2009

STATISTICAL METHODS IN MICROARRAY DATA ANALYSIS

Liping Huang
University of Kentucky, lipinglphuang@hotmail.com

ABSTRACT OF DISSERTATION

Liping Huang

The Graduate School
University of Kentucky
2009

STATISTICAL METHODS IN MICROARRAY DATA ANALYSIS

ABSTRACT OF DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Arts and Sciences
at the University of Kentucky

By
Liping Huang
Lexington, Kentucky

Co-Directors: Dr. Mai Zhou, Professor of Statistics
and Dr. Arne Bathke, Professor of Statistics
Lexington, Kentucky 2009

ABSTRACT OF DISSERTATION

STATISTICAL METHODS IN MICROARRAY DATA ANALYSIS

This dissertation includes three topics. First topic: Regularized estimation in the AFT model with high dimensional covariates. Second topic: A novel application of quantile regression for identification of biomarkers exemplified by equine cartilage microarray data. Third topic: Normalization and analysis of cDNA microarray using linear contrasts.

KEYWORDS: AFT Model, Elastic Net, cDNA Microarray, Quantile Regression, Linear Combinations

Author's signature: Liping Huang

Date: October 29, 2009

STATISTICAL METHODS IN MICROARRAY DATA ANALYSIS

By
Liping Huang

Co-Director of Dissertation: Mai Zhou

Co-Director of Dissertation: Arne Bathke

Director of Graduate Studies: William S. Griffith

Date: October 29, 2009

RULES FOR THE USE OF DISSERTATIONS

Unpublished dissertations submitted for the Doctor's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgments.

Extensive copying or publication of the dissertation in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this dissertation for use by its patrons is expected to secure the signature of each user.

Name	Date

DISSERTATION

Liping Huang

The Graduate School
University of Kentucky
2009

STATISTICAL METHODS IN MICROARRAY DATA ANALYSIS

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Arts and Sciences
at the University of Kentucky

By
Liping Huang
Lexington, Kentucky

Co-Directors: Dr. Mai Zhou, Professor of Statistics
and Dr. Arne Bathke, Professor of Statistics
Lexington, Kentucky 2009

ACKNOWLEDGMENTS

I am grateful to numerous people for their assistance during my graduate studies. In particular I am grateful to the members of my committee whom I was privileged to know both professionally and personally.

I especially wish to thank my dissertation advisor, Dr. Mai Zhou. Without his constant encouragement, support and guidance, this work would not have been possible. I would also like to thank Dr. Arne Bathke for his guidance, encouragement and friendship. He always made time for me in his busy schedule. I am grateful to Dr. Arny Stromberg for offering me the research assistantship in his group, which was an excellent learning experience for me. I am grateful to Dr. James MacLeod for his brilliant microarray related statistical questions, helpful comments, and generosity with his time. I am grateful to Dr. Constance Wood and Dr. Bill Griffith for their comments and help.

I would like to thank all the members of Dr. James MacLeod's lab: Dr. Naoki Miura, Dr. Michael Mienaltowski, Wenying Zhu, Stephen Coleman, and Rebekah Cosden.

My heartfelt thanks go to my parents for their endless encouragement and support during all my years of education. And finally I like to thank my many friends for their support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1 Introduction

1.1 Microarray Technology

A microarray is an arrangement of miniaturized test sites or 'spots' on a surface. Typically each spot is less than 250 micrometers in diameter and contains biological molecules–usually DNA or protein–which act as probes for a test sample applied to the array. The surface may be a glass slide, a plastic plate with wells or a polymer bead. The microarray format allows many tests or experiments to be performed simultaneously, in parallel, leading to the generation of huge amounts of biological information. Several features of DNA microarray technology make it particularly well suited to exploratory research. It is (relatively) cheap, flexible and universal, fast and user-friendly (Brown and Botstein, 1999). A comprehensive review of the biological and technological aspect of the microarray technology can be found in Nguyen et al. (2002).

Microarrays have many uses. For example, they can disclose the correlated loss and increase of gene expression, allowing gene interactions to be studied. And they can be used to design and screen drugs, wherein compounds that affect the expression of important genes are disclosed. The examples of microarray use are myriad because of this simple technique's power.

Two of the most commonly used microarrays for gene-expression measurements are oligonucleotide GeneChip expression arrays made by Affymetrix and custom-made cDNA arrays. Oligonucleotide arrays use small 25 base pair gene fragments as the DNA to be spotted onto an array. To combat non-specific hybridization, that is, DNA fragments, often called probes, have cross-reactivity with other genes, a second probe that is identical to the first except for a mismatched base at its centre is placed next to the first. This is called the Perfect Match/Mismatch (PM/MM) probe strategy.

Any background hybridization with the MM probe is subtracted from the PM probe signal which results in perfect hybridization.

It is important to note that in oligonucleotide arrays, only one sample is hybridized to a single array during these experiments. Such samples are prepared by extracting mRNA from a cell and turning it back into DNA through reverse transcription of the mRNA into a cDNA (see Figure 1.1). This is followed by a labeling step during which the cDNA is then transcribed to cRNA while incorporating a label (e.g., biotin). Once labeled, the sample of cRNAs can be hybridized to the array and bound by the various oligonucleotide probes. Finally, a staining reaction is performed in order to visualize the amount of hybridization.

By contrast, two samples are prepared in cDNA microarray for hybridization to the array: a control sample and an experimental sample (see Figure 1.2). mRNA are extracted from cells and reverse transcribed into cDNA. During the reverse transcription step a fluorescent dye is incorporated into the newly formed cDNA and a different dye is employed to label the different samples. For example, the control sample can be labeled with a green-fluorescing dye called Cy3 and the experimental sample labeled with a red-fluorescing dye called Cy5. The differently labeled samples are then combined and hybridized to the microarray together (see Figure 1.2). The two samples will competitively bind to the probes on the array and the sample containing more gene expression for a particular probe will win out. That is, if there is more of an mRNA transcript in the control sample than in the experimental sample (i.e., the gene is downregulated in the experiment) then more Cy3 will bind to the probe on the array and the spot will fluoresce green. Conversely, if there is more mRNA transcript in the experimental sample, the reverse will happen and the spot will fluoresce red. When the two samples have the same amount of transcript, the dyes will cancel each other out and the spot will fluoresce yellow.

When oligonucleotide and cDNA arrays are compared with one another, one will

note a few distinctions between them. Affymetrix oligonucleotide arrays are created using a combination of DNA synthesis and photolithographic techniques, whereas cDNA arrays are constructed by spotting or printing PCR products or oligonucleotides onto glass slides. Affymetrix arrays contain sets of multiple 25 oligonucleotide probes specific for each gene or expressed-sequence tag (EST), whereas spotted arrays generally contain longer cDNA probes (usually 500 to 1,000 bases) or oligonucleotide probes (usually 25 to 60 bases) for each gene.

Although a large number of genes are believed to be mostly inactive, there are many genes whose activities are associated with various physiological effects. An interesting and important task in analyzing human genomic data is to relate gene activities to phenotypic or clinical information. Hence, microarray technology has garnered considerable attention in the statistics, bioinformatics and medical communities.

1.2 Statistical Problems Associated with Microarray Data

The analysis of DNA microarrays poses a large number of statistical problems, including the normalization of the data, multiple comparisons, and small sample sizes.

Normalization involves adjusting the microarray data for effects which arise from variation in the technology rather than from biological differences between the RNA samples or between the printed probes. Much research has been done to address the problem of normalization (Yang et al., 2001; Wolfinger et al., 2001; Tseng et al., 2001; Huang et al., 2005).

The problem of multiple comparisons is that even if the statistical P-value assigned to a gene indicates that it is extremely unlikely that differential expression of this gene was due to random rather than treatment effects, the very high number of genes on an array makes it likely that differential expression of some genes in fact represent false positives. The false discovery rate (FDR) of a test, defined as the expected

proportion of false positives among the declared significant results (Benjamini and Hochberg, 1995; Keselman et al., 2002) has been proposed to address the problem of false positives in multiple comparisons in microarray data. Because of this directly useful interpretation, FDR is a more convenient scale to work with than the P-value scale. Some statistical methods have been proposed either to transform a P-value into an FDR or to compute FDR directly (Storey and Tibshirani, 2003; Aubert et al., 2004; Reiner et al., 2003; Benjamini and Hochberg, 1995; Benjamini and Yekutieli, 2001; Storey, 2002). .

The sample size n is typically small in microarray studies. The rapid accumulation of microarray gene expression data suggests that combing microarray data obtained from different studies may be a useful way to increase sample size. Recently, several methods have been proposed to combine inter-study microarray data at different levels in cancer research (Choi et al., 2003; Jiang et al., 2004; Rhodes et al., 2002; Shen et al., 2004). Instead of integrating microarray gene expression values, some methods, referred to as meta-analysis, combine results (e.g., t-statistic) of individual studies to increase the power of identifying genes differentially expressed between normal and diseased samples (Choi et al., 2003; Rhodes et al., 2002).

