USING SMOOTHING SPLINES TO SELECT SIGNIFICANT GENES IN MICROARRAYS

USING SMOOTHING SPLINES TO SELECT SIGNIFICANT GENES IN MICROARRAYS

By

Ji Li, B.Eng

A Thesis
Submitted to the School of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree
Master of Science

McMaster University

MASTER OF SCIENCE (2008) McMaster University

(Statistics) Hamilton, Ontario

TITLE:	USING SMOOTHING SPLINES TO SELECT SIGNIFICANT GENES IN MICROARRAYS
AUTHOR:	Ji Li, B.Eng (Shandong University, P.R.China)
SUPERVISOR:	Professor Angelo Canty
NUMBER OF PAGES:	ix, 66

Abstract

DNA microarray technology has been widely used in many applications such as gene discovery, disease research and drug investigation. This thesis is based on a project studying the genetics of Type 1 Diabetes.

In this thesis we introduce a method to use smoothing splines to select significant genes in microarrays. This method is based on significance analysis of microarrays (SAM). We choose upper and lower significance cut-offs based on when the numerical derivative of the spline exceeds a threshold. We declare that any genes whose observed statistics are less than the lower cut-off or greater than the upper cut-off to be significant. We also explain how to use this method to calculate the number of significant genes and estimate false discovery rates.

We use both Affymetrix and Illumina real data sets in our analysis and the results are satisfactory. We try to use the simulation study to test our method but we have a problem that we can not generate simulated data which is similar to the real microarray data.

Acknowledgements

I would like to thank my supervisor, Dr. Angelo Canty, for giving me the opportunity to work on this thesis with him and his great guidance, patience and encouragement. I also thank him for providing his code for the SAM method for me to use in this thesis. I appreciate his supervision and suggestions on the programming.

I would like to thank my husband Kevin, sons Jerry and Jason and my parents for their support and love.

I would like to thank Dr. Peter Macdonald and Dr. Tulay Koru-Sengul for agreeing be my thesis examiners. I appreciate their comments and suggestions.

I also like to thank Dr. Zhenya Ivakine and Dr. Tanya Prasolava supplying the datasets used in this thesis.

I also like to thank Genome Canada and NSERC for funding this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Background

1.1 Background in Genetics

The life process includes a wide array of molecules and macromolecules that determine the structure of the cells. Macromolecules which include Deoxyribonucleic acid (DNA), proteins and polysaccharide dominate most of the activities of life. DNA is a nucleic acid that contains the genetic instructions used in the development and functioning of all known living organisms. The main role of DNA molecules is the long-term storage of information about the construction of macromolecules, allowing them to be made exactly in the accordance with the specifications and needs of the cells.

Genes are the units of the DNA sequence that control the heritable characters of an organism. A gene can be defined as a part of DNA that is destined for a functional RNA. The whole set of genes carried by an individual cell is called its genome. Almost every cell contains a complete copy of the genome in its nucleus.

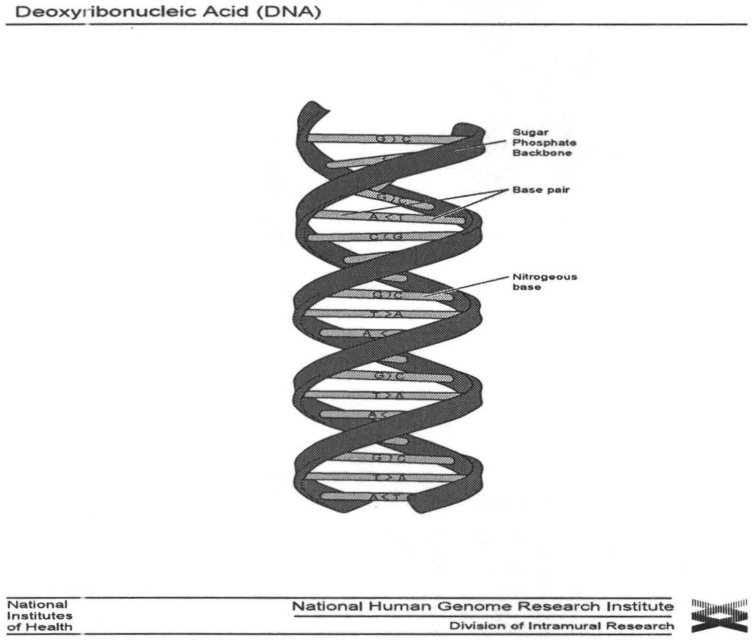

Deoxyribonucleic Acid (DNA)
Sugar Phosphate Backbone
Base pair
Nitrogeous base
National Institutes of Health
National Human Genome Research Institute
Division of Intramural Research

Figure 1.1: *The structure of Deoxyribonucleic Acid (DNA). This graphic is from http://www.accessexcellence.org.*

The genome defines the genetic construction of an organism or cell. The external appearance of an organism is the total set of characteristics displayed by an organism under a specific set of environmental factors. Today we can study the expression of many genes in an organism simultaneously using microarray technology.

DNA is constructed of chains of chemical building blocks called nucleotides. Each nucleotide consists of a phosphate group, a deoxyribose sugar molecule, and one of four different nitrogenous bases: guanine (G), cytosine (C), adenine (A), or thymine (T). The sequence of these nucleotides in DNA controls genetic information. The information stored in the sequence of nucleotides is similar to a long word in a fourletter alphabet. DNA forms a double helix of two chains of nucleotides which run in opposite direction. By Watson-Crick rules G pairs with C only and A pairs with T only (Watson and Crick, 1953). We can see them in Figure 1.1.

The process of transcription is to copy the information encoded in DNA of the genes into RNA which is a single stranded molecule. So RNA has only one nucleotide chain, not a double helix of two chains. Another difference between DNA and RNA is that RNA contains uracil (U) instead of T. That is, the RNA bases are G, C, A and U which make RNA less stable than DNA. Figure 1.2 shows the structure of RNA compared to DNA. RNAs have two general classes, messenger RNA (mRNA) and functional RNA. Messenger RNA works in the translation process and functional RNAs which are the transfer RNAs (tRNA) and the ribosomal RNA (rRNA) are part of the complex protein synthesis machinery which translates mRNA into proteins. The messenger RNA is an exact copy of the DNA coding regions since the sequence of mRNA is identical to one strand of DNA with the replacement of T by U. We can use mRNA analysis to identify polymorphisms in coding regions of DNA and measure gene expression (Lee, 2004). Figure 1.3 shows the processes of replication and transcription.

1.2 Gene Expression and Microarrays

Gene expression is the process by which mRNA and proteins are produced from the DNA of each gene. Gene expression has two stages. One stage is the transcription process of an RNA copying one strand of the DNA. The other stage is the translation process of the mRNA into protein which occurs in the cytoplasm. In the process of gene expression, RNA provides both mRNA and functional RNA (Lee, 2004).

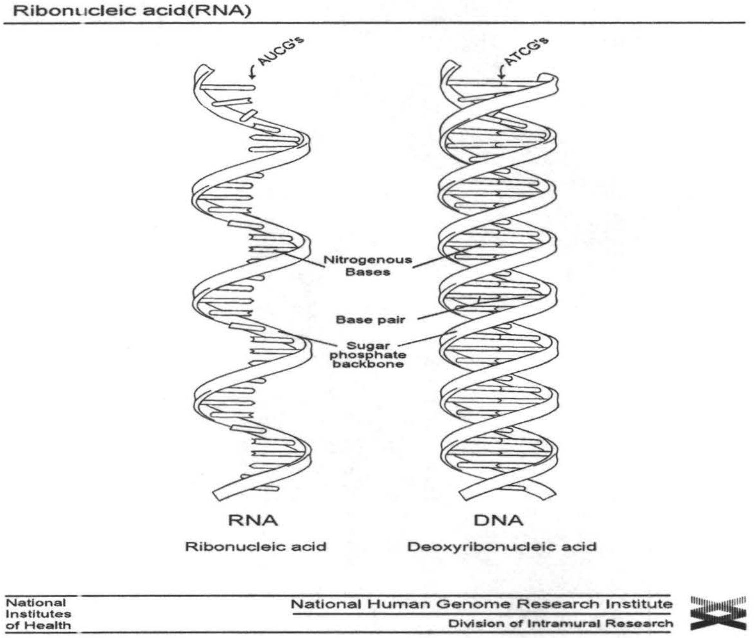


Figure 1.2: *Ribonucleic Acid (RNA).*
This graphic is from http://www.accessexcellence.org.

In the past, scientists conducted genetic analysis on a few genes at a time. With the development of DNA microarray technology, scientists can examine thousands of genes at once, which helps to find the complex relationships between genes.

When a cell is ready to make a certain protein, a segment of the double-stranded DNA, which is responsible for that code, unknots to become two single strands temporarily. The mRNAs make a complementary copy of the coding region of one single strand of DNA. The more a gene is translated into protein, the more copies of mRNA are present inside the cell. DNA microarrays can detect the level of expression of a certain gene by testing for the presence and level of mRNAs associated with that

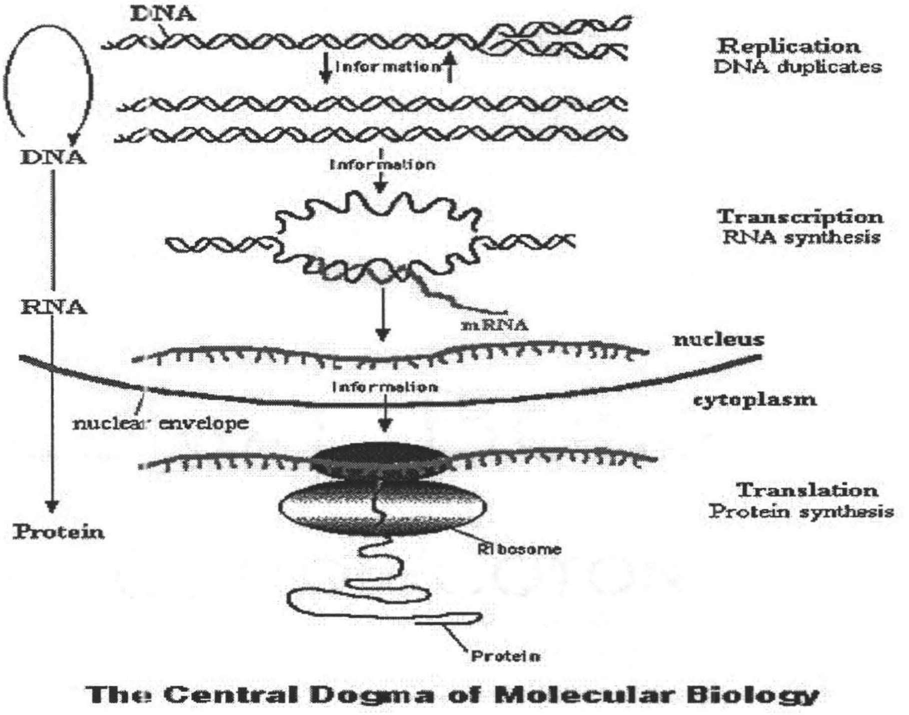


Figure 1.3: *The central Dogma of Molecular Biology.*
This graphic is from http://www.accessexcellence.org.

particular gene. First the researcher collects the mRNA molecules present in the cell and labels each mRNA molecule by attaching a fluorescent dye. Then the researcher places the labeled mRNA onto a DNA microarray slide. The mRNA will hybridize to its complimentary DNA on the microarray. The researcher can use a laser scanner to measure the areas of the fluorescent on the microarray. If a gene is very active, it produces many molecules of mRNA and the fluorescent area is very bright. On the other hand, if a gene is less active, it produces less mRNA and the area of the fluorescent spot is dark. If the gene is inactive, indicating that none of the mRNAs have hybridized to the DNA, the fluorescent area is black. Using DNA microarray technology, the researchers can examine the activity of different genes.

Affymetrix GeneChip (Affy) is the most widely used oligonucleotide array type. It can produce a large amount of chips at a reasonable cost. Affy uses masks which control the synthesis of oligonucleotides on the surface of a chip and of several hundred thousand squares, each of them containing many copies of an oligonucleotide. It produces hundreds of thousands of different oligos, and each of them appear in millions of copies. This large number of oligos are very useful in the experiment. For expression analysis, we use oligos with a length of 25 bases to detect each gene. First Affy chooses a region of each gene that seems to have a unique nucleotide sequence. For this particular region 11 to 20 oligos are chosen to be perfect matches (PM). Affy also generates 11 to 20 mismatch oligos (MM) that are identical to the PM oligos except that the 13th nucleotide is changed to its complementary nucleotide (e.g. G to C or A to T). The hybridization is not uniform since all PM oligos for each gene have different sequences (Knudsen, 2004). These two probes, PM and MM, are called a probe pair. For each probe, arrays are scanned and images are produced and analyzed. An intensity value is obtained to represent how much hybridization occurred for each oligonucleotide. For each probe set, the typical output consists of two vectors of intensity readings, one for PMs and one for MMs.

