Spring 2010

DNA microarray image segmentation using active contours without edges method

Shenghua Ni
Louisiana Tech University

DNA MICROARRAY IMAGE SEGMENTATION USING ACTIVE CONTOURS WITHOUT EDGES METHOD

By

Shenghua Ni, 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

March 2010

LOUISIANA TECH UNIVERSITY

THE GRADUATE SCHOOL

December 9, 2009
Date

We hereby recommend that the dissertation prepared under our supervision by Shenghua Ni

entitled

DNA microarray image segmentation using Active Contours Without Edges method

be accepted in partial fulfillment of the requirements for the Degree of Ph.D. In Computational Analysis and Modeling

[illegible]
Supervisor of Dissertation Research

[illegible]
Head of Department

Computational Analysis and Modeling
Department

Recommendation concurred in:

[illegible]

[illegible]

[illegible]

[illegible]

Advisory Committee

Approved:
[illegible]
Director of Graduate Studies

[illegible]
Dean of the College

Approved:
[illegible]
Dean of the Graduate School

GS Form 13a
(6/07)

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 Thesis. 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 Thesis. Further, any portions of the Thesis used in books, papers, and other works must be appropriately referenced to this Thesis.

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

Author [illegible]

Date Feb. 1, 2010

GS Form 14
(5/03)

ABSTRACT

The goal of this dissertation is to build a better segmentation method for DNA microarray image processing. Segmentation is a partitioning process used to separate a spot area from a non-spot area in DNA microarrays. It directly affects the accuracy of gene expression analysis in the data mining process that follows. A number of DNA microarray segmentation methods have been proposed in the area, but even modern segmentation methods seem to have accuracy problems. In this dissertation, I will present a segmentation method based on the Active Contours Without Edges (ACWE) algorithm and apply it to two types of DNA microarrays, complementary DNA (cDNA) and Affymetrix GeneChip. Several adjustments will be applied to the original ACWE method to use it more efficiently in the microarray processing area.

As a secondary research objective, I will improve the ACWE method by using higher order schemes in finite difference method for solving the partial differential equation (PDE). The original ACWE method used the associated Euler-Lagrange partial differential equation for the Lipschitz function Φ. It used the lower order finite difference schemes to solve the PDE. The improved ACWE method defines the higher order finite difference schemes to increase the accuracy of segmentation.

Various experimental results will be presented to show that the ACWE method is more efficient than other DNA microarray image segmentation methods.

Statistical analysis is performed to compare the newly proposed method with the previously best methods in this area. Experimental results will also be presented to show that the improved ACWE method has more accurate segmentation results than the ACWE method.

TABLE OF CONTENTS

ABSTRACT iii

LIST OF TABLES viii

LIST OF FIGURES x

ACKNOWLEDGEMENTS xiii

CHAPTER 1 INTRODUCTION 1

1.1 General Overview 1
1.2 Research Objects 3
1.3 Organization of the Dissertation 3

CHAPTER 2 BACKGROUND AND PREVIOUS WORK 5

2.1 Introduction of DNA and DNA Microarray 5
2.1.1 DNA 5
2.1.2 DNA Microarray 8
2.2 Introduction of cDNA Microarray Image Analysis Methods 11
2.2.1 cDNA Microarray Image Analysis Methods 11
2.2.2 cDNA Microarray Image Segmentation Methods 13
2.2.3 cDNA Microarray Image Information Extraction 16
2.2.4 cDNA Microarray Image Data Normalization 17
2.2.5 cDNA Microarray Image Data Analysis 19
2.3 Adaptive Shape Segmentation Methods 21
2.3.1 Watershed Method 21
2.3.2 Seeded Region Growing Method 22
2.3.3 Globally Optimal Geodesic Active Contours Method 26
2.4 Affymetrix GeneChip and Affymetrix Segmentation Method 28
2.4.1 Affymetrix GeneChip 28
2.4.2 Affymetrix GeneChip Segmentation Method 29

CHAPTER 3 ACTIVE CONTOURS WITHOUT EDGES SEGMENTATION METHOD 32

3.1 Active Contours Without Edges (ACWE) Algorithm 32
3.2 Weakness of Current Segmentation Methods in DNA Microarrays 35

3.3 Weaknesses of Level Sets Method and Active Contours Method 37
3.3.1 Original Level Sets Method Weaknesses 37
3.3.2 Original Active Contours Method Weaknesses 38
3.4 Advantages of Active Contours Without Edges Method 39
3.5 Active Contours Without Edges (ACWE) Method Application in DNA Microarrays Segmentation 40
3.5.1 Adjustments for ACWE Method 40
3.5.2 The Databases for Experiments in Applying ACWE on DNA Microarrays 40
3.5.2.1 The Databases for cDNA Microarrays 41
3.5.2.2 The Databases for Affymetrix GeneChip 41
3.5.3 The Process of Applying ACWE or cDNA Microarrays 43
3.5.4 The Process of Applying ACWE for Affymetrix GeneChip 45
3.6 Improvements for Active Contours Without Edge (ACWE) Method Application in DNA Microarrays Segmentation 48
3.6.1 The Improvement by Reducing the Length Constraint in ACWE 48
3.6.2 The Improvement by Using a Fast Algorithm in ACWE 48
3.6.3 The Improvements by Providing a More Accurate Segmentation Method Than the ACWE Method 50
3.6.4 Hybrid Methods for ACWE 52

CHAPTER 4 EXPERIMENTAL RESULTS OF DNA MICROARRAYS SEGMENTATION BASED ON THE ACWE METHOD 53

4.1 Experimental Result for cDNA Microarray 55
4.2 Experimental Result for Affymetrix GeneChip 72

CHAPTER 5 EXPERIMENTAL RESULTS OF DNA MICROARRAYS SEGMENTATION BASED ON THE IMPROVED ACWE METHOD 80

5.1 Experimental Results of Simplified ACWE (S-ACWE) Method 80
5.2 Experimental Results of a Fast ACWE (F-ACWE) Method 82
5.3 Experimental Results of Improving ACWE Methods (I-ACWE, I2-ACWE) 85
5.4 Experimental Results of Hybrid Methods for ACWE 89
5.5 Improved ACWE (I2-ACWE) Method Shows Accurate Gene Expressions at the Biological Level 90
5.6 Improved ACWE (I2-ACWE) Method Shows Accurate Gene Expressions Using Rank Sum. 100

CHAPTER 6 CONCLUSIONS 103

APPENDIX A SOURCE CODE FOR IMPROVED ACWE (I2-ACWE) SEGMENTATION METHOD 105

APPENDIX B SOURCE CODE FOR ACWE SEGMENTATION METHOD 120

REFERENCES 134

LIST OF TABLES

Table 4.1 Patch outside Spot 1 with intensity values. Yellow area is the spot area. Red figures are the boundary found by ACWE....59

Table 4.2 Spot 1 mean intensity value using different methods and software....59

Table 4.3 Sum Square Error of concentration between Affymetrix and ACWE methods for one image....76

Table 4.4 All images in one experiment SSE comparison result....77

Table 4.5 All experiments SSE comparison result....78

Table 4.6 Type of experiments and comparison among methods....78

Table 4.7 Affymetrix GeneChip template type SSE comparison results....79

Table 5.1 All experiments SSE comparison result....81

Table 5.2 Type of experiments and comparison among methods....82

Table 5.3 Affymetrix GeneChip template type SSE comparison result....82

Table 5.4 All experiments SSE comparison results....83

Table 5.5 Type of experiments and comparison among methods....84

Table 5.6 Affymetrix GeneChip template type SSE comparison result....84

Table 5.7 All experiments SSE comparison result for I-ACWE....85

Table 5.8 Types of experiments and comparison among methods for I-ACWE....86

Table 5.9 Affymetrix GeneChip template type SSE comparison result for I-ACWE....86

Table 5.10 All experiments SSE comparison result for I2-ACWE....87

Table 5.11 Type of experiments and comparison among methods for I2-ACWE....88

Table 5.12 Affymetrix GeneChip template type SSE comparison result for I2-ACWE 88

Table 5.13 Genes positive correlated with HW/BW ratio in mouse experiment. 91

Table 5.14 Genes negative correlated with HW/BW ratio in mouse experiment. 91

Table 5.15 Genes correlated with LVEF in the human experiment. 95

Table 5.16 Gene GAPDH expression range in all the experiments. 98

Table 5.17 GAPDH expression range values based on different GeneChip template types 99

Table 5.18 GAPDH expression range values based on different experimental models. 99

Table 5.19 All experiments Sum of rank comparison result for I2-ACWE 101

Table 5.20 Type of experiments and comparison among methods for I2-ACWE with sum of rank 102

Table 5.21 Affymetrix GeneChip template type sum of rank comparison result for I2-ACWE 102

LIST OF FIGURES

Figure 2.1 Gridding output of spot segmentation. 13

Figure 2.2 Histogram segmentation. 15

Figure 2.3 Adaptive shape segmentation using GOGAC. 16

Figure 2.4 Watershed segmentation. 22

Figure 2.5 Seeded region growing segmentation. 25

Figure 2.6 GOGAC segmentation using Spot 27

Figure 2.7 Probe pair. 28

Figure 2.8 Affymetrix probe cell (spot) and segmentation area (red area) 30

Figure 4.1 ACWE segmentation for the cDNA microarray 54

Figure 4.2 ACWE segmentation for the Affymetrix GeneChip 54

Figure 4.3 Original cDNA .TIFF file 55

Figure 4.4 Image after segmentation using ACWE. 56

Figure 4.5 Segmentation using GOGAC. 56

Figure 4.6 Segmentation using SRG. 57

Figure 4.7 Segmentation for Spot 1 using ACWE 58

Figure 4.8 Segmentation using spotSegmentation for Spot 1 58

Figure 4.9 3D graphic for square outside Spot 1 with intensity value. Z-axis is the intensity value. Five pixels with extreme high intensity are partitioned to the same spot foreground. 60

Figure 4.10 The linear regression between the SRG and ACWE methods. 61

Figure 4.11 The linear regression between GOGAC and ACWE methods. 62

Figure 4.12 The linear regression between FIXED circle and ACWE methods. 62

Figure 4.13 LAMBDA is a control spot and YBR145W is a gene spot. The trend is the result of intensity of the spots using different methods. 63

Figure 4.14 Boxplot for z-score normalization for spots in Figure 4.13.. 64

Figure 4.15 Boxplot of z-score normalization of log ratio using different methods. 65

Figure 4.16 Boxplot of the z-score normalization of relative difference using different methods. 66

Figure 4.17 Boxplot of spot intensity of LAMBDA with different segmentation methods. 67

Figure 4.18 Log ratio (R/G) of LABMDA of 18 microarrays using ACWE method. 68

Figure 4.19 Gene clusters with cell cycle-regulation...69

Figure 4.20 Histone genes under yeast alpha factor experiment. 70

Figure 4.21 Log ratio (R/G) of the histone genes in alpha factor using ACWE method 71

Figure 4.22 Log ratio (R/G) of histone genes in alpha factor using SRG method 71

Figure 4.23 Log ratio (R/G) of histone genes in alpha factor using GOGAC method 72

Figure 4.24 Segmentation for Affymetrix GeneChip image using ACWE 73

Figure 4.25 Linear Correlation between ACWE and Affymetrix segmentation method 73

Figure 4.26 Linear regression line between intensity values using Affymetrix method and the concentration of control genes. 75

Figure 4.27 Linear regression line between intensity values using ACWE method and the concentration of control genes. 76

Figure 5.1 Gene IGFBP-5 positive correlated with HW/BW ratio in the mouse experiment using the I2-ACWE method...92

Figure 5.2 Gene IGFBP-5 positive correlated with HW/BW ratio in the mouse experiment using affymetrix method. 92

Figure 5.3 Gene path negatively correlated with the HW/BW ratio in the mouse experiment using the I2-ACWE method. 93

Figure 5.4 Gene pah negatively correlated with the HW/BW ratio in the mouse experiment using the Affymetrix method. 94

Figure 5.5 Gene MMP3 positively correlated with LVEF in the human experiment using the I2-ACWE method. 95

Figure 5.6 Gene MMP3 positively correlated with LVEF in the human experiment using Affymetrix method. 96

Figure 5.7 Gene NPPA negatively correlated with LVEF in the human experiment using I2-ACWE method. 96

Figure 5.8 Gene NPPA negatively correlated with LVEF in the human experiment using Affymetrix method. 97

Figure 5.9 Boxplot of the gene GAPDH expression one of the experiments of human model. 97

ACKNOWLEDGEMENTS

I owe great appreciation to many people. My greatest gratitude is to my advisor, Dr. Andrei Paun, for supporting my work for years and for his invaluable guidance and generous encouragement. It is my honor to be his student. This dissertation could not have been completed without his guidance and advice. I would like to thank Dr. Weizhong Dai, Dr. Raja Nassar and Dr. Mihaela Paun for their warmhearted help. In their classes, they taught me mathematics & statistics so that I could apply the knowledge to my research work. Sincere acknowledgement is also extended to Dr. Vir V. Phoha and Dr. Ben Choi for their kindness of serving as advisory committee members.

To my friends and family, thanks for your company and for bearing with my long absences over the past five years. I would like to thank my father, Xingkang Ni, my mother, Liyuan Sheng, and my wife, Hui Wu, for their love, understanding and endless encouragement. Finally, I want to express my appreciation to all my friends. This dissertation is dedicated to all the above mentioned.

CHAPTER 1

INTRODUCTION

1.1 General Overview

A DNA microarray is an array of DNA spots. DNA strands are fastened at fixed spots on glass or plastic slides or silicon chips. A DNA microarray is a useful tool for analyzing gene expression based on the samples of genes in the spots aligned in a regular pattern. DNA microarrays provide a new technology for doing scientific experiments simultaneously with thousands of genes or entire genomes. This approach is much more efficient than the traditional experiment method which only focuses on a few genes at one time. A critical part in the gene analysis process is the effectiveness of image segmentation analysis. There are two different types of DNA microarray: complementary DNA (cDNA) microarray and Affymetrix GeneChip.

For the cDNA microarray there are four types of segmentation methods described in [1]: fixed circle segmentation, adaptive circle segmentation, adaptive shape segmentation and histogram segmentation. Some cDNA microarrays segmentation software have been developed based on the four types of segmentation methods. ScanAlyze [2] developed by Eisen in 1999 is based on the fixed circle segmentation method. This method assumes that the spot has a perfect circle shape and all spots have the same size. GenePix [3] developed by Axon Instruments Inc. in 1999 uses an adaptive circle segmentation method. This method assumes as well that the spot has a circular shape but also allows for adjusting the size of each spot. It provides more accurate results than the fixed circle method. QuantArray [4] software developed by the GSI Lumonics Corporation in 1999 gives the histogram segmentation for using the mean intensity values of pixels between the 5th and 20th percentile as the background and the mean between the 80th and 95th percentile as the foreground. Most of the histogram segmentation algorithms neglect the spatial information of pixels. The histogram segmentation algorithms also need to pre-define the threshold for the segmentation. SPOT [5] is a software developed by Y.H. Yang, M. J. Buckley, and T. P. Speed in 2002 and described in [1] and applied the adaptive shape segmentation. The software provides two types of adaptive segmentation methods. One is the Seeded Region Growing (SRG) first described in [6] and the other is the Globally Optimal Geodesic Active Contours (GOGAC) first described in [7]. The adaptive shape segmentation methods assume that the spot is adaptive in size and can be of irregular shape. Affymetrix GeneChip [8] segmentation uses a 75 percentile intensity value of each probe cell as the segmentation result for each probe cell.

Since DNA microarray segmentation is an important step in microarray image processing and current DNA microarray segmentation methods have not provided the most accurate results [1], a more accurate DNA microarray segmentation method is needed for DNA microarray data analysis. The motivation of my dissertation is to find out the most accurate ACWE segmentation method to apply it in DNA microarray and make improvements to the ACWE method to get even better segmentation results.

1.2 Research Objects

The objective of my dissertation is to build a better segmentation method for DNA microarray image processing.

Segmentation is a partitioning process used to separate a spot area from a non-spot area in DNA microarrays. It directly affects the accuracy of gene expression analysis in the data mining process that follows. A number of DNA microarray segmentation methods have been proposed in the area, but even modern segmentation methods have accuracy problems.

We present a segmentation method based on the Active Contours Without Edges (ACWE) algorithm and we apply it in two types of DNA microarrays: complementary DNA (cDNA) and Affymetrix GeneChip.

We performed several adjustments to the original ACWE method to use it more efficiently in the microarray processing area. We also improved the ACWE method to get more accuracy segmentation results.

We will present various experimental results to show that the ACWE method is better than the other DNA microarray image segmentation methods in finding out the DNA spots' boundaries. We will apply statistical tools to compare and contrast the newly proposed method with the previously best methods in this area.

1.3 Organization of the Dissertation

Chapter 1 introduces our research work and gives contents for the remaining chapters.

Chapter 2 reviews some background of the biological knowledge of DNA and DNA microarrays. Some previous work about cDNA Microarray Image Analysis

Methods, Adaptive Shape Segmentation Method, and the segmentation method for Affymetrix GeneChip are discussed in this chapter.

Chapter 3 presents the ACWE theory and its advantages compared to the previously introduced methods. In this chapter, we make the modifications to the ACWE method and develop the process to apply the ACWE method in the DNA microarray segmentation field. Also, the improvement for the ACWE method is proposed.

In Chapter 4, the experimental results of which ACWE is compared with the other methods discussed.

Finally, in Chapter 5, the improved ACWE method is shown to be much more efficient than the ACWE method itself and the other methods discussed.

CHAPTER 2

BACKGROUND AND PREVIOUS WORK

2.1 Introduction of DNA and DNA Microarray

This section provides an introduction into the molecular biology with the purpose of understanding the DNA microarray segmentation problem. In what follows, we will present history, structure and some properties of the DNA strands and the DNA microarrays.

2.1.1 DNA

DNA is the abbreviation of deoxyribonucleic acid and contains important inheritance information. DNA structure is stable and the genetic information can carry on from one generation to the next. It encodes all genetic information of an organism and all instructions needed for the functioning of that organism.

According to [9], in 1869, Friedrich Miescher first discovered the nucleic acid. During his experiment, Miescher found that a material had different properties from the protein component of the material. Since this material was separated from the cell nucleus, Miescher named it "nuclein", which is now called nucleic acid. In 1919, Phoebus Levene in [10] found the nucleic acid bases in the yeast nucleic acid through the experiment involving hydrolysis. He named these four substances Adenosinphosphoric acid, Uridinphosphoric acid, Guanosinphosphoric acid, and Cytidinphosphoric acid. Now we know these 4 bases are from the ribonucleic acid (RNA) and now are called adenine

(A), uracil (U), guanine (G) and cytosine (C). In 1952, Alfred Hershey and Martha Chase proved that DNA was the genetic material [11]. In their experiments they used a bacteriophage (phage) which was a virus to invade (infect) an E. Coli bacterium. By using the radioactive markers on DNA materials inside the phage's shell and the proteins constructing the phage's shell, they found that the real part which infected the bacteria was the DNA. Only the DNA could get into the bacterium to replicate and reconstruct multiple copies of the phage inside the bacterium cell, and the original phage shell could not enter into the bacterium cell, (since the radioactive marker for the protein in the shell could not be found inside the bacterium cell). This proved that DNA was the genetic material. In 1953, James D. Watson and Francis Crick [12] presented the first accurate DNA structure model, the double helix. This three dimensional DNA model gave a solid foundation for the subsequent research on DNA.

The DNA direction is from 5' –end to 3' –end (5 or 3 means carbon atoms in the deoxyribose). The enzyme named DNA polymerase adds complementary nucleotides to a single stranded DNA molecule to form a double stranded DNA molecule. DNA is found in the nucleus (in eukaryotes cells). It encodes and transmits genetic information to Messenger Ribonucleic Acid (mRNA) which passes the encoded instructions of making protein.

Ribonucleic acid (RNA) [13] is another type of nucleic acid found in the cytoplasm. RNA is similar to DNA with some important differences:

1. RNA is single stranded in the cell.
2. RNA contains ribose rather than deoxyribose.
3. RNA has base uracil (U) in place of thymine (T).

4. In RNA there are also four bases: A, U, C and G.

DNA has genetic information and controls the production of proteins in a cell. DNA is able to replicate and also able to mutate. Messenger RNA (mRNA) is the template working between DNA and the proteins. The information from a particular gene is transferred from a strand of DNA by forming a complementary strand of RNA. This process is called gene transcription. It transcribes a strand of DNA to a strand of complementary mRNA. A codon [14] is a triplet of nucleotides in mRNA. Transfer RNA (tRNA), which belongs to specific amino acids, match up the codons of mRNA to order the amino acids to form protein molecules in a process called translation.

There are a total of 64 RNA codons [14]. AUG is the start codon whereas the stop codons are UAA, UAG, or UGA. The mRNA sequence base between a start codon and a stop codon is called an Open Reading Frame (ORF).

The central dogma of molecular biology is described in [15]. The central dogma shows the genetic information pass from DNA through RNA to the proteins.

The replication process is for the DNA to make a copy of itself. During the process the base pairs of the two DNA strands open and each DNA strand acts as a template. Two complement strands are reproduced which achieve the DNA duplication process. The transcription process is for the DNA to make a copy for one of its strands. The transcription is a synthesis process for RNA to produce the messenger RNA (mRNA) which will be used in the translation process to create proteins. The translation process is for protein synthesis. The bases of mRNA are formed with a set of codons. The genetic information from the codons is translated into proteins.

Gene [16] is a segment of a long strand of DNA. Gene expression [17] is the process in which genetic information pass from a gene (DNA sequence) into mRNA or protein. DNA microarray is an effective tool which can be used to monitor many genes expressions at the same time.

2.1.2 DNA Microarray

A DNA microarray is an array of DNA spots. DNA strands are fastened at fixed spots on glass, plastic slides or silicon chips. In [18] a DNA microarray is a useful tool for analyzing gene expression based on the samples of genes in the spots aligned in a regular pattern. DNA microarrays provide new technology for doing scientific experiments simultaneously with thousands of genes or entire genomes. It is much more efficient than the traditional experimental method which only focuses on a few genes at a time. According to [19] the first DNA microarray prototype was created in 1989 by Stephen P. A. Fodor, who is the founder and executive chairman of Affymetrix Inc., and his colleagues.

In 1995, the first DNA microarray in gene expression analysis is proposed [20]. Microarrays were used to find out the overexpression of genes. The experiment from [20] used the plant Arabidopsis thaliana as a study object. By using complementary DNA (cDNA) microarrays, one gene named “HAT4” was figured out to be much more overexpressed than the other genes. Their experiment showed that a DNA microarray could be used to monitor gene expression faster and more effectively.

Typically with the manufacturing method, there are two types of DNA microarray: spotted and oligonucleotide. Spotted microarrays are cheap and the polymerase chain reaction (PCR) method is used to produce the sequences on the array spots. The probes in the arrays are long sequences. Oligonucleotide microarrays are expensive and the spots are made of oligonucleotides. The probes in the arrays are short sequences.

For the spotted array, the DNA sequence may or may not be known and there is little control of the amount of DNA in a spot. For the oligonucleotide array, the DNA sequence is known as a perfect match (PM) and mismatch (MM). PM and MM are paired and used as controls of DNA. Since an oligonucleotide array has more probe controls in the microarray than that of the spotted array, the oligonucleotide microarray is more efficient than the spotted microarray; this is the same reason that makes the oligonucleotide microarray more expensive than the spotted array.

Typically a complementary DNA (cDNA) microarray experiment [21] includes the following 6 steps:

- In the sample preparation step, two samples are selected. For example, one is a normal sample, the other a disease sample.
- In the nucleic acid isolation and purification step, the mRNA of the two samples are extracted.
- In the reverse transcription step, mRNA is transcribed to cDNA.
- In the hybridization step, the cDNA is tagged with fluorescent dye. Tagged cDNA sequence is hybridized to a microarray. The excess tag cDNA is washed away from a microarray.
- Next follows the laser scanning step.
- And last is the analysis step. In this step, data is extracted from the microarray.

For a cDNA microarray, the gene expression is checked between a control sample (normal sample) and an experimental sample (disease sample). These samples are labeled with the fluorescent dyes Cy3 (control) and Cy5 (experimental). Cy3 is for the green channel and has a wavelength of about 530 nm. Cy5 is for the red channel and has a wavelength of about 630 nm [1]. The tagged samples are targets and hybridized to microarray a substrate that contains probes. After washing the excess tagged tissues from the microarray, the microarray is scanned by a laser scanner at different wavelengths related to the green and red dyes. The outputs are two digital pixel intensity images stored in Tagged Image File Format (TIFF). One is a 16-bit image in Cy3 channel and the other is in the Cy5 channel. A spot represents a particular gene.

According to [22] an Affymetrix Gene Chip microarray experiment includes the following steps:

1. Sample preparation, where only one sample is selected from one GeneChip microarray.
2. Isolation and Purification, where mRNA is isolated and purified.
3. Reverse transcription, where mRNA is transcribed to cDNA.
4. In Vitro Transcription (IVT), where cDNA is transcribed to complementary RNA (cRNA) and cRNA is labeled.
5. Fragment, where cRNA is fragmented to short pieces.
6. Hybridization, where cRNA is hybridized onto microarray.
7. The laser scanning step.
8. And the analysis step, where data is extracted from the microarray.

For the Affymetrix GeneChip, after choosing a sample, the mRNA is isolated and purified from the sample. Then the mRNA follows a reverse transcription to cNDA and cDNA is labeled using In Vitro Transcription (IVT) or alternative labeling methods. After labeling, the cRNA is obtained from cDNA and hybridized to GeneChip microarray. Then the excess cRNA is washed away. After scanning with a laser scanner, the microarray will produce a 16-bit gray scale TIFF image. The last process is the data analysis for the gene expression.

There are two types of DNA microarray experiment devices. Affymetrix GeneChip experiment devices are much more expensive than those of the cDNA microarray.

2.2 Introduction of cDNA Microarray Image Analysis Methods

This section introduces the DNA microarray image analysis methods. Especially for the DNA microarray image segmentation, different methods are introduced and comparison results among these methods are also discussed in this section.

2.2.1 cDNA Microarray Image Analysis Methods

In [1], there are three steps for microarray image analysis: addressing (or gridding), segmentation, and information extraction. Addressing is the method of arranging each spot to a grid. Segmentation is the process of discriminating the spot out of the background. Information extraction is the way of computing the intensity of each spot, background intensity, etc. Image addressing is important since it provides the spot or probe cell location. Image segmentation is even more important, as it separates the spot from the background. The exact intensity value of the spot or probed cell can be accomplished in the information extraction step.

The goal of addressing (gridding) is to reliably compute the intensity of the spots (probe cells) according to the microarray layout design. It provides the geometric location of each spot or probe cell. Each spot sits in a patch which is a square or rectangle. The center of the patch in an ideal spot center and the region between the center and the boundary of the patch is used for finding out the boundary of each spot. However, in reality, some items may influence the accuracy of the geometric location for spots and probe cells. For example, the location of the grid may change between slides, the array image may rotate, the sub-array location of the image may shift, the microarray image may be contaminated, and the signal of some spots or probe cells may be weak. These situations will be addressed in the image analysis process.

Addressing methods can be divided into three main categories: (1) manual (2) semi-automated, and (3) automated. Manual and semi-automated methods require user input or adjustment. Automated methods still need user defined parameters or give out the threshold for refining the intensities. SpotSegmentation developed by Q.H. Li and C. Fraley [23] in 2005 is one of the software programs that claims to provide robust gridding processes. Figure 2.1 is the output of spot segmentation gridding. It shows the inaccuracy of the automated addressing method since some of the spots in the bottom left of the image are not in the grid. Until now, researches mostly focused on automated addressing methods for more accurate results. A complete automated addressing method has not been found yet.

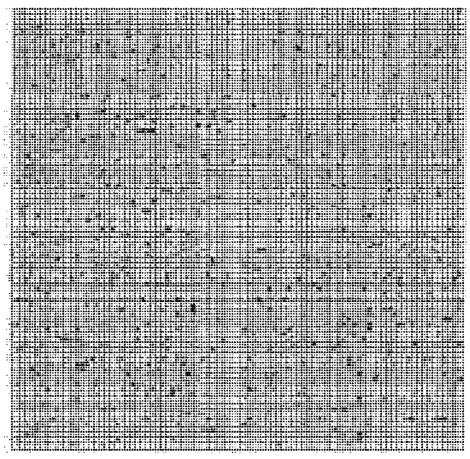

Figure 2.1 Gridding output of spot segmentation.

2.2.2 cDNA Microarray Image Segmentation Methods

Segmentation is a partition process used to separate a spot area from a non-spot area. The spot area is called foreground and the non-spot area is called background.

In [1], there are four types of segmentation methods: fixed circle segmentation, adaptive circle segmentation, adaptive shape segmentation and histogram segmentation.

Fixed Circle Segmentation Method is an ideal method. It assumes that each spot has the same size of circle shape. It was first used in [2] by ScanAlyze which is an image segmentation software developed by Eisen Lab. All spots have a round shape and the same size with the same radius (R) in the microarray image. It is easy to implement this segmentation method but the disadvantage is that the real spots may not be the circular shape or the same size. It is observed that all the spots in the figure actually have the size slightly smaller than the size of the fixed circle. Most of the time, the fixed circle segmentation cannot provide the exact spot boundary or accurate spot intensity.

The Adaptive Circle Segmentation Method is the evolution of the fixed circle segmentation method. It allows the diameter of each spot circle to be adjusted separately.

It is better than the fixed circle, since it considers that spots may have different sizes. But the disadvantage of adaptive circle still exists since the spots shape may not be circles. It is observed from the figure that if the spot size is close to circular shape, it will get pretty good segmentation results. But still for some spots in the figure which have a square shape, the segmentation result is not accurate. The adaptive circle method may figure out the similar size of the spot, but it cannot provide the accurate boundary of the spot since most spots do not have a regular circle shape. Therefore, it cannot give the spots' correct intensities either.

The Histogram Segmentation Method uses the normal distribution of the pixels' intensity percentiles around and inside each spot to segment the spot from the background. Obviously, this method neglects the particular pixels' locations. It will not give the accurate spot intensity but only the trend. For example, in [4] the QuantArray software developed by the GSI Lumonics Corporation in 1999 gives the histogram segmentation for using the mean intensity values of pixels between the 5th and 20th percentile as the background and the mean between the 80th and 95th percentile as the foreground. In [23], the spot Segmentation software also uses the histogram segmentation method. Figure 2.2 presents the output with the histogram segmentation method using the spot segmentation software. From the figure some of the spots (gray) have been segmented out (the spots that have been segmented are colored black) by the histogram segmentation method. Since the spot's actual geometric location was not taken care of, the output only gave the distribution range. The threshold of the percentile can be chosen differently among different implementation software. It cannot find the exact boundary of the spot.

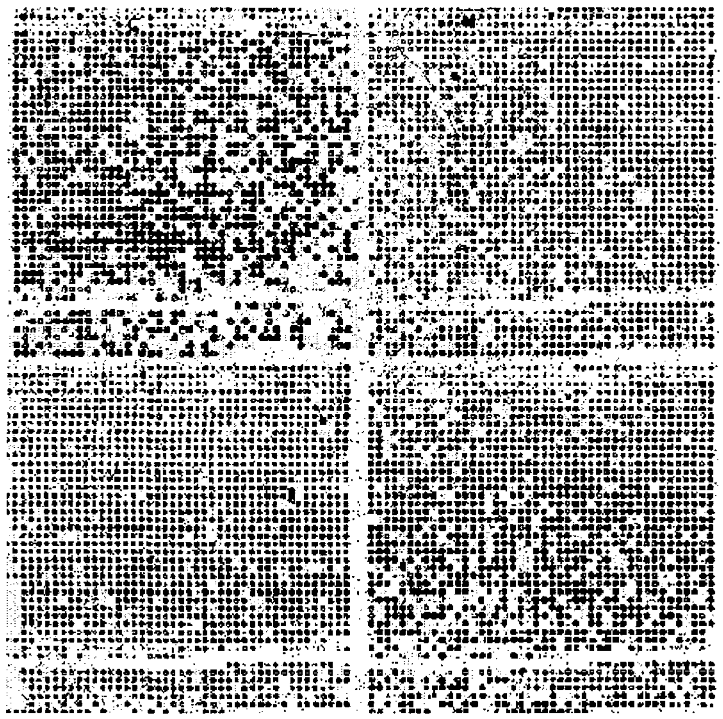

Figure 2.2 Histogram segmentation.

The Adaptive Shape Segmentation Method is designed to improve the accuracy of the segmentation process. It tries to find out the real spot boundary which separates the spot with the background. The adaptive shape segmentation is better than the other three segmentation methods, since it corrects the weaknesses of the other three methods. Seeded Region Growing (SRG) and Globally Optimal Geodesic Active Contours (GOGAC) are the two algorithms applied in the adaptive shape segmentation method. Figure 2.3 is an example of adaptive shape segmentation implemented in Spot software (developed by Y.H. Yang, M.J. Buckley and T. P. Speed in 2002) [5] using the GOGAC algorithm. The figure shows that the segmentation is based on the changing of shape of each spot. But it still shows that some of the spots' boundaries are inaccurate.