In this dissertation, we will discuss three microarray related statistical topics: first, regularized estimation in accelerated failure time (AFT) model with high-dimensional covariates; second, linear quantile regression to identify equine cartilage biomarkers; finally, planned linear combinations to normalize and analyze cDNA array data. For the first topic, we will address the problem from background, statistical methods, simulation studies, and real data application. For the second and third topics, we will organize the problems under the headings of background, results and discussion, conclusion, and methods (statistical algorithm or microarray experiments).

Figure 1.1: The use of Oligonucleotide arrays

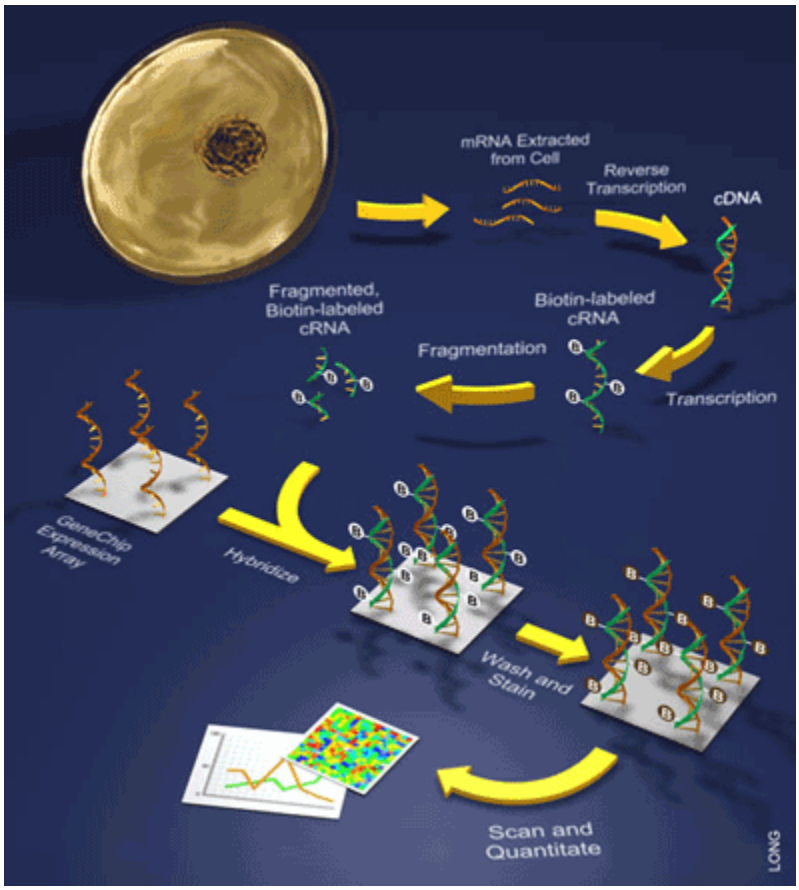
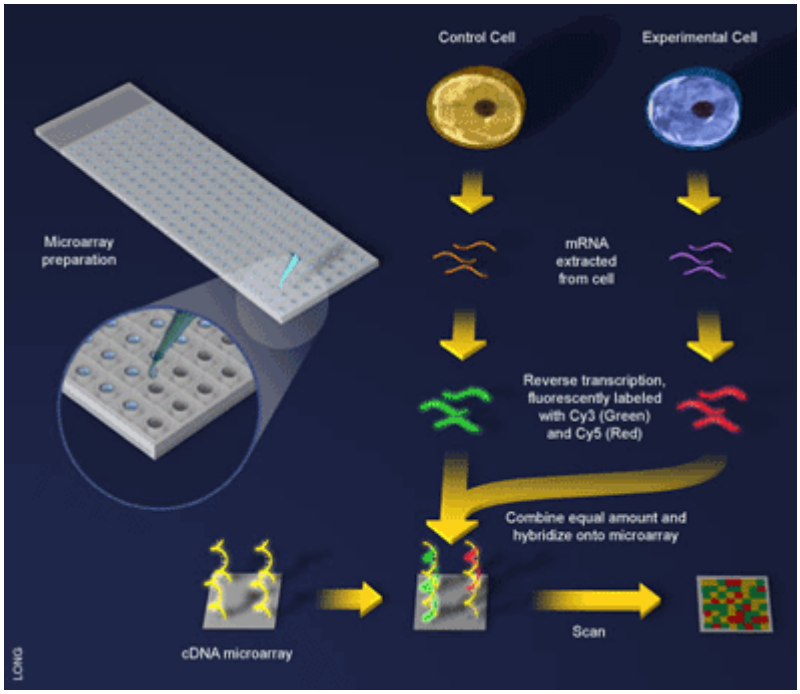


Figure 1.2: An example of a cDNA microarray experiment



Chapter 2 Regularized Estimation in Accelerated Failure Time (AFT) Model with High-dimensional Covariates

2.1 Background

Cancer is a highly complex disease which can encompass multiple genomic alterations. These changes may be inherited or somatically acquired during progression from a normal to a cancerous cell. A comprehensive characterization of all of the genetic, genomic, and epigenetic modifications associated with the cancer is critical for the understanding of the origins of tumor process, and for finding the targets of therapeutic interventions. In recent years advances in microarray technology have allowed us to measure the expression levels of tens of thousands of genes or even entire genomes simultaneously, thus reducing time and cost considerably (Lockhar and Winzler, 2000; Brown and Botstein, 1999). Cancer is an especially pertinent target of microarray technology due to the well-known fact that this disease causes, and may even be caused by, changes in gene expression. Microarrays have allowed the rapid identification of which genes are turned on and off in tumor development, resulting in a much better understanding of the disease.

Use of microarray technology, either Affymetrix or cDNA array, often leads to high-dimensional and low-sample size data settings where the number of genes typically far exceeds sample size. Table 2.1 gives an example of dimensionality of microarray data. Sample size n is equal to 100 while the number of covariates p is 10,000. It is obviously that $p \gg n$. The high dimensional property makes the statistical analysis of microarray data complicated .

Suppose that the data set has n observations with p predictors. Let $\mathbf{Y} = (y_1, \ldots, y_n)^T$ be the response and $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_p)$ be the model matrix, where $\mathbf{x}_j = (x_{1j}, \ldots, x_{nj})^T$, $j = 1, \ldots, p$ are the predictors. The linear model is $E(Y|X = x) = \beta_0 + X^T\boldsymbol{\beta}$, where

$\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$. The traditional linear regression method has problems with analyzing high dimensional data. The most prominent problems are colinearity and overfitting. The colinearity problem occurs when covariates are so highly correlated that reliable estimates of their individual regression coefficients can not be found. The overfitting problem occurs when the statistical model describes random error or noise rather than the underlying relationship. Such a model which has been overfit typically has poor predictive performance, as it exaggerates minor fluctuations in the data.

To handle such high-dimensional data, two kinds of statistical methods are proposed. One way to handle high dimensional data is dimension reduction, which searches for low-dimensional projections of the covariates to optimize the tradeoff between bias and variance and thus achieve reduced mean squared errors (MSE) (Park, 1981). Examples include principal component regression (Jolliffe, 1986) and partial least square methods (Martens and Naes, 1989). The other way to handle high dimensional data is by penalized estimation, such as ridge regression (Hoerl and Kennard, 1970); support vector machines (Vapnik, 1995); the LASSO (Tibshirani, 1996); and the gradient directed regularization method (Friedman and Popescu, 2004). We will give more detailed discussion on penalized methods in the following section.

Penalized Methods

It is well known that ordinary least squares (OLS) often performs poorly in both prediction and interpretation for high dimensional data. Penalization methods have been proposed to improve OLS. For example, ridge regression (Hoerl and Kennard, 1970) using L_2-norm ($\sum \beta^2$) is known to shrink the coefficients of correlated covariates towards each other, allowing them to borrow strength from each other. In the extreme case of k identical covariates, they each get identical coefficients, each $1/k$ the size that any single one would get if fit alone. From a Bayesian point of view, the ridge

penalty is ideal if there are many covariates and all have non-zero coefficients. One limitation of L_2-norm penalty is that it uses all covariates in the prediction and does not provide a way of selecting relevant covariates for prediction. Best subset selection in contrast produces a sparse model, but it is extremely variable because of its inherent discreteness. A promising technique called the LASSO using L_1-norm ($\sum |\beta|$) is somewhat indifferent to very correlated covariates, and will tend to pick one and ignore the rest. In the extreme case above, the LASSO problem breaks down. The LASSO penalty corresponds to a Laplace prior, which expects many coefficients to be zero or close to zero, and a small subset of non-zero coefficients. L_1-norm penalty suffers from two drawbacks (Zou and Hastie, 2005). First, the expression levels of genes that share one biological pathway are highly correlated. The L_1-norm penalty can only select one gene instead of automatically selecting the whole group of relevant and yet highly correlated genes. Second, the L_1-norm penalty can select at most n genes where n is the sample size. However, for microarray data, the number of covariates p is much larger than the sample size n (say, 10000 versus 100 as in Table 2.1). This limitation of selecting at most n genes instead of an arbitrary number of genes relevant to the clinical outcome seems unrealistic for many biomedical studies. Elastic net (Zou and Hastie, 2005) performs much like LASSO, but removes any degeneracies caused by extreme correlations. Similar to LASSO, the elastic net simultaneously does automatic variable selection and continuous shrinkage, and it can select groups of correlated variables. It is like a stretchable fishing net that retains 'all the big fish'. More generally, it creates a useful compromise between ridge and LASSO. Some details are as follows.