Affymetrix provides various chips, the MGU74Av2 mouse chip which we use in this thesis is one of these chip types. Each particular chip type can hybridize different arrays. For each of these arrays, millions of molecules of a specific probe are attached on the chip with a very small area ($400\mu m^2$). Each probe is represented by around 100 pixels at a particular location of the image. Finally the software used to process the image stores the location and two statistical summaries, which are mean and standard deviation, for each probe in a file with CEL as the extension. The information mapping probe Id's to locations on the chip are stored in a file with CDF extension. We notice that each chip type has a unique CDF file while each hybridization has a unique CEL file. So if a typical experiment has various hybridized arrays all from the same chip type, only one common CDF file is needed and various CEL files are created to store the complete probe level data (Irizarry *et al.*, 2003b). We will recall this in Chapter 3.

Illumina Inc. recently introduced long-oligonucleotide bead-based array. One difference between Affymetrix and Illumina is the oligonucleotide physical attachment. The oligonucleotides on Illumina BeadChip use a random self-assembly mechanism to be attached to microbeads. The microbeads are then put onto microarrays. Illumina arrays are randomly generated and produce on the order of 30 copies of the same oligonucleotide on the array. Another feature of the Illumina BeadChip is that all different bead types come from a master beads pool and randomly placed onto the wells on the array substrate (Luo, 2007). The Affymetrix arrays are constructed in a specific layout with each probe placed at a predefined location. Another difference between the two platforms is that multiple Illumina arrays are processed in the same way since they are placed on the same physical substrate while Affymetrix arrays are processed separately. Both Affymetrix and Illumina platforms yield highly comparable data, especially for the differentially expressed genes (Barnes *et al.*, 2005). We use both in this thesis.

1.3 Robust Multi-array Average (RMA)

In the analysis of high-density oligonucleotide arrays, we want to know how RNA populations differ in expression in reaction to genetic and environmental differences. Non-biological variation that may have many different effects on the data is also present. The variation could occur during the sample preparation, the manufacture and processing of the arrays such as labeling, hybridization and scanning. It is important to remove sources of variation between arrays of non-biological origin. Normalization is a process for reducing this variation. We use Robust Multi-array Average (RMA) for the normalization in the analysis.

Many expression measures are based on $PM - MM$ with the intention of correcting for non-specific binding and background noise. There are some problems with this approach. MM may be detecting signal as well as non-specific binding. For some probes, changing the middle base does not make a difference. Another problem is subtracting MM adds noise with no obvious gain in bias. The RMA method introduced by Irizarry *et al.* (2003a) uses only background-corrected PM values. For each PM expression, model observed PM as the sum of a signal intensity S_{ij} and a background noise N_{ij}

$$PM_{ij} = S_{ij} + N_{ij}$$

with i representing different arrays, j representing the probe. Here it is assumed that S_{ij} has an exponential distribution, N_{ij} has a normal distribution, and that S_{ij} and N_{ij} are independent. We want to adjust the PM intensities to remove the background effect. Consider $\widetilde{PM}_{ij} = E[S|PM]$ to adjust for background on the raw intensity scale using maximum likelihood. Define $f_{S,PM} = f_S(s)f_N(e_{ij} - s)$.

$$E[S|PM = e_{ij}] = \frac{\int_0^\infty s f_S(s) f_N(e_{ij} - s) ds}{\int_0^\infty f_S(s) f_N(e_{ij} - s) ds} = h(\lambda_i, \mu_i, \sigma_i; e_{ij})$$

Use maximum likelihood to get $\hat{\lambda}_i$, $\hat{\mu}_i$ and $\hat{\sigma}_i$ and estimate $\widetilde{PM}_{ij} = \hat{E}[S|PM = e_{ij}] = h(\hat{\lambda}_i, \hat{\mu}_i, \hat{\sigma}_i; e_{ij})$. Then use quantile normalization on $\widetilde{PM}_{ij}$ and $\log_2$ transform to get PM^*_{ij}. We assume that vast majority of genes should have equal expression on all arrays. The goal of quantile normalization is to make the distribution of probe intensities the same across arrays. The quantile method works well on reducing the between array variances and giving the smallest distance between arrays. Quantile normalization has three steps. First sort the probe intensities of each array. Then compute the mean over the smallest, the second smallest to the largest intensity of each array respectively. Then replace the smallest to the largest intensity of each array by the smallest to the largest value of the mean vector respectively. To obtain an expression measure, we assume that for each probe set, PM^*_{ij} follows a linear model

$$PM^*_{ij} = \alpha_i + \beta_j + \epsilon_{ij}$$

with α_i representing the log scale expression level for the probe set on array i, β_j a probe effect with the assumption that $\sum_j \beta_j = 0$ and ϵ_{ij} representing an independent identically distributed error term with mean 0. To protect against outlier probes we use median polish, a robust procedure, to estimate model parameters. The estimate of α_i is the expression measure for array i (Irizarry *et al.*, 2003a).

1.4 Description of the Data

Our experiment arises from research into Type 1 Diabetes. Type 1 diabetes (T1D) is a complex disease caused by multiple genetic and environmental risk factors. The relative frequency of T1D is quite different between populations, ranging from 0.7/100,000 people per year in Peru to 45/100,000 people per year in Finland. The rate of T1D in Canada is approximately 15/100,000 per year which is the third highest in the world. Scientist have also noticed that the relative frequency of childhood T1D has risen rapidly over the past 50 years in Finland, England and several other countries (Llanos and Libman, 1994).

Recent research has found a number of genetic regions that contribute to T1D susceptibility in mice. Idd4, Idd5 and Idd13 are three of them. We have two parental strains of mice called Non-Obese Resistant (NOR) and Non-Obese Diabetic (NOD). These two strains are identical by descent in 88% of the genome. But the rate that NOD mice get Type 1 Diabetes is 82 - 85% which is much higher than NOR mice (3 - 5%) at the age of six months for female mice. These two strains differ in the regions of Idd4, Idd5 and Idd13 that we mentioned above. Congenic strains are constructed by selective multi-generational inbreeding of these mice and derived from the parental NOD and NOR strains. For example, NOD.NOR-Idd4 strain is identical to the parental NOD strain except for region Idd4 which inherits from the NOR mice. NOR.NOD-Idd5 strain is identical to the parental NOR strain except for region Idd5 which inherits from the NOD mice. Similarly, NOR.NOD-Idd13 strain is identical to the parental NOR strain except for region Idd13 which inherits from the NOD mice. NOR.NOD-Idd5/13 is a double congenic strain which is identical to the parental NOR strain except for two regions, Idd5 and Idd13, which inherit from the NOD mice. We will use these strains in Chapter 3.

1.5 Thesis Outline

In Chapter 2 we review the Significant Analysis of Microarrays method (SAM) for selecting significant genes in microarray data and give some definitions for False Discovery Rate (FDR) and smoothing splines we use in this thesis. In Chapter 3 we give a detailed description of our datasets and method for finding significant genes using smoothing splines. In Chapter 4 we discuss our simulation study and the problems that arose with it. In Chapter 5 we make our conclusion and discuss some areas for future research.

Chapter 2

SAM, FDR and Smoothing Splines

2.1 Significance Analysis of Microarrays (SAM)

Significance analysis of microarrays (SAM) is statistical technique for testing genes for differential expression between two conditions. SAM computes a statistic d_j for each probe set j based on the normalized log-expression values. The permutations are also used in estimating false discovery rates (FDR) and q-values (Storey and Tibshirani, 2003).

SAM selects differentially expressed genes from the microarray experiments using multiple hypothesis testing. In order to do so, we have to execute hypothesis tests on all genes to see if some genes are differentially expressed. In the hypothesis tests, the null hypothesis means that there is no change in expression levels between experimental conditions while the alternative hypothesis is that there are some changes. If there is enough evidence to show the changes, we reject the null hypothesis. We can calculate the probabilities of rejecting the null hypothesis when it is true or false to make the decision.

There are four important steps to test the differential gene expression. The first is to choose a statistic that can be formed for each gene with no relevant information loss. The next step is to calculate the null distributions for the statistics assuming that each gene has a different null distribution. The third step is to choose the rejection region. Both symmetric and one-side rejection regions are acceptable here, but we prefer to use asymmetric rejection regions because we do not know how many differentially expressed genes are in the positive or negative direction. The last step is to control the false positive rate in a reasonable way.

We are interested in finding significant differentially expressed genes. For each gene we define a statistic that is a function of the data (we use t-statistic). Then we set a significance region. The gene is said to be differentially expressed if the statistic lies in the region. Otherwise it is not. Suppose that we have G genes measured on n arrays under two different experimental conditions. Let $\bar{x}_{j1}$ and $\bar{x}_{j2}$ be the average gene expression for gene j under conditions 1 and 2, and let s_j be the pooled standard deviation for gene j.

$$s_j = \sqrt{\left(\frac{1}{n_1} + \frac{1}{n_2}\right) \cdot \frac{\sum_1 (x_{ji} - \bar{x}_{j1})^2 + \sum_2 (x_{ji} - \bar{x}_{j2})^2}{n_1 + n_2 - 2}}$$

.

Here n_k is the number of arrays in condition k, and each summation is taken within its respective group. So the standard (unpaired) t-statistic for differential gene expression is:

$$t_j = \frac{\bar{x}_{j2} - \bar{x}_{j1}}{s_j}$$

We want to compare the values of t_j across all genes. In order to do so, we have to make sure that the distribution of t_j is independent of the gene expression levels. From the function above, we notice that the variance in t_j could be high at low gene expression levels due to small values of s_j. Tusher *et al.* (2001) add a positive constant s_0 to increase the value of the denominator such that the variance of t_j is independent of the gene expression levels. The modified t-statistic is :

$$d_j = \frac{\bar{x}_{j2} - \bar{x}_{j1}}{s_j + s_0} \tag{2.1}$$

where s_0 is chosen to be a certain percentile of the s_j values. Chu *et al.* (2005) give the procedure for calculating s_0:

1. Let r_i be the difference between $\bar{x}_{j1}$ and $\bar{x}_{j2}$, s_i be the pooled standard deviation for gene i, s^α be the α percentile of the s_i, define $d_i^\alpha = r_i/(s_i + s^\alpha)$.

2. Compute the 100 quantiles of the s_i values, denoted by $q_1 < q_2 ... < q_{100}$.

3. For $\alpha \in (0, 0.05, 0.10, ..., 1.0)$, compute $\text{mad}(d_i^\alpha)$ given s_i in $[q_j, q_{j+1})$ for $j = 1, 2, ..., n$, where mad is the median absolute deviation from the median, divided by 0.64. Define this value to be v_j^α. Then compute the coefficient of variation of

the v_j values, using the function

$$cv(\alpha) = \frac{\text{sd}(v_j^\alpha)}{\text{mean}(v_j^\alpha)}$$

4. Choose $\hat{\alpha}$ as the minimum value of $cv(\alpha)$, and compute $\hat{s}_0 = s^{\hat{\alpha}}$. Then $\hat{s}_0$ is used as the fixed value of s_0.

The equation 2.1 is for the situation without day effect. Affy datasets have day effect. Therefore we can not use this equation to calculate d_j. We use a method that takes blocking into account in calculation of d_j. Below is the algorithm:

1. Build a linear model $y_{ij} = \mu_j + \alpha_j \text{day}_{ij} + \beta_j \text{strain}_{ij} + \epsilon_{ij}$, where y_{ij} is a normalized $\log_2$ expression for gene j on array i, day is a factor for day and strain is a factor for strain.

2. Define

$$d_j = \frac{\hat{\beta}_j}{s_j + s_0} \tag{2.2}$$

where $\hat{\beta}_j$ is the estimate of strain effect in the linear model, s_j is the standard error of $\hat{\beta}_j$, s_0 is calculated using the same way given above.

SAM method given by Tusher *et al.* (2001) derives two cut points, $t_1 < t_2$, and uses the rejection rule $d_j < t_1$ or $d_j > t_2$. This can lead to a more powerful test in situations where more genes are overexpressed than underexpressed. The SAM procedure is

1. Compute the ordered statistics $d_{(1)} \leq d_{(2)} \cdots \leq d_{(G)}$.

2. Take B sets of permutations of group labels. For each permutation b compute statistics d_j^{*b} and corresponding order statistics $d_{(1)}^{*b} \leq d_{(2)}^{*b} \cdots \leq d_{(G)}^{*b}$. From

the set of B permutations, estimate the expected order statistics by $\bar{d}_{(j)} = (1/B)\sum_{b=1}^{B} d_{(j)}^{*b}$ for $j = 1, 2, ..., G$.