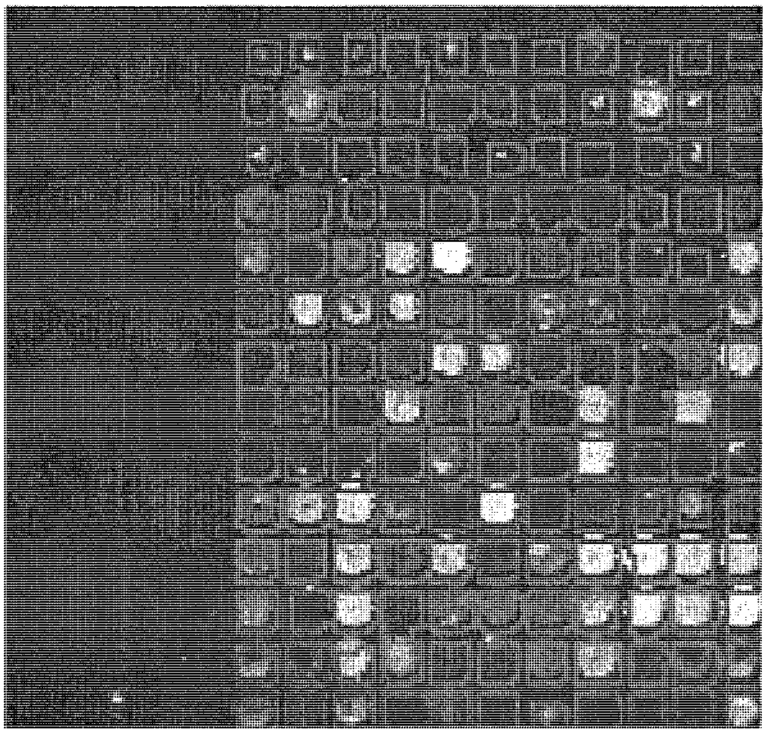

Figure 2.3 Adaptive shape segmentation using GOGAC.

2.2.3 cDNA Microarray Image Information Extraction

After the segmentation step, we can get the foreground and the background intensity values of each spot in the information extraction step.

In [1] each pixel within the foreground area of a spot patch is counted and the mean and median intensity values of the spot area are computed. The mean and median intensity values of the background are also computed.

In cDNA microarray, the ratio between the red and green channel is often used to represent gene expression. The ratio can be computed as follows in Equation 2.1:

$$ratio = \frac{Ch2}{Ch1} = \frac{CH2I - CH2B}{CH1I - CH1B}, \tag{2.1}$$

where $Ch1$ represents the green dye channel ,$Ch2$ represents the red dye channel ,$CH1I$ is the mean intensity value for the spot foreground of the green channel , *CH1B* is the median intensity value for the spot background of the green channel , *CH2I* represents the mean intensity value for the spot foreground of the red channel , and *CH2B* represents the median intensity value for the spot background of the red channel.

Information extraction step also collects the information for quality measurement.

For example, the spot size is calculated by measuring how many pixels are in a spot area. If a spot only contains 1 pixel, it may not be qualified for the further DNA data analysis.

2.2.4 cDNA Microarray Image Data Normalization

In [24], normalization is used to reduce the variation created by the cDNA microarray processing technology. The main reasons caused the variation are as follows: labeling efficiencies, different scanner settings and difference between print tips.

All these may cause the gene expressions' imbalance between the red and green channels.

$$M = \log_2 R - \log_2 G, \tag{2.2}$$

$$A = (\log_2 R + \log_2 G)/2, \tag{2.3}$$

where M is the intensity ratio, A is the average intensity, R is the background-corrected intensity value for each spot from the red channel, and G is the background-corrected intensity value for each spot from the green channel.

Equation 2.2 represents the log ratio of gene expression, and Equation 2.3 represents the average of log intensity. A plot is formed by using M as Y-axis and A as

X-axis, which we call an MA-plot. Such a plot can be used as a monitor for the imbalance gene expression between the red and green channels based on the intensity values.

According to [24], there are several normalization methods for cDNA microarray data. The first is the print tip loess normalization. The second is the composite loess normalization. The third is between array normalization.

Loess is a modern regression model different from linear regression and non-linear least square regression. In [25], loess regression is defined as locally weighted scatterplot smoothing.

For the print tip loess normalization in [24], there are three normalization methods:

1. The global loess normalization can be described as follows in Equation 2.4:

$$N = M - loess(A), \tag{2.4}$$

where N is the normalized log ratio, M is the log ratio, and *loess(A)* is the global loess regression curve.

2. The two-dimensional loess normalization is as follows in Equation 2.5:

$$N = M - loess(r, c) - loess(A), \tag{2.5}$$

where N represents the normalized log ratio, M represents the log ratio, *Loess(A)* represents the global loess regression curve, and *Loess(r,c)* represents a two-dimensional loess regression curve, where r and c represent the row and column of the spot.

3. Standardize the N values in Equation 2.6:

$$N_s = N / mad_i \tag{2.6}$$

where N represents the normalized log ratio, mad_i is the median absolute deviation of N

in the i^{th} tip group, and N_s represents the scale-normalized log ratio.

For the composite loess normalization in [24] is as follow in Equation 2.7:

$$N = M - p(A)loess_{MSP}(A) - \{1 - p(A)\}loess_i(A), \tag{2.7}$$

where $loess_{MSP}(A)$ is the loess regression curve through the microarray sample pool (*MSP*) spots, and *p(A)* is the probability of spots whose A values are less than A.

For the between array normalization, scale normalization is a useful tool. Scale normalization adjusts the data range and makes the data comparable among the arrays.

Through normalization non-biological variations can be reduced and gene expression can be compared easily among the microarrays by using the same scale.

2.2.5 cDNA Microarray Image Data Analysis

According to [26], the main purpose for data analysis is to find out which genes are differentially expressed and also to figure out the differentially expressed genes between two samples.

In [26] t-test is used in Equation 2.8 for finding the differentially expressed genes.

$$t = \frac{\overline{M}}{s / \sqrt{n}} \tag{2.8}$$

where $\overline{M}$ is the mean of the log ratio, s is the standard deviation of M, n is the replication numbers, and t is for t-test.

To figure out the differentially expressed gene between sample A and B [26], a two sample t-statistic can be used as follows in Equation 2.9:

$$t = \frac{\overline{M_A} - \overline{M_B}}{s^* \sqrt{1/n_A + 1/n_B}}, \tag{2.9}$$

where $s^* = \sqrt{a + s^2}$, $\overline{M_A}$ and $\overline{M_B}$ are the mean log ratios from two samples, n_A and n_B

are the replication numbers from two samples, s^* is the penalized pooled sample standard deviation of M, s is the standard deviation of M, a is the penalty, and t is for the two sample t-test. cDNA microarray data analysis uses t-test as an important tool for ranking the genes based on their expression differences.

Another approach of cDNA microarray data analysis is classification [26]. Classification is used to find the gene expression level similarity or dissimilarity among the samples.

In [26], two types of methods are mentioned as the tools for classification.

First, there are the cluster methods. K-mean clustering [27] is one of the cluster methods, which can classify different genes into different clusters (groups) based on their gene expression level similarity.

Second, there are the discrimination methods. Support Vector Machines (SVM) [28] is one of the discrimination methods and is an important data mining tool which is used to classify cDNA microarray gene data.

Sections 2.2.1 to 2.2.5 show the entire picture for cDNA microarray image data processing steps. Section 2.1.2 introduces how to produce the cDNA microarray images.

These sections (2.2.1-2.2.5 and 2.1.2) show how to get the useful gene expression information from these cDNA microarray images and these gene expression values are much more helpful in the cluster analysis, function prediction for future analysis.

Another type of DNA microarray (Affymetrix GeneChip) will be discussed in Section 2.4.

2.3 Adaptive Shape Segmentation Methods

In this section we will describe the adaptive shape segmentation concept and present different methods of this type of segmentation. Adaptive shape segmentation will provide more accurate results than those of the fixed circle, adaptive circle and histogram segmentation methods. The most popular adaptive shape segmentation methods are the Watershed Method, the Seeded Region Growing (SRG) Method and the Globally Optimal Geodesic Active Contours (GOGAC) Method.

2.3.1 Watershed Method

The Watershed Segmentation Method treats an image as a topographic surface. When the water enters from the minima it will flood the surface. The only visible surface after the flood is called the watershed lines. The areas segmented by the watershed lines are different catchment basins. Watershed segmentation has the weakness of over-segmenting the original image. In [29] watershed was first applied to detect and image objects contours. The watershed lines were used as an object's contours while using the variation function *g*. Equation 2.10 proposed in [29] has the following definition:

$$g(x)=\lim_{\varepsilon\to 0}\frac{Sup_{B(x,\varepsilon)}[f]-Inf_{B(x,\varepsilon)}[f]}{2\varepsilon}, \tag{2.10}$$

where *f* is the grey function of an image, $Sup_{B(x,\varepsilon)}[f]$ is the maximum value of *f* within a ball of radius ε and centered in *x*, and $Inf_{B(x,\varepsilon)}[f]$ is the minimum value of *f* in a space centered in *x* with radius ε.

Matlab was used in [30] to provide an implementation of watershed algorithm. The reason for the over-segmentation problem of the watershed segmentation was also mentioned in [30] as being caused by every regional minimum that would tend to create its own catchment basin. Figure 2.4 is an example of applying the watershed segmentation method on a cDNA microarray image using Matlab implemented in the watershed algorithm.

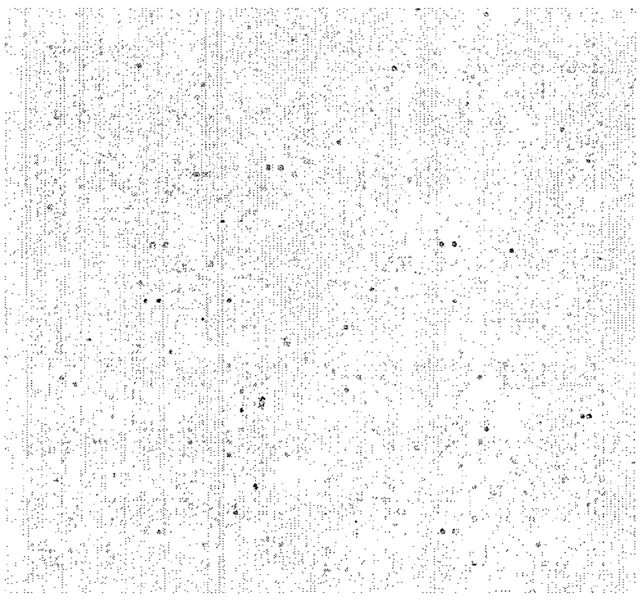
Figure 2.4 Watershed segmentation.

The result from Figure 2.4 shows that the watershed segmentation would not give the accurate segmentation to each spot in the cDNA microarray image. The over-segmentation problem was still there and also some spots had been overlooked using this method. To overcome the over-segmentation problem, some markers can be applied to some minimum local catch basins of the image. Even so, some segments do not relate to any geometric region.

2.3.2 Seeded Region Growing Method

Seeded Region Growing (SRG) Segmentation Method starts with some seeds (starting points). Then it includes the neighboring pixels and checks if they have the same intensity. This process will continue until all the pixels have been checked and each pixel will be put in the region that belongs to one of the seeds. The formed regions are connected and homogeneous. Based on the given seeds' properties, a region is defined from the difference between this region and its neighborhoods. Each region competes with other regions and grows. When growing is finished, the segmentation is also finished.

In [6] the Seeded Region Growing (SRG) Algorithm was proposed. At the beginning there were n groups of seeds, named $A_1, A_2, ..., A_n$. Pixel *x* was unassigned to the groups ($A_1, A_2, ..., A_n$), but at least one of the neighbor pixels of pixel *x* had been assigned to these groups. All pixels with the same property (itself has not been allocated, but at least one of its neighbors has been) as pixel *x* was put in the set named *T*. *T* is defined as $T = \{x \notin \bigcup_{i=1}^{n} A_i \mid N(x) \cap \bigcup_{i=1}^{n} A_i \neq \emptyset\}$, where *N(x)* collects all the neighboring pixels of pixel *x*.

If for a pixel *x*, *x* was not assigned to any one of the groups ($A_1, A_2, A_3, ..., A_n$), only one of pixel *x*'s neighbors has been assigned to ($A_1, A_2, A_3, ..., A_n$), then $i(x) \in \{1,2,...,n\}$ is used as the index so that $N(x) \cap A_{i(x)} \neq \emptyset$. The difference between pixel *x* and its neighboring pixel was defined as $\delta(x) = |g(x) - \underset{y \in A_{i(x)}}{mean}[g(y)]|$, where *g(x)* is the intensity value of pixel *x*.

If the difference is less than a tolerance value, then pixel *x* will be assigned to the same group of its neighbor pixels which has already been assigned.

If for a pixel *x*, *x* was not assigned to any one of the groups ($A_1, A_2, ..., A_n$), at least two of pixel *x*'s neighbors have been assigned to ($A_1, A_2, ..., A_n$), then *i(x)* will have the value as follows: $i(x) = \{i \mid N(x) \cap A_i \neq \emptyset \wedge \delta(x)$ is minimized$\}$, where *i*(*x*) uses *i* as its value so that $N(x) \cap A_{i(x)} \neq 0$ and chooses the minimum of the differences between pixel x and its neighboring pixels (those assigned to $(A_1, A_2, \ldots, A_n)$).

If the difference is less than a tolerance value, then pixel x will be assigned to the same group of its neighboring pixel which has already been assigned and has the minimum difference.

The SRG algorithm was provided in [6] as follows:

```
Step 1. Mark seed pixels based on their initial grouping
Step 2. Add neighbor pixels of seed pixels (the initial $T$) in the SSL (Sequentially
        Sorted List).
Step 3. While the SSL is not empty,
Step 4. Take out the first pixel $y$ from SSL
Step 5. Check the neighbors of $y$
        {
Step 6. If all neighbors (are marked but not marked with the boundary mark) of $y$
        have the same mark,
Step 7. Set $y$ to this mark
Step 8. Change running mean of related region
Step 9. Put neighbors (not marked or not in the SSL) of $y$ to the SSL by their $\delta$
        value.
Step 10. Else
Step 11. Mark $y$ with the boundary mark.
        }
```

The δ value in the algorithm was defined as follows:

If the noise in each region has the equal variance, then $\delta(x) = |g(x) - \underset{y \in A_{i(x)}}{mean}[g(y)]|$

or $\delta(x) = \left| \frac{g(x) - mean_y \in A_{i(x)}[g(y)]}{SD_{y \in A_{i(x)}}[g(y)]} \right|$, otherwise, *SD* is the standard deviation.

The weakness of the seeds' region growing is that if the seeds are chosen improperly, the segmentation result would not be accurate. Figure 2.5 presents the SRG segmentation method implemented in [5] with Spot software applied on a cDNA image. The figure shows that some spots were not correctly segmented. For example, the yellow arrow in the image point out one spot which is not correctly segmented.

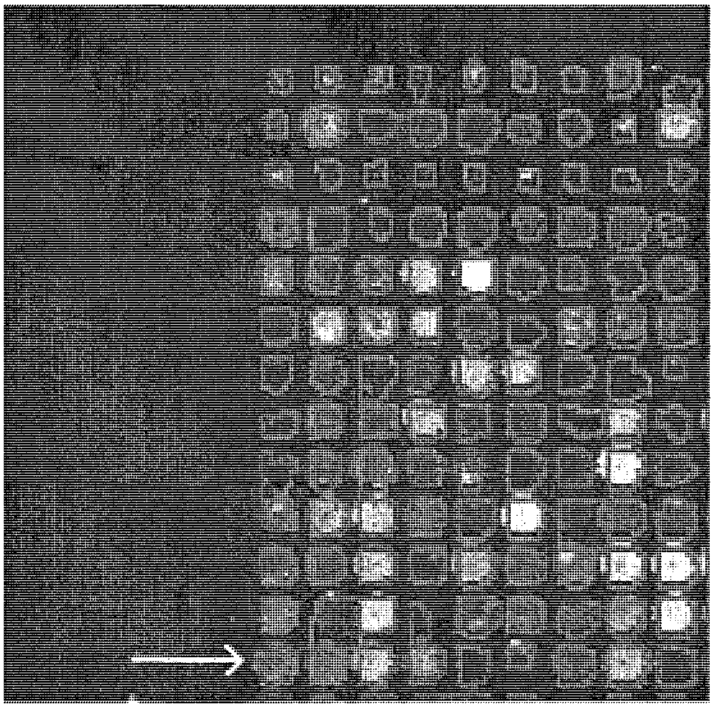

Figure 2.5 Seeded region growing segmentation.

2.3.3 Globally Optimal Geodesic Active Contours Method

Globally Optimal Geodesic Active Contours (GOGAC) was first proposed in [7]. The GOGAC segmentation method searches the geodesic active contours with globally minimal energy containing an internal point p_{int}.

In [7] a general algorithm of Globally Optimal Geodesic Active Contours is presented with the following steps:

1. Initialization:

- Assign the root search cut node R (P_{cut}^{root}) with ∞ as lower bound
- Mark P_{cut}^{root} as open
- Enqueue R

2. Priority First Search (infinite loop):

- Delete the search cut node n of least lower bound from the priority queue
- If n is marked as closed:

– Assign the minimal closed geodesic corresponding to n

Halt

- Else

  - Calculate the surface of minimal action U in the helical surface space S from the start set of n.

  - Halt the calculation early when at least one element of each end set of χ_1 and χ_2 has been checked

  - Find out the end of the geodesic: $p_{end} = \arg\inf\{U(p_{end}) \mid p_{end} \in P_{end}\}$

  - Obtain the minimal geodesic $C_{\min}$ and the start point p_{start} for n by gradient descent from p_{end} to p_{start}

- For each child χ of the search tree:
  - Assign P_{start}, P_{end} be the start set and end set of χ
  - Let χ be a lower bound $\min\{U(p_{end}) \mid p_{end} \in P_{end}\}$
  - Mark χ as closed if P_{start} and P_{end} are both located in χ and are connected in the discrete grid
  - Enqueue χ

The proposed GOGAC algorithm was implemented in Spot software [5]. The weakness of this method is that it prefers to produce circles, it cannot prevent overlap, and it is slower than SRG. Figure 2.6 shows the GOGAC segmentation method using the Spot software. We used the same original cDNA microarray image in Figure 2.5 and Figure 2.6. In Figure 2.6 the spot (the same spot pointed at by a yellow arrow in Figure 2.5) indicated by a yellow arrow showed it was not correctly segmented either.

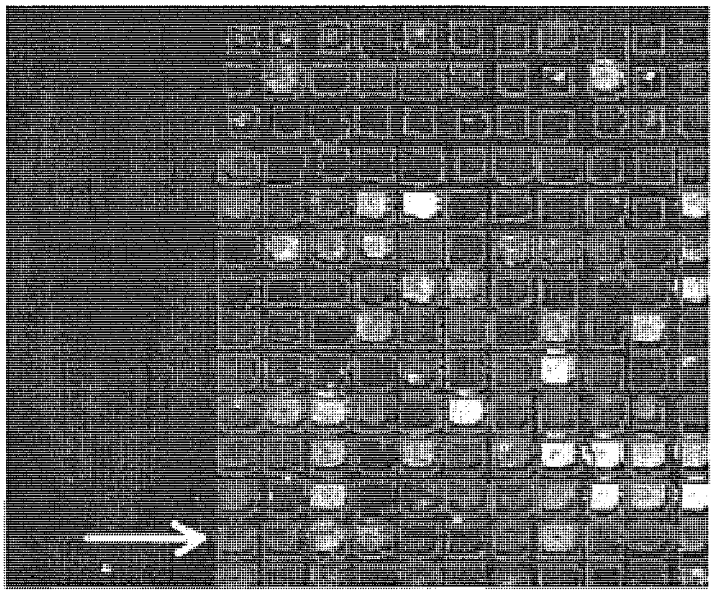

Figure 2.6 GOGAC segmentation using Spot.

2.4 Affymetrix GeneChip and Affymetrix Segmentation Method

In this section, the oligonucleotide microarray, especially Affymetrix GeneChip, is introduced. The image segementation method of Affymetrix GeneChip is also discussed.

2.4.1 Affymetrix GeneChip

The GeneChip is a microarray of short oligonucleotide sequcences created by Affymetrix Inc. In the GeneChip, the gene expression sequence is represented by 11-20 unique probe pairs (probe set). Each probe cell has a 25 mers base length. Each probe pair has a perfect match (PM) probe cell and a mismatch (MM) probe cell. The difference between MM and PM probes is on the 13th base location. Figure 2.16 is an example of a probe pair. In Figure 2.7 the perfect match probe cell has 25 mers bases in length. The mismatch probe cell has the same length. The difference between the PM cell and the MM cell was the 13th base location (PM was 'C' , MM was 'G'). The 13th base of MM was the complement of the 13th base of PM.

Perfect match (PM) AAGAATCTATGCCAGTAGTCATCTA
Mismatch (MM) AAGAATCTATGCGAGTAGTCATCTA

Figure 2.7 Probe pair.

According to [31] and [32], the database which are used for Affymetrix GeneChip image segmentation include seven different types.

For the human experiment, there are two types of GeneChip. One is HgU95Av2, the other is HgU133plus2.

HgU95Av2 contains 12,625 probe sets. Each probe set has 16 probe pairs and each probe cell has 25 mers base length.

HgU133plus2 has 54,675 probe sets. Each probe set contains 16 probe pairs and each probe cell has the base length of 25 mers.

There are four types of GeneChip in the mouse experiment. They are MgU74A, MgU74Av2, Mg430 2.0, and Mu11k Set.

MgU74A includes 12,654 probe sets. Sixteen probe pairs form a probe set. The base length for each probe cell is 25 mers.

There are 12,488 probe sets in MgU74Av2. The base length for each probe cell is 25 mers and each probe set has 16 probe pairs.

Mg430 2.0 holds 45,101 probe sets. Each probe set has 11 probe pairs. Each probe cell has a 25 mers base length.

Mu11k Set has 2 subtype: Mu11kSubA and Mu11kSubB. Mu11kSubA has 6,584 probe sets. Twenty probe pairs are in each probe set. Base length for each probe cell is 25 mers. Mu11kSubB contains 6,595 probe sets. Probe pairs in each probe set and base length of each probe cell are the same as those in Mu11kSubA.

There is only one type for a rat model. It is RgU34A.

RgU34A has 8,799 probe sets. Each probe set contains 16 probe pairs and each probe cell has a 25 mers base length.

2.4.2 Affymetrix GeneChip Segmentation Method

In the Affymetrix GeneChip image, each probe cell (each PM and MM cell) is constructed with n*n pixels. When Affymetrix method is segmentating the image for the spot intensity, it uses the inner (n-2)*(n-2) pixels. The outer boundary of 4*(n-1) pixels are excluded. The average intensity of the probe cell (spot) is computed by using the 75 percentile of the (n-2)*(n-2) pixels. Figure 2.8 is the example of the Affymetrix segmentation method of a probe cell, when n is 6. The outer 20 pixels are not included in the segmentation. The 75 percentile intensity value of the inner 16 pixels is used as the mean intensity by Affymetrix.

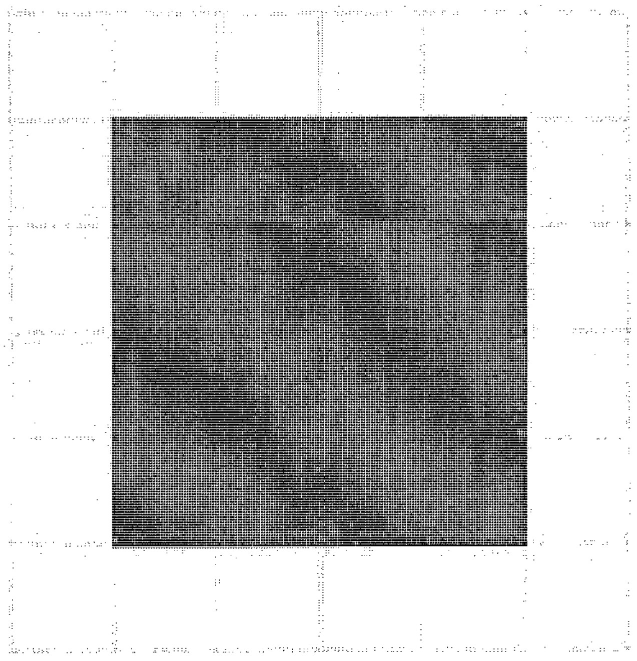

Figure 2.8 Affymetrix probe cell (spot) and segmentation area (red area).

There are different Affymetrix GeneChip file types. Some important types will be introduced since they are used in the ACWE image segmentation method.

A .DAT file is the scanned image file. A .CEL file is the cell file including the intensities and locations of the probe cells. The .CEL intensity values are calculated from the .DAT file. These two files can be used as the data source for further study. Affymetrix also provides a .CDF file which is a library file that defines the probe set and probe pairs. The .CDF file contains the maps between features, probe pairs, probe sets, and genes.

The .DAT file contains the information of the number of pixels of each row in the image, number of rows in the image, pixel coordinates of the image, the image array, etc.

The .CEL file contains the information of the cell array. It includes the coordinates of the cell, the intensity value of the cell, the standard deviation of intensity value, number of pixels in the cell, etc.

The .CDF file contains the information of gene probes. It gives out the number of probe sets, the probe sets' names, each probe pairs coordinated of a probe set, etc.

These files need to be used in the following ACWE segmentation.

CHAPTER 3

ACTIVE CONTOURS WITHOUT EDGES SEGMENTATION METHOD

In this chapter, Active Contours Without Edges Segmentation method will be discussed.

3.1 Active Contours Without Edges (ACWE) Algorithm

The segmentation I implemented was based on Active Contours Without Edges (ACWE) method, which was proposed by Tony F. Chan and Luminita A. Vese in [33]. The Chan and Vese (C-V) model segments an image by detecting the different objects boundaries through evolving a curve. The authors assumed an image was formed by two regions within and outside the objects. Their model can find objects within an image without any definition of gradient. [33] gives an algorithm as follows:

1. Initialize ϕ^0 by ϕ_0, n=0

2. Compute $c_1(\phi^n)$ and $c_2(\phi^n)$ by

$$c_1(\phi)=\frac{\int_\Omega u_0(x,y)H(\phi(x,y))dxdy}{\int_\Omega H(\phi(x,y))dxdy} \quad \text{and} \quad c_2(\phi)=\frac{\int_\Omega u_0(x,y)(1-H(\phi(x,y)))dxdy}{\int_\Omega (1-H(\phi(x,y)))dxdy}$$

3. Solve the PDE in ϕ from

$$\frac{\partial\phi}{\partial t}=\delta_\varepsilon(\phi)\left[\mu div\left(\frac{\nabla\phi}{|\nabla\phi|}\right)-\nu-\lambda_1(u_0-c_1)^2+\lambda_2(u_0-c_2)^2\right]=0 \qquad (3.1)$$

$$\text{in}\,(0,\infty)\times\Omega,\quad \phi(0,x,y)=\phi_0(x,y)\ \text{ in }\ \Omega,\quad \frac{\delta_\varepsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\vec{n}}=0\ \text{ on }\ \partial\Omega$$

where $\vec{n}$ is the exterior normal to the boundary $\partial\Omega$, and $\partial\phi/\partial\vec{n}$ is the normal derivative of ϕ at the boundary.

4. Reinitialize ϕ locally to the signed distance function to the curve (optional)
5. Check whether the solution is stationary. If not, n=n+1 and repeat

This algorithm can be explained as follows:

In Step 1, an evolving curve is initialized.

Step 2, compute the average energy inside and outside the evolving curve.

In Step 3, find out the exact evolving curve location depending on time t by solving the Partial Differential Equation.

Step 4, an optional step for reinitializing the evolving curve.

In Step 5, check whether the solution of the evolving curve is stationary or not. If not, go to Step 2 for iteration.

C-V model was implemented using finite differences equations as [33]:

$$\frac{\phi^{n+1}_{i,j}-\phi^{n}_{i,j}}{\Delta t}=\delta_h(\phi^n_{i,j})\left[\begin{array}{l}\frac{\mu}{h^2}\Delta^x_-\cdot\left(\frac{\Delta^x_+\phi^{n+1}_{i,j}}{\sqrt{(\Delta^x_+\phi^n_{i,j})^2/(h^2)+(\phi^n_{i,j+1}-\phi^n_{i,j-1})^2/(2h)^2}}\right)\\ +\frac{\mu}{h^2}\Delta^y_-\cdot\left(\frac{\Delta^y_+\phi^{n+1}_{i,j}}{\sqrt{(\phi^n_{i+1,j}-\phi^n_{i-1,j})^2/(2h)^2+(\Delta^y_+\phi^n_{i,j})^2/(h^2)}}\right)\\ -v-\lambda_1(u_{0,i,j}-c_1(\phi^n))^2+\lambda_2(u_{0,i,j}-c_2(\phi^n))^2\\ \end{array}\right] \qquad (3.2)$$

Equation 3.2 is the finite different scheme of Equation 3.1.

The C-V method is the minimization of an energy based segmentation. For example, an image denotes u_0 and the boundary denotes C_0.

The image u_0 can be divided by two regions: one that is inside the objects is denoted by u_0^i and the other outside the objects is denoted by u_0^o. Inside the objects there exists $u_0 \approx u_0^i$ (or $inside(C_0)$) and outside the objects there exists $u_0 \approx u_0^o$ (or $outside(C_0)$). The fitting function is as follows:

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

where C is the variable curve, and $c_{1,}c_2$ are the constants depending on C.

C_0 is the minimum of the fitting function as follow:

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

In the C-V model the fitting function is minimized and some more terms are added like the length of the curve and the area inside the C. The energy function $F(c_1,c_2,C)$ is defined as follows:

$$\begin{aligned}F(c_1,c_2,C)&=\mu.Length(C)+v.Area(inside(C))\\&+\lambda_1\int_{inside(C)}|u_0(x,y)-c_1|^2dxdy+\lambda_2\int_{outside(C)}|u_0(x,y)-c_2|^2dxdy,\end{aligned}$$

where $\mu\ge 0, v\ge 0, \lambda_1,\lambda_2>0$ are constants. In the later experiment, $\lambda_1=\lambda_2=1$ and $v=0$ are chosen. In this case, u is the approximate value of u_0,

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

And this particular minimization case can be handled with the level set method.

$$\begin{cases} C=\partial\omega=\{(x,y)\in\Omega:\phi(x,y)=0\}, \\ inside(C)=\omega=\{(x,y)\in\Omega:\phi(x,y)>0\}, \\ outside(C)=\omega=\{(x,y)\in\Omega:\phi(x,y)<0\},\end{cases}\quad \text{where}\quad \begin{matrix}C\subset\Omega,\\ \phi:\Omega\to R.\end{matrix}$$

In the level set method C is replaced by ϕ.

By using the Heaviside function H and Dirac function δ_0 as follows:

$H(z) = \begin{cases} 1, \ if\ z \geq 0, \\ 0, \ if\ z < 0, \end{cases}$ $\delta_0(z) = \frac{d}{dz}H(z)$, the terms in energy function $F(c_1, c_2, C)$ can be rewritten as follows:

$$Length\{\phi = 0\} = \int_\Omega |\nabla H(\phi(x,y))| dxdy = \int_\Omega \delta_0(\phi(x,y)) |\nabla \phi(x,y)| dxdy,$$

$$Area\{\phi \geq 0\} = \int_\Omega H(\phi(x,y)) dxdy,$$

$$\int_{\phi>0} |u_0(x,y) - c_1|^2 dxdy = \int_\Omega |u_0(x,y) - c_1|^2 H(\phi(x,y)) dxdy,$$

$$\int_{\phi<0} |u_0(x,y) - c_2|^2 dxdy = \int_\Omega |u_0(x,y) - c_2|^2 (1 - H(\phi(x,y))) dxdy.$$

$F(c_1, c_2, \phi)$ is as follows:

$$F(c_1, c_2, \phi) = \mu \int_\Omega \delta(\phi(x,y)) |\nabla \phi(x,y)| dxdy + v \int_\Omega H(\phi(x,y)) dxdy$$

$$+\lambda_1 \int_\Omega |u_0(x,y) - c_1|^2 H(\phi(x,y)) dxdy + \lambda_2 \int_\Omega |u_0(x,y) - c_2|^2 (1 - H(\phi(x,y))) dxdy.$$

The approximate value of u_0 is

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

3.2 Weakness of Current Segmentation Methods in DNA Microarrays

Segmentation is a partitioning process used to separate a spot area from a non-spot area. The spot area is called foreground and the non-spot area is called background. In [1], there are four types of segmentation methods: fixed circle segmentation, adaptive circle segmentation, adaptive shape segmentation and histogram segmentation.

The fixed circle segmentation is an ideal method; it assumes that each spot has the same size of circle shape. It is easy to implement this segmentation method but the disadvantages are that the real spots may not be the circular shape and may not have the same size. The fixed circle segmentation cannot provide the exact spot boundary and the accurate spot intensity most of the time.

The adaptive circle segmentation method is the evolution of the fixed circle segmentation method. It allows the diameter of each circle to be adjusted separately. It is better than the fixed circle, since it considers that spots may have different sizes. But the disadvantage of adaptive circle is that since the spots' shape could be different than circles. Adaptive circle method may figure out the similar size of the spot, but it cannot provide the accurate boundary of the spot since most spots do not have the regular circular shape. Therefore, it cannot give out the correct spots' intensities either.

The histogram segmentation method uses the normal distribution of the pixels' intensity percentiles around and inside each spot to segment the spot from the background. Obviously, this method neglects the particular pixels' locations. It will not give out the accurate spot intensity but only return the trend. Another problem of the histogram segmentation is that it is difficult to preset the threshold of the percentile. Therefore it cannot find the exact boundary of the spot.