For simplicity we assume the x_{ij} are standardized: $\sum_{i=1}^{n} x_{ij} = 0$, $1/n \sum_{i=1}^{n} x_{ij}^2 = 1$. The elastic net solves the following problem:

$$min_{(\beta_0,\boldsymbol{\beta})\in R^{p+1}}[\frac{1}{2n}\sum_{i=1}^{n}(y_i - \beta_0 - x_i^T\boldsymbol{\beta})^2 + \lambda P_\alpha(\boldsymbol{\beta})] \tag{2.1}$$

where

$$P_\alpha(\boldsymbol{\beta}) = \frac{1}{2}(1-\alpha)\sum_{j=1}^{p}\beta_j^2 + \alpha\sum_{j=1}^{p}|\beta_j|$$

We call P_α the elastic net penalty, which is a convex combination of ridge-regression penalty ($\alpha = 0$) and the LASSO penalty ($\alpha = 1$). This penalty is particularly useful in the $p \gg n$ situation, or in any situation where there are many correlated predictors. As α increases from 0 to 1, for a given λ, the sparsity of the solution of equation 2.1 (i.e., the number of coefficients equal to 0) increases monotonically from 0 to the sparsity of the LASSO solution. For all $\alpha \in [0, 1)$, the elastic net penalty function is singular (without first derivative) at 0 and is strictly convex for all $\alpha > 0$, thus having the characteristics of both the LASSO and ridge regression. Note that the LASSO penalty is convex but not strictly convex. These arguments can be seen clearly from Figure 2.1. The red curve shows that singularities at the vertexes are necessary for sparsity and that strict convex edges are important for grouping.

Cyclical coordinate descent methods have been proposed for the LASSO a number of times, but only recently was their power fully appreciated. Early references (Fu, 1998; Daubechies, et al., 2004; Van der Kooij, 2007) use coordinate descent for solving elastic net penalized regression models. Recent rediscoveries include Wu and Lange (2008a) and Friedman et al. (2007). The first paper recognized the value of solving the problem along an entire path of values for the regularization parameters, using the current estimates as warm starts. This strategy turns out to be remarkably efficient for this problem. Friedman et al. (2008) develop fast algorithms glm net based on cyclical coordinate descent for fitting generalized models with elastic net penalties. The glm net can work on very large data sets and can take advantage of sparsity in the feature set.

Consider a coordinate descent step for solving equation 2.1. That is, suppose we have estimates $\tilde{\beta}_0$ and $\tilde{\beta}_l$ for $l \neq j$, and we wish to partially optimize with respect to β_j. Denote by $R(\beta_0, \boldsymbol{\beta})$ the object function in equation 2.1. We would like to

compute the gradient at $\beta_j = \tilde{\beta}_j$, which only exists if $\tilde{\beta}_j \neq 0$. If $\tilde{\beta}_j > 0$, then

$$\frac{\partial R}{\partial \beta_j}|_{\beta=\tilde{\beta}} = -\frac{1}{N}\sum_{i=1}^{n} x_{ij}(y_i - \tilde{\beta}_0 - x_i^T\tilde{\beta}) + \lambda(1-\alpha)\beta_j + \lambda\alpha$$

A similar expression exists if $\tilde{\beta}_j < 0$. Simple calculus shows (Donoho and Johnstone, 1994; Friedman et al., 2007) that the coordinate-wise update has the form

$$\tilde{\beta}_j \leftarrow \frac{S(\frac{1}{N}\sum_{i=1}^{n} x_{ij}(y_i - \tilde{y_i}^{(j)}), \lambda\alpha)}{1+\lambda(1-\alpha)}$$

where $\tilde{y_i}^{(j)} = \tilde{\beta}_0 + \sum_{l\neq j} x_{il}\tilde{\beta}_l$ is the fitted value excluding the contribution from x_{ij}, and hence $y_i - \tilde{y_i}^{(j)}$ the partial residual for fitting β_j. Because of the standardization, $1/n\sum_{i=1}^{n} x_{ij}(y_i - \tilde{y_i}^{(j)})$ is the simple least-squares coefficient when fitting this partial residual to x_{ij}. $S(z,\gamma)$ is the soft-thresholding operator with value

$$sign(z)(|z|-\gamma)_+ = \begin{cases} z-\gamma & \text{if } z>0 \text{ and } \gamma<|z| \\ z+\gamma & \text{if } z<0 \text{ and } \gamma<|z| \\ 0 & \text{if } \gamma \geq |z|. \end{cases}$$

Thus glm net computes the simple least-squares coefficient on the partial residual, applies soft-thresholding to take care of the LASSO contribution to the penalty, and then applies a proportional shrinkage for the ridge penalty. This algorithm is suggested by Van der Kooij (2007).

If weight is added to equation 2.1, we have

$$min_{(\beta_0,\boldsymbol{\beta})\in R^{p+1}}[\frac{1}{2n}\sum_{i=1}^{n} w_i(y_i - \beta_0 - x_i^T\boldsymbol{\beta})^2 + \lambda P_\alpha(\boldsymbol{\beta})] \tag{2.2}$$

And the coordinate-wise update has the form

$$\tilde{\beta}_j \leftarrow \frac{S(\sum_{i=1}^{n} w_i x_{ij}(y_i - \tilde{y_i}^{(j)}), \lambda\alpha)}{\sum_{i=1}^{n} w_i x_{ij}^2 + \lambda(1-\alpha)}$$

The presence of weights does not change the computational costs of algorithm much, as long as the weights remain fixed.

The program *glmnet* is available in the R environment, that is, *glmnet*(X, Y, *weights*, α, *nlambda*, *lambda.min*). It is very simple to use. The program accepts

X, Y data for the regression model, and produces the regularization path over a grid of values for the tuning parameter λ. We could supply λ sequence. However, typical usage is to have the program compute its own λ sequence based on nlambda and lambda.min. nlambda is the number of λ values and default is 100. lambda.min is the smallest value for λ. The default is 0.05 if $n < p$ and 0.0001 if $n > p$. The glmnet computes the solutions for a decreasing sequence of values for λ, starting at the λ_{max} for which the entire vector $\hat{\beta} = 0$. Apart from giving a path of solutions, this scheme exploits warm starts, and leads to a more stable algorithm. From equation 2.1, we see that as λ decreases, the number of covariates entering the model increases.

For low dimensional data, $n > p$, the target function is

$$min_{(\beta_0,\boldsymbol{\beta})\in R^{p+1}} \frac{1}{2n} \sum_{i=1}^{n} w_i (y_i - \beta_0 - x_i^T \boldsymbol{\beta})^2 \tag{2.3}$$

which is similar to (2.2), but without penalty. Since $n > p$, we do not need to use any penalty to shrink some coefficients towards 0 and to perform variable selection which are required for $p \gg n$. The program lm in R could be used to estimate β_0 and $\boldsymbol{\beta}$.

Weights are required in both (2.2) and (2.3). For censored data, weights could be determined by inverse probability of censoring weighting (IPCW) which will be discussed in the next section.

AFT Model and IPCW

The problem of cancer class prediction using gene expression data, which can be formulated as predicting binary or multi-category outcomes, has been studied extensively (Alon et al., 1999; Golub et al., 1999; Alizadeh et al., 2000). There has also been active methodological research in relating gene expression profiles to censored survival phenotypes such as time to cancer recurrence or time to death. In addition to the challenge of high dimensionality of the gene expression data that all statistical methods need to deal with, another major challenge is the incomplete

survival outcome due to limited follow-up time in such studies. Because patients under a follow-up study are typically observed up to a follow-up time, some lifetimes are generally right censored, meaning that they are only known to be larger than a given follow-up time, also called a right censoring time.

The Cox regression model (Cox, 1972) is the most popular method in regression analysis for censored survival data. Much work is based on the Cox model. Li and Luan (2003), treating the negative log cox partial likelihood as a loss function, propose using kernel transformations to overcome the difficulties introduced when $p > n$. Gui and Li (2005b) suggest modification of the transformation outlined by Tibshirani (1997) as the foundation for a Cox proportional hazards variable selection approach based on the LARS (Efron et al., 2004) formulation of the LASSO penalty. A more comprehensive review of related literature can be found in Li (2008). However, the proportional hazards assumption may not be appropriate for certain applications. Also, there are no residuals in the traditional sense for the Cox model.

The accelerated failure time (AFT) model (Wei, 1992) is a useful alternative to the Cox model. The AFT model is an extension of linear regression to the analysis of survival data. It is of methodological and practical interest to develop computationally feasible methods for model fitting and variable selection in the AFT model since it has the advantage of modeling event times directly. In addition, Wei (1992) discussed some advantages of using such AFT models over the Cox regression model, including easy interpretation of the model parameters and better fits for some data sets.