3. Plot the $\bar{d}_{(j)}$ values versus the $d_{(j)}$. For a fixed Δ, find the largest gene i_1 such that $\bar{d}_{(i_1)} \leq \text{median}(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$ and $d_{(i_1)} - \bar{d}_{(i_1)} \leq -\Delta$. All genes with $d_j \leq d_{(i_1)}$ are called negative significant. In the same way, find the smallest gene i_2 such that $\bar{d}_{(i_2)} \geq \text{median}(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$ and $d_{(i_2)} - \bar{d}_{(i_2)} \geq \Delta$. All genes with $d_j \geq d_{(i_2)}$ are called positive significant.

4. Let $t_1(\Delta) = d_{(i_1)}$ and $t_2(\Delta) = d_{(i_2)}$. If there is no such i_1 exists, we set $t_1(\Delta) = -\infty$ and claim that there is no negative significant gene. If there is no such i_2 exists, we set $t_2(\Delta) = \infty$ and claim that there is no positive significant gene.

Figure 2.1 is the SAM plot of the average order statistics from the expected $\bar{d}_{(i)}$ values against the observed $d_{(i)}$ values using one of our Affymetrix microarray data sets.

2.2 False Discovery Rates (FDR) and q-values

The situation is very complicated when testing multiple hypotheses. Each gene has possible Type I and Type II errors, and the overall error rate is hard to measure. Table 2.1 lists the possible outcomes from G hypothesis tests.

Here V is the number of Type I errors, T is the number of Type II errors and $R = V + T$ is the total number of significant hypotheses. Benjamini and Hochberg (1995) first defined the false discovery rate as

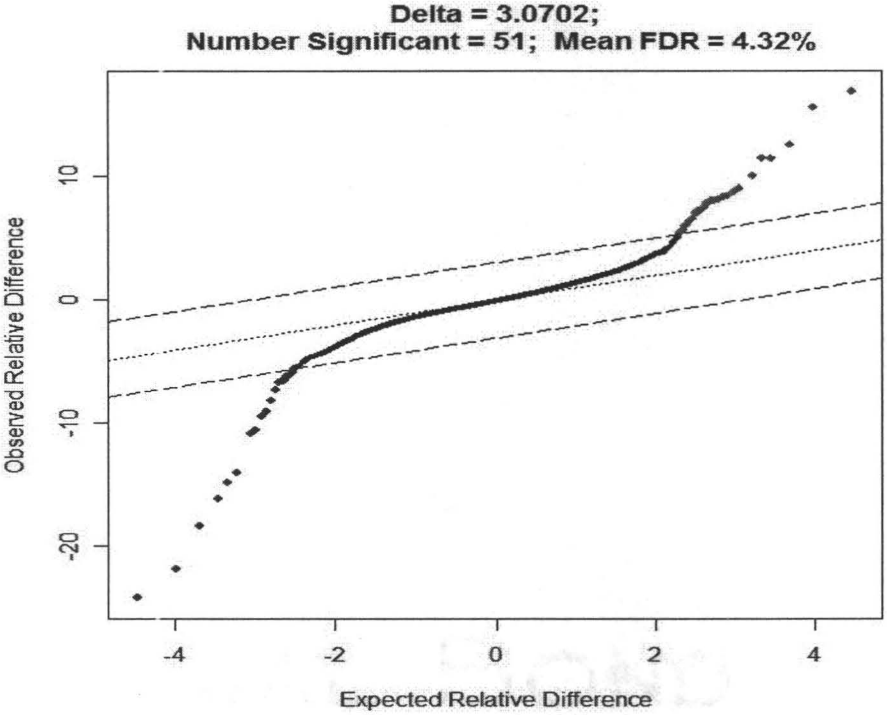


Figure 2.1: *SAM plot given* $\Delta = 3.07$

$$FDR = E\left[\frac{V}{R}|R > 0\right] \cdot Pr(\mathrm{R} > 0)$$

Storey (2002) defines a new false discovery rate, the positive false discovery rate as

$$pFDR = E\left[\frac{V}{R}|R > 0\right]$$

The pFDR conditions on the event that positive findings have occurred. There are two clear approaches to estimating the false discovery rate. The first is to fix the

	Accepted	Rejected	Total
Null True	U	V	m_0
Alternative True	T	S	m_1
Total	W	R	G

Table 2.1: *Possible outcomes from G hypothesis tests*

acceptable rate α beforehand and estimate a significance threshold to obtain this rate on average. The second is to fix the significance threshold and provide a conservative estimate of the rate using that threshold. In our study, a simple estimate of the FDR is the ratio of the average number of significant genes in B permutations and the number of significant genes. But this simple estimate tends to be biased upward. The permutations make all the genes to be non-differentially expressed. But in the data there is a proportion ($\pi_0 < 1$) of non-differentially expressed genes. To improve the estimate of the FDR, we multiply it by the estimate of π_0. We will give the algorithm in Chapter 3.

The q-value for a gene is the lowest FDR at which the gene is called significant and measures how significant the gene is. The q-value is the FDR analogue of the p-value which gives us a hypothesis testing error measure for each observed statistic with respect to pFDR. Storey (2003) gives the definition of the q-value. For each gene g, find the largest value of Δ for which that gene is significant, call this Δ_g, then the q-value is estimated as

$$\hat{q}(g) = \min_{\Delta \leq \Delta_g} \widehat{FDR}(\Delta)$$

2.3 Smoothing Splines

When we draw the SAM plot, we get a scatter plot which is hard to model using traditional parametric techniques. We are interested in the trend of the scatter plot without using any model. Using scatter plot smoothing, we can think that the vertical positions of each point to be a random variable y conditional on x with the value corresponding to the horizontal positions of the point. The trend could be a function such as $f(x) = E[y|x]$ which can be written as $y_i = f(x_i) + \epsilon_i$, $E(\epsilon_i) = 0$ in nonparametric regression. Here the function f is some unspecified smooth function that is estimated from the (x_i, y_i).

In our study, we use cubic smoothing splines. Suppose we have a set of points $(x_1, y_1), ..., (x_n, y_n)$ where the value of x are in increasing order. We use the cubic function f_i to connect the adjacent points (x_i, y_i) and (x_{i+1}, y_{i+1}), $i = 1, ..., G-1$. Then piece them together to get a curve. This curve is described as a cubic spline. The cubic function can be written as

$$f_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

for $x_i \leq x \leq x_{i+1}$, $i = 1, ..., G-1$. The first and second derivatives of the cubic function are continuous but the third derivative may be discontinuous at $x_1, ..., x_G$. A cubic smoothing spline $\hat{f}$ minimizes the penalized residual sum of squares

$$\sum_{i=1}^{G}\{y_i - \hat{f}(x_i)\}^2 + \lambda \int \{\hat{f}''(x)\}^2 dx$$

where $\lambda > 0$ is the smoothing parameter and $\hat{f}''(x)$ is the second derivative of

$\hat{f}(x)$. The first term measures the fit to the data while the second term penalizes curvature in $\hat{f}$. The smoothing parameter λ controls the relative weight of the two parts in the equation. Large values of λ produce smoother curves while small values of λ produce more wiggly curves. With $\lambda \to \infty$, the spline line approaches the least squares line. With $\lambda \to 0$, the bias decreases while variance increases. If $\lambda = 0$, the spline fit connects all the data points together. The R function *smooth.spline* uses cross-validation to calculate λ.

Cross-validation works by leaving the points (x_i, y_i) out one at a time and estimating the fit at x_i based on the remaining $G-1$ points. For $i = 1, ..., G$, we get G numbers of fits at x_i computed by taking out the ith point, denoted by $\hat{f}_{-i}(x_i; \lambda)$. We calculate the squares of the error between y_i and $\hat{f}_{-i}(x_i; \lambda)$ and get the sum of squares after we have done all the points.

The cross-validation sum of squares is constructed as

$$CV(\lambda) = \sum_{i=1}^{G} \{y_i - \hat{f}_{-i}(x_i; \lambda)\}^2$$

We compute $CV(\lambda)$ for a number of values of λ and select $\hat{\lambda}$ that minimize $CV(\lambda)$.

Recall the linear regression that a fitted model can be written as $\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$, where $\hat{\beta} = (\mathbf{X^T X})^{-1}\mathbf{X^T y}$. Then $\hat{\mathbf{y}}$ can be written as $\hat{\mathbf{y}} = \mathbf{Hy}$ where $\mathbf{H} = \mathbf{X(X^T X)^{-1} X^T}$ is known as the hat matrix. Similarly, for a smoothing parameter λ, the fitted values can be written as $\hat{\mathbf{y}} = \mathbf{S}_\lambda \mathbf{y}$, where $\mathbf{y}$ is the $G \times 1$ vector of observed responses y_i, $\hat{\mathbf{y}}$ is the $G \times 1$ vector of fitted values $\hat{y}_i = \hat{f}_\lambda(x_i)$, $\mathbf{S}_\lambda$ is a $G \times G$ matrix that depends on λ, called a smoother matrix. One definition of degrees of freedom for a linear smoother is $\mathrm{tr}(\mathbf{S}_\lambda)$, which is the sum of the diagonal elements of $\mathbf{S}_\lambda$. There is a monotonically decreasing relationship between λ and the degrees of freedom of a smoother. For each value of $\lambda > 0$, the degrees of freedom is unique (Ruppert *et al.,* 2003).

In our method, we use the degrees of freedom to do the analysis instead of λ. We rewrite the cross-validation sum of squares as

$$CV(v) = \sum_{i=1}^{G} \{y_i - \hat{f}_{-i}(x_i; v)\}^2$$

Here v is the degrees of freedom of a smoother. The value of $CV(v)$ is big for small value of v since it leads to a worse fitting spline. Large value of v leads to a too good fitting, which means $\hat{f}(x) \rightarrow y_i$. The difference between y_i and $\hat{f}(x_i; v)$ could be very small while the difference between y_i and $\hat{f}_{-i}(x_i; v)$ could be large since $\hat{f}_{-i}(x_i; v)$ does not involve (x_i, y_i) in the fitting. We can see that the plot of v against $CV(v)$ is U-shaped. We also can find $\hat{v}$ that minimize $CV(v)$. In our case, $x_i = \bar{d}_{(i)}$, $y_i = d_{(i)}$.

2.4 Permutation Tests

In our study, we want to select the genes that show a statistically significant difference in gene expression between two conditions. To get the p-value for a test of significance, we estimate the sampling distribution of the test statistic when the null hypothesis holds by permutation resampling. Permutation resamples are drawn without replacement and in a way that is consistent with the null hypothesis. A permutation test is a type of statistical significance test in which a reference distribution is obtained by calculating all possible values of the test statistic under rearrangements of the labels on the observed data. An important assumption behind a permutation test is that the observations are exchangeable under the null hypothesis.

For our Affymetrix dataset, we have 4 arrays for strain 1 and 4 arrays for strain 2 with day effect. The strain vector is (1 1 2 2 1 1 2 2) and the day vector is (1 1 1 1 2 2 2 2). For probe set j, the observations can be written as (x_{j11} x_{j11} x_{j21} x_{j21} x_{j12} x_{j12} x_{j22} x_{j22}). Here x_{j11} is for strain 1 in day 1, x_{j21} is for strain 2 in day 1 and so on. For probe set j, the null hypothesis is $\mu_{j1} = \mu_{j2}$. The total number of the null hypothesis is G since there are G probe sets in each array. The permutation matrix is a $G \times 8$ matrix. We permute the vector of condition labels. Since day effect could change the distribution of gene expression, the observations are not exchangeable between days. The permutations are done within day. Under the null hypothesis, the first four columns are exchangeable and the last four columns are exchangeable for the permutation matrix. The number B of the set of permutations is 36. For each permutation, compute the statistics and the corresponding order statistics. We repeat the calculation of the statistics for each new label for B times. Then we calculate the average order statistics from the permutations $(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$.

For our Illumina dataset, we have 4 arrays for strain 1 and 4 arrays for strain 2 without day effect. For probe set j, the observations can be written as (x_{j1} x_{j1} x_{j2} x_{j2} x_{j1} x_{j1} x_{j2} x_{j2}). Here x_{j1} is for strain 1 and x_{j2} is for strain 2. For probe set j, the null hypothesis is $\mu_{j1} = \mu_{j2}$. The total number of the null hypothesis is G. Under the null hypothesis, all the columns of the permutation matrix are exchangeable. The number of the set of permutations B is 70. Then we calculate the average order statistics from the permutations $(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$ as above.

