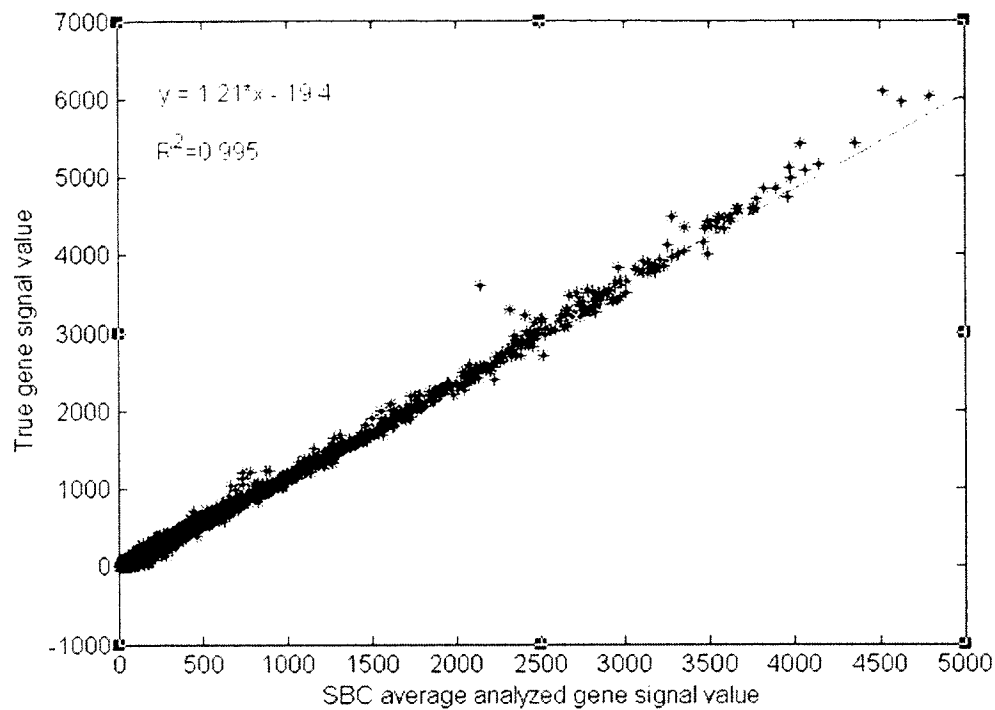


Figure 5.11 Regression plot for average SBC signal

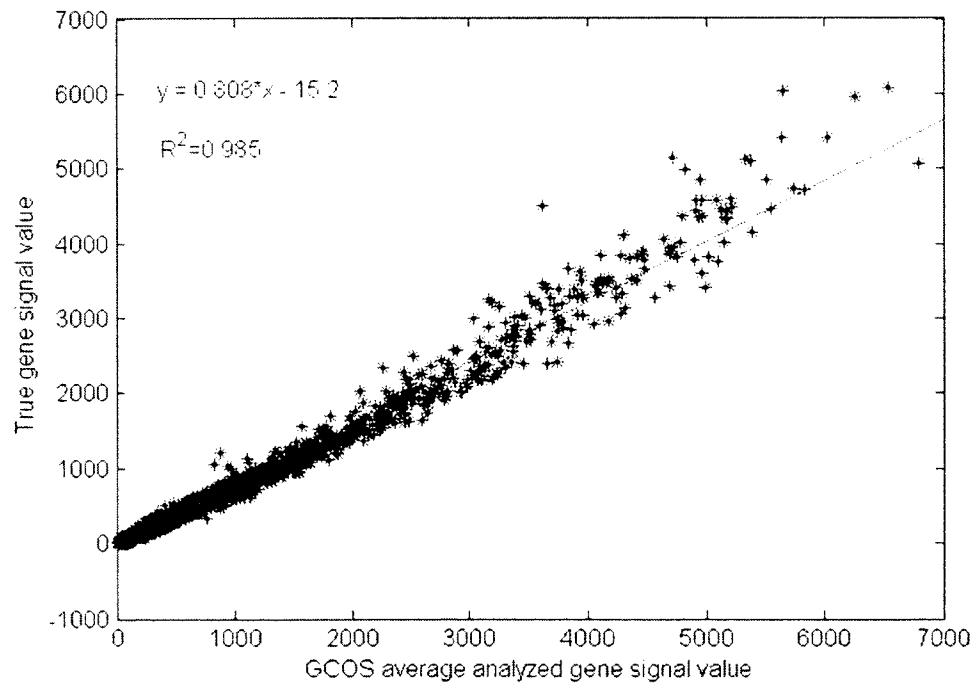


Figure 5.12 Regression plot for average GCOS signal

Figures 5.13 and 5.14 present the residual plots for these two inverse regression models to show the model of fit capability. Residuals in Figure 5.13 from the SBC average signal model were more compressed and evenly distributed around zero horizontal line. However, residuals in Figure 5.14 from the GCOS average signal model were more scattered everywhere and slowly increasing with the increasing number of the predicted value.