The AFT model is a linear regression model in which the response variable is the logarithm or a known monotone transformation of a failure time (Kalbfleisch and Prentice, 1980). Let T_i be the logarithm of the failure time and X_i a length-p covariate vector for the ith subject in a random sample of size n. The AFT model

assumes

$$T_i = \beta_0 + X_i^T \boldsymbol{\beta} + \varepsilon_i, i = 1, \ldots, n, \tag{2.4}$$

where β_0 is the intercept, $\boldsymbol{\beta} \in \mathbf{R}^p$ is the vector of regression coefficients, and ε_i is the error term. Since T_i is subject to right censoring, we can only observe (Y_i, δ_i, X_i), where $Y_i = min\{T_i, C_i\}$, C_i is the logarithm of the censoring time, and $\delta_i = I\{T_i \leq C_i\}$ is the censoring indicator. It is assumed that a random sample $(Y_i, \delta_i, X_i), i = 1, \ldots, n$, from the same distribution is available. This model has emerged as a useful alternative to the popular Cox proportional hazards model for analyzing censored data, since it provides a direct interpretation of the results in terms of quantification of survival times instead of the more abstract hazard rates. Inference procedures for the regression parameters under the AFT model include the inverse probability of censoring weighting (IPCW) method and Buckley-James method. IPCW does not require iteration in the calculation of the estimator, as opposed to the Buckley-James estimator.

Weighted least squares has been proposed by Zhou (1992b), Stute (1993, 1996). The estimator $\mathbf{b}$ can be expressed as the solution of the estimating equation

$$\sum_{i=1}^{n} w_i (Y_i - X_i^T \mathbf{b}) X_i = 0 \tag{2.5}$$

Two weighting schemes are known in the literature to determine the weights w_i. One is IPCW and the other is based on jumps of the Kaplan-Meier estimator.

Let the ordered failure or censoring times be $Y_{(i)}, i = 1, \ldots, n$, $\delta_{(i)}$ is the censoring indicator δ corresponding to the ith order statistic $Y_{(i)}$. We assume that no person can have a failure time equal to their censoring time. Then, the risk set $R(t)$ (number of persons at risk for failure at time t) can be written as

$$R(t) = \sum_{i=1}^{n} I[Y_{(i)} \geq t].$$

The Kaplan-Meier estimator $\hat{S}(t)$ of the survival function $S(t)$ is

$$\hat{S}(t) = \prod_{\{i|Y_{(i)} \leq t\}} (1 - \frac{\delta_{(i)}}{R(Y_{(i)})})$$

We can also estimate the survival function for censoring times $G(t)$, using the Kaplan-Meier approach but considering failure events as 'censored' observation and censored observations as 'failure'. The Kaplan-Meier estimator of $G(t)$ is then similar to $\hat{S}(t)$, that is,

$$\hat{G}(t) = \prod_{\{i|Y_{(i)} \leq t\}} (1 - \frac{1 - \delta_{(i)}}{R(Y_{(i)})})$$

The IPCW weight w_i is then defined as

$$w_{(i)} = \frac{\delta_{(i)}}{\hat{G}_{(t_i-)}}$$

On the other hand, Stute (1993, 1996) used the jumps in the Kaplan-Meier estimator as weights and they can be expressed as $w^*_{(i)}$, where

$$w^*_{(1)} = \frac{\delta_{(1)}}{n}, \text{ and } w^*_{(i)} = \frac{\delta_{(i)}}{n-i+1} \prod_{j=1}^{i-1} (\frac{n-j}{n-j+1})^{\delta_{(j)}}, i = 2, \ldots, n.$$

IPCW is in fact equivalent to weighting by the jumps of the Kaplan-Meier. This can be shown as follows. For all t,

$$\hat{S}(t)\hat{G}(t) = 1 - \hat{H}(t) \tag{2.6}$$

where $\hat{S}(t)$ and $\hat{G}(t)$ are defined as above and $\hat{H}(t)$ is the empirical distribution based on Y_i. Then from (2.6), we observe that when $t = Y_{(i)}$ with $\delta_{(i)} = 1$, then $w^*_{(i)}\hat{G}(t) = 1/n$, from which it follows that

$$w^*_{(i)} = \frac{\delta_{(i)}}{n\hat{G}(Y_{(i)})} = \frac{w_{(i)}}{n}$$

2.2 Statistical Methods

Problems with IPCW Method for Data with a High Censoring Rate

Wang et al. (2008) apply the elastic net method to the Buckley-James estimator for the AFT model. However, the Buckley-James approach entails an iterative least

squares procedure that suffers from convergence problems and is more computationally intensive. Huang et al. (2006) and Datta et al. (2007) apply LASSO regularization to the IPCW for the AFT model, i.e., $\frac{1}{2n}\sum_{i=1}^{n} w_i(y_i - \beta_0 - x_i^T\boldsymbol{\beta})^2 + \lambda P_\alpha(\boldsymbol{\beta})$ with weights as defined in the previous section. However, simulation studies in Li and Wang (2003) indicate that the estimator obtained by using IPCW method is biased and is particularly bad for data with a high censoring rate. In addition, there is another problem with the application of regularization methods to IPCW. That is, if the highest survival time is censored, then the two weighting schemes above yield $w_{(n)} = 0$. In fact, current studies (Huang et al., 2006; Datta et al., 2007) assign the weight of this observation as 0, which is problematic. We will use the redistribution algorithm to explain this problem in more detail.

Efron's (1967) redistribution to the right algorithm provides a useful heuristic for understanding weights obtained from jumps in the Kaplan-Meier estimator. We illustrate Efron's procedure with sample data for 10 persons undergoing maintained chemotherapy. The data is represented in Figure 2.2, where the X's denote failures and the arrows denote withdrawals (censors). Figure 2.2a and 2.2b represent the last observation fails and censors after order the survival time, respectively. If all 10 patients were failures, then the mass of each failure would be 0.10. In our sample, however, the $3rd$ departure time is censored. Its failure mass of 0.10 is therefore redistributed to the right among the remaining 7 departure times, assigning (0.1)/7 = 0.01429 to each future time, so that each future time now has its original failure mass of 0.10 plus the redistributed amount, for a total of 0.11429. The mass of 0.11429 is assigned to $4th$ and $5th$ departure time. When we reach the next censored time at t_6, the total mass of 0.11429 is redistributed to the right over the remaining 4 departure times, so that each one now has 0.11428+(0.11429)/4=0.14286. The weight of 0.14286 is assigned to t_7 and t_8. Finally, the total mass of 0.14286 assigned to the last censored time t_9 is redistributed over the one remaining failure time at

t_{10}, for a total of 0.28571 at that point (Figure 2.2a). As a check on the arithmetic of this procedure, we have weights at our 10 departure times (censor as well as failure) of 0.10, 0.10, 0, 0.11429, 0.11429, 0, 0.14286, 0.14286, 0, 0.28571. And these weights add up to 1. By contrast, if the last departure time (t_{10}) instead of t_9 is censored (Figure 2.2b), then the weight of 0.14286 of the departure time t_{10} cannot be redistributed to larger failure times since larger failure times do not exist. In this case, the weights for our 10 departure times would be 0.10, 0.10, 0, 0.11429, 0.11429, 0, 0.14286, 0.14286, 0.14286, 0. The sum of the weights is 0.85716, which is less than 1. The weight of 0.14286 for the last observation is absent in this case. For the data with a high censoring rate, for example, the data in the simulation study, when the last observation is censored, then $w_{(n)} = 0$, which implies that weight of about 0.3 is discarded. Two reasons why this is wrong are as follows. First, if $w_{(n)} = 0$, then the term of $w_{(n)}(Y_{(n)} - X_{(n)}^T\beta)^2$ is equal to 0. In other words, this term will be missing in the sum of weighted residual squares. Since $(Y_{(n)} - X_{(n)}^T\beta)^2$ is always non-negative, therefore $w_{(n)} = 0$ would lead to a smaller sum of weighted residual squares compared to the true value. This results in an inefficient estimation of β. Second, weighting the last observation as 0 implies that this observation is deleted in the study. Such deletion is not random since it always occurs at $w_{(n)}$, so that the largest observed survival time $Y_{(n)}$ is deleted. This results in a biased estimation of β and a biased estimation of the sum of weighted residual squares.

This problem also exists for low dimensional data. However, no study has yet been done to investigate the problem. Therefore in the next section, for high dimensional data, we will apply glmnet regularization to IPCW with focus on different methods of weighting the last censored observation, after ordering the survival times. We also investigate the low dimensional case.

Weighting Methods

Two weighting approaches are proposed in this section. One approach modifies $\delta_{(n)}$ (methods 1 to 4 below). The other approach fixes $\delta_{(n)} = 1$ and modifies $Y_{(n)}$ (methods 5 to 7 below).

Modifying $\delta_{(n)}$

Method 1 (weight 1): Treat the last observation as uncensored, i.e., $\delta_{(n)} = 1$, so $w_{(n)} = \frac{1}{n\hat{G}(Y_{(n)}-)}$

Method 2 (weight 0): Treat the last observation as censored, i.e., $\delta_{(n)} = 0$, so $w_{(n)} = 0$ (this is the currently used method).