Chapter 3

Select Significant Genes Using Smoothing Splines

3.1 Objective

We want to test for genes with differential expression between two conditions with or without day effect. When we refer to significant genes in the thesis, we mean those genes that show a statistically significant difference in gene expression between two conditions. In the usual SAM method, we have to select a value of Δ to get the significance region based on the distance from the line of slope 1 and calculate the number of significant genes. The problem is how to choose an appropriate Δ. The most common way is to pick an arbitrary constant which gives a reasonable $\widehat{FDR}$.

We notice that the significant genes are located in the tails of the SAM plot which are very far away from the line with slope 1. Our method is based on finding the points where the slope of the plot is much greater than 1. Our objective is to detect the significant genes based on the derivative of the smoothed SAM plot and choose the significance upper and lower cut-offs directly from the derivative. We then declare the significant genes based on these cut-offs. We will describe this method statistically in the later sections.

3.2 Dataset Description

We use three datasets in this thesis which are real datasets. Two of them are Affymetrix datasets coming from the experiments using mice. Another is Illumina dataset. We name the two Affy datasets as Zhenya and Tanya and Illumina dataset as illutanya. Dataset Zhenya has three strains consisting of NOD, NOR and NOD.NOR-Idd4. The experiment is completed in two days. There are two replicates for each strain on each day (day 1 and day 2) and for each sex (Male and Female) with 24 arrays in total. The Affymetrix Gene Chip MGU74Av2 has 12488 probe sets. Dataset Tanya has four strains consisting of NOR, NOR.NOD-Idd5, NOR.NOD-Idd13 and NOR.NOD-Idd5/13. The experiment is completed in two days with three replicates for each strain in day 1 and two replicates in day 2. There are 20 arrays in total. For the Illumina dataset, there are 5 strains consisting of NOR, NOD, NOR.NOD-Idd5, NOR.NOD-Idd13 and NOR.NOD-Idd5/13. The experiment has four replicates for each strain and 46628 probe sets without day effect.

3.3 Initial Analysis

We first read the data from the files. Tables 3.1 and 3.2 give the descriptions of two conditions for each subdataset I used in the thesis. For example, Zhenya2 dataset tests for a sex effect within the strain NOD from Affymetrix MGU74Av2 chip with 12488 probe sets. The data frame of dataset Zhenya2 for example with samples as the rows and the phenotypic variables as the columns is given in Table 3.3.

Names of Illumina datasets	two conditions
illutanya1	NOR vs NOR.NOD-Idd5
illutanya2	NOR vs NOR.NOD-Idd13
illutanya3	NOR vs NOR.NOD-Idd5/13
illutanya4	NOR.NOD-Idd5 vs NOR.NOD-Idd5/13
illutanya5	NOR.NOD-Idd13 vs NOR.NOD-Idd5/13
illutanya6	NOD vs NOR

Table 3.1: *Description for Illumina subdataset*

To perform gene expression analysis, we need to summarize the probe set data available for each gene into one expression measure. For Affy dataset, we use RMA to do the normalization. For Illumina dataset, we use quantile normalization of the mean expression for a probe to do the normalization.

Our new method is called SAMSPLINE. This method combines SAM and smoothing spline methods together and gives an easy way to select significant genes without using Δ. I will describe this method below.

First of all, we need to calculate some statistics we will use in the future steps. The main algorithm here is based on SAM. We get the observed order statistics

Names of Affymetrix datasets	two conditions
Zhenya1	Strain NOD vs NOR for Male
Zhenya2	Male vs Female for Strain NOD
Zhenya3	Stain NOD vs NOD.NOR-Idd4 for Male
Zhenya4	Male vs Female for Strain NOR
Zhenya5	Stain NOD vs NOD.NOR-Idd4 for Female
Zhenya6	Strain NOD vs NOR for Female
Zhenya7	Male vs Female for Strain NOD.NOR-Idd4
Tanya1	Strain NOR vs NOR.NOD-Idd5
Tanya2	Strain NOR vs NOR.NOD-Idd13
Tanya3	Strain NOR vs NOR.NOD-Idd5/13
Tanya4	Strain NOR.NOD-Idd5 vs NOR.NOD-Idd5/13
Tanya5	Strain NOR.NOD-Idd13 vs NOR.NOD-Idd5/13

Table 3.2: *Description for Affymetrix subdataset*

$(d_{(1)}, ..., d_{(G)})$ following the equations 2.1 and 2.2 for the situations without or with day effect respectively. Next we calculate the average order statistics $(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$ from the permutations. Since we have day effect as block for our Affy datasets, the permutations are done within blocks. The SAM plot is plotting the average order statistics from the permutations $(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$ against the observed order statistics $(d_{(1)}, ..., d_{(G)})$.

Another parameter is $\hat{\pi}_0$. $\hat{\pi}_0$ is an estimate of the proportion of unaffected genes in the dataset. We compute the 25% and 75% points of the permuted d^* values first, denoted as $q25$ and $q75$. Compute $\hat{\pi}_0 = \min(\#\{d_i \in (q25, q75)\}/(0.5G), 1)$. Here the d_i are the values for the original dataset and there are G such d_i values (Chu *et al.*,

sample	Strain	Block(day effect)
$U74Av2\_121003\_EI01T\_LH.CEL$	1	1
$U74Av2\_121003\_EI02T\_LH.CEL$	1	1
$U74Av2\_121003\_EI03T\_LH.CEL$	2	1
$U74Av2\_121003\_EI04T\_LH.CEL$	2	1
$U74Av2\_121203\_EI01T\_LH.CEL$	1	2
$U74Av2\_121203\_EI02T\_LH.CEL$	1	2
$U74Av2\_121203\_EI03T\_LH.CEL$	2	2
$U74Av2\_121203\_EI04T\_LH.CEL$	2	2

Table 3.3: *Data frame for dataset Zhenya2*

2005).

We use a cubic smoothing spline to smooth the SAM plot and find the numerical derivative of the smoothing spline over an equally spaced grid between $(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$ denoted as $(\tilde{x}_1, ..., \tilde{x}_N)$. We select a cut off value for the derivative (say 3) and draw a horizontal line. We find out that the horizontal line has at least one cross point with the derivative plot. If the cross point is on the negative side of the derivative plot, we find the first value i_1 whose expected order statistic is just smaller than the cross point and set $t_1 = d_{(i_1)}$. If the cross point is on the positive side of the derivative plot, we find the first value i_2 whose expected order statistic is just bigger than the cross point and set $t_2 = d_{(i_2)}$. We define all the genes whose observed statistics are less than t_1 to be negative significant and all the genes whose observed statistics are greater than t_2 to be positive significant. The number of significant genes is the sum of negative and positive significant genes.

Another very important result we want to get from a dataset is the estimated FDR.

We now can use the values of $\hat{\pi}_0$, the upper and lower significance cut-offs which were computed earlier to get $\widehat{FDR}$. Let d^* be the $G \times B$ permuted statistics and d be the G observed statistics. Denote $N_1 =$ Number of $\{d_j : d_j \geq t_2 \text{ or } d_j \leq t_1\}$ be the number of significant genes and $N_0 = B^{-1}\times$ Number of $\{d_j^* : d_j^* \geq t_2 \text{ or } d_j^* \leq t_1\}$ be the average number of significant genes in the B permutations. The FDR can be estimated by

$$\widehat{FDR} = \hat{\pi}_0 \frac{N_0}{N_1}$$

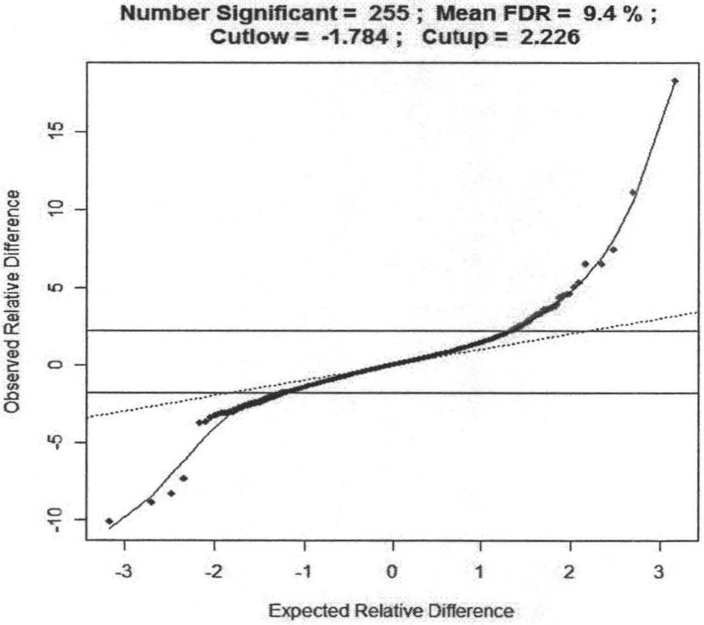


Figure 3.1: *The output of function SAM.plot*

Figure 3.1 shows the output including the SAM plot, number of significant genes, $\widehat{FDR}$ and the values of the upper and lower significance cut-offs. The two horizontal lines are the values of the upper and lower significance cut-offs. The dots that are above the upper line represent the positive significant genes and the dots below the lower line are the negative significant genes. The line through the dots is the smooth spline line.

All these calculations seem to work fine in our method. But we found a problem in the output. The problem is that the cross-validation degrees of freedom (CVdf) which is the trace of the smoother matrix is very high. For example, the CVdf for dataset Zhanya2 is 65.1. We mentioned in Chapter 2 that CVdf minimizes $CV(v)$ such that we can get a fairly good fitting and smooth spline. In our case we want not only a smooth curve but also a smooth first derivative plot. We take a look at Figure 3.2 for dataset Zhenya2. In Figure 3.2, the degrees of freedom of the upper plot is 5 which is small. The spline line does not fit the data very well although the derivative is very smooth. The lower plot is for degrees of freedom equals to CVdf 65.1. The spline line fits the data very well but the derivative plot is wiggly. We want the spline line to be a good fit and on the other hand we need a smooth derivative plot. We need to find a way to balance these two considerations to get a satisfactory result.

We create an objective function to optimize the degrees of freedom. Let $\hat{f}_v$ be the fitted spline with v degrees of freedom and $r_i(v) = d_{(i)} - \hat{f}_v(d_{(i)})$ be the residuals. At first we define the objective function to be

$$g(v) = \sum_{i=1}^{G} |r_i(v)| + \sum_{j=1}^{N} |\hat{f}_v'''(\tilde{x}_j)|$$

Here $(\tilde{x}_1, ..., \tilde{x}_N)$ is an equally spaced grid between $(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$. When we use this objective function to optimize the degrees of freedom, we found the optimized degrees of freedom is always equals to 2. Figure 3.3 shows the plot for degrees freedom against the objective value for dataset Zhenya2.

We then plot the degrees of freedom against the sum of the absolute residuals (upper) and the sum of the absolute fitted 3rd derivatives (lower) separately shown in Figure 3.4 to see the relative sizes of each sum in the objective function. We notice that the scale of sum of the absolute residuals is much smaller than the scale of sum of the absolute fitted 3rd derivatives. When we plot the degrees of freedoms against the object function, we almost get the lower plot. The minimum value for the lower plot is at 2 degrees of freedom. That is the reason we get the optimized degrees of freedom at 2 every time. We found out that the shape of the upper plot is monotonically decreasing and the shape of the lower plot is monotonically increasing. The maximum value for the upper is at 2 degrees of freedom and the maximum value for the lower plot is at CVdf. We had the idea that we can rescale the sum of the absolute fitted 3rd derivatives by dividing by a constant C.

We define C as

$$C = \frac{\sum_{j=1}^{N} |\hat{f}_{cv}'''(\tilde{x}_j)|}{\sum_{i=1}^{G} |r_i(2)|}$$

Now the new objective function is

$$g(v) = \sum_{i=1}^{G} |r_i(v)| + \frac{1}{C} \sum_{j=1}^{N} |\hat{f}_v'''(\tilde{x}_j)|$$

Then we plot the degrees of freedom against the sum of the absolute residuals (upper) and the sum of the absolute fitted 3rd derivatives divided by C (lower) separately shown in Figure 3.5 and find out that the scales for both sums are roughly the same. When we add these two monotonically decreasing and increasing plots with similar scales together, we get a function with unique interior minimum value, shown in Figure 3.6. Then the optimized degrees of freedom is the one that minimize $g(v)$.

We use this objective function to get the optimized degrees of freedom. For Zhenya2, the optimized degrees of freedom is 16.41 shown in Figure 3.6. We can see the advantage of optimizing the degrees of freedom by comparing Figure 3.2 and Figure 3.7. The spline line fitting the data in Figure 3.7 is better than the one in Figure 3.2 with $df = 5$ and the derivative is smoother than the one in Figure 3.2 with $df = 65.1$. We think that the degrees of freedom obtained by minimizing this objective function works well for balancing the fit of the spline to the data and smoothness of the first derivative plot. In our study, we only use one method to define the constant C. For different C, the optimized degrees of freedom could be different. We will discuss this further in Chapter 5.