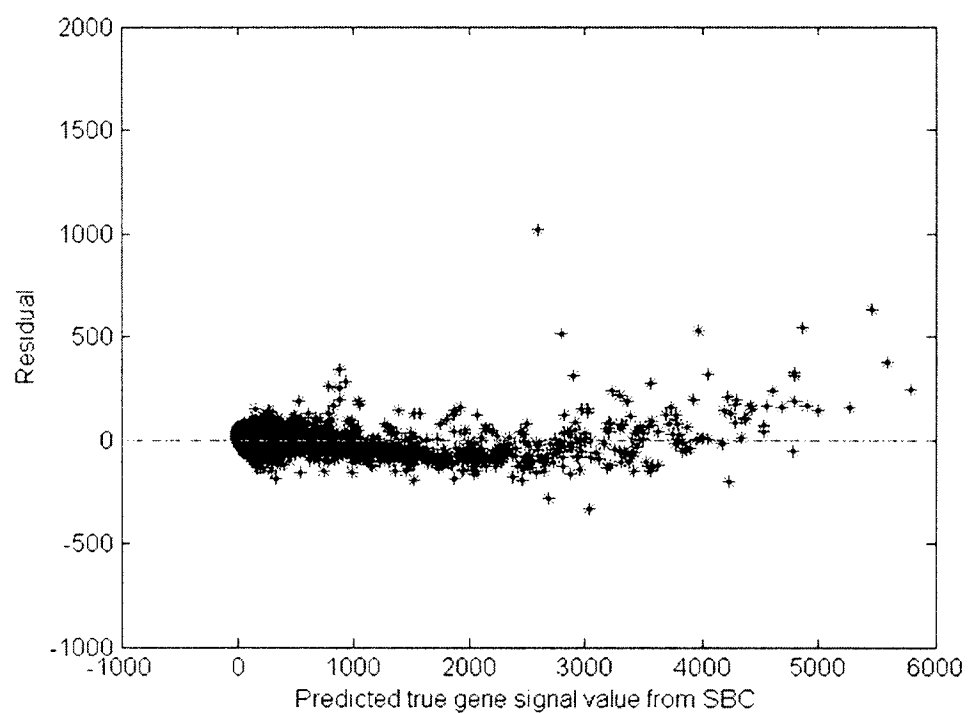


Figure 5.13 Residual plot from average SBC

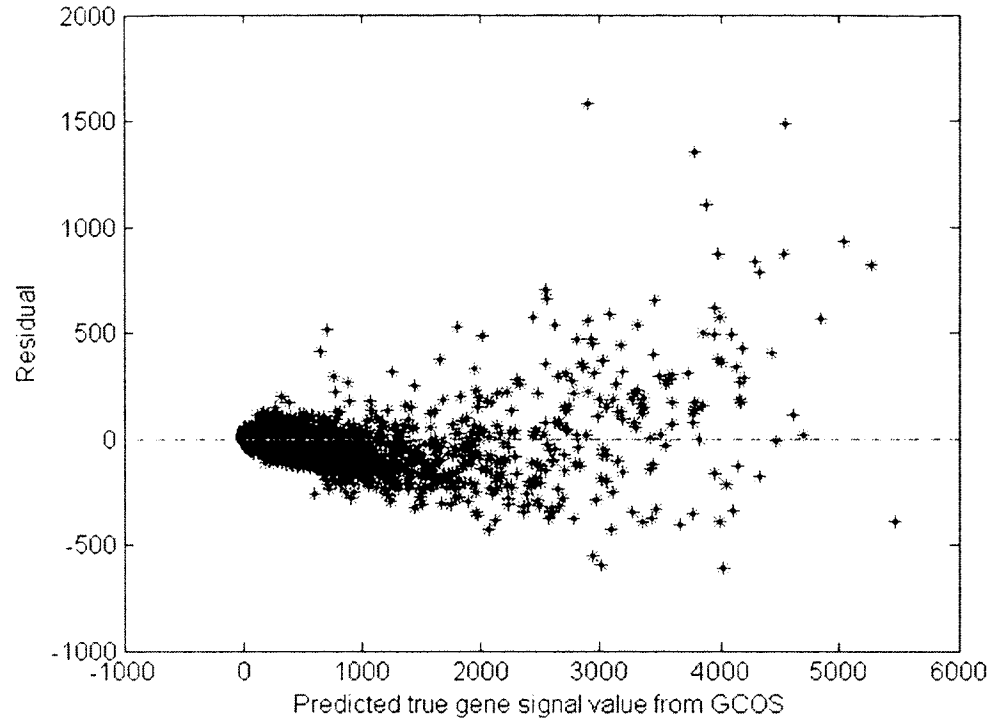


Figure 5.14 Residual plot from average GCOS

The true gene signal, the SBC average analyzed gene signal and the GCOS average analyzed gene signal were all measured for the same gene on the expression level. Therefore, straight linear regression relationship was expected between the true gene signal and the average analyzed gene signal. From Figures 5.11 and 5.12, the SBC average analyzed gene signal value was more correlated with the true gene signal value along with the expected straight line. Residuals between the true gene signal value and the average analyzed gene signal values were investigated for both SBC and GCOS. Figures 5.13 and 5.14 show that using the SBC average analyzed gene signal value to predict the reference gene signal value led to narrow residual range compressed around the zero horizontal line. However, using the GCOS average analyzed gene signal value to predict the reference gene signal value generated more scattered residual spreading over the plot. By performing the inverse regression analysis, we concluded that the SBC

average analyzed gene signal value was more stable to predict the true gene signal value. In addition, the SBC average analyzed gene signal value had a better linear fit with the true gene signal value. From the above comparison results, we achieved in standard error of performance, R squared and residual plot, the SBC average analyzed gene signal values were more accurate to approach the true gene signal value compared to the GCOS average analyzed gene signal values.

Therefore, for Canine_a genome, we concluded that the SBC analyzed gene signal value was more accurate and had less dissimilarity with true gene signal value compared to the GCOS analyzed gene signal value on average.

In our research, there were eight different Affymetrix microarray images. We started with analyzing genome Canine_a after 50 simulation replications. Similarly, 50 simulation replications were generated and analyzed for the other different Affymetrix microarray images. Next, SBC and GCOS were performed on all of these simulated images. Two sets of analyzed gene signal values obtained from SBC and GCOS were compared with the true gene signal value by statistical methods similarly as what we did above.

Table 5.27 presents the summary comparison of gene signal value obtained from SBC and GCOS for the all the eight microarray image groups, Bovine_a, Bovine_b, Canine_a, Canine_b, Vitis_a, Vitis_b, Yeast-1 and Yeast-2.

Table 5.27 Summary comparison for eight groups simulated images

Array	Simulation	Standard error of performance shows SBC better	Euclidean distance shows SBC better	Minkowski distance shows SBC better
Bovine_a	50	19	18	18
Bovine_b	50	1	1	5
Canine_a	50	30	30	31
Canine_b	50	30	30	31
Vitis_a	50	0	0	0
Vitis_b	50	17	17	17
Yeast-1	50	30	30	35
Yeast-2	50	30	30	33

In these eight groups of Affymetrix microarray images, there were four out of eight groups showing that there are more images analyzed by SBC have smaller standard error of performance, Euclidean distance and Minkowski distance compared to GCOS. Hence, for these four groups of Affymetrix microarray images, Canine_a, Canine_b, Yeast-1 and Yeast-2, SBC provided the gene signal values which were closer to the true gene signal value compared to GCOS. For the rest of images, Bovine_a, Bovine_b, Vitis_a and Vitis_b, GCOS provided more accurate results. Hence, a hybrid method which chooses dynamically between SBC and GCOS may be more useful when analyzing different Affymetrix microarray genome image.

In order to show this improvement on a more global level, we proposed the average analyzed gene signal value for both SBC and GCOS, $AverageSignal_{SBC}$ and $AverageSignal_{GCOS}$ from 50 simulated microarray images for all eight different Affymetrix microarray images. For each gene in one genome group, there were 50 analyzed values obtained from SBC and GCOS. Hence, an average mean value for each gene was calculated over 50 images for SBC and GCOS. By comparing this average

mean value of each gene to the true gene signal value, it provided us a more comprehensive understanding of how different algorithm influences the gene expression calculation on a global level.

Paired t test was performed between the average analyzed gene signal value and the true gene signal value for all eight different Affymetrix microarray images. We hypothesized that the difference between the mean of the SBC average analyzed gene signal value and the mean of the true gene signal value was zero and the difference between the mean of the GCOS average analyzed gene signal value and the true gene signal value was zero. Table 5.28 presents the paired t test results for all eight different Affymetrix microarray images between the mean of the SBC average analyzed gene signal value and the true gene signal value. Table 5.29 presents the paired t test results for all eight different Affymetrix microarray images between the mean of the GCOS average analyzed gene signal value and the true gene signal value.

Table 5.28 Paired t test for the SBC average analyzed gene signal value

Array	P-value	Decision for $\alpha = 0.05$	Decision for $\alpha = 0.01$	Decision for $\alpha = 0.001$
Bovine_a	< 0.000001	Reject	Reject	Reject
Bovine_b	< 0.000001	Reject	Reject	Reject
Canine_a	< 0.000001	Reject	Reject	Reject
Canine_b	< 0.000001	Reject	Reject	Reject
Vitis_a	< 0.000001	Reject	Reject	Reject
Vitis_b	< 0.000001	Reject	Reject	Reject
Yeast-1	< 0.000001	Reject	Reject	Reject
Yeast-2	< 0.000001	Reject	Reject	Reject

Table 5.29 Paired t test for the GCOS average analyzed gene signal value

Array	P-value	Decision for $\alpha = 0.05$	Decision for $\alpha = 0.01$	Decision for $\alpha = 0.001$
Bovine_a	< 0.000001	Reject	Reject	Reject
Bovine_b	< 0.000001	Reject	Reject	Reject
Canine_a	< 0.000001	Reject	Reject	Reject
Canine_b	< 0.000001	Reject	Reject	Reject
Vitis_a	< 0.000001	Reject	Reject	Reject
Vitis_b	< 0.000001	Reject	Reject	Reject
Yeast-1	0.0034	Reject	Reject	Accept
Yeast-2	< 0.000001	Reject	Reject	Reject

Table 5.28 shows that the difference for all eight Affymetrix microarray images between the mean of the SBC average analyzed gene signal value and the true gene signal value was significantly no zero. Table 5.29 shows that the difference for all eight Affymetrix microarray images between the mean of the GCOS average analyzed gene signal value and the true gene signal value was significantly not zero, except for Yeast-1 genome image at 0.001 significance level. Hence, we concluded that there was significance difference between the average analyzed gene signal value and the true gene signal value. Then, we implied the two sample t test between the true gene signal value and the average analyzed gene signal value. The null hypothesis and the alternative hypothesis are shown below.

$$H_0: \mu_{True} = \mu_{AverageSignaSBC}, H_1: \mu_{True} \neq \mu_{AverageSignaSBC}.$$

$$H_0: \mu_{True} = \mu_{AverageSignaGCOS}, H_1: \mu_{True} \neq \mu_{AverageSignaGCOS}.$$

We hypothesized that there was no significant difference between the mean of the true gene signal value and the mean of the average analyzed gene signal value at the

given significance levels, 0.05, 0.01 and 0.001. Table 5.30 and Table 5.31 present the two sample results for all eight Affymetrix images.