The adaptive shape segmentation method is designed to improve the accuracy of the segmentation process. It tries to find out the real spot boundary which separates the spot with the background. The adaptive shape segmentation is better than the previous three segmentation methods. Seeded Region Growing (SRG) and Globally Optimal Geodesic Active Contours (GOGAC) are the two algorithms applied in the adaptive segmentation method. We note that when we check the segmentation result, we observe that even with these methods, some of the spots' boundaries are not accurately computed.

3.3 Weaknesses of Level Sets Method and Active Contours Method

The Active Contours Without Edges method is based on the active contours model and the level sets image segmentation method. ACWE is much better than the original active contours and level sets methods.

3.3.1 Original Level Sets Method Weaknesses

A level set is a set of function *g* with *n* variables $g(y_1,...y_n) = c$, where *c* is a constant. If n=2, $g(y_1,y_2)=c$ is a level curve in a 2 D surface. When c=0, function *g* is the zero level set. For a curve *C* in boundary Ω, there exists $C = \{(y_1, y_2) \in \Omega : g(y_1, y_2) = 0\}$. *C* is the zero level set of a 2D function *g*; *g* is a level set function. The value of g is positive inside the closed curve *C* and negative outside the curve *C*.

In the level sets method, when using the finite difference method to solve the evolved curve equation, there are several weaknesses:

1. Time step should be very small, otherwise the algorithm will be unstable.
2. Without curvature constrains, the evolving curve will find the boundaries which have singularities.
3. The evolving curve cannot handle moving boundaries; additional steps should be added for inspection and handle curve combinations and separation.
4. If the evolving curve needs to be applied to 3-D images, the functions need a lot of modifications.

3.3.2 Original Active Contours Method Weaknesses

The active contours method is focused on finding the boundary of an object, which is different from the background. The idea of this method is to initialize a curve in an image and let the curve be driven by the internal and external forces to the contours of the object.

In [33] an original Snakes model is discussed for a curve C; the energy out of the curve C is defined as external energy $E_{ext}(C) = \int_0^1 |\nabla u_0(C(s))|^2 ds$. The energy inside the curve is defined as internal energy $E_{\text{int}}(C) = \alpha \int_0^1 |C'(s)|^2\, ds + \beta \int_0^1 |C''(s)|^2\, ds$.

The total energy to minimize is as follows: $E(C) = E_{\text{int}}(C) + \lambda E_{ext}(C)$.

In the Snakes model, when the initial curve is far from the boundary of the object, the evolution curve will lead to the local minimal energy, and the boundary of the object cannot be detected correctly. A new force (balloon force) is introduced into the Snakes model. The modified Snakes model of minimum energy is as follows:

$$E(C) = \alpha \int_0^1 |C'(s)|^2\, ds + \beta \int_0^1 |C''(s)|^2\, ds + \lambda \int_0^1 |\nabla u_0(C(s))|^2 ds + v \iint_\Omega dxdy.$$

The modified Snakes model reduced the sensitivity of initial curve and noise of the image, but it is needed to adjust manually for the balloon force. The geodesic active contour (GAC) model is proposed to partly solve the disadvantages of the Snakes model. It chooses β=0, so

$$E(C) = E_{\text{int}}(C) + \lambda E_{ext}(C) = \alpha \int_0^1 |C'(s)|^2 ds + \lambda \int_0^1 w|\nabla u_0(C(s))|^2 ds,$$

where $w(0) = 1$ and $\lim_{x\to\infty} w(x) = 0$.

The purpose of the GAC model is to find the object edge, at which $w(x) = 0$. But the weakness of the GAC model is that in the real image somehow the object's edge does not always have $w(x) = 0$, the evolving curve will pass through the object's boundaries. The GOGAC method is based on the original GAC model. It inherits the same weakness which will not give out the more correct boundary of objects compared with ACWE segmentation method.

3.4 Advantages of Active Contours Without Edges Method

According to [33], ACWE segmentation method does not use the edge function to determine the curve evolving to stop at the boundaries. ACWE does not rely on the edge function and it can avoid the evolving curve passing through the object's boundaries' problem.

The initial curve of the ACWE method can locate at any position within an image. But in the original Active Contours method the initial curve need to be surrounded with the objects.

For the level sets method, the image needs to be smoothed if it is noisy. But the ACWE method can detect the boundaries of objects in a very noisy image.

Compared with the current segmentation methods in DNA microarray segmentation, the ACWE segmentation method has more advantages to use.

3.5 Active Contours Without Edges (ACWE) Method Application in DNA Microarrays Segmentation

This section will describe how the ACWE method can apply to DNA microarrays segmentation.

3.5.1 Adjustments for ACWE Method

In order to apply the ACWE method to DNA microarray segmentation, some adjustments must be done.

- First, we used the ACWE to segment each spot patch one at a time. Since the large DNA microarray may contain half a million spots, if we would apply the ACWE method to the whole image at once, it would not give out the correct segmentation result and it uses a lot of memory. Also, since the ACWE will segment all the spots as a whole region, it is very difficult to extract each spot intensity value if using the whole image for segmentation.
- Second, we use the grid file as input which gives the approximate spots locations. This will help to save some computation time, since some areas in the image will be neglect since there are no spot in these areas.
- Third, we decreased the number of iterations and made computing fast.
- Fourth, we adjusted the μ value and found more tiny spots.

3.5.2 The Databases for Experiments in Applying ACWE on DNA Microarrays

There are two databases used for the experiments in applying ACWE method on DNA microarrays.

3.5.2.1 The Databases for cDNA Microarrays

The database we used is from the Stanford University Yeast Cell Cycle Analysis Project [34]. This database provides the original two-channel 16-bit gray scale TIFF images and data files generated from these two-channel TIFF files. There are four experiments in the database as follows:

1. Cln/Clb Experiments
2. Pheromone Experiments
3. cdc15 Experiments
4. Elutriation Experiments

Another database we tested is from the Stanford Microarray Database (SMD). Its webpage is http://smd.stanford.edu. We used public login to get the original TIFF image files. There are a lot of different experiments in the Stanford database.

We used the TIFF images from these databases to segment using the ACWE method and other current cDNA microarray segmentation methods, especially the adaptive shape segmentation methods. The adaptive shape segmentation methods are the most accurate segmentation method we could get before we applied the ACWE method.

We compared the differences between different methods, and we found that ACWE would provide more accurate segmentation results.

3.5.2.2 The Databases for Affymetrix GeneChip

The Affymetrix segmentation method is considered so far to be the best method for Affymetrix GeneChip microarray. In all the Affymetrix Genechip file formats, the .DAT file is the scanned image file. The .CEL file is the cell file including the intensities and locations of all the probe cells. The .CEL intensities' values are calculated from .DAT file. Affymetrix also provides the .CDF file which is a library file that defines the probe set and probe pairs. The .CDF file contains the maps between features, probe pairs, probe sets and genes.

The database we used was from Harvard Medical School. These datasets belong to the CardioGenomics Programs for Genomic Applications. From [31] the Affymetrix GeneChip microarray images were from the mouse model, rat model and human model. The mouse models from Harvard Medical School have the following 10 experiments:

1. C57BL/6 Benchmark Set for Early Cardiac Development
2. Cardiac Hypertrophy Related to the Phosphoinositide 3-Kinase Signaling Pathway
3. Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor
4. Congenital Heart Disease in Csx/Nkx2.5 mutant embryos
5. Deletion of the Nk2 specific domain of the Nkx2.5
6. Exercise Induced Hypertrophy
7. FVB Benchmark Data Set and Sex Comparison
8. Myocardial Infarction
9. Overexpression of dn-p21ras as a model system for severe dilated cardiomyopathy
10. Pressure-overload induced Cardiac Hypertrophy.

The rat model from Harvard Medical School has one experiment:

Hypertrophy and Heart Failure Through High Salt Diet and Exercise.

The human models from Harvard Medical School have the following eight experiments:

1. Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function
2. Idiopathic Cardiomyopathy
3. Ischemic Cardiomyopathy
4. Non-failing "Normal" Patient
5. Familial Cardiomyopathy Dataset
6. Hypertrophic Cardiomyopathy
7. Post-Partum Cardiomyopathy
8. Viral Cardiomyopathy

According to [31], the rat model accomplished two objectives:

It found the genes related to physiologic hypertrophy caused by exercise, which would not cause heart failure, and found the genes related to pathologic hypertrophy, caused by a high sodium diet, which would cause heart failure. The mouse model gave out the genes related to physiology hypertrophy, pathologic hypertrophy and heart failure. It cataloged different types of subsets for different causes for hypertrophy.

The human model provided genes related to different types of heart diseases. It also provided genes with normal (healthy) heart tissues for comparison.

3.5.3 The Process of Applying ACWE for cDNA Microarrays

The cDNA spotted arrays are two-channel microarrays. The microarray is hybridized with cDNA from two channels: Cy3 and Cy5. After the scanning process the microarray generates two 16 bit gray scale TIFF images for the above two channels. The process required to generate ACWE segmentation for cDNA microarrays is as follows:

1. Download two 16-bit gray scale TIFF images from online database. There are many free online cDNA image databases which provide images for analysis. The major database we use for experimenting is [34], which is a website of Stanford University Yeast Cell Cycle Analysis Project. This website provides the original two-channels TIFF files and a data file generated from these two TIFF files. We use these original files for our experiment and analysis. The other database we test with the ACWE method is from Stanford Microarray Database(SMD, http://smd.stanford.edu/). In SMD, use public login to get the original two-channel TIFF image files.

2. Generate a grid file. The data files in the database provide the grid information and this information needs to be extracted to create a grid file.

3. Create a batch file. This batch file includes all the TIFF images and corresponding grid files. The batch file is for handling all the image files automatically.

4. Apply the ACWE image segmentation. The ACWE segmentation algorithm is implemented in JAVA under Windows XP, the codes have been transformed to be used under Linux and Unix. It has been working on Louisiana Optical Network Initiative (LONI) computers under Linux and Unix operating systems. Before segmenting, the users need to choose the batch file name for the input.

5. Create an output file. The output files include two types. One is a text file which includes the foreground and background intensities of each spot. When computing the intensity values, mean and median intensity values of all pixels in each spot are computed. The other type is an image file which gives out the segmentation result (the boundary of each spot).

To compare the ACWE segmentation results with those of other cDNA microarray segmentation method, these software programs are needed for computing the same image files used in ACWE segmentation.

ScanAlyze - ScanAlyze was developed by Eisen in 1999. We use it to compute the spot intensity with the fixed circle segmentation method.

GenePix - GenePix was developed by Axon Instruments Inc. in 1999. It is used for computing the spot intensity with the adaptive circle segmentation method.

SpotSegmentation - SpotSegmentation was developed by Q.H. Li and C. Fraley in 2005. This software was used to figure out the spot intensity using the modified histogram segmentation method.

Spot - Spot was developed by Y.H. Yang, M.J. Buckley and T. P. Speed in 2002. In this software, the adaptive shape segmentation method was implemented. It provides two algorithms. One is the Seeded Region Growing (SRG) and the other is the Globally Optimal Geodesic Active Contours (GOGAC).

After using these software programs on the same images used by ACWE, a text file is created for each cDNA microarray image with the spots intensity values gotten from different method. MATLAB codes can be applied on the text file for comparison of the different segmentation methods results. For example, gene expression level (log ratio of two cDNA channels) can be computed to show which segmentation method is best.

3.5.4 The Process of Applying ACWE for Affymetrix GeneChip

Affymetrix GeneChip is different from cDNA spotted microarray. After the scanning process the microarray will generate only one 16 bit gray scale TIFF image. The process to generate ACWE segmentation for Affymetrix GeneChip is as follows:

1. Download the Affymetrix GeneChip image and data files. The database we use is from [31] Harvard Medical School CardioGenomics Programs for Application. These microarray images belong to 3 models: mouse model, rat model and human model. This database provides Affymatrix GeneChip image (.DAT) files and cell information (.CEL) files. Affymetrix Inc. provides some example images and cell information files on its own webpage. That is another database we used in the experiments.

2. Convert a .DAT file to a .TIFF file. Since .DAT file is created by Affymetrix Inc., it can only be read using Affymetrix authorized software. For example, if using JAVA, it cannot open the .DAT file. So some preprocessing must be done. MATLAB which is a software developed by MathWorks provides a Bioinformatics Toolbox. Using the Bioinformatics Toolbox, a .DAT image file can be converted to a 16-bit gray scale .TIFF image file which can be read by JAVA or other image processing software.

3. Generate a grid file. Grid information can be extracted from .DAT and .CEL files to generate a grid file.

4. Create a batch file. A batch file needs to be created for automatically segmenting all of the Affymetrix GeneChip image files. The batch file includes the converted .TIFF image file and the grid file for each Affymetrix GeneChip microarray.

5. Apply the ACWE segmentation. The segmentation program takes the batch file as input and automatically segments each probe cell in all the Affymetrix GeneChip microarray image files.

6. Create an output file. A text file with an intensity value of each probe cell and an image file for the segmentation result are generated for each Affymetrix GeneChip microarray.

To compare the ACWE segmentation results with that of Affymetrix GeneChip microarray segmentation method, these steps are needed :

Step 1. Import both ACWE segmentation results and Affymetrix GeneChip results (from .CEL file) to Microsoft Access database.

Step 2. Use MATLAB to read out the probe cell name, probe cell location and probe cell type from the .CDF file and output to a text file.

Step 3. Import the output text file in Step 2 into Microsoft Access database.

Step 4. Link both probe cell intensity values from ACWE method and Affymetrix method with the probe cell name, location and type from the output text file.

Step 5. Create an output file lists for the fields as follows:

Affy (probe cell intensity value from Affymetrix segmentation method)

ACWE (probe cell intensity value from ACWE segmentation method)

Gene Name (the gene name of the probe cell)

X (probe cell location in X-axis)

Y (probe cell location in Y-axis)

Probe Type (the probe type of probe cell, 'PM' or 'MM')

Step 6. Import the output file in Step 5 and search out all the records related to the control genes.

Step 7. Compute both the Average Differences (AD) of the Affymetrix Segmentation method and the ACWE method with the control genes.

Step 8. Compute both the Sum Square Error (SSE) of the concentration values of the Affymetrix segmentation method and ACWE segmentation method with the control genes.

Step 9. Compare the two SSE, and choose the smaller SSE to be the best segmentation result. After all these steps, we can find out which method is better.

3.6 Improvements for Active Contours Without Edge (ACWE) Method Application in DNA Microarrays Segmentation

3.6.1 The Improvement by Reducing the Length Constraint in ACWE

In order to reduce the segmentation time, we also test the possibility of a term reduction in the Chan-Vese model. We simplified the Equation 3.1 to

$$\frac{\partial \phi}{\partial t} = \delta_\varepsilon(\phi)\left[\lambda_1(u_0 - c_1)^2 + \lambda_2(u_0 - c_2)^2\right] = 0 \tag{3.3}$$

we also chose to use $\delta_\varepsilon(\phi) = 1$ for simplicity. The finite differences Equation 3.2 in [33] was simplified to

$$\begin{aligned}\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^{n}}{\Delta t} &= \left[-\lambda_1(u_{0,i,j} - c_1(\phi^n))^2 + \lambda_2(u_{0,i,j} - c_2(\phi^n))^2\right] \\ &= 2(c_1(\phi^n) - c_2(\phi^n))(u_{0,i,j} - (c_1(\phi^n) + c_2(\phi^n))/2).\end{aligned} \tag{3.4}$$

The length term in the Chan–Vese model is for the smoothing of the curve C; during the test we neglect the length term and this may cause a reduction in the accuracy of the boundary, but it will reduce the computational time because of the drop of finite difference terms in the equation. We use S-ACWE to represent this method.

3.6.2 The Improvement by Using a Fast Algorithm in ACWE

In [35], a fast algorithm was proposed to increase the segmentation speed of ACWE method. We implemented this fast algorithm and named it as the F-ACWE method.

The algorithm for F-ACWE is described as follows:

1 . Use an initial curve to partition the image into two parts. In one part $\phi=1$, in the other part $\phi=-1$.

2 . For a pixel x with intensity value y, c_1 is the average intensity value of the pixels with $\phi=1$; c_2 is the average intensity value of the pixels with $\phi=-1$; m is the number of the pixels with $\phi=1$; and n is the number of the pixels with $\phi=-1$.

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

$$\Delta F_{12} = (y-c_2)^2 \frac{n}{n+1} - (y-c_1)^2 \frac{m}{m-1}. \tag{3.5}$$

   If $\Delta F_{12} < 0$, then $\phi=-1$.

   If $\phi=-1$, then compute

$$\Delta F_{21} = (y-c_1)^2 \frac{m}{m+1} - (y-c_2)^2 \frac{n}{n-1}. \tag{3.6}$$

   If $\Delta F_{21} < 0$, then $\phi(x)=1$.

   If the length term is considered, the change of the length can be added as a term to the right hand side of ΔF_{12} and ΔF_{21}.

3 . Repeat step 2 until $F(c_1, c_2, \phi)$ remains stable.

The main purpose of this algorithm is directly computing the energy. If a pixel changed from inside the curve region to outside the curve region or the opposite, the difference of the energy of both regions would be computed. Since the ACWE method supposed that the energy inside the curve was positive and the energy outside was negative, if the pixel was inside the region and the difference was negative, then update the pixel into the outside region. It was similar for a pixel changing position from the outside to the inside region. Each pixel in the image will be swept and the iteration number for sweeping will terminate until the energy remains stable.

3.6.3 The Improvements by Providing a More Accurate Segmentation Method Than the ACWE Method

According to [36], when we proposed a method using high order finite difference scheme, it will provide a more accurate segmentation result than the ACWE method. We proposed and implemented the improved segmentation method called the I-ACWE method.

In the ACWE method, the central finite difference schemes were as follows:

$$\frac{\phi^n_{i+1,j} - \phi^n_{i-1,j}}{2h}, \tag{3.7}$$

and

$$\frac{\phi^n_{i,j+1} - \phi^n_{i,j-1}}{2h}. \tag{3.8}$$

In I-ACWE method, the central finite difference schemes would be changed from Formula 3.7 to

$$\frac{8(\phi^n_{i+1,j} - \phi^n_{i-1,j}) - (\phi^n_{i+2,j} - \phi^n_{i-2,j})}{12h}, \tag{3.9}$$

and Formula 3.8 to

$$\frac{8(\phi^n_{i,j+1} - \phi^n_{i,j-1}) - (\phi^n_{i,j+2} - \phi^n_{i,j-2})}{12h}. \tag{3.10}$$

Based on the improvement on the central finite difference schemes, we improved the forward finite difference schemes from

$$\frac{\phi_{i+1,j}^n - \phi_{i,j}^n}{h}, \tag{3.11}$$

to

$$\frac{-\phi_{i+2,j}^n + 4\phi_{i+1,j}^n - 3\phi_{i,j}^n}{2h}, \tag{3.12}$$

and from

$$\frac{\phi_{i,j+1}^n - \phi_{i,j}^n}{h}, \tag{3.13}$$

to

$$\frac{-\phi_{i,j+2}^n + 4\phi_{i,j+1}^n - 3\phi_{i,j}^n}{2h}. \tag{3.14}$$

We also improved the backward finite difference schemes from

$$\frac{\phi_{i,j}^n - \phi_{i-1,j}^n}{h}, \tag{3.15}$$

to

$$\frac{\phi_{i-2,j}^n - 4\phi_{i-1,j}^n + 3\phi_{i,j}^n}{2h}, \tag{3.16}$$

and from

$$\frac{\phi_{i,j}^n - \phi_{i,j-1}^n}{h}, \tag{3.17}$$

to

$$\frac{\phi_{i,j-2}^n - 4\phi_{i,j-1}^n + 3\phi_{i,j}^n}{2h}. \tag{3.18}$$

We called this improved segmentation method the I2-ACWE method. The spot patch size of DNA microarray is very small. Higher order schemes larger than the fourth order in central schemes will have the problem of the mesh size exceeding the spot patch boundary. The same problem happens to the forward and backward schemes when the order is higher than the second order. In the implementation of I2-ACWE method, we choose the fourth order for central schemes and the second order for forward and backward schemes.

3.6.4 Hybrid Methods for ACWE

In order to improve the accuracy of the segmentation methods, we also proposed a hybrid method for the Affymetrix DNA microarray image segmentation. We combined the ACWE method with the Affymetirx segmentation method.

We calculated the sum of square error (SSE) for Minimum, Average and Maximum of the two values of concentration obtained by the ACWE and Affymetrix methods. We observed that the Maximum gives a smaller sum of square error for the control genes, therefore a better fit.

When the Affymetrix segmentation method is a better fit (smaller SSE) than the ACWE method, we compare the Affymetrix method with the Maxium.

In Chapter 5, the experimental results will present that hybrid method has a better fit; therefore, it is a better segmentation method than the Affymetrix method.

CHAPTER 4

EXPERIMENTAL RESULTS OF DNA MICROARRAYS SEGMENTATION BASED ON THE ACWE METHOD

The ACWE method uses the C-V model. Normally, the ACWE can partition an image into several parts, but when applying it in the microarray, it needs to be partitioned into exactly two parts: intensity and background.

The C-V model has some limitations. For example, it cannot detect the texture of the image when the average intensity inside the object's boundary is the same as the average intensity outside the boundary. These limitations have no effect on applying the C-V model to the DNA microarrays' images segmentation since the intensities of spots and background are different in the microarray. We use the C-V method to compute the exact boundary of the spot. We can get the exact location of each spot before hand by using the grid file from the database. The grid file gives the location of each spot that wants to be printed on the microarray during the printing process. We use the spot patch which is the location (rectangle or square) of each spot as a sub image. Since the patch outside of every spot is the same, we only need to set up the initial function and other parameters once for PDE equations. After solving the equations we will get the exact boundary of the spot and the intensity value of that spot. To compute the background intensity value, we use the local background. The background area lies outside the boundary of the spot and inside the boundary of the spot patch. Figure 4.1 is the output of the ACWE for cDNA microarray. Figure 4.2 is the output of ACWE for Affymetrix GeneChip.

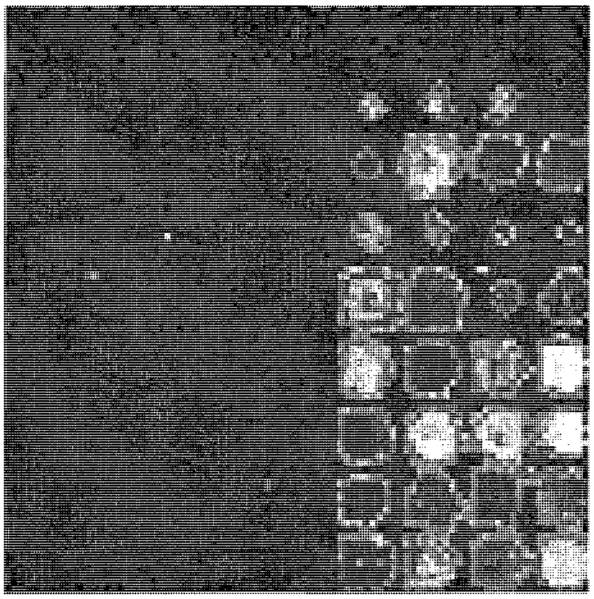

Figure 4.1 ACWE segmentation for the cDNA microarray

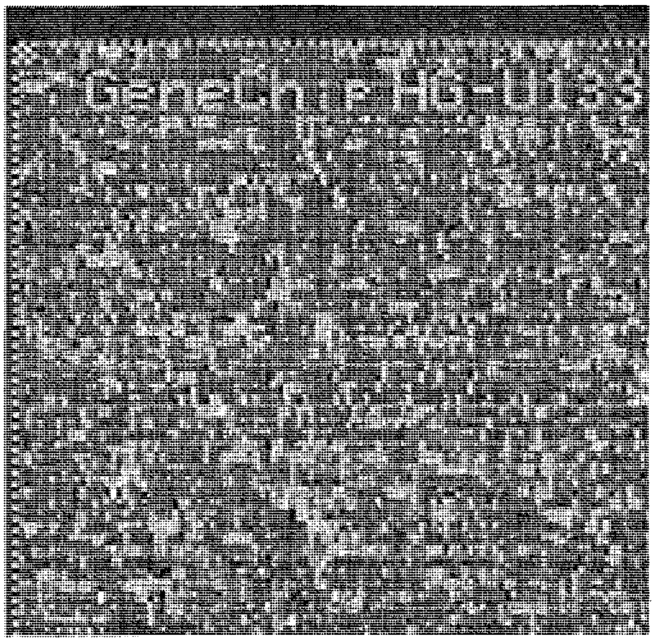

Figure 4.2 ACWE segmentation for the Affymetrix GeneChip

In the segmentation of the DNA microarray, some parameters are set as follows: $\lambda_1 = \lambda_2 = 1$, $v = 0$, $h = 1$, $\Delta t = 0.1$. There are two types of microarrays which are chosen for the ACWE segmentation. The other parameters need adjusting based on the spot size of these two different microarrays. For example, since the Affymetrix GeneChip spots are smaller than those of cDNA microarray, smaller μ, ϕ_0 are chosen for the Affymetrix GeneChip spots. Once these parameters are set in the program, there is no need to adjust the parameters during computing, since the spot's size for an image are the same based on the grid file.

4.1 Experimental Result for cDNA Microarray

We use real cDNA microarray images from [34] for segmentation. The experiment results show that ACWE is the best accurate segmentation method. We will argue this point through this section. Figure 4.3 represents an original 16-bit Tagged Image File Format (TIFF) file of a cDNA microarray image.

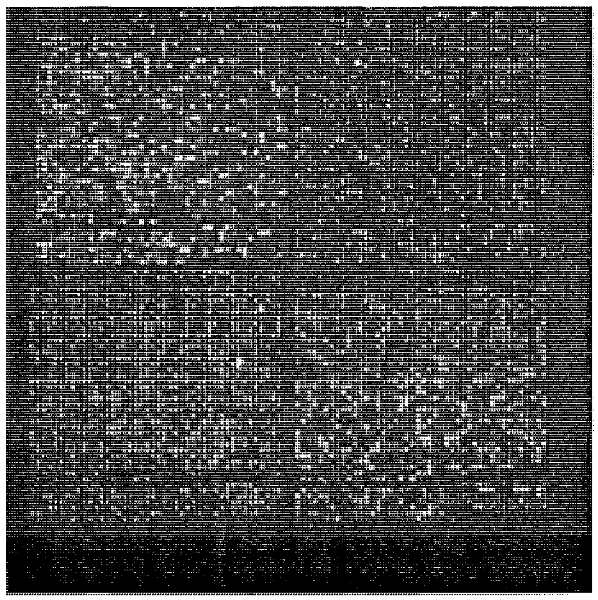

Figure 4.3 Original cDNA .TIFF file

In Figure 4.4 we present a segmentation result image using the ACWE segmentation method. The red curves around each spot represent the boundaries of the spots. The boundary of each spot in Figure 4.4 using the ACWE method gives nearly the actual boundary which can be checked by visual inspection.

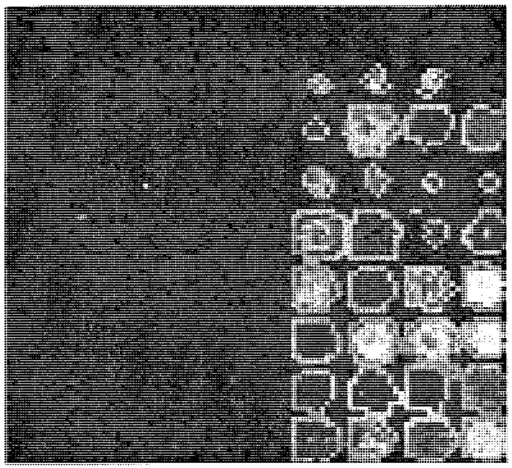

Figure 4.4 Image after segmentation using ACWE.

Figure 4.5 presents the segmentation result image obtained by using the GOGAC segmentation method. The red curve around each spot is the boundary of the spot.

Spot 1 is at the top left corner with a red line boundary.

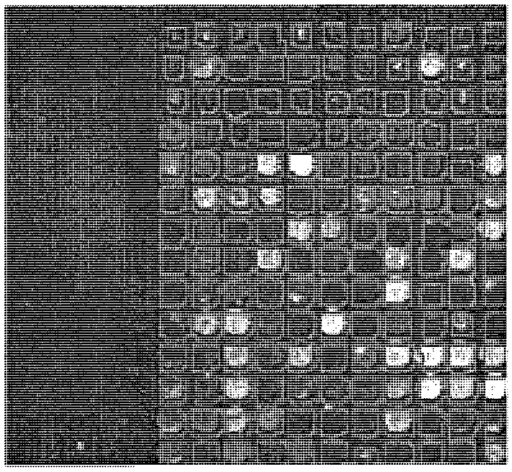

Figure 4.5 Segmentation using GOGAC.

Figure 4.6 presents the segmentation result image using the SRG segmentation method. The red curve around each spot is the boundary of the spot. Spot 1 is at top left corner with a red line boundary.

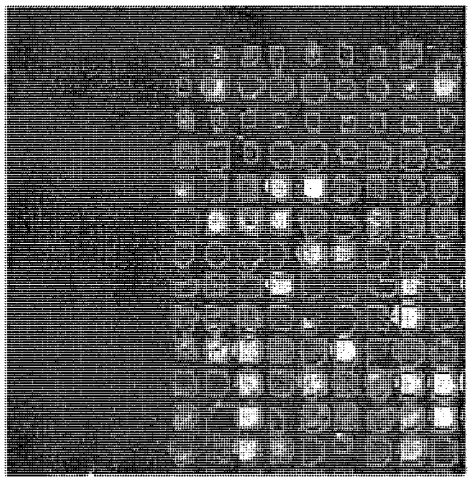
Figure 4.6 Segmentation using SRG

Comparing Figure 4.4 with Figures 4.5 and 4.6 we observe that the ACWE method gives the more accurate boundary than the previous two methods mentioned. In Figure 4.7, a segmentation image was provided by using ACWE for only one spot (Spot 1). In Figure 4.8, we provide a segmentation image using the spotSegmentation for only one spot (Spot 1). The spotSegmentation is based on histogram segmentation. Comparing Figure 4.7 and Figure 4.8, we visually observe that the ACWE method gives out a more accurate boundary than the spotSegmentation method. These two methods were segmenting the same spot from the same original image. The same spot (Spot 1) can also be found in Figure 4.5 and 4.6 at the top left corner of the images.

For more detail, we focus on one spot to analyze. Spot 1 (shown in Figure 4.7 and Figure 4.8) was chosen for analyzing the accuracy of ACWE compared to other segmentation methods.

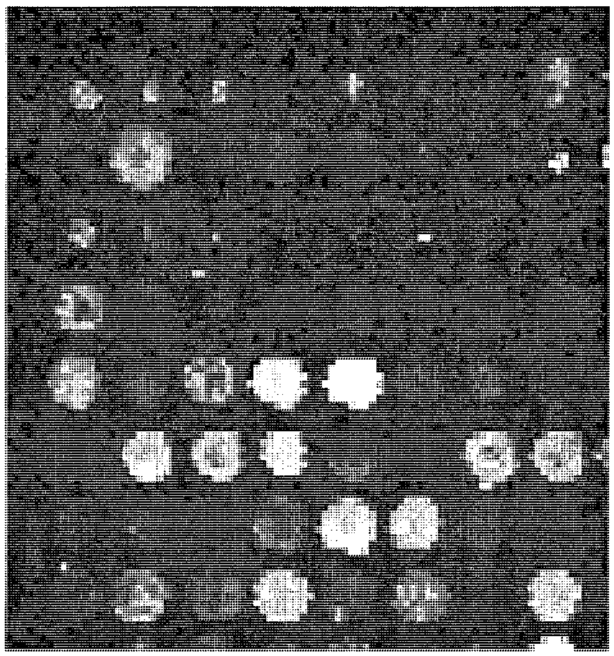

Figure 4.7 Segmentation for Spot 1 using ACWE

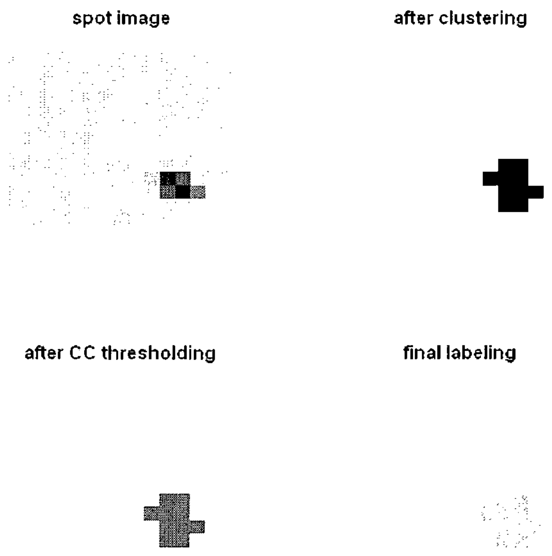

Figure 4.8 Segmentation using spotSegmentation for Spot 1

Table 4.1 shows the pixels' intensities for Spot 1. The 10 columns by 10 rows square is the patch for Spot 1. After segmentation using ACWE, the area with 5 pixels (yellow part) is the exact as Spot 1. The red figures are the boundary which separates the

Spot 1 with the background. Based on Table 7.1, the mean intensity value for Spot 1 was calculated: (2576+1648+1640+2456+1896)/5=2043.

Table 4.1 Patch outside Spot 1 with intensity values. Yellow area is the spot area. Red figures are the boundary found by ACWE.