We use the minimum degrees of freedom to the samspline function. Table 3.5 shows the output of 12 comparisons for Affy datasets and Table 3.4 shows the output of 6 comparisons for Illumina datasets. As we see from Tables 3.4 and 3.5, almost all the minimum degrees of freedom are reasonably small compared to the CVdf which means the objective function works well in finding a suitable degrees of freedom for all the datasets. The FDRs are in the reasonable range too. The figures for most of the real datasets look similar to Figure 3.7. But we still get some figures like Figures 3.8 and 3.9. Our guess is that this is due to the dataset itself. For some datasets, it is possible to have none or a very large number of significant genes.

data set	dfmin	Num.Sig	$\widehat{FDR}$	C	CVdf
illutanya1	15.34	86	1	309.29	71.34
illutanya2	14.93	120	1	322.83	66.297
illutanya3	12.33	210	0.3668	43.66	71.857
illutanya4	10.26	67	0.3697	42.37	70.313
illutanya5	10.88	24	1	47.65	73.452
illutanya6	18.66	698	0.3807	10.81	65.447

Table 3.4: *Partial output for Illumina data sets with cutoff=3*

data set	dfmin	Num.Sig	$\widehat{FDR}$	C	CVdf
Zhenya1	12.84	226	0.06099	66.16	72.60
Zhenya2	16.41	42	0.3934	234.44	65.10
Zhenya3	17.52	77	0.05188	178.35	79.39
Zhenya4	12.82	38	0.2921	86.69	63.65
Zhenya5	12.66	98	0.06362	147.36	66.64
Zhenya6	13.38	207	0.2117	57.16	61.64
Zhenya7	24.12	17	0.1438	101.33	70.13
Tanya1	17.77	1	0.01878	97.93	99.87
Tanya2	21.44	1	0.01285	44.10	101.73
Tanya3	14.65	28	0.06599	83.93	81.38
Tanya4	21.53	21	0.1524	446.77	70.00
Tanya5	24.05	5	0.2005	567.14	85.25

Table 3.5: *Partial output for Affy data sets with cutoff=3*

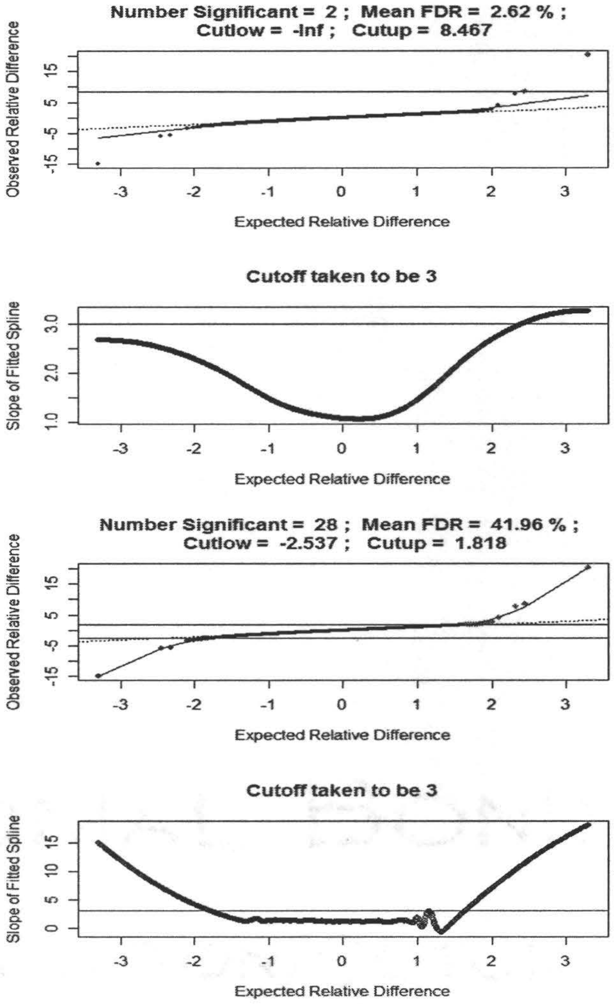


Figure 3.2: *The SAM and derivative plots for Zhenya2 with df=5 and df=65.1.*

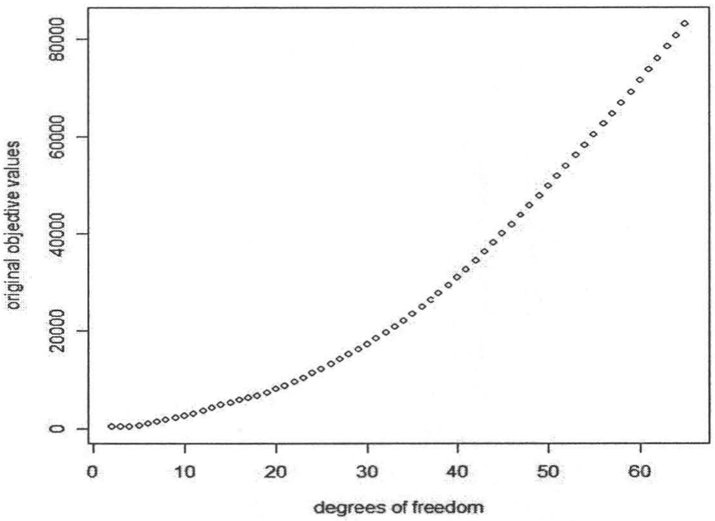


Figure 3.3: *The plot of degrees freedom against the original objective function for dataset Zhenya2*

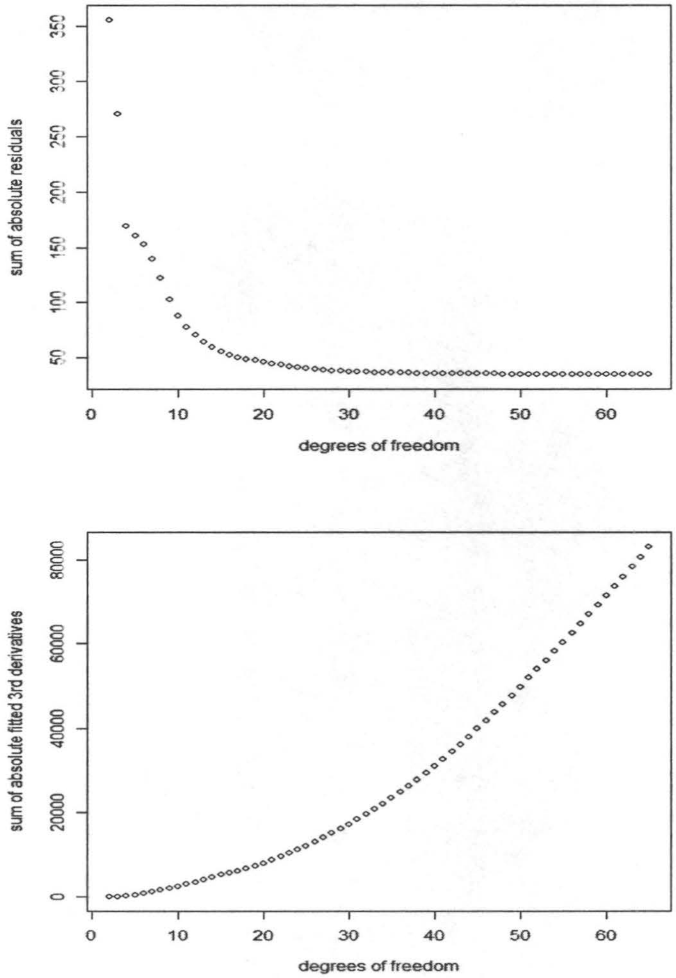
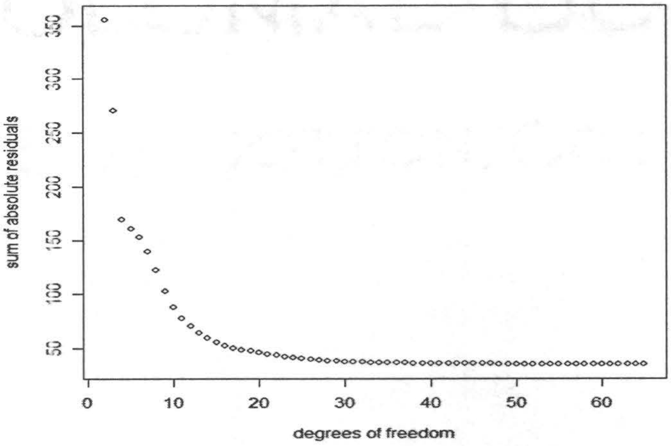


Figure 3.4: *The upper plot is the degrees freedom against sum of absolute residuals and the lower plot is the degrees freedom against sum of absolute fitted 3rd derivatives*



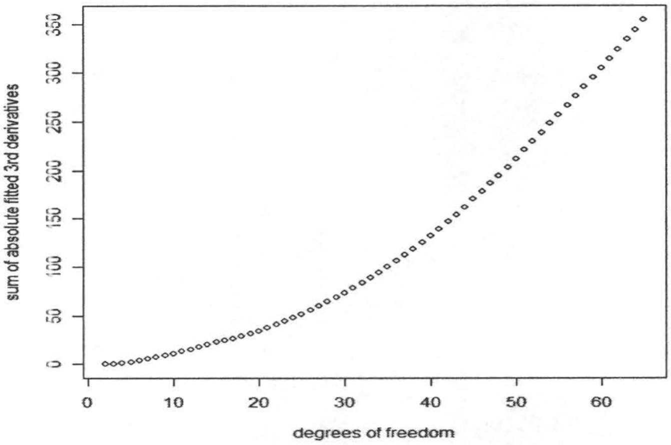


Figure 3.5: *The upper plot is the degrees freedom against sum of absolute residuals and the lower plot is the degrees freedom against sum of absolute fitted 3rd derivatives*

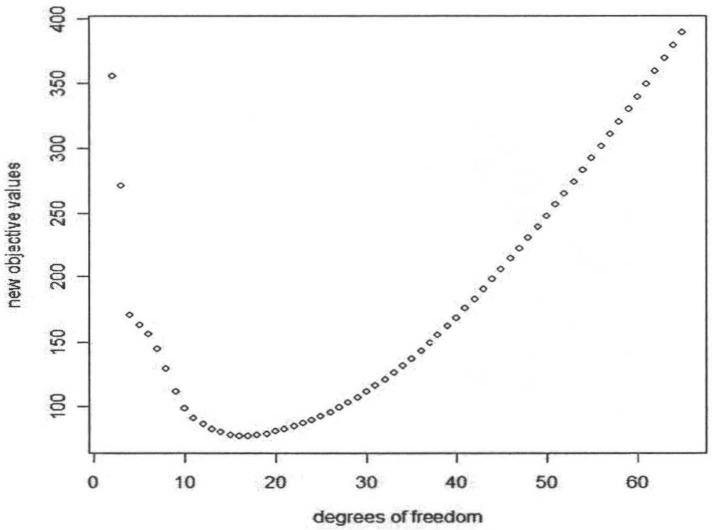


Figure 3.6: *The plot of the degrees freedom against the output of new objective function for dataset Zhenya2*