Table 5.30 Two sample t test for the SBC average analyzed gene signal value

Array	P-value	Decision for $\alpha = 0.05$	Decision for $\alpha = 0.01$	Decision for $\alpha = 0.001$
Bovine_a	< 0.000001	Reject	Reject	Reject
Bovine_b	< 0.000001	Reject	Reject	Reject
Canine_a	0.43	Accept	Accept	Accept
Canine_b	0.37	Accept	Accept	Accept
Vitis_a	< 0.000001	Reject	Reject	Reject
Vitis_b	< 0.000001	Reject	Reject	Reject
Yeast-1	0.003	Reject	Reject	Accept
Yeast-2	0.008	Reject	Reject	Accept

Table 5.31 Two sample t test for the GCOS average analyzed gene signal value

Array	P-value	Decision for $\alpha = 0.05$	Decision for $\alpha = 0.01$	Decision for $\alpha = 0.001$
Bovine_a	< 0.000001	Reject	Reject	Reject
Bovine_b	< 0.000001	Reject	Reject	Reject
Canine_a	< 0.000001	Reject	Reject	Reject
Canine_b	0.18	Accept	Accept	Accept
Vitis_a	< 0.000001	Reject	Reject	Reject
Vitis_b	< 0.000001	Reject	Reject	Reject
Yeast-1	0.37	Accept	Accept	Accept
Yeast-2	0.034	Reject	Accept	Accept

When the null hypothesis failed to be rejected, there was not enough evidence showing that there was significant difference between the mean of the true gene signal value and the mean of the average analyzed gene signal value. When the null hypothesis was rejected, that indicated that there was significant difference between the mean of the true gene signal value and the mean of the average analyzed gene signal value.

When we set the significant level at 0.05, for SBC, there were two accepted decisions for Canine_a and Canine_b. For GCOS, there were two accepted decisions for Canine_b and Yeast-1. When we set the significant level at 0.01, for SBC, there were two accepted decisions for Canine_a and Canine_b. For GCOS, there were three accepted decisions for Canine_b, Yeast-1 and Yeast-2. When we set the significant level at 0.001, for SBC, there were four accepted decisions for Canine_a, Canine_b, Yeast-1 and Yeast-2. For GCOS, there were three accepted decisions for Canine_b, Yeast-1 and Yeast-2. Hence, for the significance level at 0.001, SBC had more images showing that it was providing gene signal value which were closer to the true gene signal value as compared to GCOS.

Then, we implemented the clustering analysis among true gene signal value, averaged analyzed gene signal value from SBC and GCOS by using the same cluster technique as what we did for one simulated image analysis. Eight clustering tree plots for eight microarray images are shown in Figures 5.15 through 5.22.