296	336	280	272	368	200	272	256	256	288
336	232	280	248	272	216	344	248	344	360
248	256	288	312	240	304	272	296	264	248
296	304	264	312	304	224	272	344	216	336
336	320	296	336	720	464	296	336	272	248
336	320	336	520	2576	1648	384	280	344	304
288	296	328	368	1640	2456	1896	272	328	296
272	288	280	336	472	400	296	240	312	272
256	392	264	248	336	272	248	288	288	320
304	280	240	288	216	272	336	296	184	288

The actual mean intensity value of Spot 1 was computed by different methods and the result is as follows in Table 4.2:

Table 4.2 Spot 1 mean intensity value using different methods and software.

SPOT	ACWE	SRG	GOGAC	FIXED-Circle	spotSegmentation	spotfinder	imagene
1	2043	1352	784	414	1279	304	306

This shows that the ACWE gave more precise mean intensity values than the rest of the methods and software.

In Figure 4.9, the 3D visualization picture of Table 4.1 was presented. The intensity value of each pixel was used as Z-axis. The 5 pixels with extreme high intensities were partitioned as the spot's foreground.

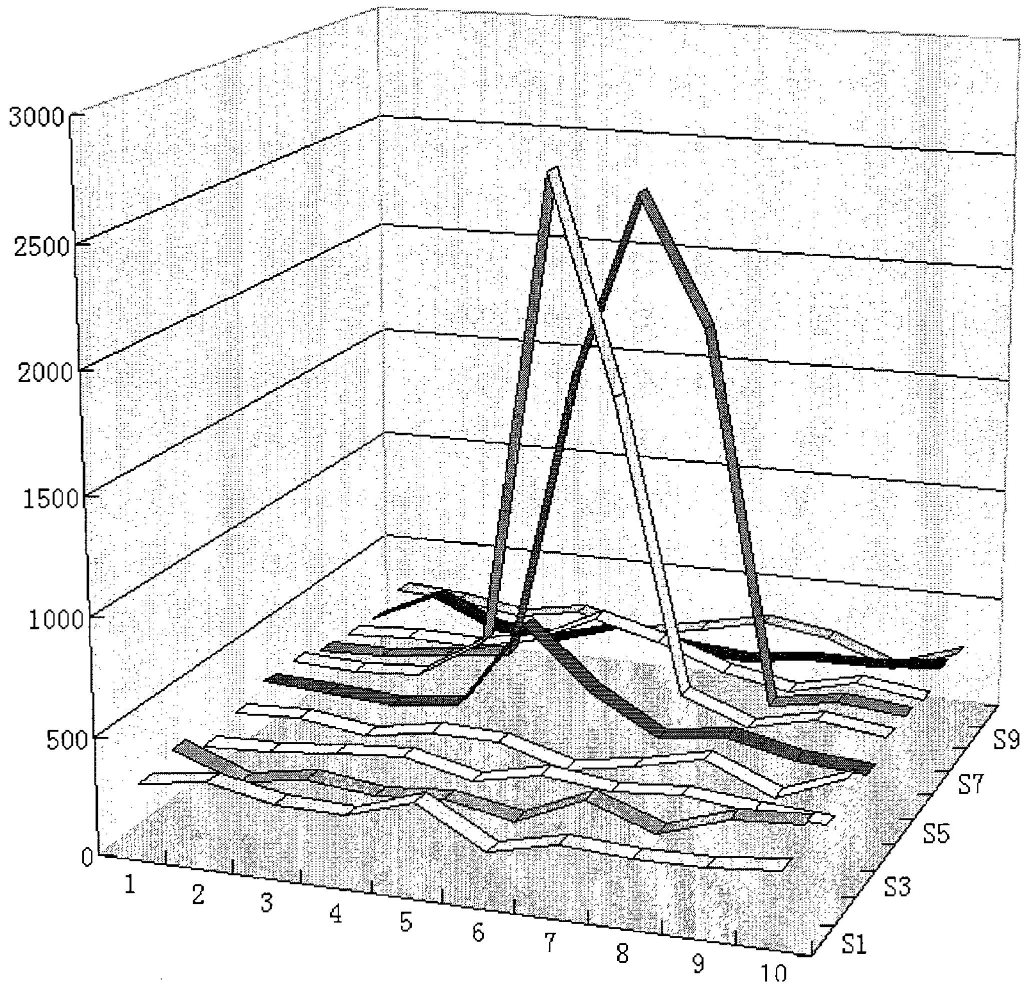

Figure 4.9 3D graphic for square outside Spot 1 with intensity value. Z-axis is the intensity value. Five pixels with extreme high intensity are partitioned to the same spot foreground.

In what follows we will perform a linear regression analysis in order to investigate the relation between the ACWE method and the SRG, GOGAC and Fixed circle methods. The same microarray image was segmented using these different methods, the spots' intensities were calculated using each of the methods. Then the linear model was determined for each case. Figure 4.10 presents the linear regression line and the model equation that explains the relation between the ACWE and SRG methods. We observe that R^2=0.9714.

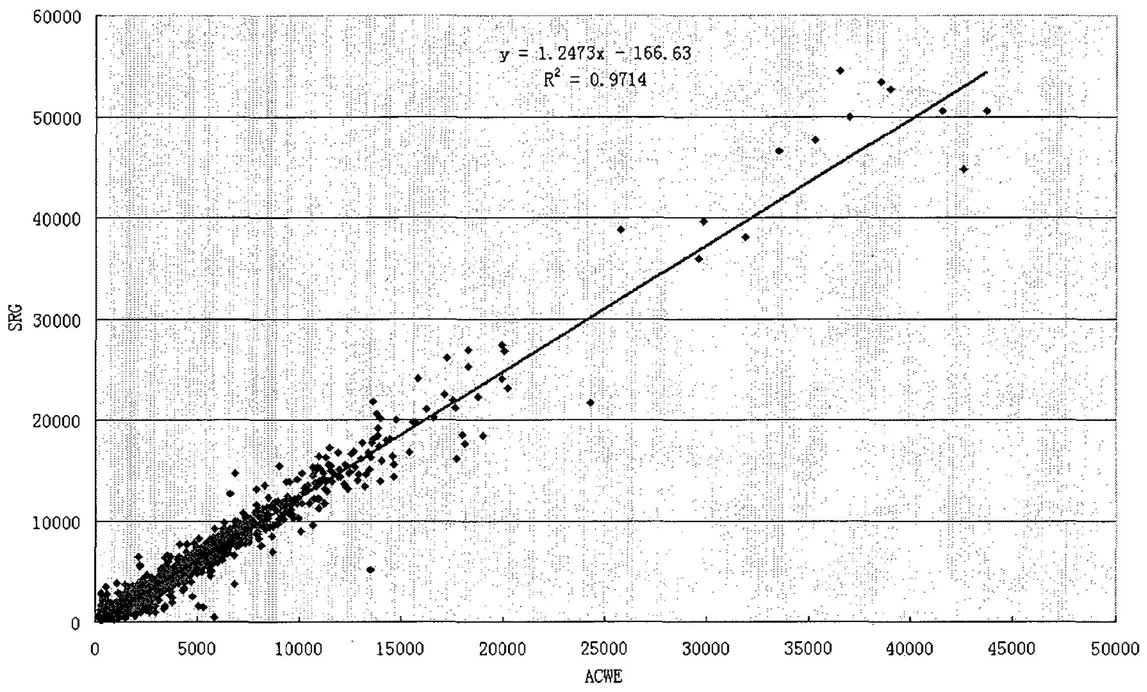

Figure 4.10 The linear regression between the SRG and ACWE methods.

Figure 4.11 presents the linear regression line and the model equation that explains the relation between the ACWE and GOGAC methods. We observe that R^2=0.9662. Similarly, Figure 4.12 presents the linear regression line and the model equation that explains the relation between the ACWE and the Fixed circle methods. We observe that R^2=0.6494.

Therefore, observing the regression lines in Figures 4.10-4.12 we conclude that ACWE was highly correlated with SRG and GOGAC and least correlated with the Fixed circle methods. Method ACWE is the most accurate method in determining the boundary of the spots.

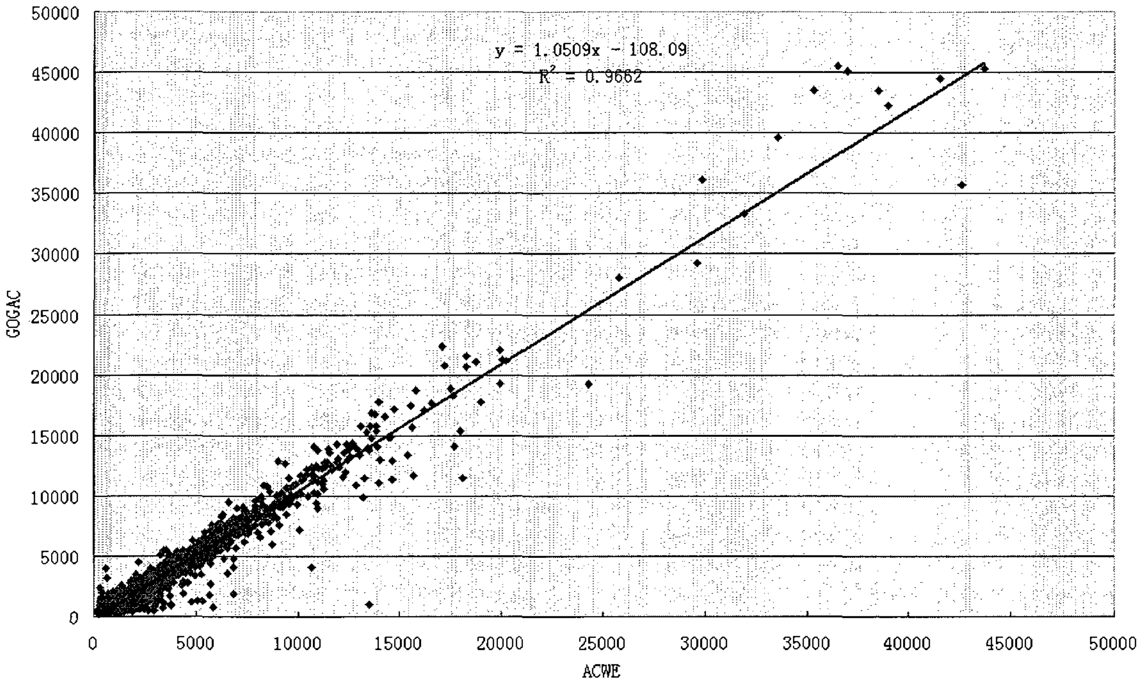

Figure 4.11 The linear regression between GOGAC and ACWE methods.

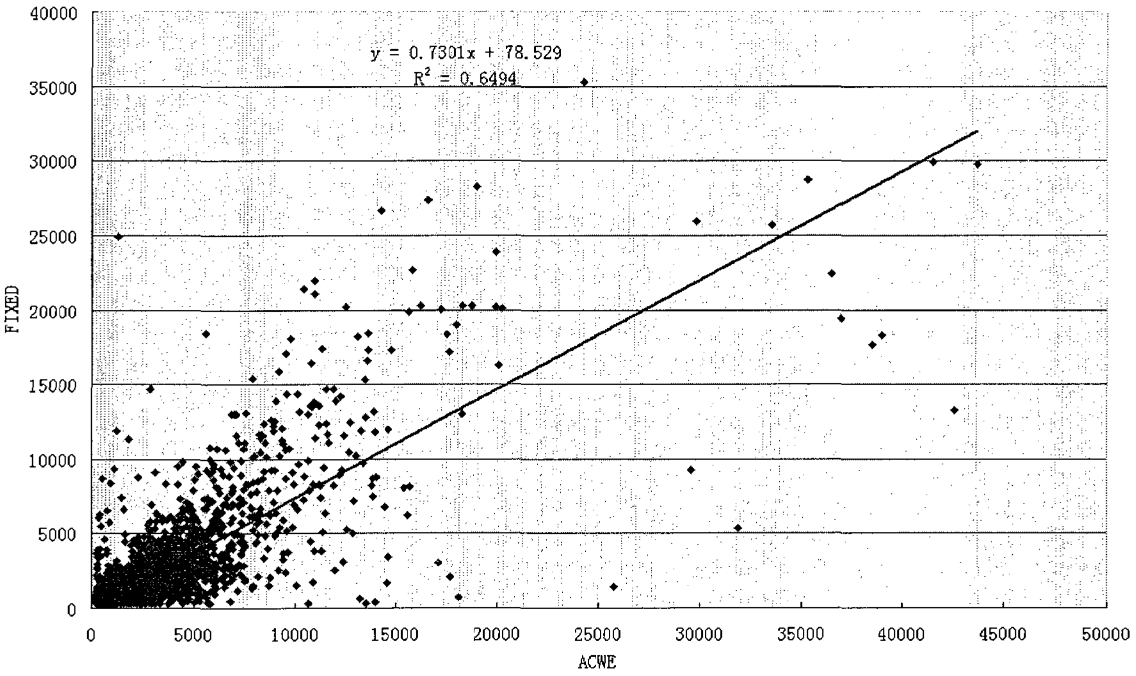

Figure 4.12 The linear regression between FIXED circle and ACWE methods.

Figure 4.13 presents the comparison between control spots and spots with the same gene. LAMBDA was a control spot and YBR145W was a gene spot (with gene named "ADH5"). For the two gene spots (Spot 2055 and Spot 3687), ACWE has less differences between them. For the two control spots (Spot 1941 and Spot 5813), the ACWE method showed more differences between the two controls.

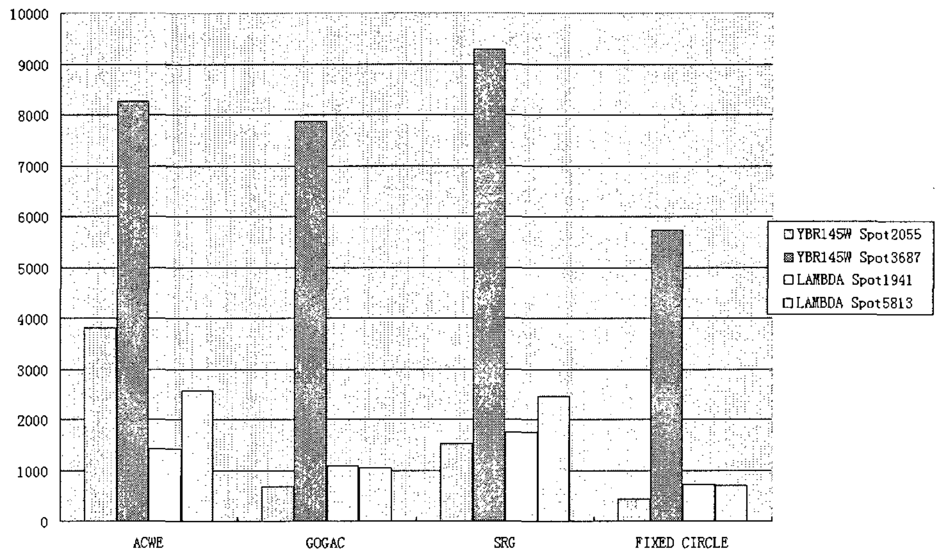

Figure 4.13 LAMBDA is a control spot and YBR145W is a gene spot. The trend is the result of intensity of the spots using different methods.

Figure 4.14 presents a boxplot for control spots named LAMBDA and the gene spots named YBR145W. It was based on different segmentation methods and software. It compared the inter-quantile range (IQR, which is the distance between 25 percentile and 75 percentile) of normalized intensity values of spots (using z-score). IQR used the middle 50% data and was not affected by the outliers. In [37], the z-score normalization rule is defined as follows: $Z_{gi} = \frac{X_{gi} - \overline{X}_g}{S_g}$, Z_{gi} is a z-score, $\overline{X}_g$ is the mean of the gth observation, and s_g is the standard deviation of the gth observation.

The result showed that the ACWE method had the smallest inter-quantile range and it was the best segmentation method for cDNA microarray segmentation.

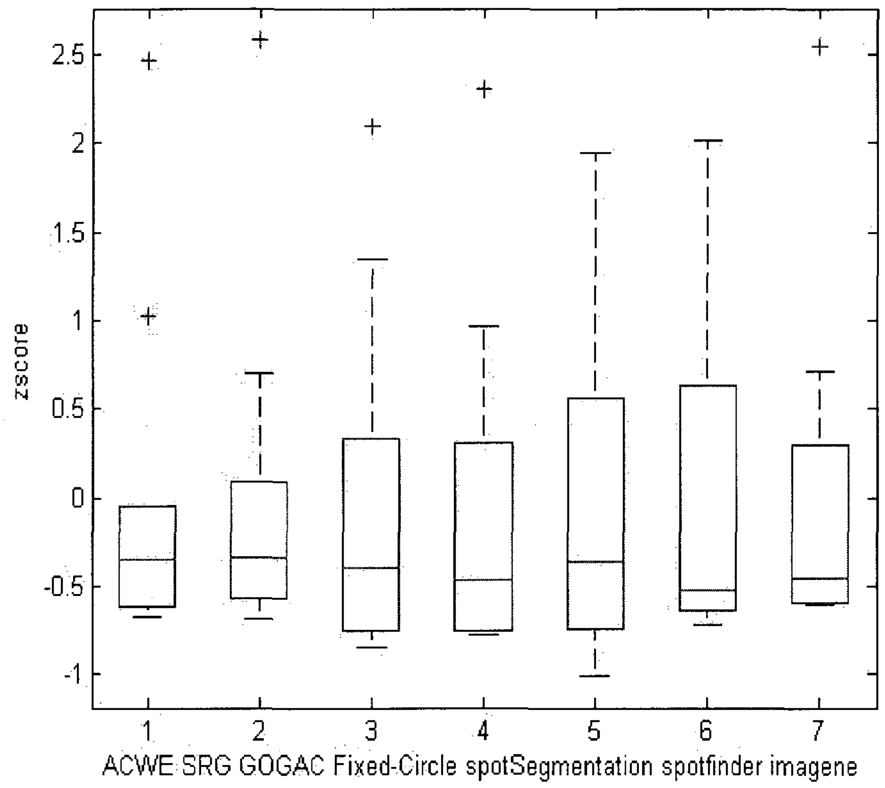

Figure 4.14 Boxplot for z-score normalization for spots in Figure 4.13.

The gth spot pixel intensity of Cy3 channel (Green) is denoted as X_g. Y_g denotes the gth spot pixel intensity of Cy5 channel (Red). The ratio R_g is equal to Y_g / X_g. This ratio represents the DNA folder change. Logarithms of this ratio is $\log_2 R_g$. Using logarithmic transformation reduces the skewness of the distribution and improves the variance estimation. In Figure 4.15 we present the box plot of the log ratio on the spots named "LAMBDA". The inter-quantile range (IQR) presented method ACWE is smaller than the ones provided by the SRG and GOGAC methods, since it showed less range which meant less gene expression difference for the control spot.

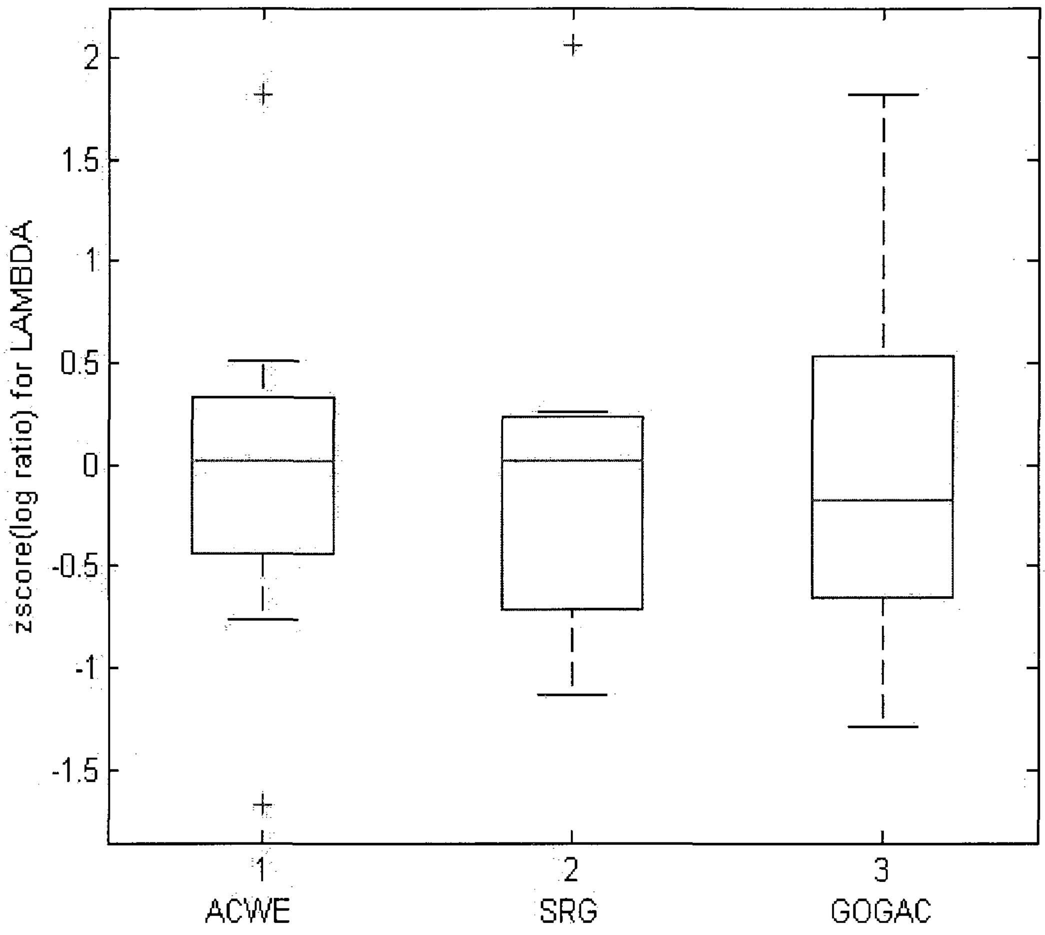

Figure 4.15 Boxplot of z-score normalization of log ratio using different methods.

In [38], the author proposed a more precise relative difference (RelDiff) calculation for cDNA microarray analysis. The relative difference was more stable than log ratio.

Relative difference is defined as follows: RelDiff=$2\frac{R-G}{R+G}$, where R is the Cy5 (red) intensity with background adjustment and G is the Cy3 (Green) intensity with background adjustment.

Figure 4.16 is the box plot for relative difference on spots named "LAMBDA". The inter-quantile range (IQR) showed that the ACWE method was the best.

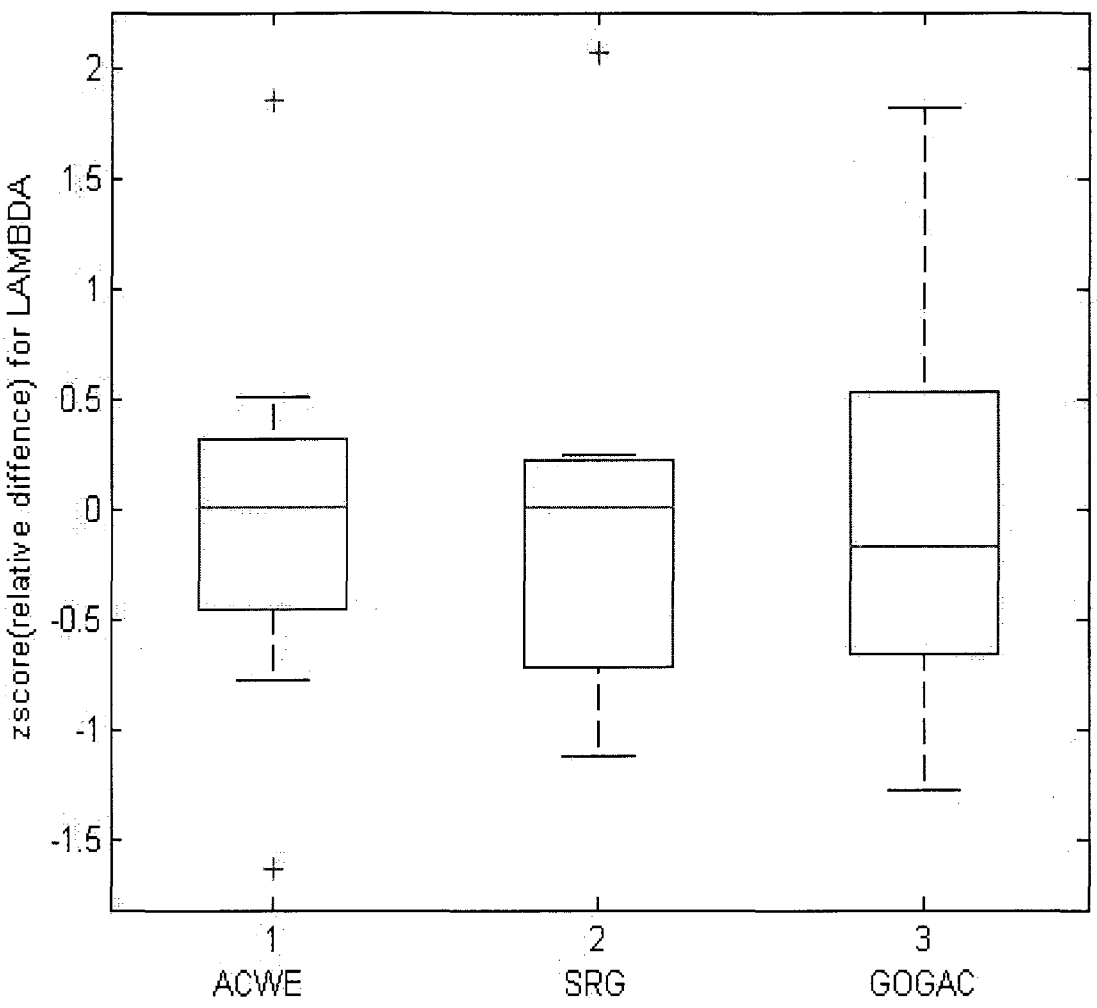

Figure 4.16 Boxplot of the z-score normalization of relative difference using different methods.

Each pair of microarrays corresponds to a single mRNA sample. The two microarrays in each pair are technical replicates as they are exposed to the same biological samples. Microarrays from different mRNA samples are biological replicates. In the experiments, we used 3 biological replicate samples, each of which has two technical replicates from two channels. The control spot named "LAMBDA" is chosen for the test. Figure 4.17 showed the "LAMBDA" control spot that expressed less difference should have a smaller inter-quantile range (IQR). There should be less difference between replicates. The figure showed that ACWE was a better method than the other two.

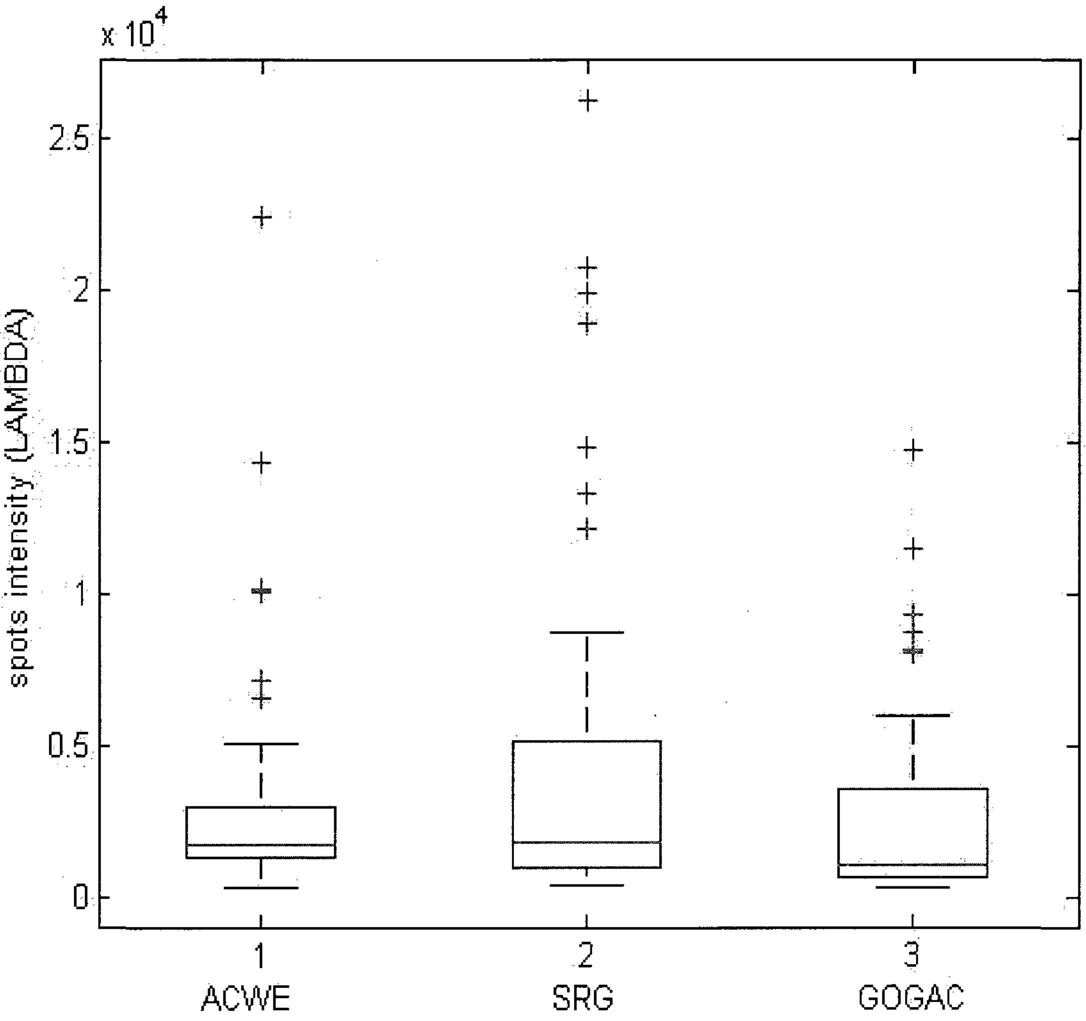

Figure 4.17 Boxplot of spot intensity of LAMBDA with different segmentation methods.

The log ratio (*R/G*) of the LAMBDA control spots was computed for each array in a serial of 18 microarrays of a yeast experiment. The result was shown in Figure 4.18. Since LAMBDA was a control spot with less expression difference, the range of log ratio (*R/G*) is less than 2 using ACWE segmentation method. This showed the computed result had the biological meanings.

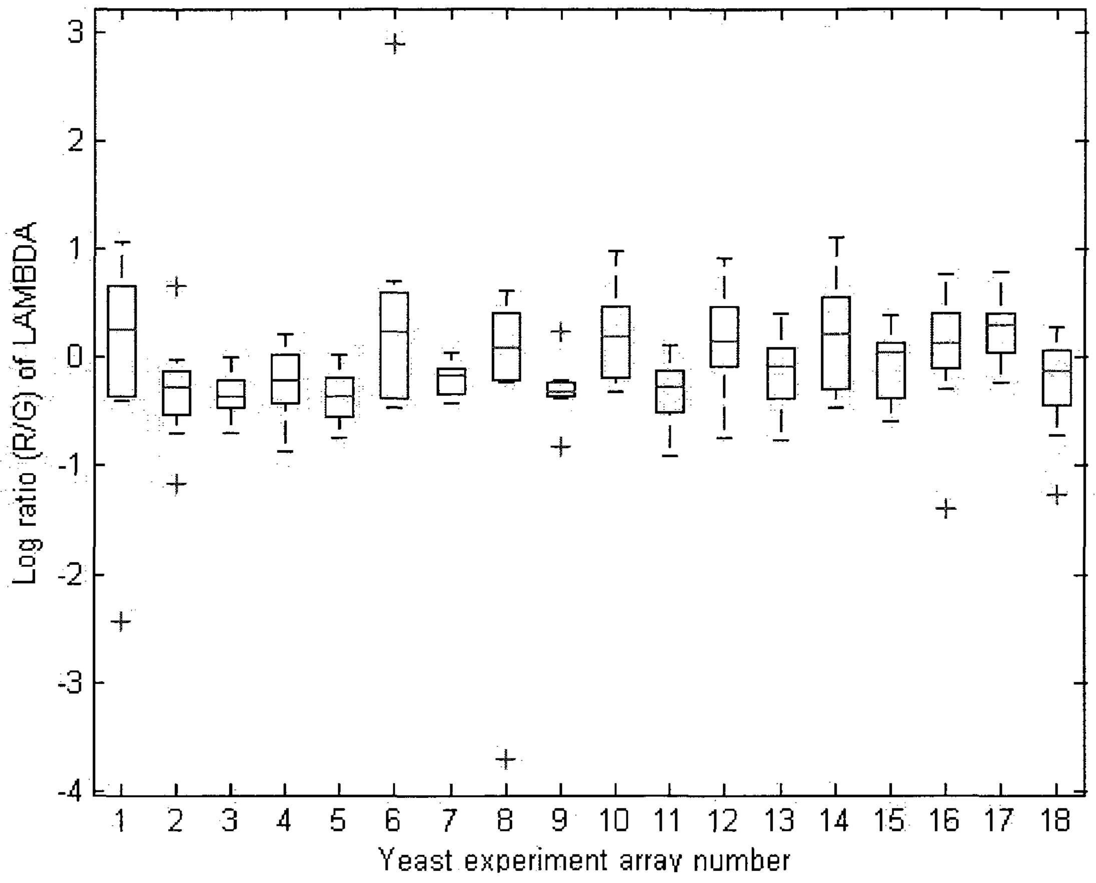

Figure 4.18 Log ratio (*R/G*) of LABMDA of 18 microarrays using ACWE method.