Method 3 (weight average): Average the weight from method 1 and weight from method 2, that is $w_{(n)} = \frac{0.5}{n\hat{G}(Y_{(n)}-)}$

Method 4 (weight depending): The explanation of this method is more involved than the explanation of methods 1 - 3. Since the last observation is censored, the true survival time is larger than the observed survival time. The ideal approach would therefore obtain an estimated survival time greater than observed. Hence, we let the residual of $Y_{(n)}$ be 0 if the estimated survival time is in the correct direction. Based on this guideline, we first set $\delta_{(n)} = 0$ and $\delta_{(n)} = 1$ and we designate the resulting two estimates of β as β^0 and β^1 respectively (Note that β^0 and β^1 absorb the intercept term). Then depending on the sign of the residuals, we have three possible cases, as follows:

$$w_{(n)} = \begin{cases} 0 & \text{if } (Y_{(n)} - X_{(n)}^T\beta^0) < 0 \\ \frac{1}{n\hat{G}(Y_{(n)}-)} & \text{if } (Y_{(n)} - X_{(n)}^T\beta^0) \geq 0 \text{ and } (Y_{(n)} - X_{(n)}^T\beta^1) \geq 0 \\ \frac{0.25}{n\hat{G}(Y_{(n)}-)} & \text{if } (Y_{(n)} - X_{(n)}^T\beta^0) \geq 0 \text{ and } (Y_{(n)} - X_{(n)}^T\beta^1) < 0 \end{cases}$$

In case 1, where $(Y_{(n)} - X_{(n)}^T\beta^0) < 0$, the direction is correct, then we consider the residual of the last observation to be 0. This implies that $w_{(n)}(Y_{(n)} - X_{(n)}^T\beta) = 0$, which in turn implies that $w_{(n)} = 0$ and we proceed accordingly. In case 2, where

$(Y_{(n)} - X_{(n)}^T\beta^0) \geq 0$ and $(Y_{(n)} - X_{(n)}^T\beta^1) \geq 0$, treating the last observation as censored pushes the estimator in the wrong direction, so we would not want to weight this observation as 0. Instead, we treat the last observation as uncensored, as in method 1 above. This still pushes the estimation in the wrong direction, but the result will be less biased than case 1 where we set $w_{(n)} = 0$. Then we calculate $w_{(n)}$ as in method 1, that is, $w_{(n)} = \frac{1}{n\hat{G}(Y_{(n)}-)}$. In the case 3, where $(Y_{(n)} - X_{(n)}^T\beta^0) \geq 0$ and $(Y_{(n)} - X_{(n)}^T\beta^1) < 0$, we need to treat the last observation as uncensored since $(Y_{(n)} - X_{(n)}^T\beta^0) \geq 0$. However, if the last observation is uncensored, then $(Y_{(n)} - X_{(n)}^T\beta^1) < 0$, so residual equal to 0 based on the guideline mentioned before, which is equivalent to $w_{(n)} = 0$. Hence, we find ourselves in an 'oscillation' as follows: $\delta_{(n)} = 0 \rightarrow$ estimator in wrong direction $\rightarrow \delta_{(n)} = 1 \rightarrow$ estimator in right direction, but residual is $0 \rightarrow$ equivalent to $\delta_{(n)} = 0$. In order to avoid this oscillation, we choose some arbitrary constant between 0 and 1, i.e., 0.25, and let $w_{(n)} = \frac{0.25}{n\hat{G}(Y_{(n)}-)}$ for case 3. Simulations suggest that case 1 and case 2 are most common, and case 3 is less common.

Modifying $Y_{(n)}$

This approach fixes $\delta_{(n)} = 1$ and uses different methods to impute $Y_{(n)}$. For the censored $Y_{(n)}$, we know that the observation will fail at some time after $Y_{(n)}$. So we let $\delta_{(n)} = 1$ and $Y_{(n)}^* = Y_{(n)} + c$, where c is a non negative number which can be determined by various methods. We then replace $Y_{(n)}$ by $Y_{(n)}^*$ in (2.2), that is

$$min_{(\beta_0,\boldsymbol{\beta})\in R^{p+1}}[[\frac{1}{2n}\sum_{i=1}^{n-1} w_i(Y_i - \beta_0 - x_i^T\boldsymbol{\beta})^2 + w_{(n)}(Y_{(n)}^* - \beta_0 - x_{(n)}^T\boldsymbol{\beta})^2] + \lambda P_\alpha(\boldsymbol{\beta})] \quad (2.7)$$

For low dimensional data, we remove the penalty so that (2.7) becomes

$$min_{(\beta_0,\boldsymbol{\beta})\in R^{p+1}}[\frac{1}{2n}\sum_{i=1}^{n-1} w_i(Y_i - \beta_0 - x_i^T\boldsymbol{\beta})^2 + w_{(n)}(Y_{(n)}^* - \beta_0 - x_{(n)}^T\boldsymbol{\beta})^2] \quad (2.8)$$

Methods 5 to 7 below use different approaches to identify c. The rationale is to impute the least influential value of $Y_{(n)}$, consistent with observed $Y_{(n)}$. Other imputation

rationales are possible, but without a model they are not 'clean'. At least the following imputation rationales are more reasonable than taking $\delta_{(n)} = 0$, which is equivalent to delete this observation.

Method 5 (Constant adding):

Step 1: A sequence of non-negative c are generated and the corresponding $Y^*_{(n)} = Y_{(n)} + c$ are imputed. If (residual of $Y_{(n)}$) < 0, then the sequence of c is generated from 0 to (|residual| +3) with an increment of 0.01. If (residual of $Y_{(n)}$) > 0, the sequence of c is generated from 0 to 100 with an increment of 1.

Step 2: For each c, a weighted residual sum of squares (wRSS) is calculated from equation 2.7 (or 2.8).

Step 3: The c that produces the smallest wRSS is chosen as the optimal value of c.

Step 4: The optimal c is added to $Y_{(n)}$ to produce an estimator of β according to equation 2.7 (or 2.8) based on updated $Y_{(n)}$.

Methods 6 and 7 below use an idea similar to Buckley-James to impute a value for $Y_{(n)}$. Buckley-James (1979) proposed a least squares estimator taking censoring into account. For un-censored data, solving the equation $\sum_i^n X_i^T(Y_i - X_i^T\beta) = 0$ or $\sum_i^n X_i^T e_i = 0$ results in the least squares estimator of β. Similarly, if $\delta = 0$, let $E(\varepsilon_i|\varepsilon_i > e_i)$ replace e_i, that is

$$\begin{aligned} E(\varepsilon_i|\varepsilon_i > e_i, X_i) &= \int_{-\infty}^{\infty} \varepsilon_i dF(\varepsilon_i|\varepsilon_i > e_i) \\ &= \int_{-\infty}^{\infty} \varepsilon_i p(x < \varepsilon_i < x + \Delta x|\varepsilon_i > e_i) \\ &= \int_{-\infty}^{\infty} \varepsilon_i \frac{p(x < \varepsilon_i < x + \Delta x, \varepsilon_i > e_i)}{p(\varepsilon_i > e_i)} \\ &= \int_{e_i}^{\infty} \varepsilon_i \frac{dF(\varepsilon_i)}{1 - F(e_i)} \\ &= \sum_{j:e_j > e_i} \frac{e_j \Delta \hat{F}(e_j)}{1 - \hat{F}(e_i)} \end{aligned} \tag{2.9}$$

where $\hat{F}(.)$ is the Kaplan-Meier estimator computed from (e_i, δ_i).

Method 6 (Conditional mean adding):

Step 1 : Set $\delta_{(n)} = 1$ and get an estimator of β from equation 2.7 (or 2.8)

Step 2: Calculate residuals based on the estimated β from Step 1 and compute the Kaplan-Meier estimator of the residuals according to equation 2.9

Step 3: Add the Kaplan-Meier estimator of the residuals to $Y_{(n)}$

Step 4: Get an improved new estimator of β based on imputed $Y_{(n)}$ from equation 2.7 (or 2.8).

Method 7 (Conditional median adding): It is the same as method of conditional mean adding except that it replaces $E(\varepsilon_i|\varepsilon_i > e_i)$ by median$(\varepsilon_i|\varepsilon_i > e_i)$ and adds median$(\varepsilon_i|\varepsilon_i > e_i)$ to $Y_{(n)}$.

2.3 Simulation Studies

$p < n$ Case

For the simulation studies we first consider the low dimensional case ($p < n$). We focus on the situation that the highest observation is censored ($\delta_{(n)} = 0$). The logarithm of the true survival time is simulated by

$$T = X^T\beta + \sigma\epsilon, \text{ where } \epsilon \sim N(0,1). \tag{2.10}$$

We choose $n = 100$, $p = 4$ with $X = (X_1, X_2, X_3, X_4)$ where $X_1 = 1$, $X_2 \sim Bin(1, 0.5)$, $X_3 \sim N(140, 60)$ and $X_4 \sim U(30, 70)$. Since $X_1 = 1$, therefore β_1 is the intercept. We choose $\beta = (100, 5, -0.5, -0.5)$ and $\sigma = 15$. The logarithm of censoring time C is generated from both a uniform distribution $U(-65, 80)$ and from an extreme value distribution, in both cases such that a 50% censoring rate is achieved. The observed log-transformed survival time is $Y = min(T, C)$. 1000 runs are simulated.