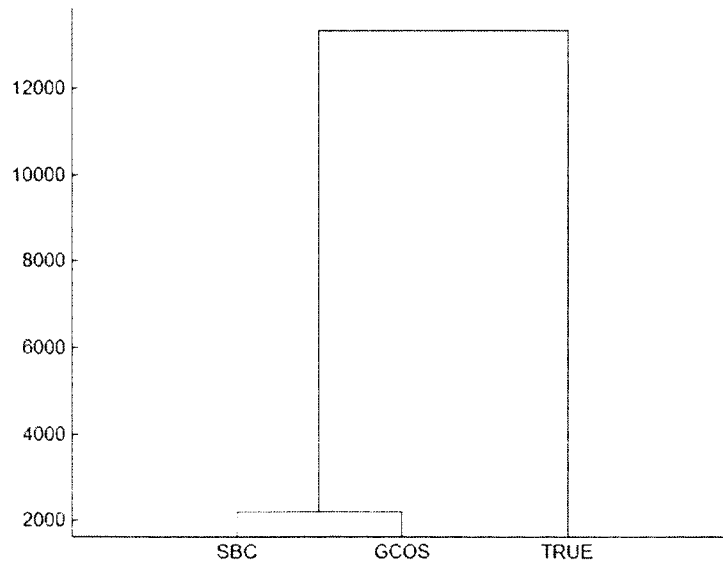


Figure 5.15 Clustering tree for Bovine_a

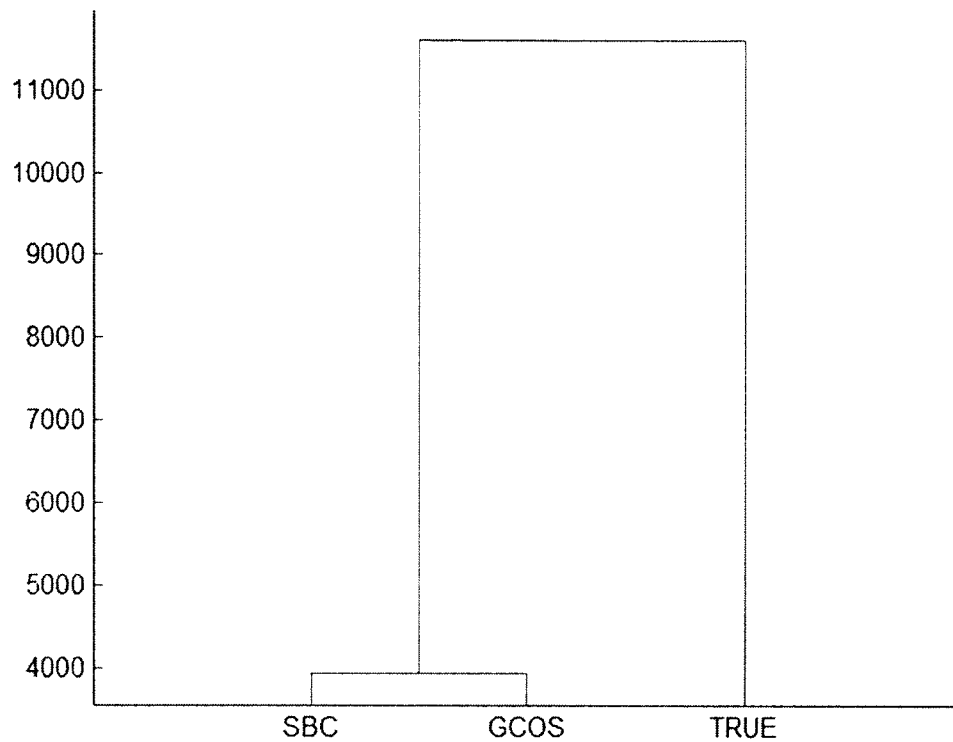


Figure 5.16 Clustering tree for Bovine_b

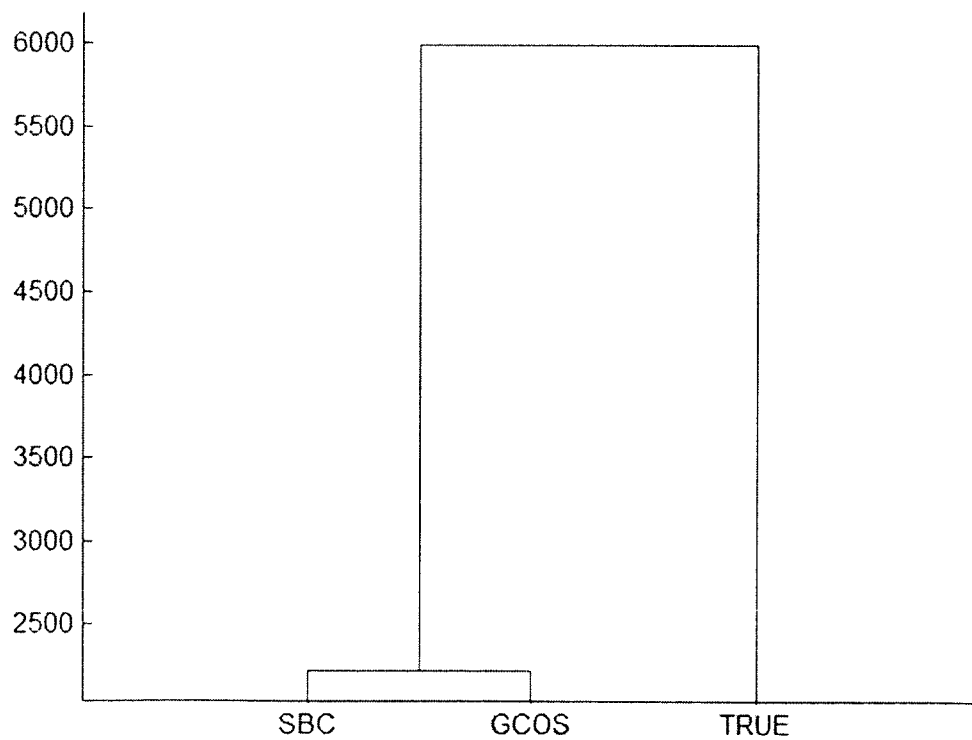


Figure 5.17 Clustering tree for Vitis_a

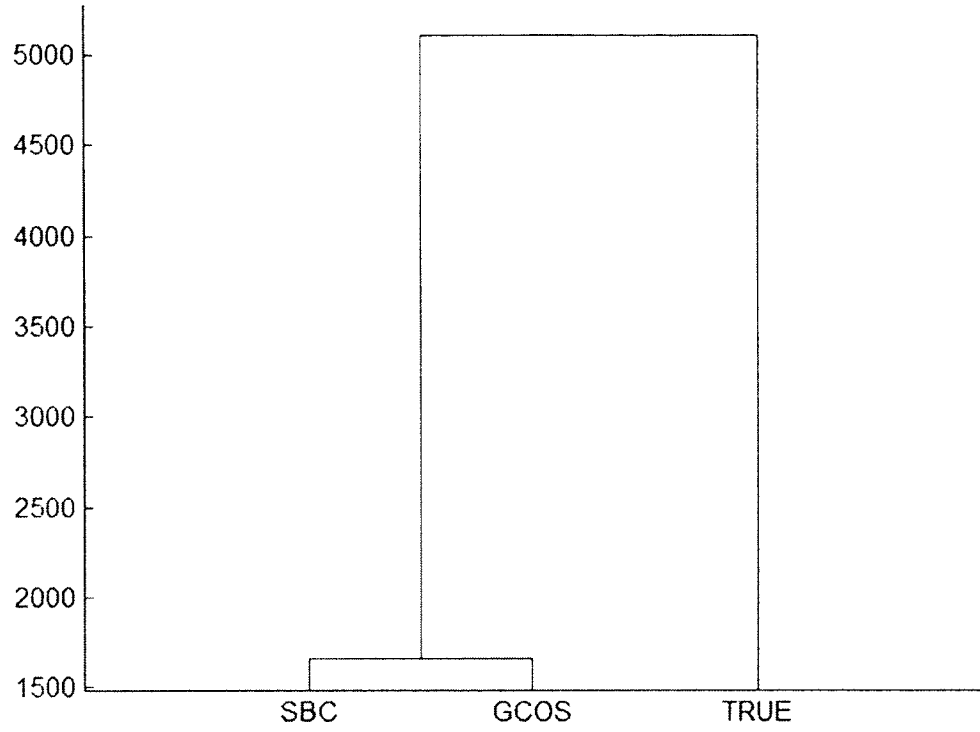


Figure 5.18 Clustering tree for Vitis_b

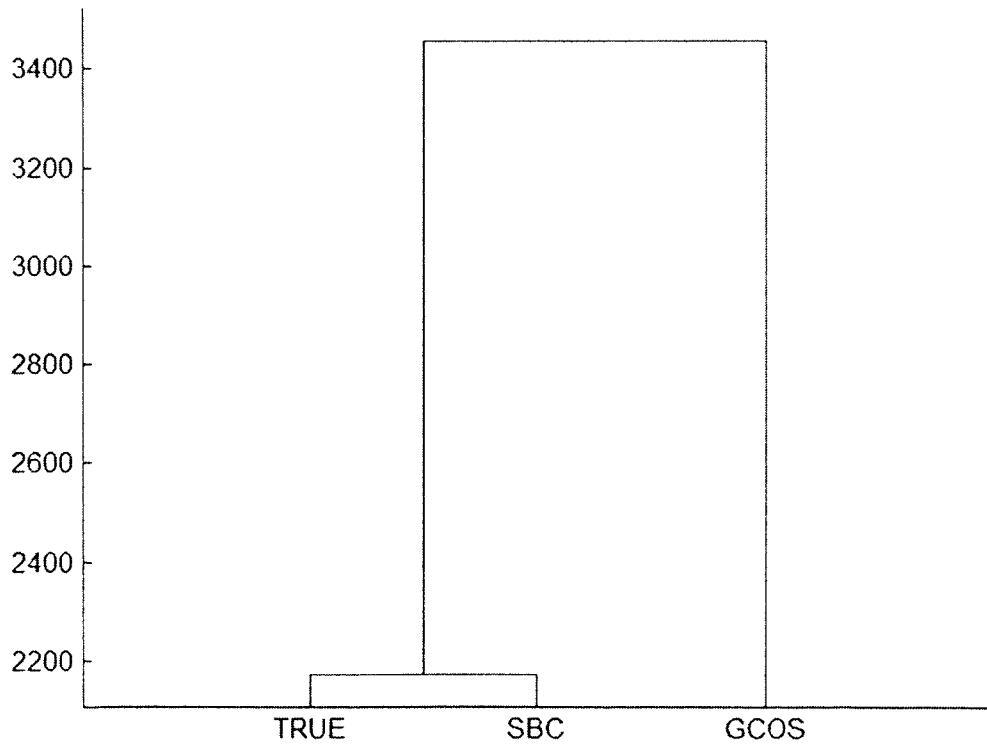


Figure 5.19 Clustering tree for Yeast-1

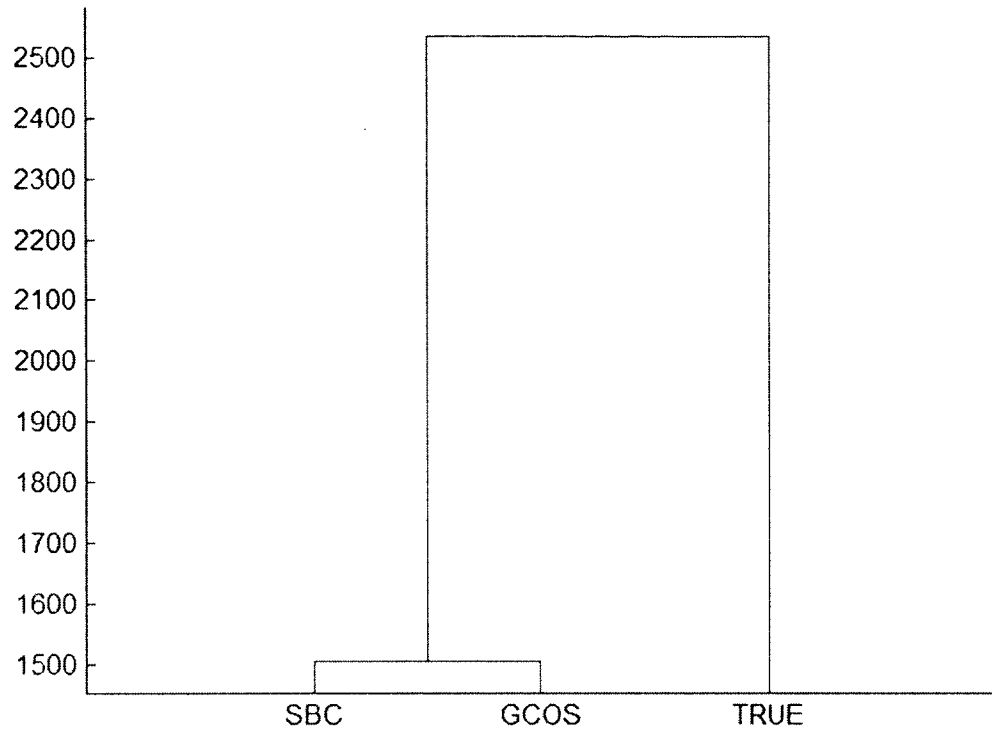


Figure 5.20 Clustering tree for Yeast-2

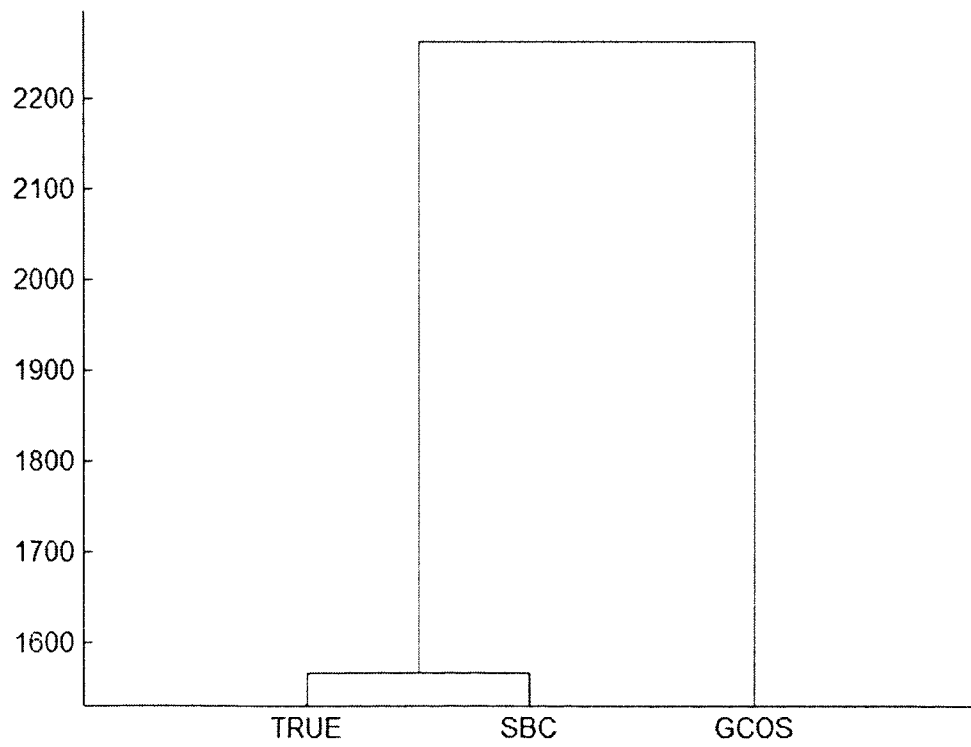


Figure 5.21 Clustering tree for Canine_a

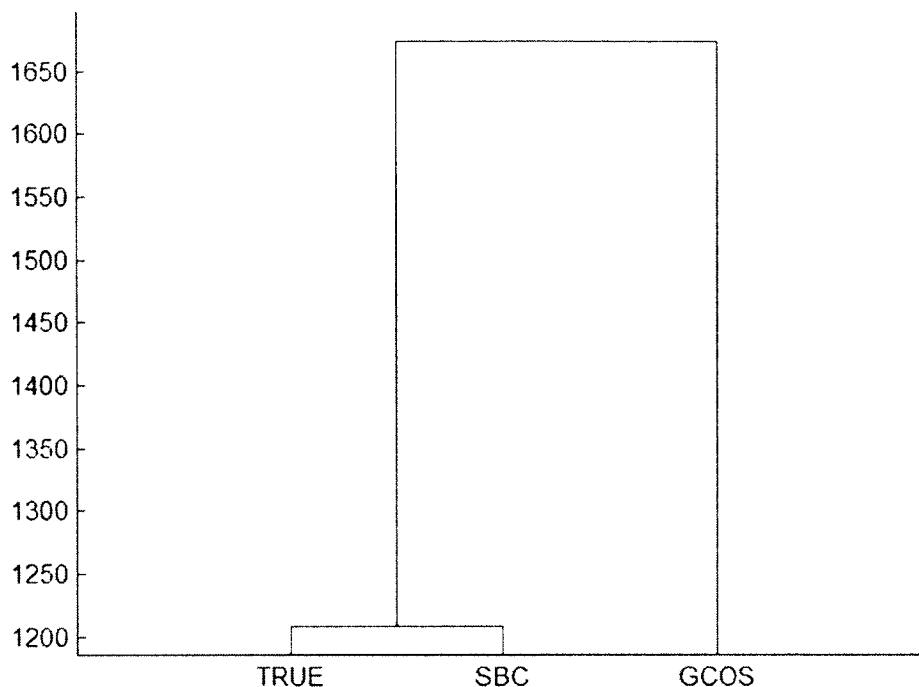


Figure 5.22 Clustering tree for Canine_b

Figures 5.19, 5.21 and 5.22 show that SBC averaged analyzed gene signal value and true gene signal value were assigned to the same group by clustering analysis. This indicated that for these three genomes, Yeast-1, Canine_a and Canine_b, there were less dissimilarity between the true gene signal value and the SBC average analyzed gene signal value compared to the GCOS average analyzed gene signal value. SBC was able to provide more accurate gene signal value for these three images. For the rest of the genome, Bovine_a, Bovine_b, Vitis_a, Vitis_b and Yeast-2, the SBC averaged analyzed gene signal value and the GCOS averaged analyzed gene signal value were assigned to the same group.

Hence, there were less dissimilarity between the SBC averaged analyzed gene signal value and GCOS averaged analyzed gene signal value as compared to the true gene signal value. SBC and GCOS stayed at the same accuracy level for the rest of five

Affymetrix microarray images on average, Bovine_a, Bovine_b, Vitis_a, Vitis_b and Yeast-2 based on the clustering analysis.

CHAPTER 6

CONCLUSIONS

The purpose of this dissertation was to construct a more accurate segmentation algorithm for Affymetrix microarray image. In Chapter 4, the Active Contours Without the Edges (ACWE) method and Segmentation Based Contours (SBC) method are presented. After modifying ACWE method, the SBC method was constructed and proposed to apply on Affymetrix microarray simulated image. The simulation method was introduced in Chapter 3, which embraces the most important biological characteristics of microarray experiments. In Chapter 5, we presented the comparison results after applying SBC and GCOS on simulated image based on the gene expression level. This gene expression comparison would bring significant impact in shedding light on cellular analysis field.