Figure 4.19 is the copy of Figure 4 (printed with the permission of the owner) from [39]. It showed the gene clusters with the cell cycle-regulated. Figure 4.20 is extracted from Figure 4.19. Figure 4.20 shows the gene clusters with cell cycle-regulated and the histone (key protein component) cluster under the alpha factor experiment. Figure 4.20 was presented as an array with 9 rows and 18 columns. Each row represents a gene and each column represents a microarray image. Histone genes show periodical regulation. It has 9 genes in Figure 4.20. Only eight genes were used in the experimental data sets which were "HTB2, HTA2, HHF1, HHF2, HHT2, HTB1, HHT1 and HTA1". A total of 18 microarray images were used in the alpha factor experiment. Each image was taken every 7 minutes in a time serial. Figure 4.20 showed a clear cell cycle in the Alpha Factor section under cluster histone. It showed "Green Red Green Red" cycle. Approximately, at minutes 0, 7, and 14, it showed green. At minutes 28, 35 and 42, it showed red. At minutes 63 and 70, it showed green again. At minutes 84, 91 and 98, it showed red again. Red showed the DNA expression was increased, black showed the DNA expression was stable. Green represented that the DNA expression was decreased.

The biological cycle was based on the experiments depending on time repeated every 7 minutes. In that experiment a total of 18 microarrays were used. This is why each row in Figure 4.20 had 18 elements.

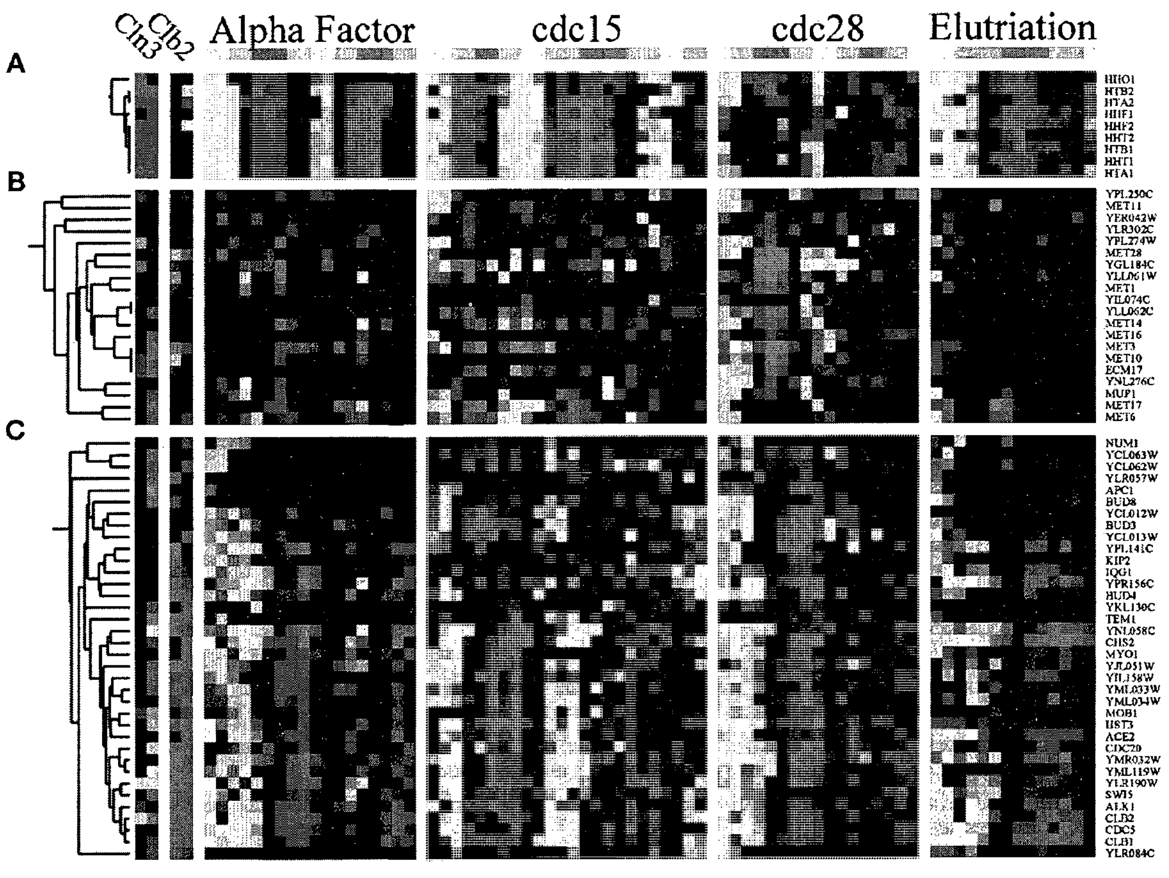

Figure 4.19 Gene clusters with cell cycle-regulation.

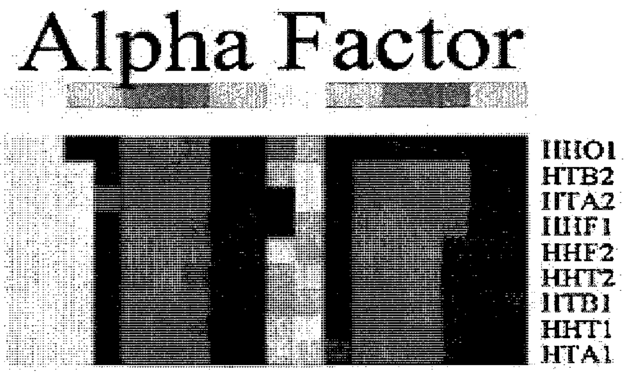

Figure 4.20 Histone genes under yeast alpha factor experiment.

Figure 4.21 shows the "Green Red Green Red" cycle which was the same as in Figure 4.20. The ratio R_g is equal to Y_g/X_g. This ratio represents the DNA folder change (DNA expression difference). X_g denotes the gth spot pixel intensity of Cy3 channel (Green). Y_g denotes the gth spot pixel intensity of Cy5 channel (Red). Logarithms of this ratio is $\log_2^{R_g}$. Using the logarithmic transformation reduces the skewness of the distribution and improves the variance estimation. Log ratio of intensities from Red and Green channel was computed to represent histone genes expression in alpha factor using the ACWE method.

In Figure 4.22, the log ratio of intensities from the Red and Green channel was computed to represent the histone gene expression in the alpha factor using the SRG method. At minute 70, it showed red. This is different from Figure 4.20.

In Figure 4.23, the log ratio of intensities from the Red and Green channel was computed to represent the histone genes expression in the alpha factor using the GOGAC method. At minute 70, it showed red. This is different from Figure 4.20.

Therefore, we can conclude that ACWE is a better segmentation method than both the SRG and GOGAC methods since it matched the exact biological cell cycle.

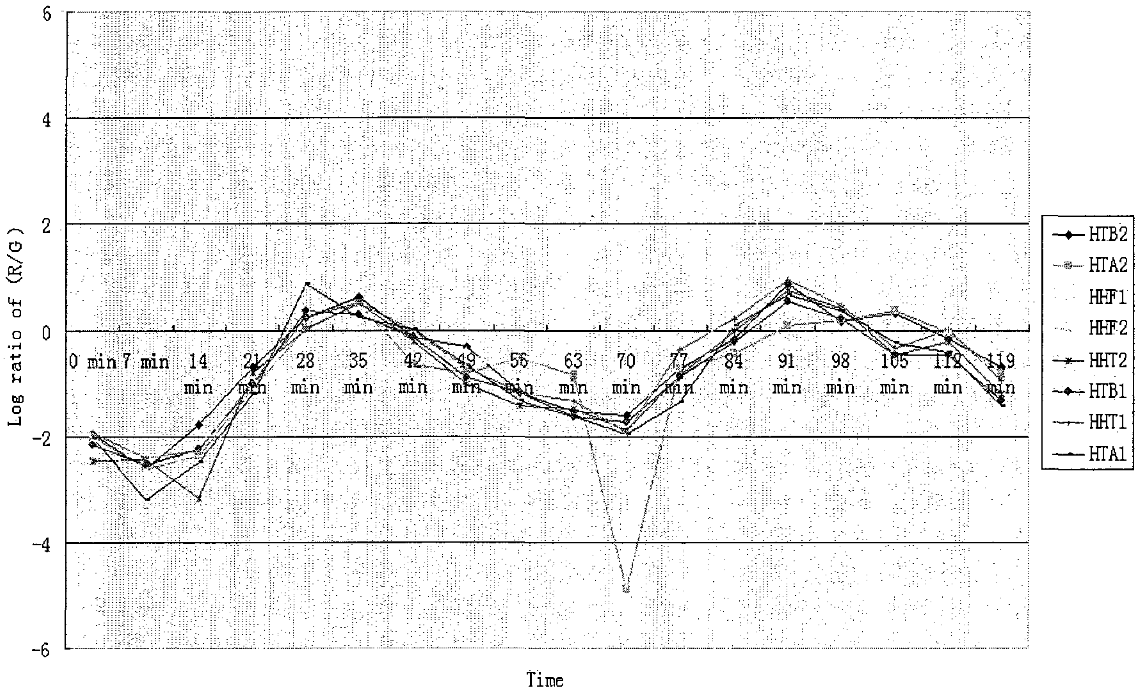

Figure 4.21 Log ratio (*R/G*) of the histone genes in alpha factor using ACWE method.

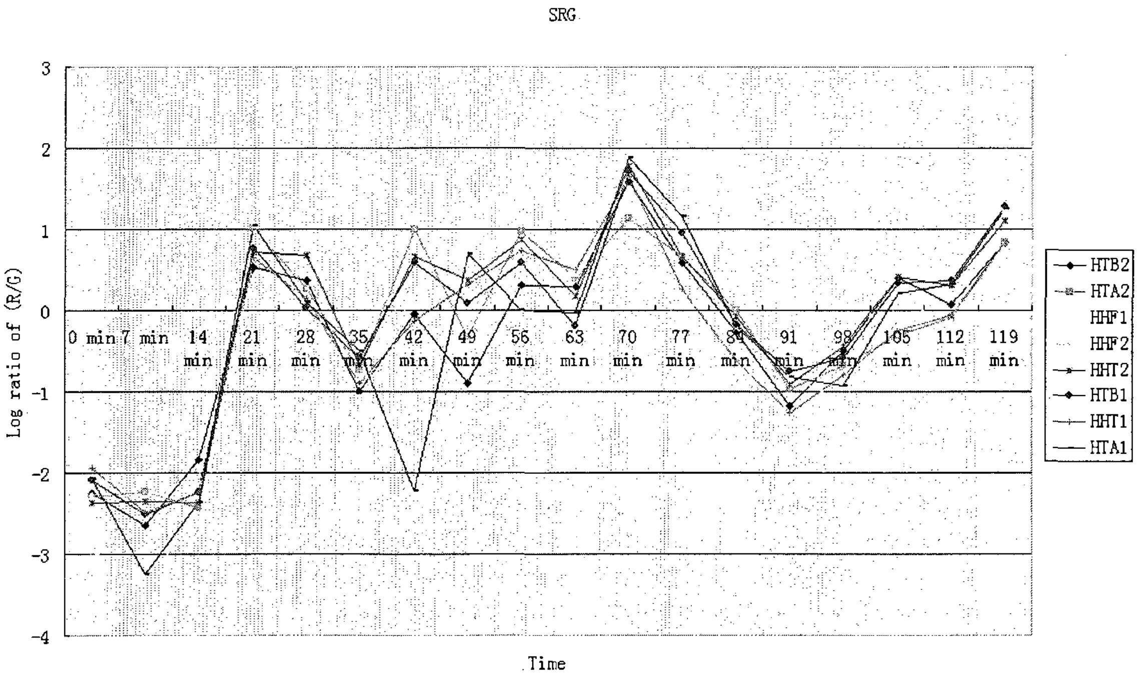

Figure 4.22 Log ratio (*R/G*) of histone genes in alpha factor using SRG method.

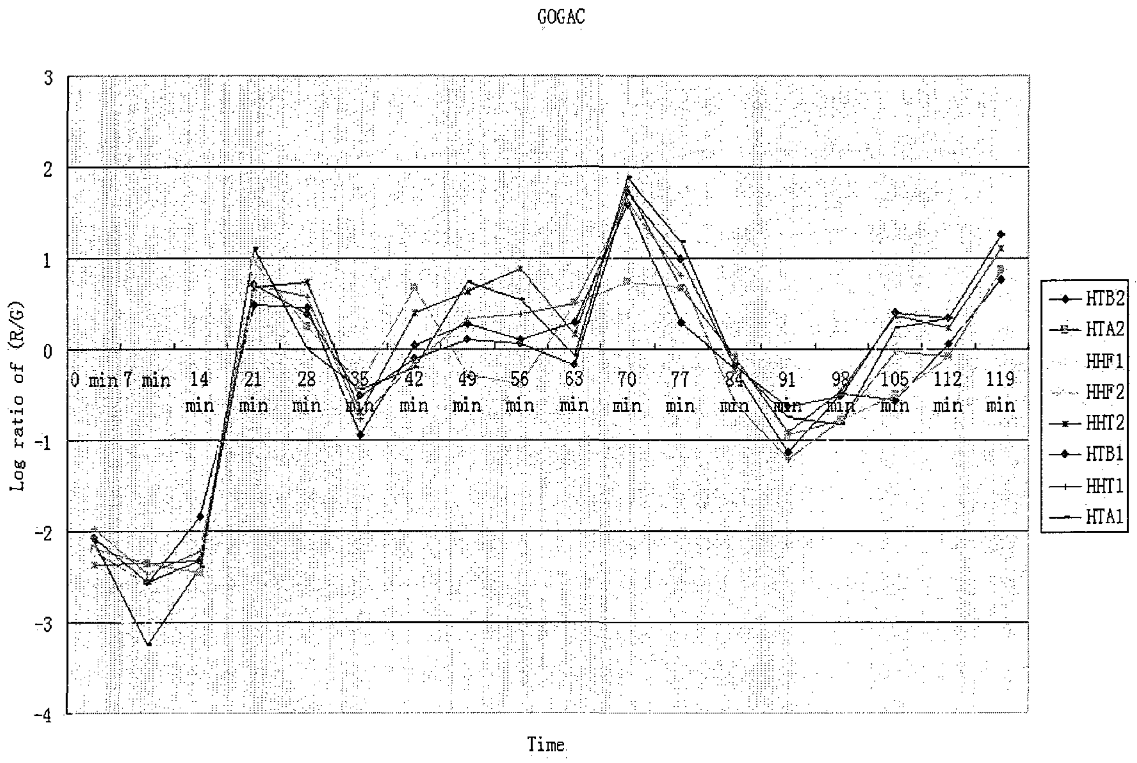

Figure 4.23 Log ratio (*R/G*) of histone genes in alpha factor using GOGAC method.

4.2 Experimental Result for Affymetrix GeneChip

Before the ACWE segmenting, the Affymatrix GeneChip .DAT file needs to be converted to a .TIFF file which can be processed by our ACWE segmentation software. The .DAT file also needs to adjust since the .DAT image might rotate a little bit. All the converting and adjusting can be done with our program which was written using Matlab with the Bioinformatics Toolbox.

We used real Affymetrix GeneChip microarray images from [31] for segmentation. The experimental results show that ACWE is the most accurate segmentation method.

In Figure 4.24 a segmentation image was provided by using ACWE for an Affymetrix GeneChip image.

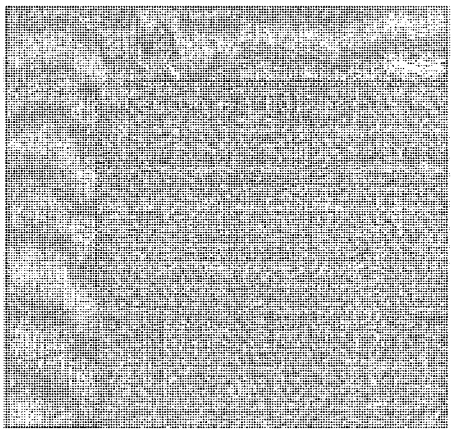

Figure 4.24 Segmentation for Affymetrix GeneChip image using ACWE

Figure 4.25 is the linear correlation between ACWE and Affymetrix segmentation method using the sample .DAT file from Affymetrix website [31] named "arabidopsisath1". These two methods are highly correlated and the linear correlation coefficient R was 0.99172. In Figure 4.25, all the cells' intensities are used for comparison.

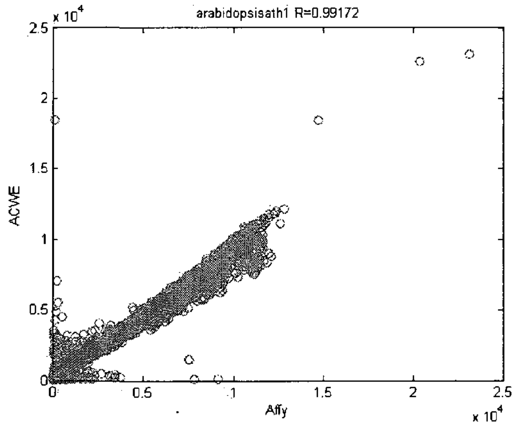

Figure 4.25 Linear Correlation between ACWE and Affymetrix segmentation method

The expression of gene in Affemetrix GeneChip can be used as the following methods according to [37].

The first method is Average Difference. $S_g = \frac{\sum_{i=1}^{m_g}(PM_{gi} - MM_{gi})}{m_g} = \frac{\sum_{i=1}^{m_g} Y_{gi}}{m_g}$, where S_g is the Average Difference signal value. Y_g represents the difference of each probe pair. m_g is the number of the probe pairs in one probe set. PM_g gives the intensity of the Perfect Match probe cell and MM_g shows the intensity of the Mismatch probe cell.

The second method is a Weighted Average Difference. $S_g = \exp(T_{biwt}(\{X_{gi}\}))$, where $T_{biwt}(\{X_{gi}\}) = \frac{\sum_{i=1}^{m_g} w_{gi} X_{gi}}{\sum_{i=1}^{m_g} w_{gi}}$, $X_{gi} = \log(Y_{gi})$, S_g is the Weighted Average Difference signal value, Y_{gi} represents the difference of each probe pair, w_{gi} is the weight using the bi-weight weighting function: $w(u) = (1-u^2)^2$ if $|u|<1$ and $w(u)=0$ if $|u| \geq 1$, and m_g is the number of the probe pairs in one probe set.

The third method is the Perfect Match Only. $S_g = \exp\left(\frac{\sum_{i=1}^{m_{gi}} \log(PM_{gi})}{m_g}\right)$ where S_g is the signal value. PM_{gi} gives the intensity of the Perfect Match probe cell.

There are two groups of control genes that can be used in comparison of which the segmentation method is better.

In [40], Affymetrix gave one group of spike control genes named Hybridization Controls; it contained BioB, BioC, BioD, and Cre for four genes. The concentrations for these genes were 1.5 pM, 5 pM, 25 pM, and 100 pM. The average difference signals of these four genes should be linearly correlated with these four concentrations with a linear coefficient of R=1 in theory.

The other group of control genes Affymetrix provided in [41] was Labeling Control. There are four genes in that group: Lys, Phe, Thr, and Dap. Respectively, they have the concentration as follows: 1:100000, 1:50000, 1:25000, and 1:6667. In theory, the average difference signals of these four genes should be highly correlated given their concentrations with a linear correlation coefficient R equal to 1.

Combining these two groups of control genes into one large group, linear regression was performed using these eight control genes. Using the concentration for control genes we fitted two regression lines for the Affymetrix segmentation method and the ACWE method. We calculated SSE1 (Sum Square Error for the Affymetrix segmentation method), SSE2 (Sum Square Error for ACWE method) to determine which method was better.

Figure 4.26 and Figure 4.27 showed different segmentation methods applied on the same image file named "Nk2-sd_null_8b". By comparing the linear coefficient R in both figures, we found the intensity values from the ACWE method had a higher correlation with the concentration of the control genes. R square in Figure 4.26 (Affymetrix method) was 0.9019 and 0.9949 in Figure 4.27 (ACWE method).

The value of R should be 1: the closer the better.

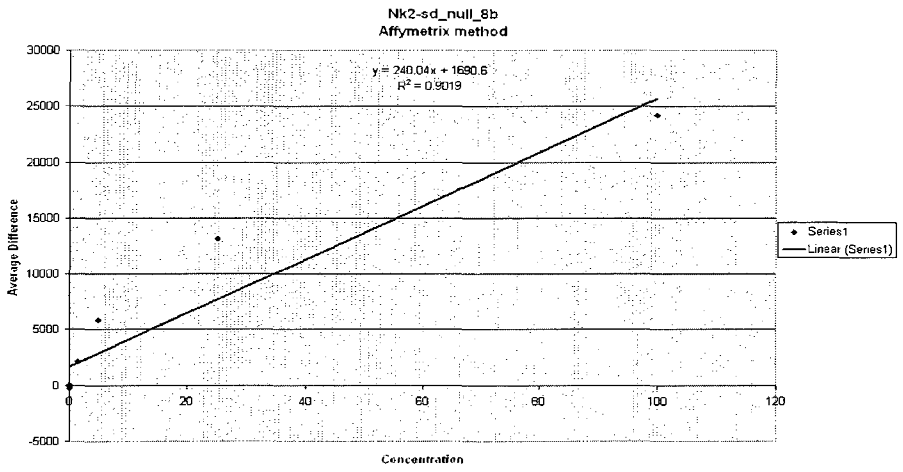

Figure 4.26 Linear regression line between intensity values using Affymetrix method and the concentration of control genes.

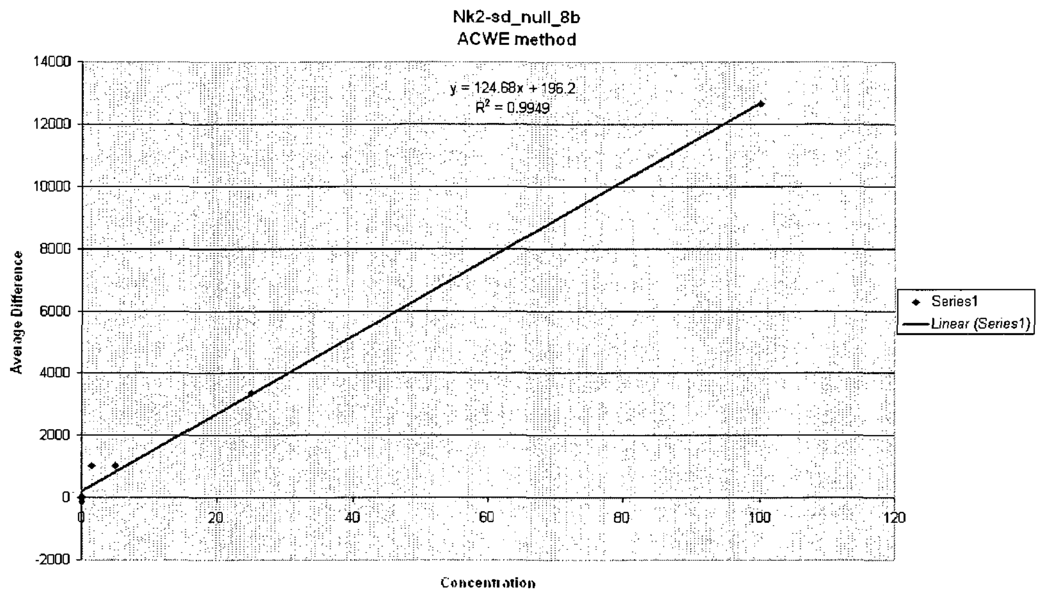

Figure 4.27 Linear regression line between intensity values using ACWE method and the concentration of control genes.

Table 4.3 provides the result of Sum Square Error (SSE) of concentration from the Affymetrix method (SSE1=923.3815536) and ACWE method (SSE2=43.19708603) for one image named "Nk2-sd_null_8b". By comparing the SSE1 and SSE2 values, we observe that ACWE has a smaller SSE value than that of the Affymetrix method. The result shown in the ACWE had a smaller SSE, which means the ACWE method was the better segmentation method.

Table 4.3 Sum Square Error of concentration between Affymetrix and ACWE methods for one image.

Gene Name	Concentration	Affymetrix	ACWE	C1	C2	C-C1	C-C2			SSE1(Affymetrix)	SSE2(ACWE)
bioB	1.5	2127.017	1025.717	1.8181	6.653165	-0.3181	-5.15317	0.101187	26.55511	923.3815536	43.19708603
bioC	5	5874.585	1024.775	17.43037	6.645613	-12.4304	-1.64561	154.514	2.708041		
bioD	25	13143.08	3378.275	47.71071	25.52194	-22.7107	-0.52194	515.7766	0.272417		
cre	100	24181.82	12628.05	93.69778	99.71006	6.30222	0.289942	39.71798	0.084067		
lys	0.00001	-6.74	-23.4333	-7.07107	-1.76158	7.071081	1.761586	50.00019	3.103186		
phe	0.00002	60.81667	53.06667	-6.78963	-1.14801	6.789652	1.148026	46.09938	1.317963		
thr	0.00004	-65.225	11.11667	-7.31472	-1.48447	7.314758	1.484507	53.50569	2.203761		
dap	0.000149993	-224.675	-132.533	-7.97898	-2.63662	7.979133	2.636766	63.66656	6.952537		

Table 4.4 presents a table of comparing the results of SSE between the Affymetrix method and the ACWE method for a group of images from one experiment. SSE1 was for Affymetrix and SSE2 was for ACWE. For a group of images in the same experiment named "Non-failing Normal Patient", of all the 14 images, 12 of them showed the ACWE method has a smaller SSE than Affymetrix and therefore is a better choice for a segmentation method.

Table 4.4 All images in one experiment SSE comparison result.

Experiment name	Picture Name	SSE1 (Affymetrix)	SSE2 (M-ACWE)	SSE1>SSE2
Non-failing "Normal" Patient	PAN_2	38.27691209	32.77832087	1
Non-failing "Normal" Patient	PAN_3	36.28320103	35.96082502	1
Non-failing "Normal" Patient	PAN_5	46.88678481	45.15868231	1
Non-failing "Normal" Patient	PAN_112-1	48.38131101	45.45383283	1
Non-failing "Normal" Patient	PAN_118	75.10650964	69.26868915	1
Non-failing "Normal" Patient	PAN_148	78.74727054	54.61473739	1
Non-failing "Normal" Patient	PAN_200	93.21929361	92.14273772	1
Non-failing "Normal" Patient	PAN_249	67.01296917	62.05873328	1
Non-failing "Normal" Patient	PAN_291	36.5763059	29.09979072	1
Non-failing "Normal" Patient	PAN_294	61.48699358	66.08901362	0
Non-failing "Normal" Patient	PAN_300	55.08412502	44.49878051	1
Non-failing "Normal" Patient	PAN_322	90.12749255	86.26224659	1
Non-failing "Normal" Patient	PAN_325	52.89713986	55.28467159	0
Non-failing "Normal" Patient	PAN_326	88.25998143	76.74630714	1

Table 4.5 shows all the experiments in the database ([31]). There were a total of 19 experiments on that website and a total of 377 images were segmented using both the Affymetrix segmentation and ACWE methods. The results show that 189 out of 377 images (more than half of the images) proved that the ACWE method was better. Also considering the experiment dependent effect, we checked by individual experiment. Eleven out of 19 experiments showed that the ACWE method was better and 1 of 19 experiments showed these two methods were a tie.

Table 4.5 All experiments SSE comparison result.

Experiment number	Experiment name	Images in experiment	Images showed M-ACWE method was better than Affymetrix	Ratio
1	Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function	16	11	68.75%
2	Idiopathic Cardiomyopathy	25	8	32.00%
3	Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor	9	5	55.56%
4	C57BL/6 Benchmark Set for Early Cardiac Development	36	20	55.56%
5	Post-Partum Cardiomyopathy	4	2	50.00%
6	Familial Cardiomyopathy	5	3	60.00%
7	Hypertrophic Cardiomyopathy	5	3	60.00%
8	Viral Cardiomyopathy	6	2	33.33%
9	Non-failing "Normal" Patient	14	12	85.71%
10	Ischemic Cardiomyopathy	30	13	43.33%
11	Cardiac hypertrophy related to the PI3 Kinase Signaling Pathway (v2)	9	5	55.56%
12	Myocardial Infarction	59	35	59.32%
13	Congenital Heart Disease in Csx/Nkx2.5 mutant embryos	18	10	55.56%
14	Deletion of the Nk2 specific domain of the Nkx2.5	24	11	45.83%
15	FVB Benchmark and Sex Comparison	24	11	45.83%
16	Pressure-overload induced cardiac hypertrophy in FVB mice	36	15	41.67%
17	Exercised Induced Hypertrophy	30	17	56.67%
18	Mice over-expressing dn-p21ras as a model system for severe dilated cardiomyopathy	3	2	66.67%
19	Hypertrophy and Heart Failure Through High Salt Diet and Exercise	24	4	16.67%
		377	189	50.13%

In Table 4.6, 2 out of 3 experiment model types showed that the ACWE method was better. In the human model, 4 out of 8 experiments showed ACWE was better and 1 of 8 experiments showed ACWE was equal to the Affymetrix method. In the mouse model, 7 out of 10 experiments showed that the ACWE method was better. In the rat model, only 1 experiment was in the model; it didn't prefer to ACWE. The mouse model and the human model are the main models that Harvard Medical School uses in the projects. In the mouse model the ACWE method shows the significant advantage over the Affymetrix method. Also in the human model, the ACWE method is still better than Affymetrix.

Table 4.6 Type of experiments and comparison among methods

Experiment model	Total experiments	Experiments showed M-ACWE method was better
Human	8	4 (1 more experiment showed M-ACWE was the same as Affymetrix method)
Mouse	10	7
Rat	1	0

Table 4.7 showed there were a total of seven Affymetrix GeneChip templates used for the total nineteen experiments of Harvard Medical School datasets. Three out of 7 templates showed the ACWE method was better than the Affymetrix method. Three out of 7 templates showed ACWE was the same as that of the Affymetrix method. Only 1 template showed the ACWE method was worse.

Table 4.7 Affymetrix GeneChip template type SSE comparison results

Affymetrix GeneChip Template Type	Total experiments	Experiments showed S-ACWE method was better
HgU95Av2	1	1
HgU133plus2	7	3 (1 more experiment showed S-ACWE was the same as Affymetrix method)
Mu11k Set	2	1
MgU74A	4	2
MgU74Av2	3	3
Mg430 2.0	1	1
RgU34A	1	0

Therefore, we can conclude that the ACWE method is better than the Affymetrix segmentation method since it has less SSE in concentration of control genes.

CHAPTER 5

EXPERIMENTAL RESULTS OF DNA MICROARRAYS SEGMENTATION BASED ON THE IMPROVED ACWE METHOD

In the previous chapters, we have already shown that the ACWE method is a better method when compared to the current DNA microarray segmentation methods. In this chapter, we will discuss the improvements we made to let the original ACWE segmentation method be more accurate or faster.

5.1 Experimental Results of Simplified ACWE (S-ACWE) Method

In Section 3.6.1, the S-ACWE method is discussed. In this section the results of the S-ACWE method are analyzed.

During the experiment, we still used the same database as in Chapter 4. We compared the sum squared errors of the Affymetrix method and the S-ACWE method.

Table 5.1 shows all of the experiments in the database [31]. There were a total of 19 experiments in that website and a total of 377 images were segmented using both the Affymetrix segmentation and S-ACWE methods.

The results showed that 163 out of the 377 images (less than half of the images) conclude that the S-ACWE method was better. Also, considering the experiment dependent effect we checked by individual experiments.

Six out of 19 experiments showed S-ACWE method was better and 3 of 19 experiments showed these two methods were a tie.

Table 5.1 All experiments SSE comparison result.

Experiment number	Experiment name	Images in experiment	Images showed S-ACWE method was better than Affymetrix	Ratio
1	Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function	16	7	43.75%
2	Idiopathic Cardiomyopathy	25	4	16.00%
3	Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor	9	4	44.44%
4	C57BL/6 Benchmark Set for Early Cardiac Development	36	18	50.00%
5	Post-Partum Cardiomyopathy	4	2	50.00%
6	Familial Cardiomyopathy	5	3	60.00%
7	Hypertrophic Cardiomyopathy	5	3	60.00%
8	Viral Cardiomyopathy	6	2	33.33%
9	Non-failing "Normal" Patient	14	11	78.57%
10	Ischemic Cardiomyopathy	30	11	36.67%
11	Cardiac hypertrophy related to the PI3 Kinase Signaling Pathway (v2)	9	5	55.56%
12	Myocardial Infarction	59	30	50.85%
13	Congenital Heart Disease in Csx/Nkx2.5 mutant embryos	18	10	55.56%
14	Deletion of the Nk2 specific domain of the Nkx2.5	24	10	41.67%
15	FVB Benchmark and Sex Comparison	24	12	50.00%
16	Pressure-overload induced cardiac hypertrophy in FVB mice	36	13	36.11%
17	Exercised Induced Hypertrophy	30	14	46.67%
18	Mice over-expressing dn-p21ras as a model system for severe dilated cardiomyopathy	3	1	33.33%
19	Hypertrophy and Heart Failure Through High Salt Diet and Exercise	24	3	12.50%
		377	163	43.24%

In Table 5.2, 0 out of 3 experiment model types showed that the S-ACWE method was better. In the human model, 3 out of 8 experiments showed S-ACWE was better and 1 of 8 experiments showed S-ACWE was equal to Affymetrix method. In the mouse model, 3 out of 10 experiments showed S-ACWE method was better and 2 of 8 experiments showed S-ACWE was equal to the Affymetrix method. In the rat model, 0 out of 1 experiment model types showed that the S-ACWE method was better. S-ACWE method was not accurate in all the 3 experiment models compared with the Affymetrix segmentation method.