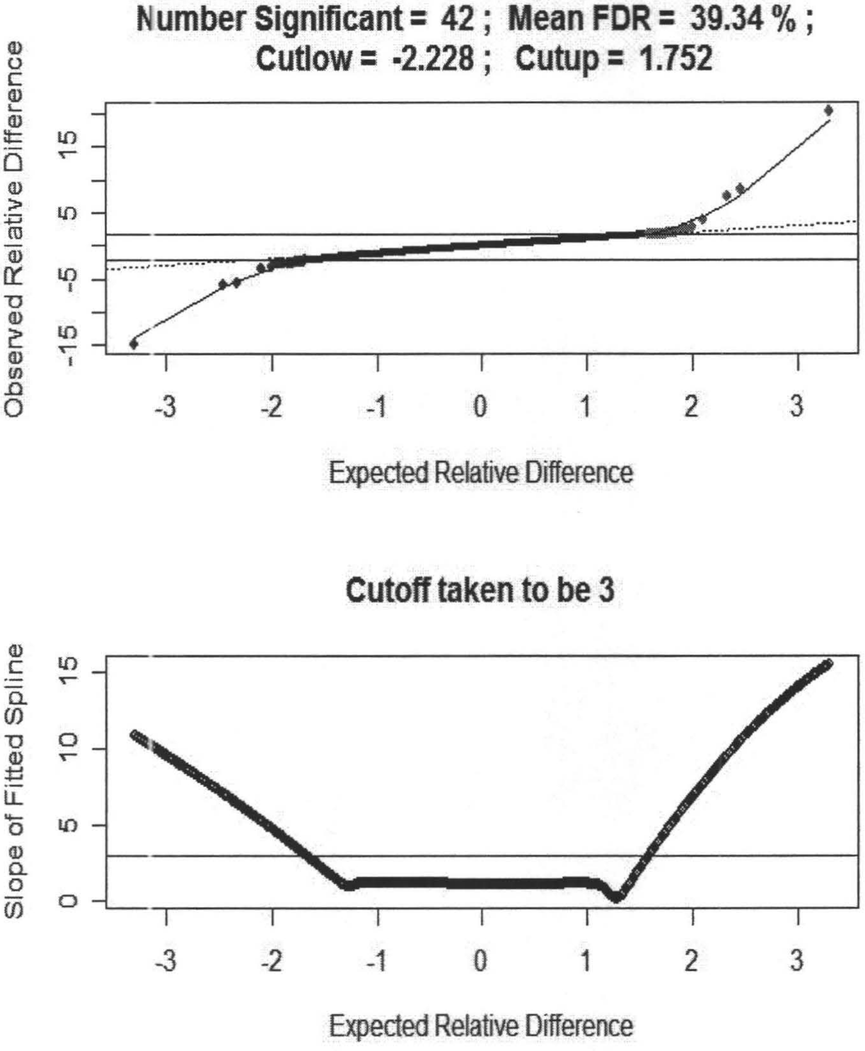
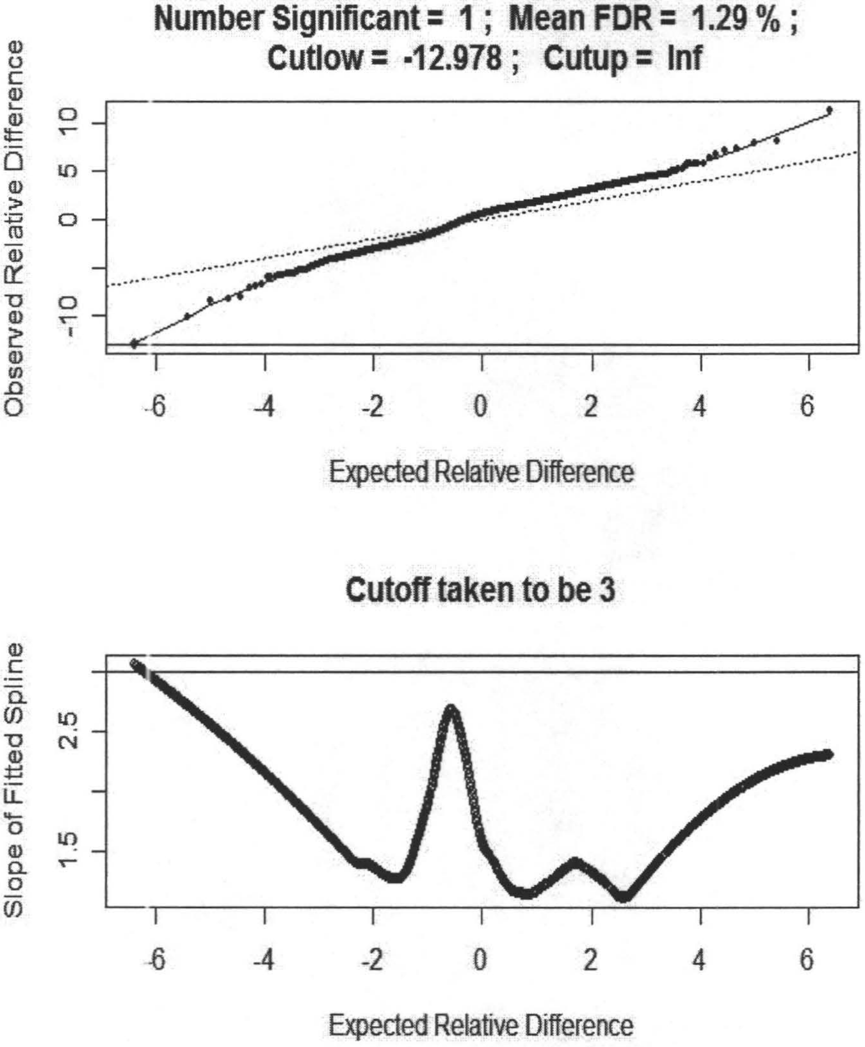


Figure 3.7: *The output of function samspline for data set Zhenya2 with minimum degrees of freedom*



Number Significant = 1 ; Mean FDR = 1.29 % ;
Cutlow = -12.978 ; Cutup = Inf

Cutoff taken to be 3

Figure 3.8: *The output of function samspline for data set Tanya2*

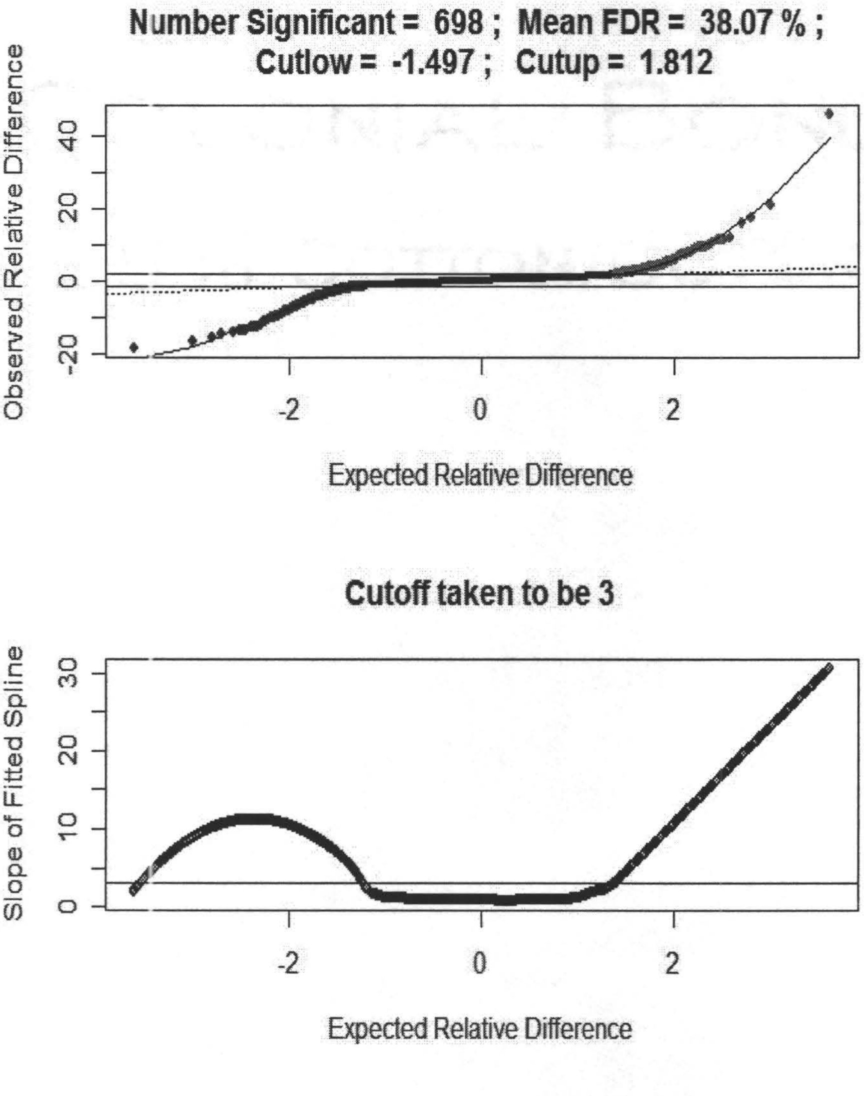


Figure 3.9: *The output of function samspline for Illumina data set illutanya6*

Chapter 4

Simulation

We are interested in significant genes. Using the method above, we did get some significant genes. But we have no idea if they are really significant or not. We wish to use a simulation study to test our method.

First we generate the data set. We use two ways to generate the data set. One is using a t distribution with 3 degrees of freedom and the other is using a standard normal distribution. For each distribution, we set two strains, strain 1 and strain 2 with 5 replicates each. Each sample has $G = 10000$ genes. For strain 1, the means for the observations are μ_i and the standard deviations are σ_i, $i = 1, ..., G$, where $\mu_i \sim$ Normal $(5, 1)$ and $\sigma_i^2 \sim \chi_1^2$. For strain 2, 95% of the observations have mean μ_i and the standard deviations σ_i while 5% of them have mean $\mu_i + d_i$ and the standard deviations σ_i, where half of d_i are $-3\sigma_i$ and half are $3\sigma_i$. These 5% of the observations are differentially expressed marked by row numbers from 1 to G_0. The R code to generate simulated data is given in Appendix E. Since we know which genes are truly differentially expressed, we can calculate the real FDR to see if our method is accurate.

We define the real FDR as the ratio of the number of false significant genes and the total number of significant genes. After we get the numbers of significant genes, we calculate how many row numbers of the significant genes are greater than G_0, these are false significant genes. The real FDR is this number divided by the total number of significant genes.

The results below are based on a small simulation with 5 replicates and 100 permutations. Table 4.1 lists the results for data simulated from a t distribution with 3 degrees of freedom which is rescaled to have standard deviation equal to σ_i. with $G = 10000$ and $G_0 = 500$. For this data, we first set the cut off value equals to 3 and found out the numbers of significant genes ranged from 700 to 800, which are much higher than the number of truly significant genes and hence the real FDRs are very high (i.e. 0.4). The estimated FDRs are extremely small (i.e. 0.03) which means they badly underestimate the truth. Even when we increase the cut off value from 3 to 4, the numbers of significant genes and the real FDR are still bigger than the number of truly significant genes but lower than the ones with cut off value equal to 3. We also find out that over 95% of the truly significant genes are included in the genes calculated to be significant. This shows that our method is working well on selecting significant genes. But also it selects some genes which are not significant to be significant. Figure 4.1 shows the plots for t distribution simulated data. The upper one is the plot with the cut off value equal to 3 and the lower one with the cut off value equal to 4. We notice that the SAM plot is similar to the real microarray data but the derivative plot is different.

We think the FDR estimation problem may be caused by the t distribution since it is a heavy tailed distribution. To test this we also use the normal distribution to generate the simulated data. Table 4.2 shows the partial results for $G = 10000$ with $G_0 = 500$ and $G = 20000$ with $G_0 = 1000$. We notice that the real FDR are still high. We increase the gene numbers from 10000 to 20000, the results are similar. Figure 4.2 shows the plots for normal distributed simulation data with $G = 10000$ and 20000 respectively. We can see that the derivative plot is quite different from the ones using real data, especially in the tails. For most of the real datasets, the tails of the derivative plots are going up not down. We conclude that since we can not get the simulated data similar to the real data, our simulation study is not successful. We conclude that further research is needed to generate simulated data which mimics real microarray data.

	cutoff=3			cutoff=4		
dfmin	Num.Sig	$\widehat{FDR}$	RealFDR	Num.Sig	$\widehat{FDR}$	RealFDR
16.41	839	0.0367	0.4291	662	0.0185	0.2885
19.38	800	0.0302	0.4087	645	0.0167	0.2791
13.98	872	0.0446	0.4426	668	0.0221	0.2889
15.38	814	0.0345	0.4140	633	0.0179	0.2669
16.53	787	0.0301	0.3901	612	0.0156	0.2451

Table 4.1: *Simulation study for t distribution with* $G = 10000$ *and* $G_0 = 500$ *for cutoff=3 and 4*

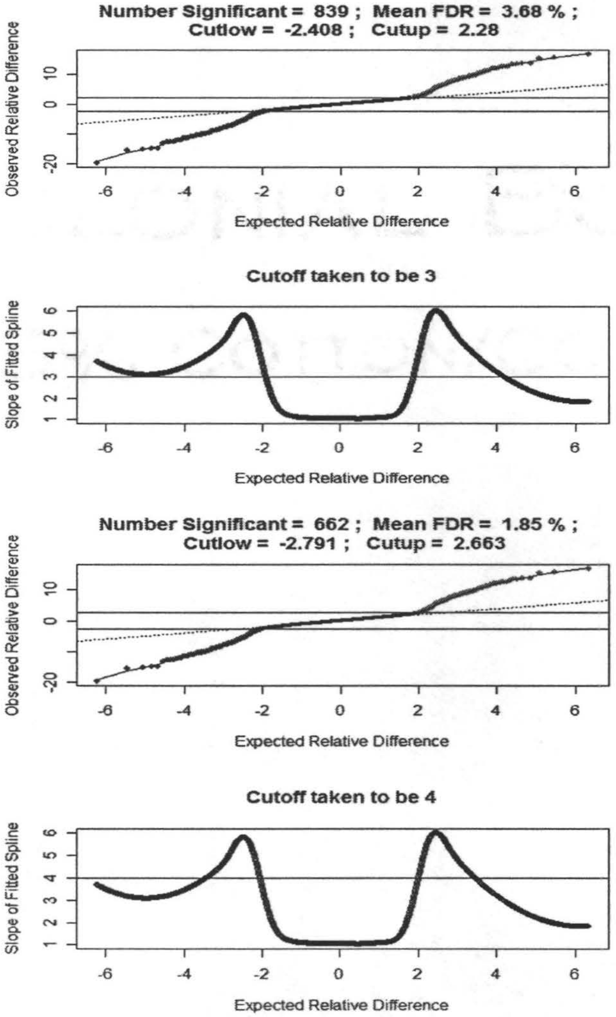


Figure 4.1: *The figure for t distribution simulation data with G=10000 G_0=500. The upper plot with cutoff=3 and the lower plot with cutoff=4*