From all eight groups of different microarray images downloaded from Affymetrix sample test database, we replicated the generation of an image 50 times. We applied SBC and GCOS on all of these simulated images to obtain the gene signal expression values. Statistical analysis was performed based on the gene signal expression values instead of the intensity level. In this case, we were able to have a comprehensive understanding on how different algorithm will influence the gene expression calculation. In Chapter 5, we presented the dissimilarity distance metrics, sum squares of errors

measurement, standard error of performance, paired t test, two sample t test, inverse regression analysis and cluster analysis among three sets of gene signal values, which were true gene signal values, gene signal values obtained from SBC and gene signal values obtained from GCOS. Additionally, we computed the average gene signal values for each gene from the analyzed gene signal value obtained from both SBC and GCOS in each Affymetrix microarray image group. Similar statistical analysis was conducted on these average gene signal values, such as dissimilarity distance metrics, sum squares of errors measurement, standard error of performance, paired t test, two sample t test, inverse regression analysis and cluster analysis.

Based on all the comprehensive comparison results obtained from the above analyses, the SBC method provided more accurate gene signal values for Canine_a, Canine_b, Yeast-1 and Yeast-2. The GCOS method provided more accurate gene signal value for Bovine_a, Bovine_b, Vitis_a and Vitis_b. Hence, we concluded that SBC provided more accurate segmentation intensity values for some genome Affymetrix microarray images as compared to the GCOS. For the rest of the images analyzed in our research, GCOS had a better capability. However, even a small improvement in microarray image segmentation process would lead to a significant impact on genome expression analysis. For future research, we would like to propose a new hybrid method which will dynamically choose between SBC and GCOS based on the types of genome to which the microarray image belongs. This new hybrid method will possibly yield a greater improvement on detecting interested genes for disease diagnostics and disease control.