Table 5.2 Type of experiments and comparison among methods

Experiment model	Total experiments	Experiments showed S-ACWE method was better
Human	8	3 (1 more experiment showed S-ACWE was the same as Affymetrix method)
Mouse	10	3 (2 more experiment showed S-ACWE was the same as Affymetrix method)
Rat	1	0

Table 5.3 showed there were a total of seven Affymetrix GeneChip templates used for the nineteen experiments in Harvard Medical School datasets. One out of 7 templates showed the S-ACWE method was better than the Affymetrix method. One out of 7 templates showed S-ACWE was the same as that of the Affymetrix method. Five templates showed that the S-ACWE method was worse.

Table 5.3 Affymetrix GeneChip template type SSE comparison result

Affymetrix GeneChip Template Type	Total experiments	Experiments showed S-ACWE method was better
HgU95Av2	1	0
HgU133plus2	7	3 (1 more experiment showed S-ACWE was the same as Affymetrix method)
Mu11k Set	2	1
MgU74A	4	0 (1 more experiment showed S-ACWE was the same as Affymetrix method)
MgU74Av2	3	2
Mg430 2.0	1	0 (1 more experiment showed S-ACWE was the same as Affymetrix method)
RgU34A	1	0

Through the experimental results, we found that the S-ACWE did not provide the accuracy we were looking for. It also means that the length constraint in the ACWE method is highly related with the segmentation accuracy.

5.2 Experimental Results of a Fast ACWE (F-ACWE) Method

In Section 3.6.2, the F-ACWE method is presented. After using all the same 19 experiments for testing by the F-ACWE method, we got the following results.

Table 5.4 showed all the experiments in the database [31]. There were a total of 19 experiments in that website and a total of 377 images were segmented using both

Affymetrix segmentation and F-ACWE methods. The results showed 195 out of 377 images (more than 51.72% of the images) showed the F-ACWE method was better. Also considering the experiment dependent effect, we checked by individual experiment. Nine out of 19 experiments showed the F-ACWE method was better and 3 of 19 experiments showed these two methods were a tie.

Table 5.4 All experiments SSE comparison results

Experiment number	Experiment name	Images in experiment	Images showed F-ACWE method was better than Affymetrix	Ratio
1	Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function	16	8	50.00%
2	Idiopathic Cardiomyopathy	25	7	28.00%
3	Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor	9	5	55.56%
4	C57BL/6 Benchmark Set for Early Cardiac Development	36	20	55.56%
5	Post-Partum Cardiomyopathy	4	1	25.00%
6	Familial Cardiomyopathy	5	3	60.00%
7	Hypertrophic Cardiomyopathy	5	4	80.00%
8	Viral Cardiomyopathy	6	3	50.00%
9	Non-failing "Normal" Patient	14	12	85.71%
10	Ischemic Cardiomyopathy	30	15	50.00%
11	Cardiac hypertrophy related to the PI3 Kinase Signaling Pathway (v2)	9	6	66.67%
12	Myocardial Infarction	59	38	64.41%
13	Congenital Heart Disease in Csx/Nkx2.5 mutant embryos	18	10	55.56%
14	Deletion of the Nk2 specific domain of the Nkx2.5	24	10	41.67%
15	FVB Benchmark and Sex Comparison	24	11	45.83%
16	Pressure-overload induced cardiac hypertrophy in FVB mice	36	17	47.22%
17	Exercised Induced Hypertrophy	30	18	60.00%
18	Mice over-expressing dn-p21ras as a model system for severe dilated cardiomyopathy	3	1	33.33%
19	Hypertrophy and Heart Failure Through High Salt Diet and Exercise	24	6	25.00%
		377	195	51.72%

In Table 5.5, 2 out of 3 experiment model types showed that the F-ACWE method was better. In the human model, 3 out of 8 experiments showed F-ACWE was better and 3 of 8 experiments showed that F-ACWE was equal to the Affymetrix method. In the mouse model, 6 out of 10 experiments showed that the F-ACWE method was better. In the rat model, since only 1 experiment was in the model, it did not prefer to the F-ACWE.

The mouse model and the human model are the main models that Harvard Medical School uses in these projects. In the mouse model, the F-ACWE method shows the significant advantage over the Affymetrix method. Also in the human model, the F-ACWE method is still better than Affymetrix.

Table 5.5 Type of experiments and comparison among methods

Experiment model	Total experiments	Experiments showed F-ACWE method was better
Human	8	3 (3 more experiment showed F-ACWE was the same as Affymetrix method)
Mouse	10	6
Rat	1	0

Table 5.6 showed there were a total of seven Affymetrix GeneChip templates used for the total nineteen experiments of Harvard Medical School datasets. Three out of 7 templates showed F-ACWE method was better than the Affymetrix method. One out of 7 templates showed F-ACWE was the same as that of the Affymetrix method. Three templates showed that the F-ACWE method was worse.

Table 5.6 Affymetrix GeneChip template type SSE comparison result

Affymetrix GeneChip Template Type	Total experiments	Experiments showed F-ACWE method was better
HgU95Av2	1	0
HgU133plus2	7	3 (2 more experiment showed F-ACWE was the same as Affymetrix method)
Mu11k Set	2	1
MgU74A	4	1
MgU74Av2	3	3
Mg430 2.0	1	1
RgU34A	1	0

Overall, the F-ACWE method can provide an accurate segmentation result and can significantly reduce the segmentation speed since we do not need to solve the partial differential equations like we did using the ACWE method. It can replace the ACWE method when large Affymetrix images need to be segmented.

5.3 Experimental Results of Improving ACWE Methods (I-ACWE, I2-ACWE)

In Section 3.6.3, I-ACWE and I2-ACWE methods are proposed by using the high order finite different schemes.

After using the same Harvard Medical School database for experiments, we got the results as follows:

Table 5.7 showed all the experiments in the database ([31]). There were a total of 19 experiments in that website and a total of 377 images were segmented both using the Affymetrix segmentation and I-ACWE methods. From the results, 229 out of 377 images (more than 60 percent of the images) showed that the I-ACWE method was better. Also considering the experiment dependent effect, we checked by individual experiments. Fourteen out of 19 experiments showed I-ACWE method was better and 1 of 19 experiments showed these two methods were a tie.

Table 5.7 All experiments SSE comparison result for I-ACWE

Experiment number	Experiment name	Images in experiment	Images showed I-ACWE method was better than Affymetrix	Ratio
1	Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function	16	11	68.75%
2	Idiopathic Cardiomyopathy	25	14	56.00%
3	Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor	9	5	55.56%
4	C57BL/6 Benchmark Set for Early Cardiac Development	36	17	47.22%
5	Post-Partum Cardiomyopathy	4	3	75.00%
6	Familial Cardiomyopathy	5	3	60.00%
7	Hypertrophic Cardiomyopathy	5	2	40.00%
8	Viral Cardiomyopathy	6	3	50.00%
9	Non-failing "Normal" Patient	14	12	85.71%
10	Ischemic Cardiomyopathy	30	16	53.33%
11	Cardiac hypertrophy related to the PI3 Kinase Signaling Pathway (v2)	9	6	66.67%
12	Myocardial Infarction	59	41	69.49%
13	Congenital Heart Disease in Csx/Nkx2.5 mutant embryos	18	7	38.89%
14	Deletion of the Nk2 specific domain of the Nkx2.5	24	10	41.67%
15	FVB Benchmark and Sex Comparison	24	18	75.00%
16	Pressure-overload induced cardiac hypertrophy in FVB mice	36	28	77.78%
17	Exercised Induced Hypertrophy	30	17	56.67%
18	Mice over-expressing dn-p21ras as a model system for severe dilated cardiomyopathy	3	3	100.00%
19	Hypertrophy and Heart Failure Through High Salt Diet and Exercise	24	13	54.17%
		377	229	60.74%

In Table 5.8, 3 out of 3 experiment model types showed that the I-ACWE method was better. In the human model, 6 out of 8 experiments showed I-ACWE was better and 1 of 8 experiments showed I-ACWE was equal to Affymetrix method. In the mouse model, 7 out of 10 experiments showed that the I-ACWE method was better. For the rat model, I-ACWE gave better results.

Table 5.8 Types of experiments and comparison among methods for I-ACWE

Experiment model	Total experiments	Experiments showed I-ACWE method was better
Human	8	6 (1 more experiment showed I-ACWE was the same as Affymetrix method)
Mouse	10	7
Rat	1	1

Table 5.9 showed there were a total of seven Affymetrix GeneChip templates used for the nineteen experiments of Harvard Medical School datasets. Five out of 7 templates showed that the I-ACWE method was better than the Affymetrix method.

Table 5.9 Affymetrix GeneChip template type SSE comparison result for I-ACWE

Affymetrix GeneChip Template Type	Total experiments	Experiments showed I-ACWE method was better
HgU95Av2	1	1
HgU133plus2	7	5 (1 more experiment showed I-ACWE was the same as Affymetrix method)
Mu11k Set	2	0
MgU74A	4	4
MgU74Av2	3	3
Mg430 2.0	1	0
RgU34A	1	1

By far, the improved ACWE (I-ACWE) method is even more accurate in segmenting than the ACWE method.

For I2-ACWE method, after experimenting with the Harvard datasets, we got the results as follows:

Table 5.10 showed all the experiments in the database ([31]). There was a total of 19 experiments in that website and a total of 377 images were segmented both using Affymetrix segmentation and I2-ACWE methods. From the results, 229 out of 377 images (more than 60 percent of the images) showed I2-ACWE method was better. Also considering the experiment dependent effect, we checked by individual experiment showed these two methods were tie.

Table 5.10 All experiments SSE comparison result for I2-ACWE

Experiment number	Experiment name	Images in experiment	Images showed I2-ACWE method was better than Affymetrix	Ratio
1	Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function	16	11	68.75%
2	Idiopathic Cardiomyopathy	25	14	56.00%
3	Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor	9	5	55.56%
4	C57BL/6 Benchmark Set for Early Cardiac Development	36	17	47.22%
5	Post-Partum Cardiomyopathy	4	3	75.00%
6	Familial Cardiomyopathy	5	3	60.00%
7	Hypertrophic Cardiomyopathy	5	3	60.00%
8	Viral Cardiomyopathy	6	3	50.00%
9	Non-failing "Normal" Patient	14	12	85.71%
10	Ischemic Cardiomyopathy	30	17	56.67%
11	Cardiac hypertrophy related to the PI3 Kinase Signaling Pathway (v2)	9	6	66.67%
12	Myocardial Infarction	59	41	69.49%
13	Congenital Heart Disease in Csx/Nkx2.5 mutant embryos	18	6	33.33%
14	Deletion of the Nk2 specific domain of the Nkx2.5	24	10	41.67%
15	FVB Benchmark and Sex Comparison	24	17	70.83%
16	Pressure-overload induced cardiac hypertrophy in FVB mice	36	28	77.78%
17	Exercised Induced Hypertrophy	30	17	56.67%
18	Mice over-expressing dn-p21ras as a model system for severe dilated cardiomyopathy	3	3	100.00%
19	Hypertrophy and Heart Failure Through High Salt Diet and Exercise	24	13	54.17%
		377	229	60.74%

In Table 5.11, 3 out of 3 experiment model types showed that the I2-ACWE method was better. In the human model, 7 out of 8 experiments showed that the I2-ACWE was better and 1 of 8 experiments showed that the I2-ACWE was equal to Affymetrix method. In the mouse model, 7 out of 10 experiments showed that the I2-ACWE method was better. For the rat model, I2-ACWE was better.

Table 5.11 Type of experiments and comparison among methods for I2-ACWE

Experiment model	Total experiments	Experiments showed I2-ACWE method was better
Human	8	7 (1 more experiment showed I2-ACWE was the same as Affymetrix method)
Mouse	10	7
Rat	1	1

Table 5.12 shows that there were seven Affymetrix GeneChip templates used for the total nineteen experiments out of the Harvard Medical School datasets. Five out of 7 templates showed that the I2-ACWE method was better than the Affymetrix method.

Table 5.12 Affymetrix GeneChip template type SSE comparison result for I2-ACWE

Affymetrix GeneChip Template Type	Total experiments	Experiments showed I2-ACWE method was better
HgU95Av2	1	1
HgU133plus2	7	6 (1 more experiment showed I2-ACWE was the same as Affymetrix method)
Mu11k Set	2	0
MgU74A	4	4
MgU74Av2	3	3
Mg430 2.0	1	0
RgU34A	1	1

The comparison results showed that the I2-ACWE method is even more accurate in segmenting than the I-ACWE method. The accurate intensity data provided are much more helpful in the cluster analysis function prediction of the data mining for future analysis.

5.4 Experimental Results of Hybrid Methods for ACWE

Section 3.6.4 presents hybrid methods for ACWE.

In order to improve the accuracy of the segmentation methods, we also proposed a hybrid method for the Affymetrix DNA microarray image segmentation. We combined the ACWE method with the Affymetirx segmentation method.

We calculated the sum of square error for Minimum, Average and Maximum of the two values of concentration obtained by the ACWE and Affymetrix methods. We observed that the maximum gives a smaller sum of square error, therefore a better fit. Two hundred forty-three out of 377 (64.46%) showed that the maximum had a better fit than the Affymetrix method.

We calculated the sum of square error for Minimum, Average and Maximum of the two values of concentration obtained by the F-ACWE and Affymetrix methods. We observed that the Maximum gives a smaller sum of square error, therefore a better fit. Two hundred twenty-three out of 377 (59.15%) showed Maximum had a better fit than the Affymetrix method.

We calculated the sum of square error for Minimum, Average and Maximum of the two values of concentration obtained by the I-ACWE and Affymetrix methods. We observed that the Maximum gives a smaller sum of square error, therefore a better fit. Two hundred eighty-two out of 377 (74.8%) showed Maximum had a better fit than the Affymetrix method.

We calculated the sum of square error for Minimum, Average and Maximum of the two values of concentration obtained by the I2-ACWE and Affymetrix methods. We observed that the Maximum gives a smaller sum of square error, therefore a better fit.

Two hundred eighty-two out of 377 (74.8%) showed Maximum had a better fit than the Affymetrix method.

Therefore, we can conclude that the ACWE, F-ACWE, I-ACWE and I2-ACWE methods are better than the Affymetrix segmentation method since they have less sum of square error in concentration of control genes. By using the hybrid methods, the more accurate segmentation results can be obtained.

5.5 Improved ACWE (I2-ACWE) Method Shows Accurate Gene Expressions at the Biological Level

The Harvard Medical School Cardio Genomics program published some research results related to some high expressed genes. In [42], the authors presented that some genes were highly correlated with heart weight to body weight ratio (HW/BW). These results were obtained from the mouse experiment. HW/BW uses mg/g as a unit. During the experiment, HW/BW ratio was recorded for each mouse. The genes highly correlated with HW/BW ratio were presented by the authors.

Table 5.13 showed that 10 genes were highly positively correlated with the HW/BW ratio. R was the linear coefficient between the gene and HW/BW ratio. The genes expression values were computed based on different segmentation methods (Affymetrix method and I2-ACWE method). The comparison results showed that I2-ACWE method yielded more accurate R values than those of the Affymetrix method. Six out of 10 genes showed that the I2-ACWE method gave higher positive correlation coefficient values.

Table 5.13 Genes positive correlated with HW/BW ratio in mouse experiment.

Gene	R of HW/BW (Affymetrix)	R of HW/BW (I2-ACWE)	R (I2-ACWE>Affymetrix)
IGFBP-5	0.7792	0.7848	1
Glucokinase	0.8226	0.7947	0
Serine proteinase inhibitor, clade F1	0.7355	0.7517	1
Mlc, alkali, fast skeletal muscle	0.8519	0.8521	1
Calsequestrin 1	0.8261	0.8228	0
Procollagen, type 8, alpha 1	0.7238	0.7326	1
P4ha2	0.7367	0.6851	0
Hsp105	0.8241	0.8246	1
Lectin, galactose binding, soluble 1	0.7288	0.7304	1
Clusterin, Clu	0.7729	0.7669	0

Some genes were negatively correlated with the HW/BW ratio. In the mouse experiment, there were 3 genes highly negative correlated with HW/BW. Table 8.14 showed 2 out of 3 genes presented higher linear coefficient R values using I2-ACWE method than the Affymetrix method.

Table 5.14 Genes negative correlated with HW/BW ratio in mouse experiment.

Gene	R of HW/BW (Affymetrix)	R of HW/BW (I2-ACWE)	R (I2-ACWE>Affymetrix)
Hypothetical protein MGC37568	-0.9303	-0.9338	1
Prkab1	-0.9141	-0.8397	0
Pah	-0.8394	-0.8527	1

IGFBP-5 is one of the genes which were positively correlated with the HW/BW ratio. In Figure 5.1, a linear regression analysis was done, the gene IGFBP-5 expression value that was obtained by the I2-ACWE method was highly positively correlated with the HW/BW ratio.

In Figure 5.2, a linear regression analysis was performed; the gene IGFBP-5 expression value obtained by the Affymetrix method was highly positively correlated with the HW/BW ratio.

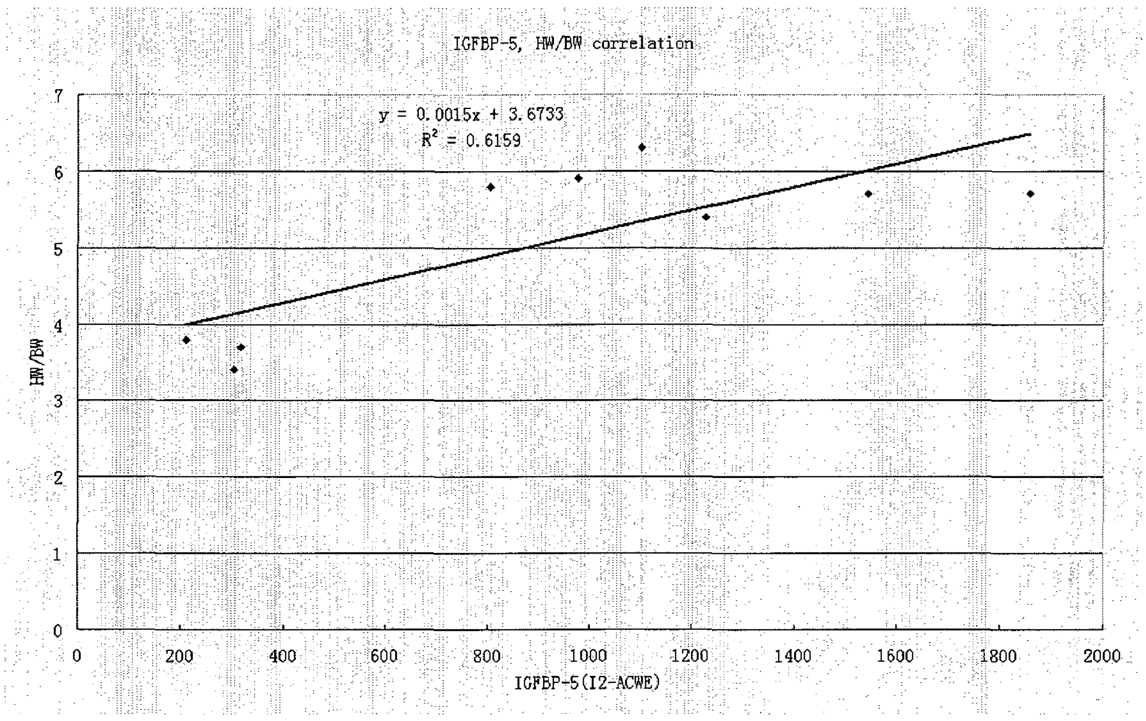

Figure 5.1 Gene IGFBP-5 positive correlated with HW/BW ratio in the mouse experiment using the I2-ACWE method.

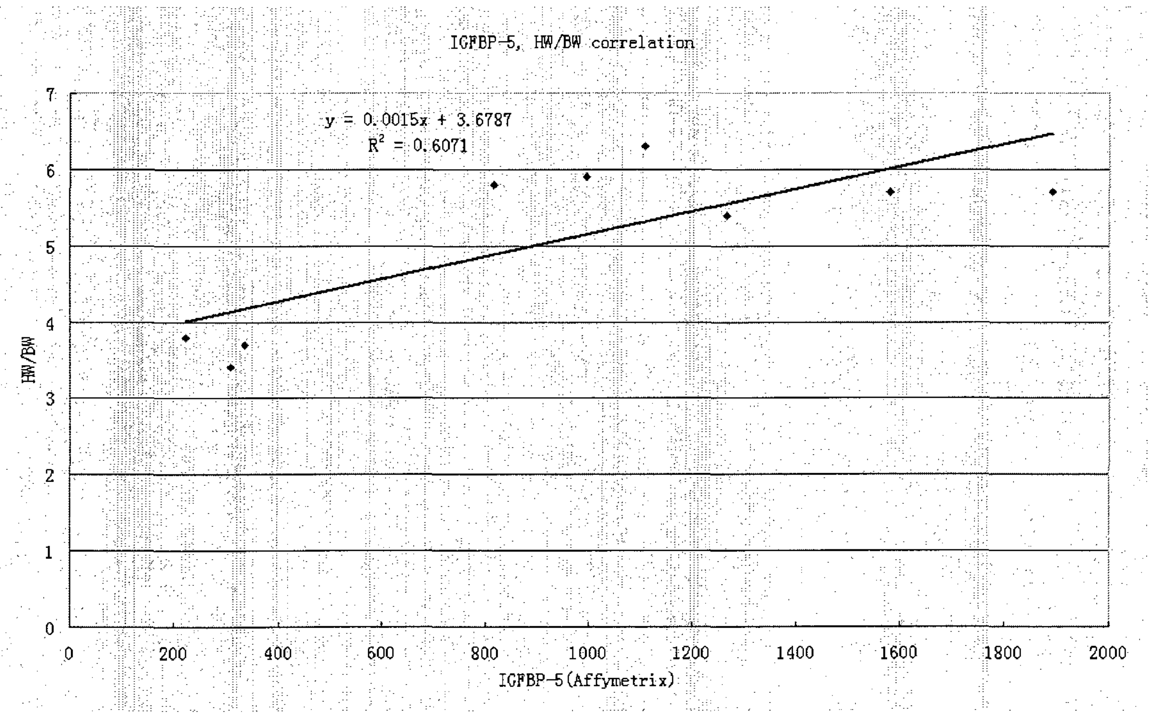

Figure 5.2 Gene IGFBP-5 positive correlated with HW/BW ratio in the mouse experiment using affymetrix method.

By comparing the two linear regression coefficients, the I2-ACWE method provides more accuracy in this case, since the R square of the I2-ACWE method is larger than that of the Affymetrix method.

PAH is one of the genes which are negatively correlated with the HW/BW ratio. In Figure 5.3, a linear regression analysis was done. The gene pah expression value obtained by the I2-ACWE method was highly negatively correlated with the HW/BW ratio.

In Figure 5.4, a linear regression analysis was done. The gene pah expression value obtained by the Affymetrix method and was highly negatively correlated with the HW/BW ratio.

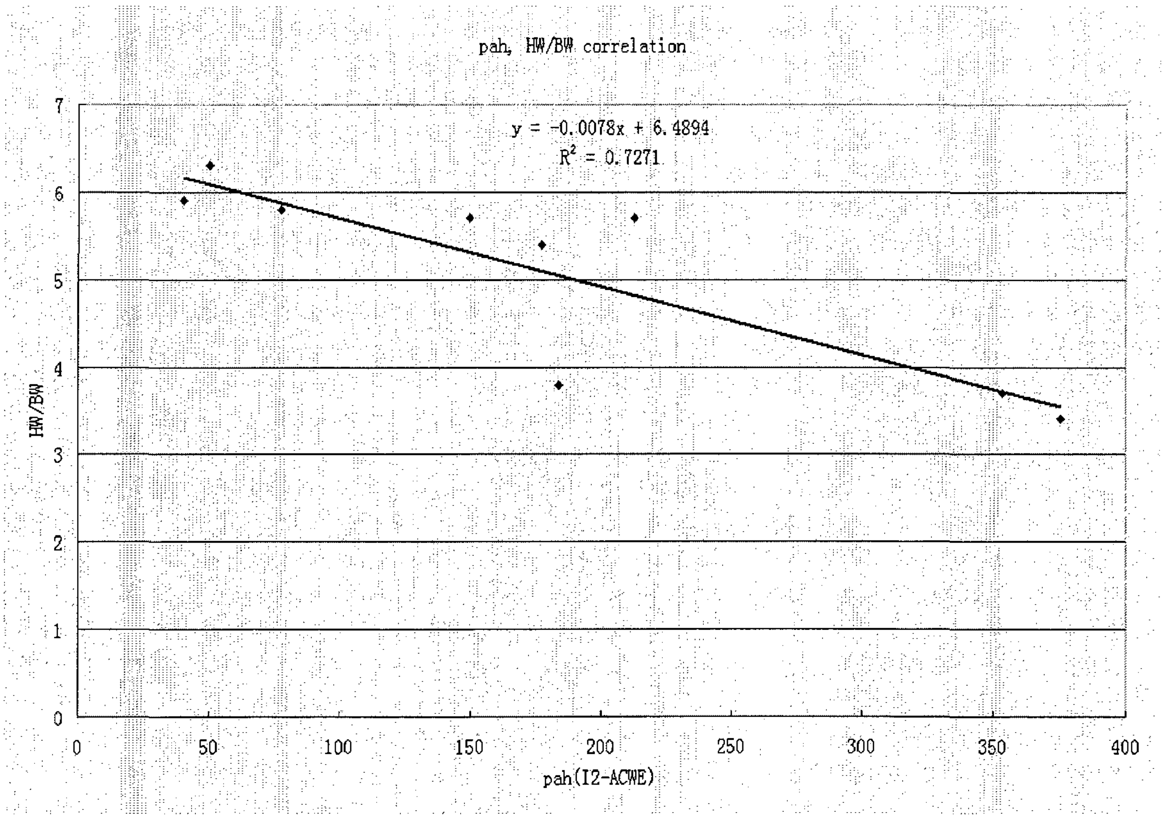

Figure 5.3 Gene path negatively correlated with the HW/BW ratio in the mouse experiment using the I2-ACWE method.

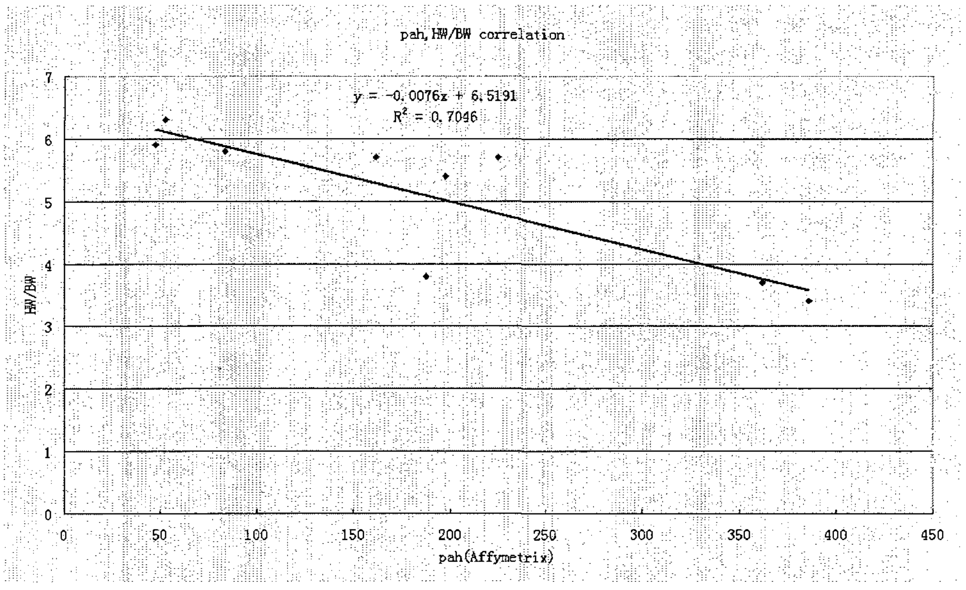

Figure 5.4 Gene pah negatively correlated with the HW/BW ratio in the mouse experiment using the Affymetrix method.

Comparing the R square value in Figures 5.3 and 5.4, it shows I2-ACWE method has a larger R square value than the Affymetrix method.

We conclude that in the mouse experiment, the I2-ACWE method provides more accurate segmentation results than the Affymetrix method.

For the human experiment, Left Ventricular Ejection Fraction (LVEF) is used as a marker of left ventricular systolic function. Heart diseases will reduce the value of LVEF. In [43,44,45,46,47], some genes were presented in correlation with LVEF.

Chemokine receptor 1 (CCR1), chemokine ligand 3 (CCL3), chemokine ligand 4 (CCL4), chemokine receptor 4 (CXCR4), glycogen synthase kinase 3 beta (GSK3B), matrix metallopeptidase 3 (MMP3), natriuretic peptide precursor B (NPPB), and natriuretic peptide precursor A (NPPA) were genes correlated with LVEF.

In Table 5.15, there were 8 genes correlated with LVEF based on the research results from [43-47]. Four genes were positively correlated with LVEF and other 4 genes were negatively correlated with LVEF. In the positive correlated genes, 4 out of 4 showed that the I2-ACWE method gave a higher correlation. In the negatively correlated genes, 3 out of 4 showed that the I2-ACWE method gave a higher correlation. Therefore, we could conclude that the I2-ACWE method would provide more accurate segmentation results.

Table 5.15 Genes correlated with LVEF in the human experiment.

Gene	R of LVEF (Affymetrix)	R of LVEF (I2-ACWE)	R (I2-ACWE>Affymetrix)
CCR1	-0.1913	-0.2254	1
CXCR4	-0.0447	-0.1466	1
NPPB	-0.1353	-0.2159	1
NPPA	-0.1237	-0.2126	1
CCL3	0.1587	0.1597	1
CCL4	0.0332	0.0346	1
GSK3B	0.2681	0.2498	0
MMP3	0.0825	0.3317	1

By comparing Figures 5.5 and 5.6, the I2-ACWE method gave the higher positive correlation coefficient.

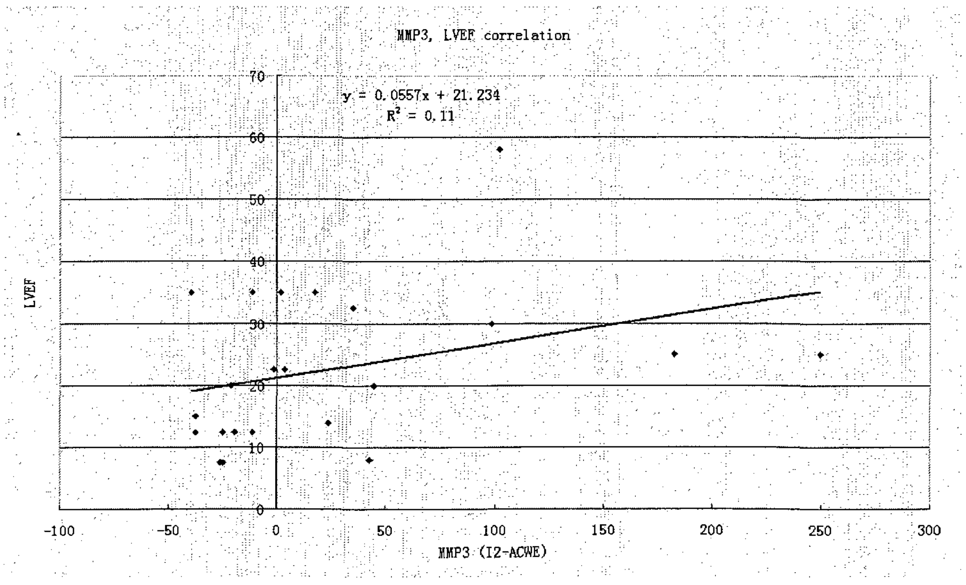

Figure 5.5 Gene MMP3 positively correlated with LVEF in the human experiment using the I2-ACWE method.

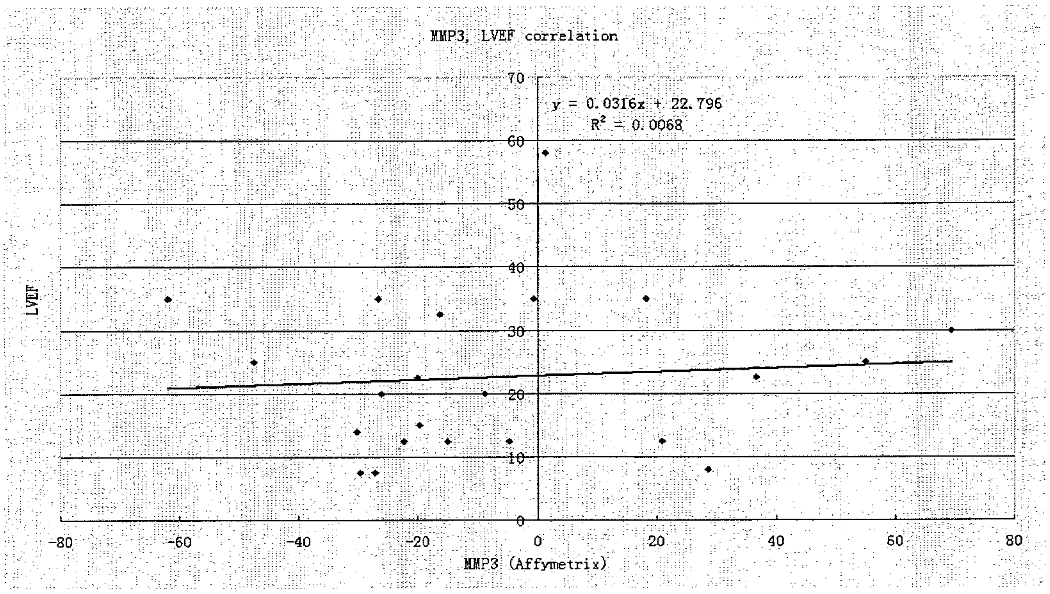

Figure 5.6 Gene MMP3 positively correlated with LVEF
in the human experiment using Affymetrix method.