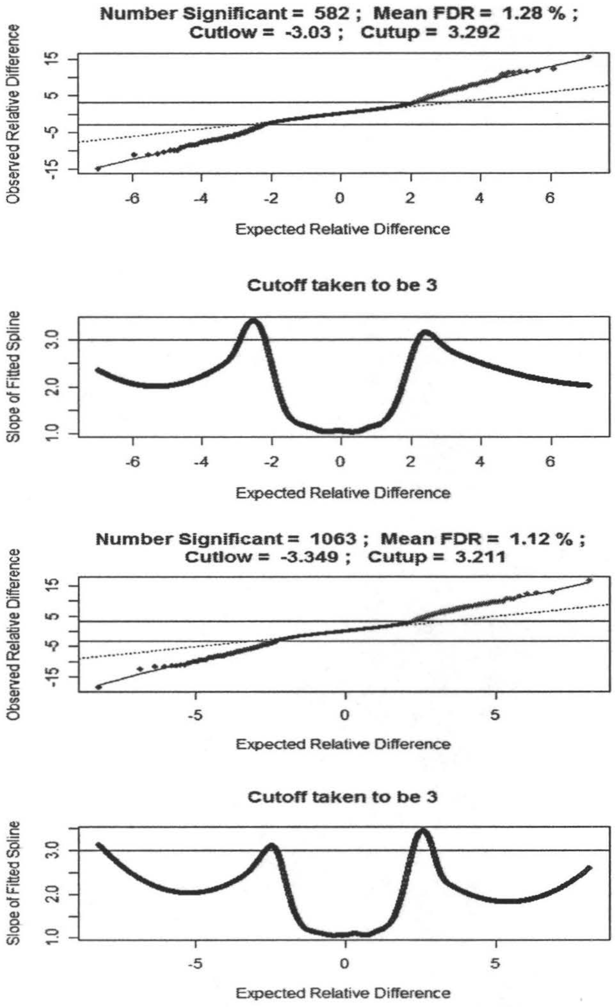


Figure 4.2: *The figure for normal distribution simulation data with cutoff=3. The upper plot with G=10000 G_0=500 and the lower plot with G=20000 G_0=1000*

G=10000 G_0=500			G=20000 G_0=1000		
Num.Sig	$\widehat{FDR}$	RealFDR	Num.Sig	$\widehat{FDR}$	RealFDR
333	0.0107	0.2744	1161	0.0131	0.2282
521	0.0106	0.1938	1063	0.0112	0.1797
672	0.0186	0.2842	1288	0.0176	0.2694
556	0.0119	0.1871	1121	0.0098	0.2096
582	0.0128	0.2216	1171	0.0144	0.2263

Table 4.2: *Simulation study for Normal distribution with* $G = 10000$ *and* 20000

Chapter 5

Discussion and Future Work

In this thesis, we introduce a new method SAMSPLINE to select significant genes. SAM methodology chooses the significance region based on the distance from the line of slope 1 and the distances are the same for both positive and negative significance. But the positive and negative significance could be different. Our method chooses upper and lower significant cut off points directly from the derivative and declares that any genes with their observed statistics greater than the upper cut-off are positive significant and any genes with their observed statistics less than the lower cut-off are negative significant. The biologist can use the list of significant genes for their Type 1 Diabetes study. The objective of Type 1 Diabetes research is understand the autoimmune response that results in the death of b-islet cells and to identify the genes that control this process in mice models and in human patients.

We notice that we can get good results using this method for some datasets (say Figure 3.7). But for some datasets we can not get good results since the derivative plots are not as good as we expect (say Figure 3.8 and 3.9). Another thing is the value of cutoff horizontal line. As we can see from Figure 3.8, if we set the cutoff value to be a fixed number, we have no idea if it is the best choice for different datasets. Different datasets have different smoothing splines and the slopes of fitted splines are different. If we set the cut off line equals to 3 for all datasets, it will affect the results. But how to calculate the cutoff value?

We also considered an automatic method for calculating the cutoff value. First fit a spline to the plot of the average of expected order statistics $(\bar{d}_{(1)}, ..., \bar{d}_{(G)})$ against $(d^b_{(1)}, ..., d^b_{(G)})$ for each permutation $b = 1, ..., B$. For each spline find the numerical derivatives using the method we described before. Then find an upper 95% pointwise envelope for the derivative curves. Then choose the cutoff value as the maximum value of the envelope. We use this method to find the cutoff value for Affy datasets and Table 5.1 lists the results. From Table 5.1 we find out that all the cutoff values are around 2 to 3 and $\widehat{FDR}$ are very high. So we claim that this method is not working well. In the future we wish to find a better way to calculate the cutoff value based on the data set rather than a fixed value.

data set	dfmin	cutoff	Num.Sig	$\widehat{FDR}$
Zhenya1	12.84	1.6855	1516	0.2413
Zhenya2	16.41	1.9883	72	0.4054
Zhenya3	17.52	1.5103	717	0.3402
Zhenya4	12.82	3.3836	30	0.2997
Zhenya5	12.66	1.6903	744	0.2013

Table 5.1: *The output for Affy data sets without setting cutoff value*

Another problem is the value of C in the objective function. We can see from Figure 5.1 that for different values of C, the optimized degrees of freedom could be different. We do not know if our method to calculate C is a good one or not. We wish a method could be developed to test it.

The last problem we noticed is from Table 3.4. The $\widehat{FDR}$ are equal to 1 for three datasets. We then calculate the number of false significant genes for each permutation and get its distribution. Table 5.2 shows the distributions for the Illumina datasets and the numbers of significant genes respectively.

Name	Number	Min	1stQu	Median	Mean	3rdQu	Max	Num.sig
Illutanya1	70	16	45	93.5	288	408	3044	86
Illutanya2	70	4	25.25	100	250	275	2768	120
Illutanya5	70	0	3	8.5	52.1	24.25	1086	24
Illutanya3	70	0	3	10.5	79.47	41.5	1614	210
Illutanya4	70	0	0	1.5	24.77	4	726	67
Illutanya6	70	10	46.25	119	265	252	3083	698

Table 5.2: *The Distribution of the number of significant genes for Illumina datasets*

We find out for the first three datasets, the means are greater than the numbers of significant genes. So when we calculate $\widehat{FDR}$, it is greater than 1. But $\widehat{FDR}$ can not be greater than 1, the program sets it to be 1. We also notice that the medians are much smaller than the means and the maximum values are huge. For the other three datasets, the means are smaller than the number of significant genes. Therefore we can get a reasonable $\widehat{FDR}$. We can have a better look through Figure 5.2. Figure 5.2 is the histogram of the distribution of the number of significant genes for dataset illutanya2. We can see there are some outliers that are greater than 2500. Figure 5.3 shows the histogram for dataset illutanya4 with $\widehat{FDR}$ equal to 0.3807. The value and number of the outliers are both small. Some statistician use median false discovery rate instead of mean false discovery rate. If we use median $\widehat{FDR}$ here, it would reduce the values of $\widehat{FDR}$ for those three datasets above. But for the other datasets, the $\widehat{FDR}$ could be extremely small since the median values are always much less than the mean values in our case. So we prefer mean $\widehat{FDR}$ instead of median $\widehat{FDR}$.

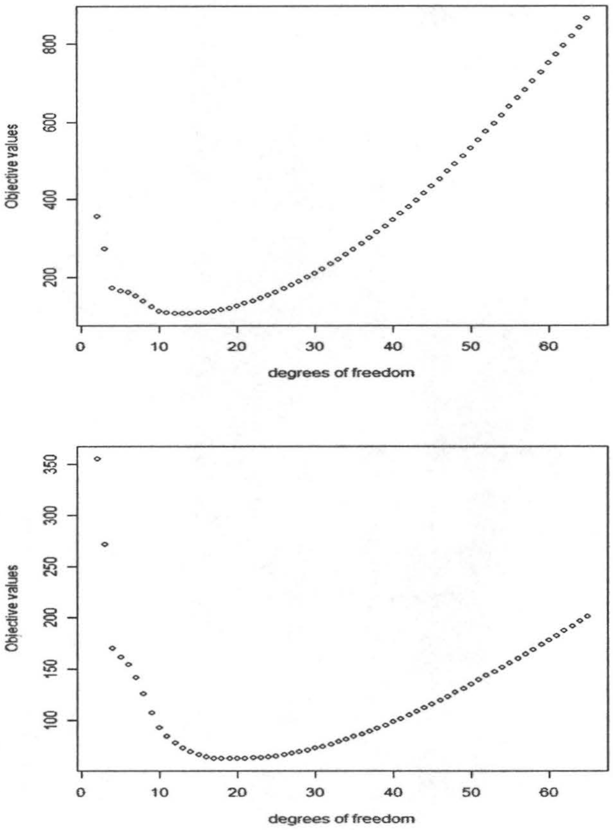


Figure 5.1: *The plot of the degrees freedom against the output of objective function for* $C = 100$ *and* 500 *for dataset Zhenya2*

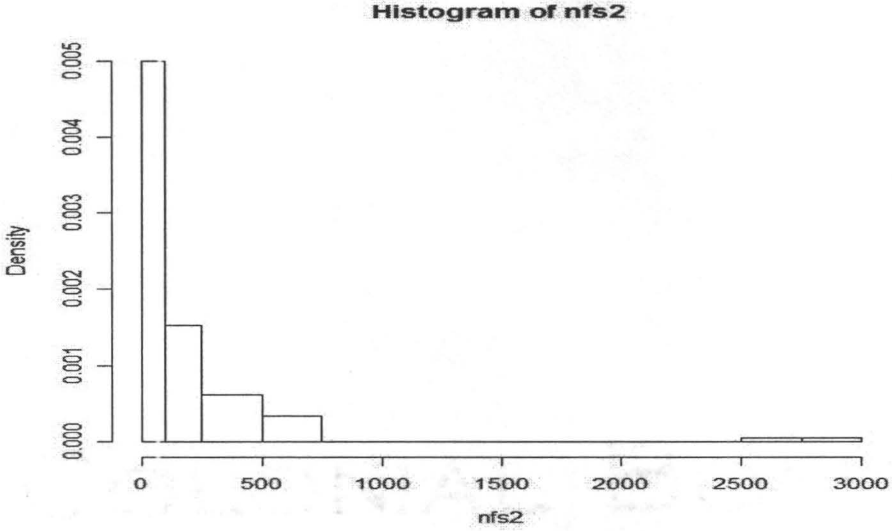


Figure 5.2: *The histogram of the distribution of the number of significant genes in permutations for Illutanya2*

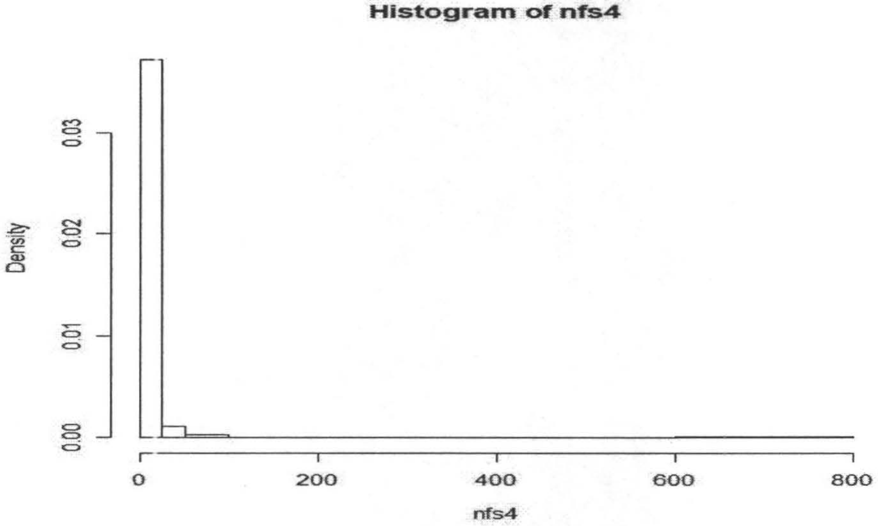


Figure 5.3: *The histogram of the distribution of the number of significant genes in permutations for Illutanya4*

Appendix A

R Code for SAMSPLINE

# The function *samspline* is the main function for our method. It fits a cubic smoothing spline to the SAM plot, choose the upper and lower significance cut-offs, calculate the numbers of significant genes and FDR and draw the SAM and derivative plots.