APPENDIX A

SOURCE CODE FOR SEGMENTATION METHOD


```

// import the libraries using in the program
import java.awt.Transparency;
import java.awt.image.*;
import java.io.*;
import javax.media.jai.*;
import javax.swing.*;
import java.awt.event.*;
import com.sun.media.jai.codec.*;

// main class of image segmentation
public class ImageSeg
{

// get a file from input
public static File getFileForFilename(String filename)
throws IOException {
    File fi = new File(filename);

    if (!fi.exists())
        throw new IOException(fi.getName() + " not found");
    if (!fi.isFile())
        throw new IOException(fi.getName() + " is not a file");
    return fi;
}

// main method
public static void main(String[] args)
{
    String filename=null;
    // Prompt for user to choose the file
    JFileChooser chooser = new JFileChooser();
    chooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
    int returnVal = chooser.showOpenDialog(null);

    // Confirm the user choosing file.
    if(returnVal == JFileChooser.APPROVE_OPTION) {
        //System.out.println("You chose to open this file: " +
        //    chooser.getSelectedFile().getName());
        filename=chooser.getSelectedFile().getPath();
    }
    int xpels=0, ypels=0 ; //initialize the variables
    double [] intensity ;
    try {
        // get the input file
        File fi1 = getFileForFilename(filename);
        //System.out.println("Segmenting...");

```

```

FileInputStream fin1 = new FileInputStream(fi1);
    BufferedReader bin1 = new BufferedReader(new InputStreamReader(fin1));
    String line1 = bin1.readLine(); //read in each line of the input file
while (line1!= null )
    {
String[] result1 = line1.split("\\t");
    String filename1=null,filename2=null,filename3=null,filename4=null;
for (int z=0; z<result1.length; z++)
    {
        filename1=result1[0];
        filename2=result1[1];
    }
// create output files name
    filename3=filename1.replaceAll(".tif","_segment.tif");
    filename4=filename1.replaceAll(".tif","_output.txt");
// create output image file
    RasterImage pi = JAI.create("fileload", filename1);
    Raster imagedata=pi.getData();
    WritableRasterwr=null ;
    WritableRaster imagedata1=pi.copyData(wr);
    intensity = new double [100];
// create an new object from class segment
    segmentmyseg;
    myseg = new segment();
    try {
        intzzz=1;
        File fi = getFileForFilename(filename2);
        FileInputStream fin = new FileInputStream(fi);
        BufferedReader bin = new BufferedReader(new InputStreamReader(fin));
        String line = bin.readLine();
        BufferedWriter out = new BufferedWriter(new FileWriter(filename4));
// using progress bar
        ProgressMonitorInputStream pin
        = new ProgressMonitorInputStream(null, fi.getName(), fin);
        ProgressMonitor pm = pin.getProgressMonitor();

        File file = new File(filename2);
        FileReaderfr = new FileReader(file);
        LineNumberReaderln = new LineNumberReader(fr);
        int count = 0;
        while (ln.readLine() != null){
            count++;
        }
        pm.setMaximum(count);
        while (line!= null )
            {

```

```

// initialize the variables
String[] result = line.split("\t");
intstartx=0;
intstarty=0;
intlastx=0;
intlasy=0;
for (int z=0; z<result.length; z++)
{
    // get the location from the gridding file
    starty=Integer.parseInt(result[0]);
    startx=Integer.parseInt(result[1]);
    lasty=Integer.parseInt(result[2]);
    lastx=Integer.parseInt(result[3]);
}
    for (int y=starty;y<=lasty;y++)
        for (int x=startx;x<=lastx ;x++)
            {
                inti,j;
                xpels=lastx-startx+1; // pixels in X-axis
                ypels=lasty-starty+1; // pixels in Y-axis
                intensity=new double[xpels*ypels];
                for (j=0;j<ypels;j++)
                    for (i=0;i<xpels;i++)
                        intensity[i+xpels*j]=imagedata.getSample(i+startx,j+starty,0);
                // get the intensity value
            }
        // segmenting using the methods from class segment
        myseg.create(xpels, ypels, startx, starty, lastx, lasty, intensity);
        myseg.initialize(xpels, ypels, startx, starty, lastx, lasty, intensity);
if (xpels*ypels>=100)
    {
        myseg.set_dt_e_w(0.1,1,0.0251);
        myseg.set_init_curve(3);
    }
else
    {
        myseg.set_dt_e_w(0.1,1,0.01);
        myseg.set_init_curve(1);
    }
myseg.segment();
out.write(Double.toString(myseg.areainfo(startx+1,starty+1,lastx-1,lasty-1)));
out.write("\t");
if (myseg.areainfo(startx,starty,lastx,lasty)!=0)
out.write(Double.toString(myseg.area_intensitymean(startx,starty,lastx,lasty)));

```

```

else
out.write(Double.toString(myseg.background_intensitymedian(startx,starty,lastx,lasty)));
out.write("\t");
if (myseg.areainfo(startx+1,starty+1,lastx-1,lasty-1)!=0)
out.write(Double.toString(myseg.area_intensity_75pvalue(startx+1, starty+1, lastx-1,
lasty-1)));
else
out.write(Double.toString(myseg.background_intensity_75pvalue(startx, starty, lastx,
lasty)));
out.write("\t");
out.write(Double.toString(myseg.background_intensitymedian(startx,starty,lastx,lasty)));
out.write("\n");
// output to the output text file
pm.setProgress(zzz);
zzz=zzz+1;
{
inti,j;
for(j=0;j<myseg.ypels;j++)
for (i=0;i< myseg.xpels-1;i++)
{
if
(myseg.sign(myseg.area_mapping[i+myseg.xpels*j])!=myseg.sign(myseg.area_mapping[
i+1+myseg.xpels*j]))
{
if (myseg.sign(myseg.area_mapping[i+myseg.xpels*j])<0)
imagedata1.setSample(i+myseg.startx,j+myseg.starty,0,60000);
else
imagedata1.setSample(i+1+myseg.startx,j+myseg.starty,0,60000);
}
}
}
}
inti,j;
for(j=0;j<myseg.ypels-1;j++)
for (i=0;i<myseg.xpels;i++)
{
if
(myseg.sign(myseg.area_mapping[i+myseg.xpels*j])!=myseg.sign(myseg.area_mapping[
i+myseg.xpels*(j+1)]))
{
if (myseg.sign(myseg.area_mapping[i+myseg.xpels*j])<0)
imagedata1.setSample(i+myseg.startx,j+myseg.starty,0,60000);
else
imagedata1.setSample(i+myseg.startx,j+1+myseg.starty,0,60000);
}
}
}
}

```

```

    }
    {
int j;
for(j=0;j<myseg.ypels;j++)

    {
        if (myseg.sign(myseg.area_mapping[0+myseg.xpels*j])>=0)
            imagedata1.setSample(0+myseg.startx,j+myseg.starty,0,60000);
        if (myseg.sign(myseg.area_mapping[myseg.xpels-1+myseg.xpels*j])>=0)
            imagedata1.setSample(myseg.xpels-1+myseg.startx,j+myseg.starty,0,60000);
    }
    }
}
inti;
for(i=0;i<myseg.xpels;i++)

    {
        if (myseg.sign(myseg.area_mapping[i+myseg.xpels*0])>=0)
            imagedata1.setSample(i+myseg.startx,0+myseg.starty,0,60000);
        if (myseg.sign(myseg.area_mapping[i+myseg.xpels*(myseg.ypels-1)])>=0)
            imagedata1.setSample(i+myseg.startx,myseg.ypels-1+myseg.starty,0,60000);
    }
    }
    // draw the boundary of each cell in the output image file
    line = bin.readLine();
}
out.close();
}
catch (IOException e) {          // Trap exception
System.err.println(e.toString()); // Display error
}
BufferedImage bi = new BufferedImage(pi.getColorModel(), imagedata1, true, null);
RenderedImage op =JAI.create("filestore",bi,filename3,"TIFF");
    line1 = bin1.readLine();
}
}
catch (IOException e) {          // Trap exception
System.err.println(e.toString()); // Display error
}
}
//System.out.println("Finished!");
int response = JOptionPane.showOptionDialog(
null          // Center in window.
, "The whole segmentation process has finished" // Message
, null          // Title in titlebar
, JOptionPane.DEFAULT_OPTION // Option type
, JOptionPane.PLAIN_MESSAGE // messageType