Figures 5.7 and 5.8 showed that NPPA was negatively correlated with LVEF. The correlation coefficient of the I2-ACWE was better than that of the Affymetrix.

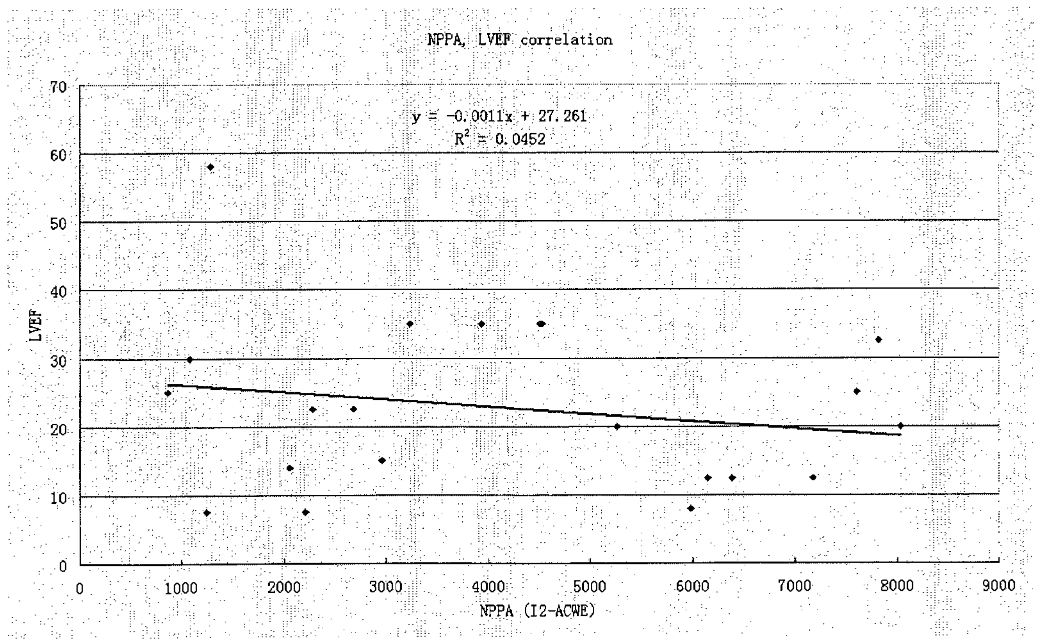

Figure 5.7 Gene NPPA negatively correlated with LVEF
in the human experiment using I2-ACWE method.

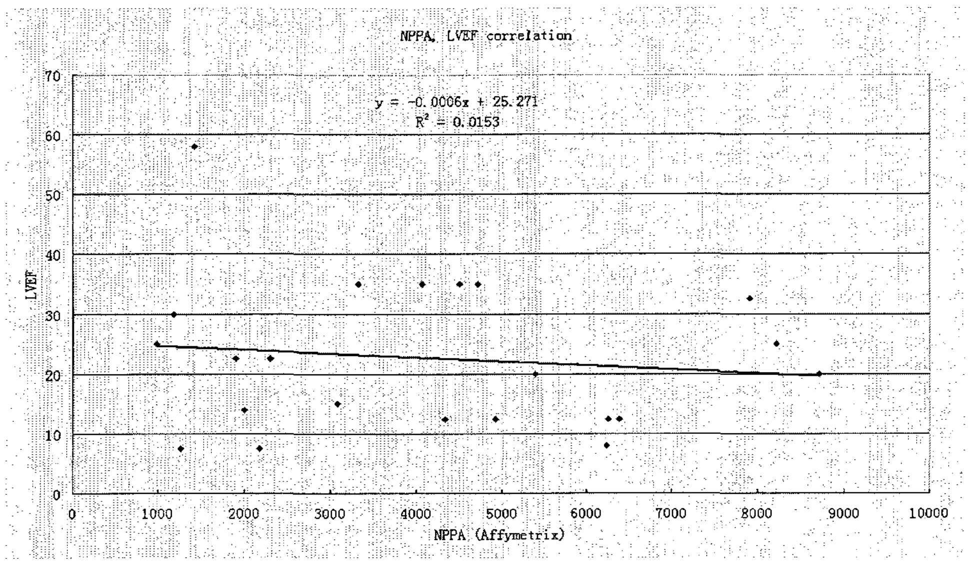

Figure 5.8 Gene NPPA negatively correlated with LVEF in the human experiment using Affymetrix method.

Therefore, the I2-ACWE showed more accurate results in the human experiment.

Glyceraldehyde-3-phosphate dehydrogenase catalyzes (GAPDH) is a house keeping gene. As a house keeping gene the range of expression of GAPDH should be small and GAPDH should be relatively stable in the changing of gene expression.

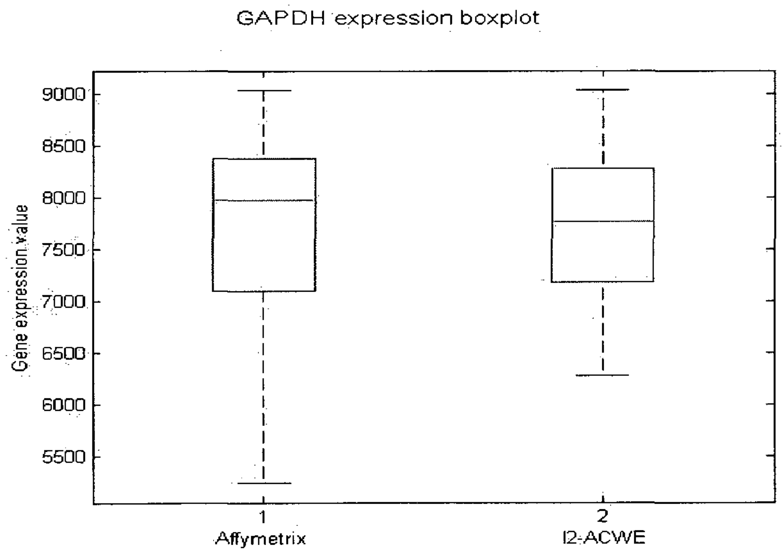

Figure 5.9 Boxplot of the gene GAPDH expression one of the experiments of human model.

Figure 5.9 showed the different GAPDH expression ranges obtained from the Affymetrix method and the I2-ACWE. In this example, we found I2-ACWE gave the more accurate segmentation result, since the inter-quantile range of GAPDH using the I2-ACWE method is smaller than that of Affymetrix method.

In Table 5.16, the GAPDH expression data range was computed by using every image in each experiment. Each experiment has two GAPDH expression range values using the Affymetrix method and I2-ACWE method. In a total of 19 experiments, 11 of 19 showed that the data range values obtained by the I2-ACWE method had smaller values, which meant the I2-ACWE method had accurate segmentation results. Eight of 19 showed that the Affymetrix method had smaller range values. By comparison, we conclude that the I2-ACWE method has better segmentation results.

Table 5.16 Gene GAPDH expression range in all the experiments.

Experiment number	Experiment name	Affymetrix has smaller GAPDH data range	I2-ACWE has smaller GAPDH data range
1	Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function	✓	
2	Idiopathic Cardiomyopathy		✓
3	Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor		✓
4	C57BL/6 Benchmark Set for Early Cardiac Development		✓
5	Post-Partum Cardiomyopathy	✓	
6	Familial Cardiomyopathy	✓	
7	Hypertrophic Cardiomyopathy		✓
8	Viral Cardiomyopathy		✓
9	Non-failing "Normal" Patient	✓	
10	Ischemic Cardiomyopathy		✓
11	Cardiac hypertrophy related to the PI3 Kinase Signaling Pathway (v2)		✓
12	Myocardial Infarction		✓
13	Congenital Heart Disease in Csx/Nkx2.5 mutant embryos	✓	
14	Deletion of the Nk2 specific domain of the Nkx2.5		✓
15	FVB Benchmark and Sex Comparison	✓	
16	Pressure-overload induced cardiac hypertrophy in FVB mice		✓
17	Exercised Induced Hypertrophy		✓
18	Mice over-expressing dn-p21ras as a model system for severe dilated cardiomyopathy	✓	
19	Hypertrophy and Heart Failure Through High Salt Diet and Exercise	✓	
		8	11

Based on Table 5.17, 3 out of 7 templates showed I2-ACWE gave smaller range values, 2 out of 7 showed I2-ACWE and Affymetrix were a tie. Only 2 out of 7 showed that the I2-ACWE method was worse. By comparison, I2-ACWE method was better than Affymetrix method based on GeneChip templates.

Table 5.17 GAPDH expression range values based on different GeneChip template types.

Affymetrix GeneChip Template Type	Total experiments	Experiments showed I2-ACWE method had smaller GAPDH expression range
HgU95Av2	1	0
HgU133plus2	7	4
Mu11k Set	2	1
MgU74A	4	2
MgU74Av2	3	3
Mg430 2.0	1	1
RgU34A	1	0

By analyzing Table 5.18, I2-ACWE was better in the mouse model. I2-ACWE was the same as Affymetrix in the human model. I2-ACWE was worse in the rat model, but there was only one experiment in the rat model. I2-ACWE was better than Affymetrix based on the experimental models.

Table 5.18 GAPDH expression range values based on different experimental models.

Experiment model	Total experiments	Experiments showed I2-ACWE method had smaller GAPDH range
Human	8	4
Mouse	10	7
Rat	1	0

In conclusion, I2-ACWE could provide more accurate segmentation results in the biology gene expression level.

5.6 Improved ACWE (I2-ACWE) Method Shows Accurate Gene Expressions Using Rank Sum.

A rank sum is to compare variability of the gene expression values obtained from I2-ACWE and Affymetrix methods.

These eight control genes (BioB, BioC, BioD, Cre, Lys, Phe, Thr, and Dap) are used in comparison.

We first compute $[y_i - mean(y)]^2$ using the gene expression values from the Affymetrix method. The index i is from 1 to 8 and y_i represents the gene expression value of each control gene. Mean(y) is the average gene expression value of all the eight control genes.

Then we compute $[x_i - mean(x)]^2$ using the gene expression values from I2-ACWE method. The index i is from 1 to 8 and x_i represents the gene expression value of each control gene. Mean(x) is the average gene expression value of all the eight control genes.

After that, we compared the value of $[y_i - mean(y)]^2$ with that of $[x_i - mean(x)]^2$. If $[y_i - mean(y)]^2 > [x_i - mean(x)]^2$, we assign rank value 2 to ith gene in the Affymetrix group and rank value 1 to ith gene in the I2-ACWE group. If $[y_i - mean(y)]^2 <$ $[x_i - mean(x)]^2$,, we assign rank 1 to ith gene in the Affymetrix group and rank value 2 to ith gene in the I2-ACWE group.

If all eight control genes are finished for the comparison and rank assignment, the sum of rank values are computed base on Affymetrix and I2-ACWE groups.

The larger sum of rank values means the gene expression values are not closer to the mean gene expression values. The smaller sum of rank values should be better.

Table 5.19 showed all the experiments in the database ([31]). There were totally 19 experiments in that website and a total of 377 images were segmented both using Affymetrix segmentation and I2-ACWE methods. From the results, 271 out of 377 images (more than 70 percent of the images) showed I2-ACWE method has smaller sum of rank. Also considering the experiment dependent effect, we checked by individual experiment showed these two methods were tie.

Table 5.19 All experiments Sum of rank comparison result for I2-ACWE

Experiment number	Experiment name	Images in experiment	Sum of rank I2-ACWE > Affymetrix	Sum of rank I2-ACWE = Affymetrix	Sum of rank I2-ACWE < Affymetrix	Ratio
1	Aortic Stenosis, Congestive Cardiomyopathy and Normal Left Ventricular Function	16	12	2	2	75.00%
2	Idiopathic Cardiomyopathy	25	20	3	2	80.00%
3	Cardiac Hypertrophy Induced by the Insulin-like Growth Factor 1 Receptor	9	9	0	0	100.00%
4	C57BL/6 Benchmark Set for Early Cardiac Development	36	27	4	5	75.00%
5	Post-Partum Cardiomyopathy	4	4	0	0	100.00%
6	Familial Cardiomyopathy	5	5	0	0	100.00%
7	Hypertrophic Cardiomyopathy	5	5	0	0	100.00%
8	Viral Cardiomyopathy	6	5	0	1	83.33%
9	Non-failing "Normal" Patient	14	12	1	1	85.71%
10	Ischemic Cardiomyopathy	30	27	0	3	90.00%
11	Cardiac hypertrophy related to the PI3 Kinase Signaling Pathway (v2)	9	9	0	0	100.00%
12	Myocardial Infarction	59	57	2	0	96.61%
13	Congenital Heart Disease in Csx/Nkx2.5 mutant embryos	18	3	0	15	16.67%
14	Deletion of the Nk2 specific domain of the Nkx2.5	24	9	0	15	37.50%
15	FVB Benchmark and Sex Comparison	24	17	2	5	70.83%
16	Pressure-overload induced cardiac hypertrophy in FVB mice	36	19	3	14	52.78%
17	Exercised Induced Hypertrophy	30	30	0	0	100.00%
18	Mice over-expressing dn-p21ras as a model system for severe dilated cardiomyopathy	3	1	1	1	33.33%
19	Hypertrophy and Heart Failure Through High Salt Diet and Exercise	24	0	0	24	0.00%
		377	271	18	88	71.88%

In Table 5.20, 2 out of 3 experiment model types showed that the I2-ACWE method was better of having smaller sum of rank. In the human model, 8 out of 8 experiments showed that the I2-ACWE was better. In the mouse model, 7 out of 10 experiments showed that the I2-ACWE method was better and 1 of 10 experiments showed that the I2-ACWE was equal to the Affymetrix method. For the rat model, the Affymtrix method was better.

Table 5.20 Type of experiments and comparison among methods for I2-ACWE with sum of rank

Experiment model	Total experiments	Experiments showed I2-ACWE method had smaller sum of rank
Human	8	8
Mouse	10	7 (1 more showed I2-ACWE was the same as Affymetrix)
Rat	1	0

Table 5.21 shows that there were seven Affymetrix GeneChip templates used for all nineteen experiments out of the Harvard Medical School datasets. Five out of 7 templates showed that the I2-ACWE method was better than the Affymetrix method by having a smaller sum of rank.

Table 5.21 Affymetrix GeneChip template type sum of rank comparison result for I2-ACWE

Affymetrix GeneChip Template Type	Total experiments	Experiments showed I2-ACWE method had smaller sum of rank
HgU95Av2	1	1
HgU133plus2	7	7
Mu11k Set	2	0
MgU74A	4	3 (1 more showed I2-ACWE was the same as Affymetrix)
MgU74Av2	3	3
Mg430 2.0	1	1
RgU34A	1	0

The comparison results showed that the I2-ACWE method is more accurate in segmenting than the Affymetrix method.

CHAPTER 6

CONCLUSIONS

As described in Chapter 3 the ACWE method in theory can provide more accurate segmentation results than the current segmentation methods applied in the DNA microarray segmentation field. ACWE method has not been successfully applied to large DNA microarray segmentation processes. We applied the algorithm in the DNA microarray field and experimented with the cDNA microarray and the Affymetrix GeneChip. In these two types of microarrays in Chapter 4, we showed that the ACWE method was more accurate than the currently existing cDNA microarray segmentation methods such as SRG , GOGAC, etc. ACWE was also more accurate than the Affymetrix segmentation method, the details of the experiments' results were provided in Chapter 4.

In Chapter 5, we proposed several improvement methods based on ACWE. We used experiments to show that the improved ACWE method (I2-ACWE) was much more accurate than the ACWE method. I2-ACWE was also better than the current cDNA microarray segmentation methods and the Affymetrix method.

We also did the experiment using a Hybrid method to improve the accuracy of the segmentation. For example, in Chapter 4, 189 out of 377 (over half of the total images) images showed that the ACWE was better than the Affymetrix method. The 188 images show that the Affymetrix is better. We calculated the sum of square error for minimum, average and maximum values of the two concentration values obtained by the ACWE and Affymetrix methods. We observed that the maximum value gives a smaller sum of square error, and therefore a better fit. Two hundred forty-three out of 377 (64.46%) showed the maximum had a better fit than the Affymetrix method. This example showed that by hybridizing the ACWE and Affymetrix methods, we got a new segmentation method and it would provide more accurate segmentation results.

In the dissertation, it can be concluded that the ACWE method is more efficient than the current cDNA microarray segmentation methods and the Affymetrix GeneChip segmentation method. The ACWE method can be improved by utilizing higher order terms and hybridization with other methods to provide more accurate segmentation results.

Although the ACWE and improved ACWE segmentation methods cost more computing time, they provide more accurate segmentation results than the other DNA microarray segmentation methods. These DNA microarray images only need to be segmented once and put in the database, only if these images have the most accurate segmentation results. Since these are biological experiments with application in medicine, even a small gain in accuracy is important, and more important than adding more time.

APPENDIX A

SOURCE CODE FOR IMPROVED ACWE (I2-ACWE) SEGMENTATION METHOD

```
/*This program is improvement of Active Contours Without Edges (I2-ACWE)
   segmentation method by using higher order finite different schemes.
   Proposed and implemented by Shenghua Ni.
*/
class segment
{
        // initial variables
        int   xpels, ypels ;
         int   startx, starty ;
         int   lastx, lasty ;
         double c1, c2 ;
         int   n_toreinit, n_doreinit ;
         double [t] sign_d ;
         double [t] area_mapping ;
         double [t] gridcombine_mapping ;
         double [t] forw_dx, back_dx, forw_dy, back_dy, cent_dx, cent_dy ;
         double [t] intensity ;
         double h ;
         double dt ;
         double e ;
         double w ;
         // The Dirac delta funtion
         double dirac(double d)
         {
            double result=1/(Math.PI*e*(1+(d/e)*(d/e)));
            return result;
         }
         void initsigned_dist(double h,int   m,int   n,int   a)
         {
            double [t] center ;
            center = new double[8];
            double r ;
            int    i, j ;
            center[t0]=0;
            center[12]=0;
            center[t0]=Math.floor(m/2*h);
            center[12]=Math.floor(n/2*h);
            r=Math.min((m*h-center[t0]-a*h),(n*h-center[12]-a*h));
            r=Math.max(r,0);
            for (j=0; j<ypels;j++)
               for (i=0;i<xpels;i++)
      sign_d[ti+xpels*j]=r-Math.sqrt(Math.pow((center[t0]-i*h),2)+Math.pow((center[12]-
  j*h),2));
         }
         //compute c1,c2 value using average
         void meanc1_c2()
         {
            int    i, j, counter ;
            double sum1, sum2 ;
            sum1=0;
            sum2=0;
            counter=0;
            for (j=0;j<ypels;j++)
               for (i=0;i<xpels;i++)
               {
                 if (sign_d[ti+xpels*j] >= 0)
                    {
                    counter=counter+1;
                    sum1=sum1+intensity[ti+xpels*j];
                    }
                 else
                    sum2=sum2+intensity[ti+xpels*j];
                 }
            if (counter    !=    0)
               c1=sum1/counter;
            if ((xpels*ypels-counter)    !=    0)
               c2=sum2/(xpels*ypels-counter);
         }
         void get_diff_results()
         {
            int    i, j ;
            for (j=2; j<ypels-2;j++)
               for (i=2;i<xpels-2;i++)
                 {
                 //forw_dx[ti+xpels*j]=(sign_d[ti+1+xpels*j]-sign_d[ti+xpels*j])/h;
                 //if (forw_dx[ti+xpels*j] == 0)
                 //    forw_dx[ti+xpels*j]=Math.pow(2,-23);
```

```
forw_dx[ti+xpels*j]=(4*sign_d[ti+1+xpels*j]-3*sign_d[ti+xpels*j]-sign_d[ti+2+xpels*j]
)/(2*h);
                    if (forw_dx[ti+xpels*j] == 0)
                       forw_dx[ti+xpels*j]=Math.pow(2,-23);
                    //back_dx[ti+xpels*j]=(sign_d[ti+xpels*j]-sign_d[ti-1+xpels*j])/h;
                    //if (back_dx[ti+xpels*j] == 0)
                    //    back_dx[ti+xpels*j]=Math.pow(2,-23);

back_dx[ti+xpels*j]=(3*sign_d[ti+xpels*j]-4*sign_d[ti-1+xpels*j]+sign_d[ti-2+xpels*j])
/(2*h);
                    if (back_dx[ti+xpels*j] == 0)
                       back_dx[ti+xpels*j]=Math.pow(2,-23);
                    //cent_dx[ti+xpels*j]=(sign_d[ti+1+xpels*j]-sign_d[ti-1+xpels*j])/(2*h);
                    //if (cent_dx[ti+xpels*j] == 0)
                    //    cent_dx[ti+xpels*j]=Math.pow(2,-23);

cent_dx[ti+xpels*j]=(8*(sign_d[ti+1+xpels*j]-sign_d[ti-1+xpels*j])-(sign_d[ti+2+xpels*
j]-sign_d[ti-2+xpels*j]))/(12*h);
                    if (cent_dx[ti+xpels*j] == 0)
                       cent_dx[ti+xpels*j]=Math.pow(2,-23);
                    //forw_dy[ti+xpels*j]=(sign_d[ti+xpels*(j+1)]-sign_d[ti+xpels*j])/h;
                    //if (forw_dy[ti+xpels*j] == 0)
                    //    forw_dy[ti+xpels*j]=Math.pow(2,-23);

forw_dy[ti+xpels*j]=(4*sign_d[ti+xpels*(j+1)]-3*sign_d[ti+xpels*j]-sign_d[ti+xpels*(j+
2)])/(2*h);
                    if (forw_dy[ti+xpels*j] == 0)
                       forw_dy[ti+xpels*j]=Math.pow(2,-23);
                    //back_dy[ti+xpels*j]=(sign_d[ti+xpels*j]-sign_d[ti+xpels*(j-1)])/h;
                    //if (back_dy[ti+xpels*j] == 0)
                    //    back_dy[ti+xpels*j]=Math.pow(2,-23);

back_dy[ti+xpels*j]=(3*sign_d[ti+xpels*j]-4*sign_d[ti+xpels*(j-1)]+sign_d[ti+xpels*(j-
2)])/(2*h);
                    if (back_dy[ti+xpels*j] == 0)
                       back_dy[ti+xpels*j]=Math.pow(2,-23);

//cent_dy[ti+xpels*j]=(sign_d[ti+xpels*(j+1)]-sign_d[ti+xpels*(j-1)])/(2*h);
                    //if (cent_dy[ti+xpels*j] == 0)
                    //    cent_dy[ti+xpels*j]=Math.pow(2,-23);

cent_dy[ti+xpels*j]=(8*(sign_d[ti+xpels*(j+1)]-sign_d[ti+xpels*(j-1)])-(sign_d[ti+xpels
*(j+2)]-sign_d[ti+xpels*(j-2)]))/(12*h);
                    if (cent_dy[ti+xpels*j] == 0)
                       cent_dy[ti+xpels*j]=Math.pow(2,-23);
                    }
```

```
}
void set_dt_e_w(double p_dt,double p_e,double p_w)
{
  dt=p_dt;
  e=p_e;
  w=p_w*255*255;
}
void set_init_curve(int   a)
{
  initsigned_dist(h,xpels,ypels,a);
}
void create(int   x, int y,int   sx1,int sy1,int sx2,int sy2,double [t] data_intensity)
{
  int   i, j ;
  dt=0.1;
  e=1;
  w=0.01*255*255;
  h=1;
  n_toreinit=40;
  n_doreinit=8;
  xpels=x;
  ypels=y;
  startx=sx1;
  starty=sy1;
  lastx=sx2;
  lasty=sy2;

  area_mapping=new double[txpels*ypels];
  sign_d=new double[txpels*ypels];
  forw_dx=new double[txpels*ypels];
  forw_dy=new double[txpels*ypels];
  back_dx=new double[txpels*ypels];
  back_dy=new double[txpels*ypels];
  cent_dx=new double[txpels*ypels];
  cent_dy=new double[txpels*ypels];
  intensity=new double[txpels*ypels];
  initsigned_dist(h,xpels,ypels,4);
  for (j=0;j<ypels;j++)
     for (i=0;i<xpels;i++)
     intensity[ti+xpels*j]=data_intensity[ti+xpels*j];
}
 void segment()
{
  int   i, j ;
  double t ;
  double[t] ea, fa, ga, ha ;
```

```
            int   t_max=10 ;
            ea = new double[txpels*ypels];
            fa = new double[txpels*ypels];
            ga = new double[txpels*ypels];
            ha = new double[txpels*ypels];
           //sign_d = new double[txpels*ypels];
            t=0;

            while (t <= t_max)
              {
              meanc1_c2();
              get_diff_results();
              for (j=2;j<ypels-2;j++)
               for (i=2;i<xpels-2;i++)
                    {

ea[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(forw_dx[ti+xpels*j]*forw
_dx[ti+xpels*j]+cent_dy[ti+xpels*j]*cent_dy[ti+xpels*j]));

fa[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(back_dx[ti+xpels*j]*back
_dx[ti+xpels*j]+cent_dy[ti+xpels*j]*cent_dy[ti+xpels*j]));

ga[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(forw_dy[ti+xpels*j]*forw
_dy[ti+xpels*j]+cent_dx[ti+xpels*j]*cent_dx[ti+xpels*j]));

ha[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(back_dy[ti+xpels*j]*back
_dy[ti+xpels*j]+cent_dx[ti+xpels*j]*cent_dx[ti+xpels*j]));
                    }
              for (j=2;j<ypels-2;j++)
               for (i=2;i<xpels-2;i++)
              {

sign_d[ti+xpels*j]=(sign_d[ti+xpels*j]+ea[ti+xpels*j]*sign_d[ti+1+xpels*j]
            +fa[ti+xpels*j]*sign_d[ti-1+xpels*j]+ga[ti+xpels*j]*sign_d[ti+xpels*(j+1)]
            +ha[ti+xpels*j]*sign_d[ti+xpels*(j-1)]+dt*dirac(sign_d[ti+xpels*j])

*(-(intensity[ti+xpels*j]-c1)*(intensity[ti+xpels*j]-c1)+(intensity[ti+xpels*j]-c2)*(intensi
ty[ti+xpels*j]-c2)))
            /(1+ea[ti+xpels*j]+fa[ti+xpels*j]+ga[ti+xpels*j]+ha[ti+xpels*j]);
                }
            /*
              for (i=1;i<xpels-1;i++)
                 {
                 sign_d[txpels+i]=sign_d[ti+2*xpels];
               sign_d[ti+xpels*(ypels-2)]=sign_d[ti+xpels*(ypels-3)];
                 }
```

```
        for (j=1;j<ypels-1;j++)
          {
          sign_d[t1+xpels*j]=sign_d[t2+xpels*j];
         sign_d[txpels-2+xpels*j]=sign_d[txpels-3+xpels*j];
          }
      */

        for (i=0;i<xpels;i++)
          {
          sign_d[ti]=sign_d[ti+xpels];
         sign_d[ti+xpels*(ypels-1)]=sign_d[ti+xpels*(ypels-2)];
          }

        for (j=0;j<ypels;j++)
          {
          sign_d[t0+xpels*j]=sign_d[t1+xpels*j];
         sign_d[txpels-1+xpels*j]=sign_d[txpels-2+xpels*j];
          }

      //  if ((Math.floor(t/dt)%n_toreinit == 0) && (t   !=   0))
      //      reinitial(n_doreinit);
        t=t+dt;
        }

      for (j=0;j<ypels;j++)
          for (i=0;i<xpels;i++)
              area_mapping[ti+xpels*j]=sign_d[ti+xpels*j];
        ea=null;
        fa=null;
        ga=null;
        ha=null;
    }
    void reinitial(int   n)
    {
      int   i, j, k ;
      double[t] grad_d ;

      grad_d = new double[txpels*ypels];
      for (k=1;k<n+1;k++)
          for (j=1;j<ypels-1;j++)
              for (i=1;i<xpels-1;i++)
               {

  grad_d[ti+xpels*j]=Math.sqrt(((sign_d[ti+1+xpels*j]-sign_d[ti-1+xpels*j])/(2*h))*((
sign_d[ti+1+xpels*j]-sign_d[ti-1+xpels*j])/(2*h))+((sign_d[ti+xpels*(j+1)]-sign_d[ti+xp
els*(j-1)])/(2*h))*((sign_d[ti+xpels*(j+1)]-sign_d[ti+xpels*(j-1)])/(2*h)));

    sign_d[ti+xpels*j]=sign_d[ti+xpels*j]+dt*(sign(sign_d[ti+xpels*j])*(1-grad_d[ti+xp
els*j]));
                    }
          }
        double sign(double arg1)
        {
          double result;
          if (arg1 < 0)
          result = -1;
          else if (arg1 > 0)
             result = 1.0;
          else
             result = 0.0;
          return result;
        }

         void  adjust_boundary(int  direct,double  step,int   sel_startx,int  sel_starty,int
sel_lastx,int sel_lasty)
          {
             int   i, j ;
             int   x, y ;
             int   x1, x2, y1, y2 ;
             if ((sel_startx > startx))
                x1=sel_startx;
             else
                x1=startx;
             if ((sel_starty > starty))
                y1=sel_starty;
             else
                y1=starty;
             if ((sel_lastx < lastx))
                x2=sel_lastx;
             else
                x2=lastx;
             if ((sel_lasty < lasty))
                y2=sel_lasty;
             else
                y2=lasty;
             x=x2-x2+1;
             y=y2-y1+1;
             for (j=0;j<ypels;j++)
                 for (i=0;i<xpels;i++)
                      area_mapping[ti+xpels*j]=sign_d[t0];
```

```
      if (direct == -1)
        {
             for (j=0;j<y;j++)
              for (i=0;i<x;i++)
              {
              if (sign_d[ti+x1-startx+xpels*(j+y1-starty)] < step)
                area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=-1;
              else
                area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=1;
              }
        }
      else if (direct == 1)
        for (j=0;j<y;j++)
          for (i=0;i<x;i++)
            if (sign_d[ti+x1-startx+xpels*(j+y1-starty)] > step)
              area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=1;
            else
              area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=-1;
  }
double areainfo(int   sel_startx, int sel_starty, int sel_lastx, int sel_lasty)
{
    double result;
      int   i, j ;
      int   x, y ;
      int   x1, x2, y1, y2 ;
      if ((sel_startx > startx))
        x1=sel_startx;
      else
        x1=startx;
      if ((sel_starty > starty))
        y1=sel_starty;
      else
        y1=starty;
      if ((sel_lastx < lastx))
        x2=sel_lastx;
      else
        x2=lastx;
      if ((sel_lasty < lasty))
        y2=sel_lasty;
      else
        y2=lasty;
      x=x2-x1+1;
      y=y2-y1+1;
      result=0;
      for (j=0;j<y;j++)
        for (i=0;i<x;i++)
```

```
            {

            if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
               result=Math.round(result+1);
            }
        return result;
    }
    double area_intensitymean(int   sel_startx, int sel_starty, int sel_lastx, int sel_lasty)
    {
        double result;
        int   i, j ;
        int   x, y ;
        int   x1, x2, y1, y2 ;
        int   n ;
        if ((sel_startx > startx))
           x1=sel_startx;
        else
           x1=startx;
        if ((sel_starty > starty))
           y1=sel_starty;
        else
           y1=starty;
        if ((sel_lastx < lastx))
           x2=sel_lastx;
        else
           x2=lastx;
        if ((sel_lasty < lasty))
           y2=sel_lasty;
        else
           y2=lasty;
        x=x2-x1+1;
        y=y2-y1+1;
        result=0;
        n=0;
        for (j=0;j<y;j++)
            for (i=0;i<x;i++)
            {
              if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
                 {
                     result=result+intensity[ti+x1-startx+xpels*(j+y1-starty)];
                       n=n+1;
                 }
              }
        if (n   !=   0)
           result=Math.round(result/n);
        return result;
        }
        void initialize(int  x, int y, int sx1, int sy1, int sx2, int sy2,double [t]
data_intensity)
          {
            int  i, j ;
            dt=0.1;
            e=1;
            w=0.025*255*255;
            h=1;
            n_toreinit=40;
            n_doreinit=8;
          xpels=x;
          ypels=y;
          startx=sx1;
          starty=sy1;
          lastx=sx2;
          lasty=sy2;
            area_mapping = new double[txpels*ypels];
            sign_d=new double[txpels*ypels];
            forw_dx=new double[txpels*ypels];
            forw_dy=new double[txpels*ypels];
            back_dx=new double[txpels*ypels];
            back_dy=new double[txpels*ypels];
            cent_dx=new double[txpels*ypels];
            cent_dy=new double[txpels*ypels];
            intensity=new double[txpels*ypels];
            initsigned_dist(h,xpels,ypels,4);
            for (j=0; j< ypels;j++)
                for (i=0; i<xpels;i++)
                        intensity[ti+xpels*j]=data_intensity[ti+xpels*j];
          }
          double  area_intensity_75pvalue(int   sel_startx,int  sel_starty,int  sel_lastx,int
sel_lasty)
          {
            double result;
            int i,j,k;
            int x,y;
            int x1,x2,y1,y2;
            int n;
            double [t] data;
            if ((sel_startx > startx))
               x1=sel_startx;
            else
               x1=startx;
            if ((sel_starty > starty))
               y1=sel_starty;
```

```
else
   y1=starty;
if ((sel_lastx < lastx))
   x2=sel_lastx;
else
   x2=lastx;
if ((sel_lasty < lasty))
   y2=sel_lasty;
else
   y2=lasty;
x=x2-x1+1;
y=y2-y1+1;
n=0;
for (j=0;j<y;j++)
    for (i=0;i<x;i++)
    {
      if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
      n=n+1;
      }
k=0;
if (n   !=   0)
   {
   data=new double[tn];
   for (j=0;j<y;j++)
    for (i=0;i<x;i++)
    {
         if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
            {
            data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
            k=k+1;
            }
       }
   quicksort(data);
   result=data[t(int) Math.round(n*0.75)-1];
   }
else
   {
   data=new double[tx*y];
   for (j=0;j<y;j++)
    for (i=0;i<x;i++)
         {
         data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
         k=k+1;
         }
   quicksort(data);
   result=data[t(int) Math.round(x*y*0.75)-1];
```

```
        }
        return result;
    }
    public static void quicksort(double[t] a) {
      shuffle(a);                              // to guard against worst-case
      quicksort(a, 0, a.length - 1);
}

// quicksort a[tleft] to a[tright]
public static void quicksort(double[t] a, int left, int right) {
      if (right <= left) return;
      int i = partition(a, left, right);
      quicksort(a, left, i-1);
      quicksort(a, i+1, right);
}

// partition a[tleft] to a[tright], assumes left < right
private static int partition(double[t] a, int left, int right) {
      int i = left - 1;
      int j = right;
      while (true) {
            while (a[t++i]<a[tright])         // find item on left to swap
                  ;                                   // a[tright] acts as sentinel
            while (a[tright]<a[t--j])        // find item on right to swap
                  if (j == left) break;             // don't go out-of-bounds
            if (i >= j) break;                       // check if pointers cross
            exch(a, i, j);                            // swap two elements into place
      }
      exch(a, i, right);                             // swap with partition element
      return i;
}

// exchange a[ti] and a[tj]
private static void exch(double[t] a, int i, int j) {
      double swap = a[ti];
      a[ti] = a[tj];
      a[tj] = swap;
}

// shuffle the array a[t]
private static void shuffle(double[t] a) {
      int N = a.length;
      for (int i = 0; i < N; i++) {
            int r = i + (int) (Math.random() * (N-i));     // between i and N-1
            exch(a, i, r);
      }
```

```
}
        double    background_intensitymedian(int        sel_startx,int    sel_starty,int
sel_lastx,int sel_lasty)
        {
        double result;
        int i,j,k;
        int x,y;
        int x1,x2,y1,y2;
        int n;
        double [t] data;
        if ((sel_startx > startx))
          x1=sel_startx;
        else
          x1=startx;
        if ((sel_starty > starty))
          y1=sel_starty;
        else
          y1=starty;
        if ((sel_lastx < lastx))
          x2=sel_lastx;
        else
          x2=lastx;
        if ((sel_lasty < lasty))
          y2=sel_lasty;
        else
          y2=lasty;
        x=x2-x1+1;
        y=y2-y1+1;
        result=0;
        n=0;
        for (j=0;j<y;j++)
           for (i=0;i<x;i++)
           {
             if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
              n=n+1;
             }
        k=0;
        if (n   !=   0)
          {
          data=new double[tn];
          for (j=0;j<y;j++)
           for (i=0;i<x;i++)
             {
               if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
                 {
                 data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
```

```
                k=k+1;
                }
              }
          quicksort(data);
          if ((n%2) == 0)
            result=Math.round(data[tn/2]);
          else
            result=Math.round((data[tMath.round(n/2)]+data[t(n-1)/2])/2);
          }
          return result;
      }
        double    background_intensity_75pvalue(int       sel_startx,int    sel_starty,int
sel_lastx,int sel_lasty)
      {
        double result;
        int i,j,k;
        int x,y;
        int x1,x2,y1,y2;
        int n;
        double [t] data;
        if ((sel_startx > startx))
          x1=sel_startx;
        else
          x1=startx;
        if ((sel_starty > starty))
          y1=sel_starty;
        else
          y1=starty;
        if ((sel_lastx < lastx))
          x2=sel_lastx;
        else
          x2=lastx;
        if ((sel_lasty < lasty))
          y2=sel_lasty;
        else
          y2=lasty;
        x=x2-x1+1;
        y=y2-y1+1;
        n=0;
        for (j=0;j<y;j++)
           for (i=0;i<x;i++)
           {
             if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
             n=n+1;
             }
        k=0;
        if (n   !=   0)
           {
           data=new double[tn];
           for (j=0;j<y;j++)
            for (i=0;i<x;i++)
           {
                 if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
                    {
                    data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
                    k=k+1;
                    }
                }
     quicksort(data);
           result=data[t(int) Math.round(n*0.75)-1];
           }
        else
           {
           data=new double[tx*y];

           for (j=0;j<y;j++)
            for (i=0;i<x;i++)
                 {
                 data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
                 k=k+1;
                 }
           quicksort(data);
           result=data[t(int) Math.round(x*y*0.75)-1];
           }
           return result;
      }
}
```