For prediction performance, since the true survival time for each subject is available in simulated data, we use relative prediction error (RPE) obtained from an independent test data set, where $RPE = (1/n)\sum_{i=1}^{n}(T_i - \beta_0 - X_i^T\hat{\beta})^2/\sigma^2$ and $\hat{\beta}$ is obtained from a training data set. For each of the following simulations, we generate two independent data sets: a training set of size n_1 for model fitting and a test set of size n_2 for RPE calculation. In this simulation we choose $n_1 = 100$ and $n_2 = 400$. The bias, variance, MSE and RPE are summarized in Table 2.2.

The column 'uncensored' contains the true survival time for each patient. That is why it has smallest bias, variance, MSE for all βs, and smallest RPE as well. Columns 3 - 9 compare 7 different methods for handling the last censored patient. The methods in columns 3 - 6 modify $\delta_{(n)}$ and the methods in columns 7 - 9 modify $Y_{(n)}$. The method of conditional mean adding gives the smallest bias for all β's among 7 methods. The method of weight 0 gives the smallest variance for all β's. Weight depending method gives the smallest mean square error (MSE) for β_1 and β_3, while weight 0 method gives the smallest MSE for β_2 and β_4. Since weight 0 method discards the largest observation ('outlier'), it behaves like a shrinkage estimator, which may make the variance smaller than other 6 methods. MSE is determined by bias and variance. If the proportion of variance is large, then MSE is mostly determined by variance. However, if proportion of bias is large, then bias dominates the MSE. That is one possible reason that weight depending method obtains smallest MSE for β_1 and β_3 since bias has a big influence on MSE, and weight 0 method obtains smallest MSE for β_2 and β_4 since variance contributes more to the MSE.

The methods of weight depending and constant adding end up with exactly the same results. Recall that if $R_{(n)} < 0$ ($R_{(n)}$ represents residual of last censored patient), method of weight depending keeps $w_{(n)} = 0$. Method of constant adding chooses c to be $\mid R_{(n)} \mid$ and adds c back to $Y_{(n)}$ which makes updated $R_{(n)} = 0$. The last term in equation 2.8 is $w_{(n)}R_{(n)}$. Hence, either $w_{(n)} = 0$ or $R_{(n)} = 0$ results in same effect:

throwing away the last term in equation 2.8. If $R_{(n)} > 0$, method of constant adding chooses c = 0 and changes $\delta_{(n)} = 1$. It is the same as method of weight depending which modifies $\delta_{(n)}$ to be 1.

Similar patterns are observed for censoring times simulated by the extreme value distribution (Table 2.3). However, in this case the method of weight average gives the smallest variance for all βs and the smallest MSE for β_2 and β_4.

$p >> n$ Case

Group Selection of Correlated Covariates

We now consider the high dimensional case ($p >> n$). As before we focus on the situation that the highest observation is censored. We use 50 samples simulated from model 2.10 with $p = 120$ and $\sigma = 15$. The first 60 coefficients are nonzero and are drawn from $N(3, 0.5)$; and their values are then fixed for all simulation runs. The remaining 60 coefficients are set to zero. The covariate matrix X is generated from a multivariate normal distribution with mean zero and covariance matrix as

$$\mathbf{\Sigma} = \begin{pmatrix} \Sigma_0 & & & \\ & \Sigma_0 & 0.2J & \\ & 0.2J & \Sigma_0 & \\ & & & \Sigma_0 \end{pmatrix}$$

where Σ_0 is a 30×30 matrix with diagonal elements equal to 1 and off-diagonal elements equal to 0.7; and J is a 30×30 matrix with all elements equal to 1. The logarithm of censoring time C is generated from both a uniform distribution $U(-190, 190)$ and from an extreme value distribution to yield a 50% censoring rate. 1000 runs are simulated. Let number of λ to be 100 to include as many covariates as possible in the model. For each covariate we record the frequency of being selected among 1000 simulation runs, the mean and the standard deviation. We assess $\alpha = 0, 0.1, 0.5, 0.9, 1$. The results are summarized in Table 2.4.

We use method of weight depending in Table 2.4 to illustrate the selection frequency at different α values. The coefficients in blocks 1 and 2 are nonzero, so the corresponding covariates in these two blocks should be selected. By contrast, the coefficients in blocks 3 and 4 are zero, so the covariates in these two blocks should not be included in the model. $\alpha = 0$ (ridge regression) selects almost all covariates in all 4 blocks. By contrast, $\alpha = 1$ (LASSO) includes essentially no covariate in blocks 3 and 4, but it only selects about one third of the covariates in blocks 1 and 2. $\alpha = 0.1$ gives a good compromise: about 80%(24.3/30 or 24.4/30) covariates in blocks 1 and 2 are correctly selected while about 5% (1.7/30 or 1.2/30) of the covariates in blocks 3 and 4 are falsely selected. Similar patterns are observed in the uncensored case and in the other 6 methods.

The uncensored case selects the largest number of informative covariates and selects the smallest number of non-informative covariates for all α's.

Since $\alpha = 0.1$ is a good compromise between LASSO and ridge regression, we compare 7 imputation methods for $\alpha = 0.1$. The method of weight depending performs the best among the 4 methods modifying $\delta_{(n)}$. The method of constant adding performs the best among the 3 methods modifying $Y_{(n)}$. In fact, the method of constant adding performs best among all 7 methods. The methods of weight depending and constant adding do not perform identically (as they did in the low dimensional case).

Similar results (Table 2.5) are obtained where the log of the censoring time C is generated from an extreme value distribution.

Comparison of Prediction Performance

We now compare the prediction performance of the different methods. We generate log-transformed survival time from equation 2.10. We also generate log-transformed censoring time from both an extreme value distribution and from a uniform distribu-

tion (U (-20,20) in example 1 and U (-190,190) in example 2). In all 4 cases such that a 50% censoring rate is achieved. We use relative prediction error (RPE) mentioned in low dimensional case to evaluate the prediction performance.

Example 1 and example 2 represent the case of $p > n$ with several groups of correlated covariates as follows:

Example 1 has the same simulation setting as the example of group selection of correlated covariates with $n_1 = 50$ and $n_2 = 400$.

Example 2 considers two groups of moderately correlated covariates with $n_1 = 50$, $n_2 = 400$, and $p = 120$. We choose $\sigma = 5$ and set the first six slope parameters to be $(3, 3, 2, 3, 3, 2)$ and all other 114 slope parameters to be zero. The first three covariates comprise a group and the next three comprise another group. Within each group, the pairwise correlation between any two predictors X_{j1} and X_{j2} is 0.5.

We conduct 1000 simulations for each example. The RPE values and corresponding standard deviations are listed in Table 2.6 for uniform censoring distribution and Table 2.7 for extreme value censoring distribution.

The uncensored case always obtains not only the smallest RPE, but also the smallest standard deviation for any (α, λ) pair in study. The method of weight depending obtains the smallest RPE as well as the smallest standard deviation among the 4 methods modifying $\delta_{(n)}$. When λ is large, corresponding to only a few covariates entering the model, the method of weight depending obtains exactly the same RPE as method of weight 1. As more and more covariates enter the model, method of weight depending obtains a smaller RPE than method of weight 1. The comparisons of methods modifying $\delta_{(n)}$ are shown in Figure 2.3a. Method of weight 0 or current method (black line) always has the largest RPE. The method of weight 1 (green line) overlays with method of weight depending (red line) for large λ but falls above the method of weight depending for small λ. For methods modifying $Y_{(n)}$, method of conditional mean adding performs better than other 2 methods when λ is large.

However, when λ is small, method of constant adding performs better than the other two. In Figure 2.3b, method of conditional mean adding (dark red line) obtains smallest RPE at $25th$ and $50th$ λ, but thereafter rises and falls above the method of constant adding (dark green line) at $100th$ λ.

Here we just evaluate 5 particular values of λ ($10th$, $25th$, $50th$, $75th$ and $100th$ λ) for each fixed α. For all methods, as the value of λ decreases ($10th$ $\lambda \rightarrow 25th$ $\lambda \rightarrow 50th$ $\lambda \rightarrow 75th$ $\lambda \rightarrow 100th$ λ), the number of non-zero covariates increases and RPE decreases, representing a better prediction.

Similar results are observed for censoring times generated from an extreme value distribution as shown in Table 2.7.

Example 2 (Table 2.8 and Table 2.9) has results a little bit different from example 1. For methods modifying $\delta_{(n)}$, method of weight 1 and weight average perform similarly and both are better than method of weight depending in prediction. Three methods modifying $Y_{(n)}$ obtain close RPEs with method of conditional mean adding performing slightly better than the other two methods.

Summary of Simulation Studies

We propose 7 methods (4 methods modify $\delta_{(n)}$ and 3 methods modify $Y_{(n)}$) to improve the estimation and prediction of the IPCW method with AFT model. The improvement is especially profound for data with a high censoring percentage.