# Arguments

stats is a matrix with the values of observed order statistics.

di.perm is the order statistics from the permutations.

cutoff is the value of the horizontal line we draw on the derivative plot. In most cases, we set it equal to 3.

# Values

cut.up is the upper significance cut-off.

cut.low is the lower significance cut-off.

num.sig is the total number of significant genes.

$MeanFDR$ is the false discovery rate.

siggene is a list of significant genes with the values of their observed order statistics, s0 and the average order statistics from the permutations.

spline is the output from function *smooth.spline* including components of *spar*, *lambda*, equivalent degrees of freedom and so on.

```
samspline <- function(stats, di.perm, pi0=calc.pi0(Obs, di.perm),
          cutoff, alpha=0.05, spacing=0.01, plot=T, ...) {
   Obs <- stats$d.stat
   Exp <- stats$Expected
   sp = smooth.spline(Exp, Obs, ...)
   X <- (floor(min(Exp)/spacing):ceiling(max(Exp)/spacing))*spacing
   Y <- predict(sp, X)$y
   Xmid <- X[-1]-diff(X)/2
   Dhat <- diff(Y)/diff(X)
   if (missing(cutoff)) {
     R <- ncol(di.perm)
     Exp1 <- sort(Exp)
     derivs <- matrix(NA,nrow=length(Xmid), ncol=R-2)
     for (i in 1:(R-2)) {
       sp1 = smooth.spline(Exp1, di.perm[,i+1], ...)
       Y1 <- predict(sp1, X)$y
       derivs[,i] <- diff(Y1)/diff(X)
     }
     env = apply(derivs, 1, max)
     cutoff <- max(env)
   }
   else {
     env=cutoff
     derivs=NULL
   }
   negs <- which(Xmid<0 & Dhat>=cutoff)
   if (length(negs) > 0) {
     xneg <- max(Xmid[negs])
     i.neg <- max(which(Exp<=xneg))
     cut.low <- Obs[i.neg]
   }
   else {
     xneg <- min(Xmid)
     Eneg <- min(Exp)
     i.neg <- which.min(Exp)
     cut.low <- -Inf
```

```
  }
  pos <- which(Xmid>0 & Dhat>=cutoff)
  if (length(pos)>0) {
    xpos <- min(Xmid[pos])
    i.pos <- max(which(Exp>=xpos))
    cut.up <- Obs[i.pos]
  }
  else {
    xpos <- max(Xmid)
    Epos <- max(Exp)
    cut.up <- Inf
  }
  samplot <- SAM.plot(stats, di.perm, cut.low, cut.up, pi0)
  if (plot) {
 op <- par(no.readonly=T)
  par(mfrow=c(2,1))
 samplot <- SAMplot(stats, di.perm, cut.low, cut.up, pi0)
 lines(sp)
   plot(Xmid, Dhat, xlab="Expected Relative Difference",
        ylab="Slope of Fitted Spline",
        main=paste("Cutoff taken to be",round(cutoff,4)))
       abline(h=cutoff)
   par(op)
 }
 return(list(cut.up=cut.up, cut.low=cut.low, NumberSignificant =
 samplot$NumberSignificant, MeanFDR=samplot$MeanFDR,
 siggene=samplot$siggene, spline=sp))
}
```

Appendix B

R Code for Selecting Significant Genes

# The function *Signif* is to calculate the number of significant genes.

# Arguments

stats is a matrix with the values of observed order statistics.

expect is the average of expected order statistics.

cut.low and *cut.up* are the upper and lower significance cut off points calculated in function *samspline*.

# Values

num.sig is the number of significant genes consisted of the numbers of total significant genes, positive significant genes and negative significant genes.

siggene is a list of significant genes with the values of their observed order statistics, s0 and the average order statistics from the permutations.

```
Signif <- function(stats, expect,cut.low, cut.up) {
  obs <- stats$d.stat
  pos.sig <- which(obs >= cut.up)
  np <- length(pos.sig)
  neg.sig <- which(obs <= cut.low)
  nn <- length(neg.sig)
  ind <- which(obs >= cut.up|obs <= cut.low)
  siggene <- stats[ind,]
 return(list(num.sig = c(nn+np, np, nn), siggene=siggene))
}
```

Appendix C

R Code for Finding the Optimized Degree of Freedom

# The function *dfreed* is the objective function used in the function *optimize* to find the optimized degrees of freedom.

# Arguments

x is the variable of degrees of freedom from 2 to CVDF.

stats is a matrix with the values of observed order statistics.

perms is a permutation matrix with the values of expected order statistics.

const is the result calculated in the function dfspline.

# Values

sumsum is the sum of two sums. one is the sum of the absolute values of residuals of spline at degrees of freedom x and the other is the sum of the absolute values of fitted values of the predicted spline at *Xmid* divided by *const* at degrees of freedom x.

# *dfmin* is the optimized degrees of freedom.

```
dfreed <- function(x, stats, perms, const, spacing=0.01)
       {
 spline = samspline(stats, perms$permutations, cutoff=3,
          df=x, plot=F, spacing)
 Exp = stats$Expected
 X =(floor(min(Exp)/spacing):ceiling(max(Exp)/spacing))*spacing
 Xmid = X[-1]-spacing/2
 ppd = predict(spline$spline, Xmid, deriv=3)
 resid = residuals(spline$spline)
 absresid = abs(resid)
 absderiv = abs(ppd$y)
 sumsum = sum(absresid )+sum( absderiv)/const
 return(sumsum)
  }

dfmin = optimize(dfreed,c(2,cvdf),stats=stats,perms=perms, const =
const)$minimum
```

Appendix D

R Code for dfspline Function

# The function *dfspline* is our main function to combine all the functions together and get the results. It has three parts. Calculate *const*, the optimized degrees of freedom and use the optimized degrees of freedom to calculate the numbers of significant genes, FDR and return the results.

# Arguments

stats is a matrix with the values of observed order statistics.

perms is a permutation matrix with the values of expected order statistics.

# Values

results includes the optimized degrees of freedom, the number of significant genes, false discovery rate, constant and CVdf.

sig.gene is a list of significant genes with the values of their observed order statistics, s0 and the average order statistics from the permutations.

spline is the output from function *smooth.spline* including components of *spar*, *lambda*, equivalent degrees of freedom and so on.

```
dfspline <- function(stats, perms, spacing=0.01){
    spline1 = samspline(stats, perms$permutations, cutoff=3, plot=F, spacing)
    cvdf = spline1$spline$df Exp = stats$Expected
    X = (floor(min(Exp)/spacing):ceiling(max(Exp)/spacing))*spacing
    Xmid = X[-1]-spacing/2
    ppd = predict(spline1$spline, Xmid, deriv=3)
    absderiv = abs(ppd$y)
    sum2 = sum(absderiv)
    spline2 = samspline(stats, perms$permutations, df=2, cutoff=3,
                plot=F, spacing)
    resid = residuals(spline2$spline)
    absresid = abs(resid)
    sum1 = sum(absresid)
    const = sum2/sum1
    dfmin = optimize(dfreed,c(2,cvdf),stats=stats,perms=perms, const =
            const)$minimum
    spline = samspline(stats, perms$permutations, cutoff=3,
            df=dfmin,plot=F)
    out <- c(dfmin, spline$num.sig, spline$MeanFDR, const, cvdf)
    names(out) <- c("dfmin", "Num.Sig", "MeanFDR", "C", "CVdf")
    return(list(results=out, sig.gene=spline$siggene,
          deriv=spline$deriv, spline=spline$spline))
            }
```

Appendix E

R Code for Simulation Study

# The function *datageneratet* is to generate random dataset using t distribution with 3 degrees of freedom. The function *datagenerate* is to generate random dataset using a standard normal distribution.

# Arguments

G is the total number of simulated genes.

$G0$ is the number of differently expressed genes.

nr is the number of replicates of the same experiments.

# Value

The output *exdata* is the random dataset generated.

```
datageneratet <- function(mu, sigma, G, G0, nr ){
mean <- rnorm(G,mu,sigma)
sds <- sqrt(rchisq(G,1))
d1 <- rep(-3,length=G0/2)
d2 <- rep(3,length=G0/2)
d <- c(d1, d2, rep(0, G-G0))*sds
strain1 <- matrix(rt(nr*G,3),ncol=nr)*sds/sqrt(3)+mean
strain2 <- matrix(rt(nr*G,3),ncol=nr)*sds/sqrt(3)+mean+d
exdata <- cbind(strain1, strain2)
```

```
 strain <- rep(1:2, each=nr)
 row.names(exdata) <- (1:G)
 exdata
 }

datagenerate <- function(mu, sigma, G, G0, nr ){
 mean <- rnorm(G,mu,sigma)
 sds <- sqrt(rchisq(G,1))
 d1 <- rep(-3,length=G0/2)
 d2 <- rep(3,length=G0/2)
 d <- c(d1, d2, rep(0, G-G0))*sds
 strain1 <- matrix(rnorm(nr*G, rep(mean, nr), rep(sds, nr)),ncol=nr)
 strain2 <- matrix(rnorm(nr*G, rep(mean+d, nr), rep(sds, nr)),ncol=nr)
 exdata <- cbind(strain1, strain2)
 strain <- rep(1:2, each=nr)
 row.names(exdata) <- (1:G)
 exdata
 }
```

Bibliography

[1] Barnes, M., Freudenberg, J., Thompson, S., Aronow, B. and Pavlidis, P. (2005) Experimental comparison and cross-validation of the Affymetrix and Illumina gene expression analysis platforms, *Nucleic Acids Research* 33 : 5914-5923.

[2] Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing, *Journal of the Royal Statistical Society, Series B* 85: 289-300.

[3] Chu, G., Narasimhan, B., Tibshirani, B. and Tusher, V. (2005) *SAM "Significance Analysis of Microarrays" Users guide and technical document,* http://www-stat.stanford.edu/ tibs/SAM/sam.pdf

[4] Irizarray, R. A., Hobbs, B., Collin, F., Beazer-Barclay Y. D., Antonellis, K. J., Svherf, U. and Speed, T. P. (2003a) Exploration, Normalization and Summaries of High Density Oligonucleotide Array Probe Level Data. *Biostatistics 2003*, 4(2): 249-264.

[5] Irizarray, R. A., Gautier, L. and Cope, L. M. (2003b) An R Package for Analyses of Affymetrix Oligonucleotide Arrays. In *The Analysis of Gene Expression Data:*

Methods and Software (Parmigiani, G., Garrett, E. S., Irizarray, R. A. and Zeger, S. L. editors) Springer-Verlag New York.

[6] Knudsen, S. (2004). *Guide to Analysis of DNA Microarray Data*, Second Edition. John Wiley and Sons, New York.

[7] Lee, M. T. (2004). *Analysis of Gene Expression*, Kluwer Academic Publishers.

[8] Llanos, G. and Libman, I. (1994). Diabetes in the Americas. *Bulletin of the Pan American Health Organization*, 28: 285-301.

[9] Luo, Y. (2007). Comparison between Affymetrix and Illumina gene expression microarray platforms. *A Thesis for the Degree Master of Science*, McMaster University.

[10] Ruppert, D., Wand, M. P. and Carroll, R. J. (2003). *Semiparametric Regression*, Cambridge University Press.

[11] Storey, J. D. (2002) A direct approach to false discovery rates. *Journal Of The Royal Statistical Society Series B*, 64: 479-498.

[12] Storey, J. D. (2003) The positive false discovery rate: A Bayesian interpretation and the q-value. *Annals of Statistics*, 31: 2013-2035.

[13] Storey, J. D. and Tibshirani, R. (2003) SAM Thresholding and False Discovery Rates for Detecting Differential Gene Expression in DNA Microarrys. In *The Analysis of Gene Expression Data: Methods and Software* (Parmigiani, G., Garrett, E. S., Irizarray, R. A. and Zeger, S. L. editors) Springer-Verlag New York.

[14] Tusher, V., Tibshirani, R. and Chu, C. (2001). Significance analysis of microarrays applied to transcriptional responses to ionizing radiation. *Proceedings of the National Academy of Sciences*, 98: 5116-5121.

[15] Watson, J. D. and Crick, F. H. C. (1953). A Structure for Deoxyribose Nucleic Acid. *Nature*, 171: 737-738.