APPENDIX B

SOURCE CODE FOR ACWE SEGMENTATION METHOD

```
/*This program is the implementation of Active Contours Without Edges (ACWE)
   segmentation method proposed by Tony F. Chan and Luminita A. Vese.
   Modified and implemented by Shenghua Ni.
*/

class segment
{
        // initial variables
        int    xpels, ypels ;
         int    startx, starty ;
         int    lastx, lasty ;
         double c1, c2 ;
         int    n_toreinit, n_doreinit ;
         double [t] sign_d ;
         double [t] area_mapping ;
         double [t] gridcombine_mapping ;
         double [t] forw_dx, back_dx, forw_dy, back_dy, cent_dx, cent_dy ;
         double [t] intensity ;
         double h ;
         double dt ;
         double e ;
         double w ;

         // The Dirac delta funtion
         double dirac(double d)
         {
           double result=1/(Math.PI*e*(1+(d/e)*(d/e)));
             return result;
         }

         void initsigned_dist(double h,int    m,int    n,int    a)
         {
           double [t] center ;
```

```
            center = new double[8];
            double r ;
            int    i, j ;
            center[t0]=0;
            center[12]=0;
            center[t0]=Math.floor(m/2*h);
            center[12]=Math.floor(n/2*h);
            r=Math.min((m*h-center[t0]-a*h),(n*h-center[12]-a*h));
            r=Math.max(r,0);

            for (j=0; j<ypels;j++)
               for (i=0;i<xpels;i++)
sign_d[ti+xpels*j]=r-Math.sqrt(Math.pow((center[t0]-i*h),2)+Math.pow((center[12]-j*h)
,2));
         }
         //compute c1,c2 value using average
         void meanc1_c2()
         {
            int    i, j, counter ;
            double sum1, sum2 ;
            sum1=0;
            sum2=0;
            counter=0;

            for (j=0;j<ypels;j++)
               for (i=0;i<xpels;i++)
               {
                 if (sign_d[ti+xpels*j] >= 0)
                    {
                    counter=counter+1;
                    sum1=sum1+intensity[ti+xpels*j];
                     }
                 else
                    sum2=sum2+intensity[ti+xpels*j];
               }
            if (counter    !=    0)
               c1=sum1/counter;
            if ((xpels*ypels-counter)    !=    0)
               c2=sum2/(xpels*ypels-counter);
         }
         void get_diff_results()
         {
            int    i, j ;
            for (j=1; j<ypels-1;j++)
               for (i=1;i<xpels-1;i++)
                 {
```

```
                forw_dx[ti+xpels*j]=(sign_d[ti+1+xpels*j]-sign_d[ti+xpels*j])/h;
                if (forw_dx[ti+xpels*j] == 0)
                   forw_dx[ti+xpels*j]=Math.pow(2,-23);
                back_dx[ti+xpels*j]=(sign_d[ti+xpels*j]-sign_d[ti-1+xpels*j])/h;
                if (back_dx[ti+xpels*j] == 0)
                   back_dx[ti+xpels*j]=Math.pow(2,-23);

                cent_dx[ti+xpels*j]=(sign_d[ti+1+xpels*j]-sign_d[ti-1+xpels*j])/(2*h);
                if (cent_dx[ti+xpels*j] == 0)
                   cent_dx[ti+xpels*j]=Math.pow(2,-23);
                forw_dy[ti+xpels*j]=(sign_d[ti+xpels*(j+1)]-sign_d[ti+xpels*j])/h;
                if (forw_dy[ti+xpels*j] == 0)
                   forw_dy[ti+xpels*j]=Math.pow(2,-23);
                back_dy[ti+xpels*j]=(sign_d[ti+xpels*j]-sign_d[ti+xpels*(j-1)])/h;
                if (back_dy[ti+xpels*j] == 0)
                   back_dy[ti+xpels*j]=Math.pow(2,-23);

cent_dy[ti+xpels*j]=(sign_d[ti+xpels*(j+1)]-sign_d[ti+xpels*(j-1)])/(2*h);
                if (cent_dy[ti+xpels*j] == 0)
                   cent_dy[ti+xpels*j]=Math.pow(2,-23);
                 }
        }
        void set_dt_e_w(double p_dt,double p_e,double p_w)
        {
           dt=p_dt;
           e=p_e;
           w=p_w*255*255;
        }
        void set_init_curve(int  a)
        {
           initsigned_dist(h,xpels,ypels,a);
        }
        void create(int   x, int y,int   sx1,int sy1,int sx2,int sy2,double [t] data_intensity)
        {
           int   i, j ;
           dt=0.1;
           e=1;
           w=0.01*255*255;
           h=1;
           n_toreinit=40;
           n_doreinit=8;
           xpels=x;
           ypels=y;
           startx=sx1;
           starty=sy1;
           lastx=sx2;
        lasty=sy2;
        area_mapping=new double[txpels*ypels];
        sign_d=new double[txpels*ypels];
        forw_dx=new double[txpels*ypels];
        forw_dy=new double[txpels*ypels];
        back_dx=new double[txpels*ypels];
        back_dy=new double[txpels*ypels];
        cent_dx=new double[txpels*ypels];
        cent_dy=new double[txpels*ypels];
        intensity=new double[txpels*ypels];
        initsigned_dist(h,xpels,ypels,4);
        for (j=0;j<ypels;j++)
            for (i=0;i<xpels;i++)
            intensity[ti+xpels*j]=data_intensity[ti+xpels*j];
    }
     void segment()
    {
        int   i, j ;
        double t ;
        double[t] ea, fa, ga, ha ;
        int   t_max=10 ;
        ea = new double[txpels*ypels];
        fa = new double[txpels*ypels];
        ga = new double[txpels*ypels];
        ha = new double[txpels*ypels];
      //sign_d = new double[txpels*ypels];
        t=0;
        while (t <= t_max)
           {
           meanc1_c2();
           get_diff_results();

           for (j=1;j<ypels-1;j++)
            for (i=1;i<xpels-1;i++)
                 {
ea[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(forw_dx[ti+xpels*j]*forw
_dx[ti+xpels*j]+cent_dy[ti+xpels*j]*cent_dy[ti+xpels*j]));
fa[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(back_dx[ti+xpels*j]*back
_dx[ti+xpels*j]+cent_dy[ti+xpels*j]*cent_dy[ti+xpels*j]));
ga[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(forw_dy[ti+xpels*j]*forw
_dy[ti+xpels*j]+cent_dx[ti+xpels*j]*cent_dx[ti+xpels*j]));
ha[ti+xpels*j]=dt*dirac(sign_d[ti+xpels*j])*w/(h*h*Math.sqrt(back_dy[ti+xpels*j]*back
_dy[ti+xpels*j]+cent_dx[ti+xpels*j]*cent_dx[ti+xpels*j]));
                 }
           for (j=1;j<ypels-1;j++)
            for (i=1;i<xpels-1;i++)
            {

sign_d[ti+xpels*j]=(sign_d[ti+xpels*j]+ea[ti+xpels*j]*sign_d[ti+1+xpels*j]
           +fa[ti+xpels*j]*sign_d[ti-1+xpels*j]+ga[ti+xpels*j]*sign_d[ti+xpels*(j+1)]
           +ha[ti+xpels*j]*sign_d[ti+xpels*(j-1)]+dt*dirac(sign_d[ti+xpels*j])
*(-(intensity[ti+xpels*j]-c1)*(intensity[ti+xpels*j]-c1)+(intensity[ti+xpels*j]-c2)*(intensi
ty[ti+xpels*j]-c2)))
           /(1+ea[ti+xpels*j]+fa[ti+xpels*j]+ga[ti+xpels*j]+ha[ti+xpels*j]);
             }

            for (i=0;i<xpels;i++)
              {
              sign_d[ti]=sign_d[ti+xpels];
             sign_d[ti+xpels*(ypels-1)]=sign_d[ti+xpels*(ypels-2)];
              }
            for (j=0;j<ypels;j++)
              {
              sign_d[t0+xpels*j]=sign_d[t1+xpels*j];
             sign_d[txpels-1+xpels*j]=sign_d[txpels-2+xpels*j];
              }
            if ((Math.floor(t/dt)%n_toreinit == 0) && (t   !=   0))
              reinitial(n_doreinit);
            t=t+dt;
            }
          for (j=0;j<ypels;j++)
            for (i=0;i<xpels;i++)
                area_mapping[ti+xpels*j]=sign_d[ti+xpels*j];
            ea=null;
            fa=null;
            ga=null;
            ha=null;
        }
        void reinitial(int   n)
        {
          int   i, j, k ;
          double[t] grad_d ;

          grad_d = new double[txpels*ypels];
          for (k=1;k<n+1;k++)
              for (j=1;j<ypels-1;j++)
                  for (i=1;i<xpels-1;i++)
                   {
   grad_d[ti+xpels*j]=Math.sqrt(((sign_d[ti+1+xpels*j]-sign_d[ti-1+xpels*j])/(2*h))*((
sign_d[ti+1+xpels*j]-sign_d[ti-1+xpels*j])/(2*h))+((sign_d[ti+xpels*(j+1)]-sign_d[ti+xp
els*(j-1)])/(2*h))*((sign_d[ti+xpels*(j+1)]-sign_d[ti+xpels*(j-1)])/(2*h)));
   sign_d[ti+xpels*j]=sign_d[ti+xpels*j]+dt*(sign(sign_d[ti+xpels*j])*(1-grad_d[ti+xp
els*j]));
                        }
          }
        double sign(double arg1)
        {
          double result;
          if (arg1 < 0)
          result = -1;
          else if (arg1 > 0)
             result = 1.0;
          else
             result = 0.0;
          return result;
        }
         void  adjust_boundary(int  direct,double  step,int    sel_startx,int  sel_starty,int
sel_lastx,int sel_lasty)
          {
             int   i, j ;
             int   x, y ;
             int   x1, x2, y1, y2 ;
             if ((sel_startx > startx))
                x1=sel_startx;
             else
                x1=startx;
             if ((sel_starty > starty))
                y1=sel_starty;
             else
                y1=starty;
             if ((sel_lastx < lastx))
                x2=sel_lastx;
             else
                x2=lastx;
             if ((sel_lasty < lasty))
                y2=sel_lasty;
             else
                y2=lasty;
             x=x2-x2+1;
             y=y2-y1+1;
             for (j=0;j<ypels;j++)
                 for (i=0;i<xpels;i++)
                      area_mapping[ti+xpels*j]=sign_d[t0];
             if (direct == -1)
                {
                     for (j=0;j<y;j++)
                      for (i=0;i<x;i++)
                      {
```

```
            if (sign_d[ti+x1-startx+xpels*(j+y1-starty)] < step)
               area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=-1;
            else
               area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=1;
            }
      }
   else if (direct == 1)
      for (j=0;j<y;j++)
         for (i=0;i<x;i++)

            if (sign_d[ti+x1-startx+xpels*(j+y1-starty)] > step)
               area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=1;
            else
               area_mapping[ti+x1-startx+xpels*(j+y1-starty)]=-1;
 }
double areainfo(int   sel_startx, int sel_starty, int sel_lastx, int sel_lasty)
{
   double result;
    int   i, j ;
    int   x, y ;
    int   x1, x2, y1, y2 ;
    if ((sel_startx > startx))
       x1=sel_startx;
    else
       x1=startx;
    if ((sel_starty > starty))
       y1=sel_starty;
    else
       y1=starty;
    if ((sel_lastx < lastx))
       x2=sel_lastx;
    else
       x2=lastx;
    if ((sel_lasty < lasty))
       y2=sel_lasty;
    else
       y2=lasty;
    x=x2-x1+1;
    y=y2-y1+1;
    result=0;
    for (j=0;j<y;j++)
       for (i=0;i<x;i++)
          {
          if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
             result=Math.round(result+1);
          }
```

```
            return result;
        }
        double area_intensitymean(int   sel_startx, int sel_starty, int sel_lastx, int sel_lasty)
        {
            double result;
            int   i, j ;
            int   x, y ;
            int   x1, x2, y1, y2 ;
            int   n ;
            if ((sel_startx > startx))
               x1=sel_startx;
            else
               x1=startx;
            if ((sel_starty > starty))
               y1=sel_starty;
            else
               y1=starty;
            if ((sel_lastx < lastx))
               x2=sel_lastx;
            else
               x2=lastx;
            if ((sel_lasty < lasty))
               y2=sel_lasty;
            else
               y2=lasty;
            x=x2-x1+1;
            y=y2-y1+1;
            result=0;
            n=0;
            for (j=0;j<y;j++)
                for (i=0;i<x;i++)
                {
                  if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
                     {
                         result=result+intensity[ti+x1-startx+xpels*(j+y1-starty)];
                           n=n+1;
                     }
                  }
            if (n   !=   0)
               result=Math.round(result/n);
            return result;
        }
        void initialize(int   x, int y, int sx1, int sy1, int sx2, int sy2,double [t]
data_intensity)
         {
            int   i, j ;
```

```
  dt=0.1;
  e=1;
  w=0.025*255*255;
  h=1;
  n_toreinit=40;
  n_doreinit=8;
xpels=x;
ypels=y;
startx=sx1;
starty=sy1;
lastx=sx2;
lasty=sy2;
  area_mapping = new double[txpels*ypels];
  sign_d=new double[txpels*ypels];
  forw_dx=new double[txpels*ypels];
  forw_dy=new double[txpels*ypels];
  back_dx=new double[txpels*ypels];
  back_dy=new double[txpels*ypels];
  cent_dx=new double[txpels*ypels];
  cent_dy=new double[txpels*ypels];
  intensity=new double[txpels*ypels];
  initsigned_dist(h,xpels,ypels,4);
  for (j=0; j< ypels;j++)
      for (i=0; i<xpels;i++)
              intensity[ti+xpels*j]=data_intensity[ti+xpels*j];
}

double  area_intensity_75pvalue(int   sel_startx,int  sel_starty,int  sel_lastx,int
sel_lasty)
{
  double result;
  int i,j,k;
  int x,y;
  int x1,x2,y1,y2;
  int n;
  double [t] data;
  if ((sel_startx > startx))
     x1=sel_startx;
  else
     x1=startx;
  if ((sel_starty > starty))
     y1=sel_starty;
  else
     y1=starty;
  if ((sel_lastx < lastx))
     x2=sel_lastx;
```

```
    else
       x2=lastx;
    if ((sel_lasty < lasty))
       y2=sel_lasty;
    else
       y2=lasty;
    x=x2-x1+1;
    y=y2-y1+1;
    n=0;
    for (j=0;j<y;j++)
        for (i=0;i<x;i++)
        {
          if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
          n=n+1;
          }
    k=0;
    if (n   !=   0)
       {
       data=new double[tn];

       for (j=0;j<y;j++)
        for (i=0;i<x;i++)
        {
             if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) > 0)
                {
                data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
                k=k+1;
                }
           }
       quicksort(data);
       result=data[t(int) Math.round(n*0.75)-1];
       }
    else
       {
       data=new double[tx*y];
       for (j=0;j<y;j++)
        for (i=0;i<x;i++)
             {
             data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
             k=k+1;
             }
       quicksort(data);
       result=data[t(int) Math.round(x*y*0.75)-1];
       }
       return result;
  }
```

```
    public static void quicksort(double[t] a) {
     shuffle(a);                                // to guard against worst-case
     quicksort(a, 0, a.length - 1);
}
// quicksort a[tleft] to a[tright]
public static void quicksort(double[t] a, int left, int right) {
     if (right <= left) return;
     int i = partition(a, left, right);
     quicksort(a, left, i-1);
     quicksort(a, i+1, right);
}

// partition a[tleft] to a[tright], assumes left < right
private static int partition(double[t] a, int left, int right) {
     int i = left - 1;
     int j = right;
     while (true) {
          while (a[t++i]<a[tright])        // find item on left to swap
               ;                                     // a[tright] acts as sentinel
          while (a[tright]<a[t--j])       // find item on right to swap
               if (j == left) break;             // don't go out-of-bounds
          if (i >= j) break;                       // check if pointers cross
          exch(a, i, j);                            // swap two elements into place
     }
     exch(a, i, right);                            // swap with partition element
     return i;
}



// exchange a[ti] and a[tj]
private static void exch(double[t] a, int i, int j) {

     double swap = a[ti];
     a[ti] = a[tj];
     a[tj] = swap;
}

// shuffle the array a[t]
private static void shuffle(double[t] a) {
     int N = a.length;
     for (int i = 0; i < N; i++) {
          int r = i + (int) (Math.random() * (N-i));     // between i and N-1
          exch(a, i, r);
     }
    }


        double    background_intensitymedian(int        sel_startx,int    sel_starty,int
sel_lastx,int sel_lasty)
        {
        double result;
        int i,j,k;
        int x,y;
        int x1,x2,y1,y2;
        int n;
        double [t] data;

        if ((sel_startx > startx))
          x1=sel_startx;
        else
          x1=startx;
        if ((sel_starty > starty))
          y1=sel_starty;
        else
          y1=starty;
        if ((sel_lastx < lastx))
          x2=sel_lastx;
        else
          x2=lastx;
        if ((sel_lasty < lasty))
          y2=sel_lasty;
        else
          y2=lasty;

        x=x2-x1+1;
        y=y2-y1+1;

        result=0;
        n=0;
        for (j=0;j<y;j++)
           for (i=0;i<x;i++)
           {

             if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
              n=n+1;
             }

        k=0;
        if (n   !=   0)
          {
            data=new double[tn];
            for (j=0;j<y;j++)
             for (i=0;i<x;i++)
               {

                 if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
                   {
                   data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
                   k=k+1;

                   }
               }
            quicksort(data);
            if ((n%2) == 0)
              result=Math.round(data[tn/2]);
            else
              result=Math.round((data[tMath.round(n/2)]+data[t(n-1)/2])/2);
             }
            return result;
        }

          double   background_intensity_75pvalue(int      sel_startx,int    sel_starty,int
sel_lastx,int sel_lasty)
        {

          double result;
          int i,j,k;
          int x,y;
          int x1,x2,y1,y2;
          int n;
          double [t] data;

          if ((sel_startx > startx))
            x1=sel_startx;
          else
            x1=startx;
          if ((sel_starty > starty))
            y1=sel_starty;
          else
            y1=starty;
          if ((sel_lastx < lastx))
            x2=sel_lastx;
          else
            x2=lastx;
          if ((sel_lasty < lasty))
            y2=sel_lasty;
          else
            y2=lasty;
          x=x2-x1+1;
          y=y2-y1+1;

          n=0;
          for (j=0;j<y;j++)
              for (i=0;i<x;i++)
              {
                if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
                n=n+1;
                }
          k=0;
          if (n   !=   0)
            {
            data=new double[tn];

            for (j=0;j<y;j++)
             for (i=0;i<x;i++)
             {
                   if (sign(area_mapping[ti+x1-startx+xpels*(j+y1-starty)]) <= 0)
                    {
                    data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
                    k=k+1;
                    }
               }
    quicksort(data);
            result=data[t(int) Math.round(n*0.75)-1];
            }
          else
            {
            data=new double[tx*y];
            for (j=0;j<y;j++)
             for (i=0;i<x;i++)
                 {
                 data[tk]=intensity[ti+x1-startx+xpels*(j+y1-starty)];
                 k=k+1;
                 }
            quicksort(data);
            result=data[t(int) Math.round(x*y*0.75)-1];
              }
            return result;
      }
  }
```

REFERENCES

[1] Y. H. Yang, M. J. Buckley, S. Dudoit, and T. P. Speed, "Comparison of methods for image analysis on cDNA microarray data", *Journal of Computational & Graphical Statistics*, Vol. 11, pp.108-136, November 1, 2002.

[2] Eisen Lab. *ScanAlyze software* [Online]. Available: http://rana.lbl.gov/EisenSoftware.htm

[3] Axon Instruments. *GenePix software* [Online]. Available: http://www.axon.com/GN_GenePixSoftware.html

[4] O. Demirkaya, M. H. Asyali, and M. M. Shoukri, "Segmentation of cDNA Microarray Spots Using Markov Random Field Modeling", *Bioinfomatics*, 21(13), pp.2994-3000, 2005.

[5] CSIRO Mathematical and Information Sciences. *Spot software* [Online]. Available: http://www.hca-vision.com/Spot_Documentation/Spot.pdf

[6] R. Adams and L. Bischof, "Seeded Region Growing", *IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Computer Society*, Vol. 16, pp .641–647, 1994.

[7] B. Appleton and H. Talbot, "Globally optimal geodesic active contours", *Journal of Mathematical Imaging and Vision*, Vol. 24, pp.83-130, July 2006.

[8] J. M. Arteaga-Salas, H. Zuzan, W. B. Langdon, G. J. G. Upton, and Andrew P. Harrison. *An overview of image-processing methods for Affymetrix GeneChips* [Online]. Available: http://www.cs.ucl.ac.uk/staff/W.Langdon/ftp/papers/Arteaga-Salas_2008_BiB.pdf

[9] R. Dahm, "Discovering DNA: Friedrich Miescher and the early years of nucleic acid research", *Human genetics*, Vol. 122 (6), pp.565-581, January 2008.

[10] P. A. Levene, "The Structure of Yeast Nucleic Acid. IV. Ammonia Hygrolysis", *Journal of Biological Chemistry*, Vol. 40, pp.415-424, 1919.

[11] A. D. Hershey and M. Chase, "Independent functions of viral protein and nucleic acid in growth of bacteriophage", *The Journal of General Physiology*, Vol. 36,pp. 39-56, 1952.

[12] J. D. Watson and F. H. C. Crick, "A Structure for Deoxyribose Nucleic Acid", *Nature*, Vol. 171, pp.737-738, April 25, 1953.

[13] J. Barciszewski, B. Frederic, and C. Clark. *RNA Biochemistry and Biotechnology*, Springer, 1999.

[14] F. J. Taylor and D. Coates, "The code with the codons", *BioSystems*, Vol. 22 (3), pp.177-187, 1989.

[15] Ghent University. *The Central Dogma of Molecular Biology* [Online]. Available: http://users.ugent.be/~avierstr/principles/centraldogma.html

[16] H. Pearson, "Genetics: What is a gene?", *Nature*, Vol. 441 (7092), pp. 298-401, 2006.

[17] D. Kosman, J. Reinitz, and D. H. Sharp. *Automated Assay of Gene Expression at Cellular Resolution* [Online]. Available: http://www.nslij-genetics.org/microarray/kosman98.pdf

[18] National Center for Biotechnology Information. *Just the Facts: A Basic Introduction to the Science Underlying NCBI Resources* [Online]. Available: http://www.ncbi.nlm.nih.gov/About/primer/microarrays.html

[19] Affymetrix Inc. *DNA microarray prototype* [Online]. Available: http://www.affymetrix.com/corporate/media/image_library/image_library_3.affx?onloadforward=/corporate/media/image_library/high_res/prototype.zip

[20] M. Schena, D. Shalon, R.W. Davis, and P.O. Brown, "Quantitative Monitoring of Gene Expression Patterns with a Complementary DNA Microarray," *Science*, Vol. 270, pp. 467-470, Oct. 1996.

[21] J. Buhler. *Anatomy of a Comparative Gene Experssion Study* [Online]. Available: http://www.cs.wustl.edu/~jbuhler/research/array/

[22] German Cancer Research Center. *Affymetrix Technology* [Online]. Available: http://www.dkfz.de/gpcf/affy_technology.html

[23] Washington State University. *spotSegmentation software* [Online]. Available: http://www.bioconductor.org/packages/bioc/1.6/src/contrib/html/spotSegmentation.html

[24] G. K. Smyth and T. Speed, "Normalization of cDNA microarray data", *Elsevier Science, Methods*, Vol. 31, pp. 265-273, 2003.

[25] W. S. Cleveland and S. J. Devlin, "Locally-Weighted Regression: An Approach to Regression Analysis by Local Fitting", *Journal of the American Statistical Association*, Vol. 83 (403), pp.596-610, 1998.

[26] G. K. Smyth, Y. H. Yang, and T. Speed, "Statistical Issues in cDNA Microarray Data Analysis", *Methods in Molecular Biology*, Vol. 224, pp. 111-136, 2003.

[27] J. A. Hartigan. *Clustering Algorithms*, Wiley, 1975.

[28] D. Meyer, F. Leisch, and K. Hornik, "The Support Vector Machine under Test", *Neurocomputing*, Vol. 55 (1-2), pp.169-186, 2003.

[29] S. Beucher and C. Lantuéjoul, *Use of watersheds in contour detection, Proceedings of the International workshop on image processing, real-time edge and motion detection, pp.17-21, September 1979, France.*

[30] S. Eddins. *The Watershed Transform Strategies for Image Transform* [Online]. Available:http://www.mathworks.com/company/newsletters/news_notes/win02/watershed.html

[31] NHLBI Program for Genomic Applications, Harvard Medical School. *Genomics of Cardiovascular Development, Adaptation, and Remodeling* [Online]. Available: http://www.cardiogenomics.org

[32] Affymetrix Inc. *GeneChip Operating Software* [Online]. Available: http://www.affymetrix.com

[33] T. F. Chan and L. A. Vese, "Active Contours Without Edges", *IEEE Transactions On Image Processing*, Volume 10, Issue 2, pp.266-277, February 2001.

[34] Stanford University. *Yeast Cell Cycle Analysis Project* [Online]. Available: http://genome-www.stanford.edu/cellcycle/data/rawdata/individual.html

[35] B. Song and T. Chan, "A Fast Algorithm for Level Set Based Optimization", *UCLA Computational and Applied Mathematics Reports*, Vol. 02-68, 2002.

[36] S. K. Lele, "Compact Finite Difference Schemes with Spectral-like Resolution", *Journal of Computational Physics*, Vol. 103, pp.16-42, 1992.

[37] D. Amaratunga and J. Cabrera, *Exploration and Analysis of DNA Microarray and Protein Array Data*, Wiley-Interscience, 2004.

[38] A. Ultsch. *Is log ratio a good value for identifying differential expressed genes in microarray experiments?* [Online]. Available: http://www.uni-marburg.de/fb12/datenbionik/pdf/pubs/2003/ultsch03log

[39] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, "Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization", *Molecular Biology of the Cell*, Vol. 9, Issue 12, pp.3273-3297, December 1998.

[40] Affymetrix Inc. GeneChip Expression Analysis Data Analysis Fundamentals, pp38.

[41] Affymterix Inc. *Affymetrix Expression Console Software Version 1.0 - User Guide*, pp98.

[42] J. R. McMullen, T. Shioi, W. Y. Huang, L. Zhang, O. Tarnavski, E. Bisping, M. Schinke, S. Kong, M. C. Sherwood, J. Brown, L. Riggi, P. M. Kang, and S. Izumo, "The insulin-like growth factor 1 receptor induces physiological heart growth via the phosphoinositide 3-kinase(p110 α) pathway", *The Journal of biological chemistry*, Vol. 279, pp.4782-4793, 2004.

[43] J. K. Damas, L. Gullestad, H. Aass, S. Simonsen, J. G. Fjeld, L. Wikeby, T. Ueland, H. G. Eiken, S. S. Froland, and P. Aukrust, "Enhanced Gene Expression of Chemokines and Their Corresponding Receptors in Mononuclear Blood Cells in Chronic Heart Failure—Modulatory Effect of Intravenous Immunoglobulin", *Journal of the American College of Cardiology*, Vol. 38, pp.187-193, 2001.

[44] A. Talvani, M. O. C. Rocha, A. L. Ribeiro, R. C. Oliveria, and M. M. Teixeira, "Chemokine Receptor Expression on the Surface of Peripheral Blood Mononuclear Cells in Chagas Disease", *The Journal of Infectious Diseases*, Vol.189, pp.214-220, 2004.

[45] S. Brokat, J. Thomas, L. R. Herda, C. Knosalla, R. Pregla, M. Brancaccio, F. Accornero, G. Tarone, R. Hetzer, and V. R. Zagrosek, "Altered melusin expression in the hearts of aortic stenosis patients", *European Journal of Heart Failure*, Vol. 9, pp.568-573, 2007.

[46] E. Rydberg, P. Gudmundsson, L. Kennedy, L. Erhardt, and R. Willenheimer, "Left atrioventricular plane displacement but not left ventricular ejection fraction is influenced by the degree of aortic stenosis", *Heart*, Vol. 90, pp.1151-1155, 2004.

[47] D. E. Lanfear, J. M. Stolker, S. Marsh, M. W. Rich, and H. L. McLeod, "Natriuretic peptide receptor 3 genotype modulates the relationship between B type natriuretic peptide and left ventricular end-diastolic pressure", *Therapy*, Vol. 3, No. 6, pp.765-771, 2006.