For the low dimensional case, method of weight depending performs best among the methods modifying $\delta_{(n)}$ in terms of estimation bias and prediction error. For methods modifying $Y_{(n)}$, method of conditional mean adding obtains the least bias and method of constant adding obtains smallest RPE.

Elastic net regularization is applied to the weighted residual sum of squares in high dimensional data. Method of constant adding performs best in selecting correlated covariates among all 7 methods. In example 1, method of weight depending

obtains smallest RPE among methods modifying $\delta_{(n)}$. For methods modifying $Y_{(n)}$, method of conditional mean adding obtains smallest RPE when few covariates enter the model and method of constant adding obtains smallest RPE when many covariates are included in the model. By contrast, in example 2, for methods modifying $\delta_{(n)}$, methods of weight 1 and weight average perform similarly in prediction and both are better than method of weight depending. Three methods modifying $Y_{(n)}$ perform similarly in prediction.

We recommend these modifications to the IPCW method at all times. The additional computational cost that the two methods brought about is minimal to modest.

2.4 Squamous Cell Lung Carcinoma Data Analysis

Non-small-cell lung cancers (NSCLC) compose about 80% of all lung carcinomas with squamous cell carcinomas (SCC) and adenocarcinoma representing the majority of these tumors. Although patients with early-stage NSCLC typically have a better outcome, 35% to 50% will relapse within 5 years after surgical treatment. The goal of the Michigan squamous cell lung carcinoma study is to predict the survival of early-stage lung cancer patients using microarray gene expression data. The data can be download from Gene Expression Omnibus (GEO) website (http://www.ncbi.nlm.nih.gov/sites/GD Sbrowser?acc=GDS2373). In this study, 130 lung SCC samples from 129 individual patients (LS-71 and LS-136 were duplicate samples from different areas of the same tumor) from all stages of squamous cell lung carcinoma are evaluated. The censor rate is 48.1% with the last subject being censored after sorting the survival time. RNA samples are analyzed by using Affymetrix U133A microarray chips. Since LS-71 and LS-136 are duplicate, the average of log gene expression levels of these two samples were used for analysis. Gene expression values are log transformed. Those probe sets expressed at extremely low levels were excluded from the final data set, that is, all probe sets with a measure of their 75th

percentile value less than 100 were excluded from further study. The rest of 18082 probe sets are assessed by running univariate weighted linear regression (treating last censored subject to be dead) on all 129 subjects, 1000 probe sets with the smallest P-values are selected (corresponding to P-value cut off at 0.0000297).

Subjects are randomly divided into a training set that has 65 subjects with the last subject to be censored after sorting the survival time and a test set that has 64 subjects. Starting with these 1000 probe sets, the AFT model is fitted by glmnet using different last observation weighting methods on the training data set.

We use the V-fold cross-validation (Stone, 1974; Wahba, 1990) to determine the tuning parameters. For a pre-defined integer V, partition the data randomly into V nonoverlapping subsets of equal sizes. We define the cross-validation (CV) score and the AIC (Akaike, 1973) type of score as

$L = \frac{1}{2n} \sum_{i=1}^{n} w_i (y_i - \beta_0 - X_i^T \beta)^2$

CV score $= \sum_{v=1}^{V} [L(\hat{\beta}^{(-v)}) - L^{(-v)}(\hat{\beta}^{(-v)})]$

AIC score = log (CV score)+2K/n

Here $\hat{\beta}^{(-v)}$ is glmnet estimator of β based on the data without the vth subset, $L^{(-v)}$ is the function L evaluated without the vth subset. K is the number of nonzero coefficients in $\hat{\beta}$ for fixed (α, λ). α and λ are chosen to minimize the AIC score. For this data set, we let α range from 0 to 1 with 0.1 increment. For each fixed α, 100 program automatically generated λs are fitted. Hence one out of 1100 pairs of (α, λ) is selected to minimize AIC score. The optimal pair of (α, λ) determined by the training set is (0.5, 93).

Methods of weight 0 and weight 1 are fitted to training data set. Weight 0 selects 35 probe sets and weight 1 selects 30 probe sets with 25 probe sets are in common. The βs obtained from methods of weight 0 and weight 1 are used to predict the survival time in test data set respectively. 3-year survival time is used as cut off to distinguish the high risk group from low risk group. A subject is assigned to the

high-risk group if the predicted survival time is less than 3 years, or to the low-risk group otherwise. Kaplan-Meier curves for these two groups are plotted in Figure 2.4 and log rank tests are performed.

Figure 2.4 shows that method of weight 1 uses less probe sets but separates survival curves of high/low risk groups better than method of weight 0. When we look at the cut off survival time at 3 years corresponding to 7 in the scale of log(days), this separation also performs better using method of weight 1 than using method of weight 0 (the length of black vertical line is larger in weight 1 than that in weight 0).

The identified 30 probe sets from method of weight 1 could be the lung cancer biomarkers, a small subset of genes that distinguish normal tissue from cancer tissue. The biomedical processes that these probe sets are involved in deserve the further investigation.

The observed survival time after log-transformation is 8.37. Methods of weight 0 and weight 1 get estimated survival time equal to 8.21 and 8.35 respectively. Both methods underestimate the true survival time, but method of weight 1 obtains less bias. According to the method of weight depending, this training data set belongs to the second case in the method of weight depending. Hence, the method of weight depending prefer to treat last observation as 'death', which is the same as method of weight 1.

We also apply method of constant adding to this data set, it turns out that c is equal to 0, implying that constant adding is equivalent to method of weight 1. The methods of conditional mean adding and conditional median adding are also applied to this data set. The method of conditional mean adding adds 5.44 days and method of conditional median adding adds 14.45 days to $Y_{(n)}$, respectively. Based on updated $Y_{(n)}$, (α, λ) are selected using minimizing AIC score mentioned above. It turns out that (α, λ) are (0.8, 100) and (1, 99) for methods of conditional mean adding and conditional median adding, respectively. The β obtained from these two methods are

used to predict survival time of patients in test data set. Using method of conditional mean adding, the separation between high/low risk groups is better than method of weight 0 and worse than method of weight 1. The high/low risk groups are not well separated for method of conditional median adding and logrank test shows the differences of survival time between two groups are not significant (Figure 2.4). From simulation study, we know that as value of λ decreases, i.e., $100th$ λ, the method of constant adding performs better in prediction than methods of conditional mean and conditional median adding. In this data set, the $99th$ or $100th$ λ is chosen by cross validation, which makes the method of constant adding (c = 0) better predict survival time than the other two methods modifying $Y_{(n)}$ (methods of conditional mean/median adding). The result for method of conditional median adding is much worse than method of conditional mean adding. One possible reason is the survival time in this data set could be normally distributed, adding conditional median is not as robust as adding conditional mean.

Therefore, for this particular training data set, method of weight 1 performs better than method of weight 0 (current method). Method of constant adding performs better than methods of conditional mean and conditional median adding. Methods of weight depending and constant adding yield same results as method of weight 1.

Table 2.1: The dimensionality of the microarray data

patient	X_1	X_2	...	X_{10000}	Y
1	100	500	$\cdots$	1500	151
2	500	300	$\cdots$	900	75
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
100	1000	550	$\cdots$	1000	57

Here $X_j, j = 1, \ldots, 10000$ is jth gene expression level, $Y = (Y_1, \ldots, Y_{100})$ is response variable.

Table 2.2: Comparison of the different methods using uniform censoring time

$\beta's$		Methods						
	uncensored	weight 1	weight 0	weight average	weight depending	constant adding	mean adding	median adding
Bias								
β_1	-0.05	-5.30	-6.07	-5.76	-2.52	-2.52	0.25	-1.20
β_2	0.05	-0.31	-0.22	-0.29	-0.12	-0.12	-0.02	-0.08
β_3	-0.001	0.023	0.025	0.024	0.010	0.010	-0.002	0.004
β_4	0.003	0.026	0.029	0.028	0.012	0.012	-0.001	0.007
Variance								
β_1	62.3	208.5	172.9	179.0	182.5	182.5	214.5	201.1
β_2	8.5	23.8	22.3	22.3	24.0	24.0	28.0	25.7
β_3	0.0007	0.0023	0.0017	0.0028	0.0017	0.0017	0.0021	0.0019
β_4	0.018	0.048	0.044	0.045	0.047	0.047	0.054	0.050
MSE								
β_1	62.3	236.6	209.7	212.2	188.8	188.8	214.5	202.5
β_2	8.5	23.9	22.4	22.4	24.0	24.0	28.0	25.7
β_3	0.0007	0.0028	0.0023	0.0024	0.0018	0.0018	0.0021	0.0020
β_4	0.018	0.049	0.045	0.046	0.047	0.047	0.054	0.050
RPE	1.017	1.107	1.096	1.096	1.085	1.085	1.099	1.092

Table 2.3: Comparison of the different methods using censoring time with extreme value distribution