```

```
        , null           // Icon (none)
        , null           // Button text as above.
        , null           // Default button's label
    );
    if (response==0)
    {
        System.exit(0);
    }
    if (response==-1)
    {
        System.exit(0);
    }
    //System.exit(0);
}
}
```

APPENDIX B

SOURCE CODE FOR WRITING IMAGE TO DAT FILE

```
function affywritedat_new(image,gridname,head1,newdat)
```

```

    [px,px_1]=size(image);
    %[A1,head1,dump1]=affyreaddat(datname);
    grid=load(gridname);
    [r,c]=size(grid);
    n=sqrt(r);
    ul=[grid(1,2),grid(1,1)];
    ur=[grid(n,4),grid(n,1)];
    ll=[grid((n*(n-1)+1),2),grid((n*(n-1)+1),3)];
    lr=[grid(n*n,4),grid(n*n,3)];

    fid=fopen(newdat,'w','l');
    head=writehead(fid,image,head1,ul,ur,ll,lr);
    for i=1:px
        for j=1:px_1
            %tmp=fwrite(fid,uint16(image(i,j)),'uint16');
            fwrite(fid,uint16(image(i,j)),'uint16');
        end
    end
    %fwrite(fid,uint16(image),'uint16');
    fclose(fid);

```

```
function head=writehead(fid,image,head1,ul,ur,ll,lr)
% head=fread(fid,512,'uint8');
```

```

    [px,px_1]=size(image);

    head.type=fwrite(fid,head1.type,'uint8');
    head.pixperline=fwrite(fid,px_1,'uint16');
    head.nolines=fwrite(fid,px,'uint16');
    head.pixels=fwrite(fid,px*px_1,'uint32');
    head.minpixvalue=fwrite(fid,min(image(:)),'uint32');
    head.maxpixvalue=fwrite(fid,max(image(:)),'uint32');
    head.meanpixvalue=fwrite(fid,mean(image(:)),'double');
    head.stdpix=fwrite(fid,std(image(:)),'double');

    head.nopixperrow=fwrite(fid,head1.nopixperrow,'uchar');
    head.norows=fwrite(fid,head1.norows,'uchar');
    head.pixwidth=fwrite(fid,head1.pixwidth,'uchar');
    head.pixheight=fwrite(fid,head1.pixheight,'uchar');
    head.scanspeed=fwrite(fid,head1.scanspeed,'uchar');
    head.temperature=fwrite(fid,head1.temperature,'uchar');
    head.laserpower=fwrite(fid,head1.laserpower,'uchar');
    head.datetime=fwrite(fid,head1.datetime,'uchar');

```



```
head.subfield=fwrite(fid, head1.subfield,'uchar');

head.meandcoffset=fwrite(fid,head1.meandcoffset,'double');
head.stddcoffset=fwrite(fid,head1.stddcoffset,'double');
    head.dcdoffsetsamples=fwrite(fid,head1.dcdoffsetsamples,'uint32');

head.xy_ul=fwrite(fid,ul,'int16');
head.xy_ur=fwrite(fid,ur,'int16');
head.xy_lr=fwrite(fid,lr,'int16');
head.xy_ll=fwrite(fid,ll,'int16');
head.cellmargin=fwrite(fid,head1.cellmargin,'uint16');
    a=head1.name;
    [r,c]=size(a);
    n=154-c;
    b='h';
for i=1:n
    a=strcat(a, b);
end
    head.name=fwrite(fid,a,'uchar');
```

APPENDIX C

SOURCE CODE FOR WRITING OUTPUT TO CEL FILE

```

path=pwd;
file=dir(fullfile(path, '*.cel'));
f_size=size(file);
for (i=1:f_size(1));
f_file=fullfile(path,file(i).name);
acwe_f=strrep(lower(f_file),'.cel','_output.txt');

outcel_f=strrep(lower(f_file),'.cel','_output.cel');

fid = fopen(f_file);
fid1=fopen (outcel_f,'w');
cel=affyread(f_file);
[A,B,ACWE,C]=textread(acwe_f,'%f%f%f%f','headerlines',0);

%while ~feof(fid);
for (i=1:24)
tline=fgets(fid);
fwrite(fid1, tline);
end
%end

for (i=1:cel.NumProbes)

tline=fgets(fid);

[token1, remain1] = strtok(tline);
[token2, remain2] = strtok(remain1);
[token3, remain3] = strtok(remain2);
[token4, remain4] = strtok(remain3);
[token5, remain5] = strtok(remain4);
%fwrite(fid1, '%12.8f\n',g(3));
fprintf(fid1, '%3d\t',str2num(token1));
fprintf(fid1, '%3d\t',str2num(token2));
%fprintf(fid1, '%s\t',token3);
fprintf(fid1, '%s\t',num2str(ACWE(i)));
fprintf(fid1, '%s\t',token4);
fprintf(fid1, '%3d\n',str2num(token5));

end
while ~feof(fid);
tline=fgets(fid);
fwrite(fid1, tline);
end
end
fclose(fid);
fclose(fid1)

```

REFERENCES

- [1] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter, "Molecular Biology of the Cell," in *Garland Science*, pp. 1-125, 2007.
- [2] A. D. Hershey and M. Chase, "Independent functions of viral protein and nucleic acid in growth of bacteriophage," in *The Journal of General Physiology*, vol. 36, pp. 39-56, 1952.
- [3] J. D. Watson and F. H. C. Crick, "A Structure for Deoxyribose Nucleic Acid," in *Nature*, pp. 737-738, 1953.
- [4] "Molecular Biology." Available: <http://cnce.tumblr.com/post/9844643186/central-dogma-of-biology-sequential-information>. [Accessed: 15-Nov-2012].
- [5] S.C. Lakhota, "What is a Gene," in *Resonance*, vol. 2, pp. 44-53, 1997.
- [6] M. V. Rockman and L. Kruglyak, "Genetic of global gene expression," in *Nature Review Genetics*, vol. 7, pp. 862-872, 2006.
- [7] "Microarray." Available: <http://www.ncbi.nlm.nih.gov/About/primer/microarrays.html>. [Accessed: 15-Nov-2012].
- [8] "Affymetrix." Available: <http://www.Affymetrix.com/>. [Accessed: 15-Nov-2012].
- [9] D. Lockhart, H. Dong, M. Byrne, M. Follettie, M. Gallo, M. Chee, M. Mittmann, C. Wang, M. Kobayashi and H. Horton, "Expression monitoring by hybridization to high-density oligonucleotide arrays," In *Nature Biotechnology*, vol. 14, pp. 1675-1680, 1996.
- [10] M. Schena, D. Shalon, R.W. Davis, and P.O. Brown, "Quantitative Monitoring of Gene Expression Patterns with a Complementary DNA Microarray," in *Science*, vol. 270, pp. 467-470, 1996.
- [11] Y. H. Yang, M. J. Buckley, S. Dudoit and T. P. Speed, "Comparison of methods for image analysis on cDNA microarray data," in *Journal of Computational & Graphical Statistics*, vol. 11, pp. 108-136, 2002.

- [12] H. Gohlmann and W. Talloen, "Gene expression studies using Affymetrix microarrays," in *Mathematical and Computational Biology*, pp. 1-100, 2009.
- [13] "cDNA." Available:
<http://www.scq.ubc.ca/spot-your-genes-an-overview-of-the-microarray/>. [Accessed: 15-Nov-2012].
- [14] "cDNA." Available:
<http://www.cs.wustl.edu/~jbuhler/research/array/>. [Accessed: 15-Nov-2012].
- [15] R. Lipshutz, S. Fodor, T. Gingeras, and D. Lockhart, "High density synthetic oligonucleotide arrays," in *Nature Genetics*, suppl. 21, pp. 20±24, 1999.
- [16] "Probe Level Design." Available:
<http://www.dkfz.de/gpcf/24.html>. [Accessed: 15-Nov-2012].
- [17] "Experiment." Available:
<http://angerer.swissbrain.org/archive/2002/11/>. [Accessed: 15-Nov-2012].
- [18] L. J. Kricka and J. Cabrera, "Microarray Technology and Applications: An All-Language Literature Survey Including Books and Patents," in *Clinical Chemistry*, vol. 48, no. 8, pp. 1479-1482, 2001.
- [19] D. Amaratunga and J. Cabrera, *Exploration and analysis of DNA microarray and protein array data*, New York, Wiley, 2004.
- [20] T. Bergemann, R. Laws, F. Quiaoit and P. Zhao, "A statistically driven approach for image segmentation and signal extraction in cDNA microarrays," in *Journal of Computational Biology*, vol. 11, no. 4, pp. 695-713, 2004.
- [21] O. Demirkaya, M. H. Asyali and M. M. Shoukri, "Segmentation of cDNA microarray spots using Markov random field modeling," in *Bioinformatics*, vol. 21, no. 3, pp. 2994-3000, 2005.
- [22] R. Gottardo, J. Bsesag, M. Stephens and A. Murua, "Probabilistic segmentation and intensity estimation for microarray images," in *Biostatistics*, vol. 7, no. 1, pp. 85-99, 2006.
- [23] H. Zuzan, C. Blanchette, H. Dressman, E. Huang, S. Ishida, J. R. Marks, J. R. Nevins, R. Spang, M. West, V. E. Johnson, "Estimation of Probe Cell Locations in High-density Synthetic-oligonucleotide DNA Microarrays," in *Journal of American Statistical Association*, pp. 611-631, 2001.
- [24] R. Simon, E. Korn, L. M. McChane, M. D. Radmacher, G. W. Wright and Y. Zhao, *Design and analysis of DNA microarray investigations*, New York, Springer, 2003.

- [25] R. Adams and L. Bischof, "Seeded region growing," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 641-647, 1994.
- [26] Y. Chen, E. Dougherty and M. Bittner, "Ratio-based decisions and the quantitative analysis of cDNA microarray images," in *Journal Biomedical Optics*, vol. 2, no. 4, pp. 364-374, 1997.
- [27] T. F. Chan and L. A. Vese, "Active contours without edges," in *IEEE Transactions on Image Processing*, vol. 10, pp. 266-276, 2001.
- [28] "Gene Expression." Available:
http://media.Affymetrix.com/support/technical/whitepapers/sadd_whitepaper.pdf.
 [Accessed: 15-Nov-2012].
- [29] J. N. McClintick and H. J. Edenberg, "Effects of filtering by Present call on analysis of microarray experiments," in *BMC Bioinformatics*, pp.7-49, 2006.
- [30] S. D. Pepper, E. K. Saunders, L. E. Edwards¹, C. L. Wilson^{and} and C. J. Miller. "The utility of MAS5 expression summary and detection call Algorithms," in *BMC Bioinformatics*, pp. 8-273, 2007.
- [31] M. Nykter, T. Aho, M. Ahdesmäki, P. Ruusuvoori, A. Lehmuusola and O. Harja, "Simulation of microarray data with realistic characteristics," in *BMC Bioinformatics*, pp. 7-349, 2006.
- [32] "Simulation Model." Available:
<http://www.cs.tut.fi/sgn/csb/mamodel/>. [Accessed: 15-Nov-2012].
- [33] "Sample data." Available:
http://www.Affymetrix.com/support/technical/sample_data/demo_data.affx. [Accessed: 15-Nov-2012].
- [34] W. Black, M. Kaern, C. Cantor and J. Collins, "Noise in eukaryotic gene expression," in *Nature*, vol. 422, pp. 633-637, 2003.
- [35] H. B. Fraser, A. E. Hirsh, G. Giaever, J. Kumm and M. B. Eisen, "Noise minimization in eukaryotic gene expression," in *PLOS Biology*, vol. 2, pp. 137, 2004.
- [36] Y. Tu, G. Stolovitzky and U. Klein, "Quantitative noise analysis for gene expression microarray experiments," in *Proceedings of the National Academy of Sciences*, vol. 99, no. 22, pp. 14031-14036, 2002.
- [37] H. Cho and J. K. Lee, "Bayesian hierarchical error model for analysis of gene expression data," in *Bioinformatics*, vol. 20, no. 13, pp. 2016-2025, 2004.

- [38] R. O. Dror, J. G. Murnick, N. J. Rinald, V. D. Marinescu, R. M. Rifkin and R. A. Young. "Bayesian estimation of transcript levels using a general model of array measurement noise," in *Journal of Computational Biology*, vol. 10, pp. 433-1452, 2003.
- [39] D. M. Rocke and B. Durbin, "A model for measurement error for gene expression array," in *Journal of Computational Biology*, vol. 8, no. 6, pp. 557-569, 2001.
- [40] A. J. Hartemink, D. K. Gifford, T. S. Jaakkola and R. A. Young, "Maximum-likelihood estimation of optimal scaling factors for expression array normalization," in *Proceedings of the International Biomedical Optics Symposium*, pp. 132-140, 2001.
- [41] A. M. K. Hein, S. Richardson, H. C. Causton, G. K. Ambler and P. J. Green, "PJ: BGX: A fully Bayesian integrated approach to the analysis of Affymetrix GeneChip data," in *Biostatistics*, vol. 6, no. 3, pp. 349-373, 2005.
- [42] C. T. Ekstrom, S. Bak, C. Kristensen and M. Rudemo, "Spot shape modeling and data transformations for microarrays," in *Bioinformatics*, vol. 20, no. 14, pp. 2270-2278, 2005.
- [43] M. Moelich and T. Chan, "Tracking objects with the Chan-Vese algorithm," in *Computational Applied Mathematics*, Los Angeles, 2003.
- [44] A. Almhdie, P. L. Pereira, S. Meme, C. Colombier, V. Brault, F. Szeremeta, B. T. Doan, R. Ledee, R. Harba, Y. Herault, J. C. Belaeil and C. Leger, "Chan-vease based method to segment mouse brain MRI images: application to cerebral malformation analysis in Trisomy 21," in *17th European Signal processing Conference*, Glasgow, 2009.
- [45] N. Salman, "Image Segmentation and Edge detection based on Chan-Vese Algorithm," in *The International Arab Journal of Information Technology*, vol. 3, no. 1, 2006.
- [46] O. Rousseau, "An iterative active contours algorithm applied to hearth segmentation," in *4th Montreal Scientific computing days*, Montreal, 2007.
- [47] S. Lakshmi and V. Sankaranarayanan. "A study of edge detection techniques for segmentation computing approaches," in *IJCA Special Issue on "Computer Aided Soft Computing Techniques for Imaging and Biomedical Applications"*, vol. CASCT, pp. 35-41, 2010.
- [48] J. Quackenbush, "Computational analysis of microarray data," in *Nature Reviews Genetics*, pp. 418-427, 2001.

- [49] J. N. McClintick and H. J. Edenberg, "Effects of filtering by Present call on analysis of microarray experiments," in *BMC Bioinformatics*, pp. 7-49, 2006.
- [50] S. Drghici, "Statistics and data analysis for microarrays using R and Bioconductor," in *Chapman & Hall/CRC Mathematical & Computational Biology*, pp. 568-634, 2011.
- [51] G. Glazko and A. Mushegian, "Measuring gene expression divergence: the distance to keep," in *Biology Direct*, vol. 5, no. 1, 2010.
- [52] M. A. Levenstien, Y. Yang and J. Ott, "Statistical significance for hierarchical clustering in genetic association and microarray expression studies," in *BMC Bioinformatics*, pp. 4-62, 2003.
- [53] "Clustering Analysis." Available:
<http://www.invitrogen.com/site/us/en/home/References/Ambion-Tech-Support/transcriptome-analysis/tech-notes/analysis-of-microarray-data.html>.
[Accessed: 15-Nov-2012].
- [54] A.M.C. Davies and T. Fearn, "Back to basics: calibration statistics," in *Spectroscopy Europe*, vol. 18, no.2, pp. 31-32, 2006.
- [55] K. H. Ng and A. H. Pooi, "Calibration Intervals in Linear Regression Models," in *Communications in Statistics - Theory and Methods*, vol. 37, no. 11, pp. 1688-1696, 2008.
- [56] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, New York, Springer, 2001.
- [57] "Sampling Error." Available:
<http://www.experiment-resources.com/sampling-error.html>. [Accessed: 15-Nov-2012].