$\beta's$	uncensored	weight 1	weight 0	weight average	Methods weight depending	constant adding	mean adding	median adding
Bias								
β_1	0.30	-3.68	-4.46	-4.10	-1.62	-1.62	0.62	-0.64
β_2	-0.03	-0.16	-0.22	-0.19	-0.08	-0.08	0.02	-0.04
β_3	-0.000	0.017	0.021	0.019	0.007	0.007	-0.003	0.003
β_4	-0.003	0.018	0.018	0.018	0.007	0.007	-0.003	-0.003
Variance								
β_1	62.5	175.0	171.1	163.0	165.9	165.9	181.0	172.7
β_2	9.2	21.7	20.8	20.6	22.0	22.0	25.1	23.1
β_3	0.0007	0.0019	0.0018	0.0017	0.0017	0.0017	0.0019	0.0018
β_4	0.017	0.042	0.042	0.041	0.042	0.042	0.045	0.043
MSE								
β_1	62.6	188.5	191.0	179.8	168.6	168.6	181.4	173.1
β_2	9.2	21.7	20.8	20.7	22.0	22.0	25.1	23.1
β_3	0.0007	0.0022	0.0023	0.0020	0.0018	0.0018	0.0019	0.0018
β_4	0.017	0.042	0.042	0.041	0.042	0.042	0.045	0.043
RPE	1.025	1.095	1.097	1.091	1.087	1.087	1.094	1.089

Table 2.4: Selection frequency at different α's (uniform censoring)

Block Number	α 0	0.1	0.5	0.9	1
Uncensored					
Block 1	30.0(0.00)	27.9(1.46)	18.7(2.13)	15.6(2.03)	15.2(1.99)
Block 2	30.0(0.00)	27.8(1.44)	18.6(2.13)	15.6(2.04)	15.2(2.01)
Block 3	23.9(5.55)	0.8(1.43)	0.2(0.64)	0.2(0.56)	0.2(0.56)
Block 4	19.0(5.14)	0.3(0.76)	0.1(0.35)	0.1(0.32)	0.1(0.31)
Method of weight 1					
Block 1	29.9(0.50)	24.0(2.89)	12.7(2.52)	9.2(2.11)	8.7(2.03)
Block 2	29.9(0.55)	23.6(2.81)	12.8(2.45)	9.1(2.11)	8.6(2.08)
Block 3	24.7(3.99)	2.0(2.34)	0.5(0.96)	0.4(0.78)	0.4(0.74)
Block 4	23.3(4.06)	1.5(1.87)	0.3(0.73)	0.3(0.59)	0.3(0.59)
Method of weight 0					
Block 1	29.8(0.59)	23.7(2.89)	12.6(2.45)	9.3(2.07)	8.8(2.07)
Block 2	29.8(0.81)	23.6(2.81)	12.6(2.39)	9.4(2.14)	8.8(2.10)
Block 3	24.4(3.94)	2.0(2.34)	0.6(1.01)	0.4(0.80)	0.4(0.75)
Block 4	23.2(3.90)	1.5(1.87)	0.4(0.82)	0.3(0.65)	0.3(0.62)
Method of weight average					
Block 1	29.9(0.45)	23.8(2.92)	12.7(2.46)	9.3(2.07)	8.8(2.07)
Block 2	29.9(0.64)	23.7(2.98)	12.8(2.41)	9.4(2.14)	8.8(2.10)
Block 3	24.3(3.97)	1.9(2.22)	0.5(0.98)	0.4(0.80)	0.4(0.75)
Block 4	22.8(3.99)	1.3(1.75)	0.4(0.73)	0.3(0.65)	0.3(0.62)
Method of weight depending					
Block 1	29.9(0.50)	24.3(2.78)	13.0(2.44)	9.4(2.07)	8.9(2.03)
Block 2	29.9(0.55)	24.4(2.84)	13.1(2.40)	9.4(2.11)	8.9(2.10)
Block 3	24.7(3.99)	1.7(2.20)	0.5(0.92)	0.4(0.74)	0.3(0.71)
Block 4	23.3(4.06)	1.2(1.63)	0.3(0.66)	0.2(0.56)	0.2(0.55)
Method of constant adding					
Block 1	29.9(0.50)	25.0(2.82)	13.1(2.49)	9.3(2.09)	8.8(2.01)
Block 2	29.9(0.55)	24.9(2.86)	13.2(2.38)	9.3(2.06)	8.7(2.00)
Block 3	24.7(3.99)	1.6(2.22)	0.4(0.87)	0.3(0.69)	0.3(0.66)
Block 4	23.3(4.06)	1.0(1.54)	0.2(0.56)	0.2(0.49)	0.2(0.46)
Method of mean adding					
Block 1	29.9(0.61)	24.2(2.93)	12.9(2.52)	9.3(2.07)	8.8(2.01)
Block 2	29.8(0.66)	24.1(3.09)	12.9(2.46)	9.3(2.08)	8.7(2.06)
Block 3	24.9(3.89)	1.7(2.26)	0.5(0.95)	0.3(0.72)	0.3(0.71)
Block 4	23.5(3.98)	1.1(1.61)	0.3(0.64)	0.2(0.51)	0.2(0.49)
Method of median adding					
Block 1	29.9(0.49)	24.2(2.94)	12.8(2.49)	9.2(2.08)	8.7(2.03)
Block 2	29.9(0.58)	24.1(3.12)	12.9(2.45)	9.2(2.08)	8.7(2.04)
Block 3	24.7(3.99)	1.7(2.23)	0.5(0.96)	0.4(0.72)	0.3(0.72)
Block 4	23.3(4.06)	1.1(1.61)	0.3(0.68)	0.2(0.55)	0.2(0.52)

Table 2.5: Selection frequency at different α's (extreme value censoring)

Block Number	α = 0	0.1	0.5	0.9	1
Uncensored					
Block 1	30.0(0.00)	27.9(1.42)	18.8(2.13)	15.7(2.04)	15.2(2.05)
Block 2	30.0(0.00)	27.9(1.38)	18.6(2.10)	15.6(1.98)	15.1(1.98)
Block 3	23.7(5.48)	0.8(1.38)	0.2(0.60)	0.2(0.56)	0.2(0.55)
Block 4	19.4(5.27)	0.4(0.89)	0.1(0.45)	0.1(0.38)	0.1(0.37)
Method of weight 1					
Block 1	29.9(0.43)	24.2(3.07)	12.9(2.59)	9.4(2.15)	8.8(2.08)
Block 2	29.9(0.43)	24.0(2.99)	12.8(2.53)	9.3(2.14)	8.7(2.10)
Block 3	24.3(4.07)	1.7(2.19)	0.5(0.91)	0.4(0.71)	0.3(0.70)
Block 4	23.3(3.97)	1.3(1.79)	0.4(0.75)	0.3(0.64)	0.3(0.62)
Method of weight 0					
Block 1	29.9(0.64)	23.8(2.92)	12.7(2.54)	9.4(2.09)	8.9(2.03)
Block 2	29.9(0.50)	23.8(2.63)	12.6(2.36)	9.3(2.08)	8.8(2.02)
Block 3	24.2(3.95)	1.9(2.31)	0.5(0.98)	0.4(0.82)	0.4(0.78)
Block 4	23.2(3.78)	1.5(1.89)	0.4(0.81)	0.3(0.68)	0.3(0.66)
Method of weight average					
Block 1	30.0(0.64)	23.8(2.92)	12.7(2.54)	9.4(2.09)	8.9(2.03)
Block 2	29.9(0.50)	23.8(2.63)	12.6(2.36)	9.3(2.08)	8.8(2.02)
Block 3	24.2(3.95)	1.9(2.31)	0.5(0.98)	0.4(0.82)	0.4(0.78)
Block 4	23.2(3.77)	1.5(1.89)	0.4(0.81)	0.3(0.68)	0.3(0.66)
Method of weight depending					
Block 1	29.9(0.43)	24.5(2.73)	13.2(2.50)	9.5(2.11)	9.0(2.06)
Block 2	29.9(0.43)	24.5(2.62)	13.0(2.43)	9.4(2.14)	8.9(2.09)
Block 3	24.3(4.08)	1.6(2.15)	0.4(0.85)	0.3(0.71)	0.3(0.69)
Block 4	23.3(3.97)	1.2(1.71)	0.3(0.71)	0.2(0.57)	0.2(0.56)
Method of constant adding					
Block 1	29.9(0.43)	25.1(2.74)	13.3(2.54)	9.4(2.10)	8.9(2.06)
Block 2	29.9(0.43)	24.9(2.71)	13.2(2.46)	9.3(2.14)	8.8(2.07)
Block 3	24.3(4.08)	1.5(2.19)	0.4(0.83)	0.3(0.65)	0.3(0.65)
Block 4	23.3(3.97)	1.0(1.70)	0.3(0.67)	0.2(0.51)	0.2(0.52)
Method of mean adding					
Block 1	29.9(0.57)	24.4(2.98)	13.1(2.55)	9.4(2.14)	8.9(2.12)
Block 2	29.9(0.55)	24.2(3.89)	12.9(2.51)	9.4(2.15)	8.8(2.10)
Block 3	24.6(4.02)	1.6(2.17)	0.4(0.90)	0.3(0.68)	0.3(0.66)
Block 4	23.4(3.97)	1.2(1.79)	0.3(0.77)	0.3(0.64)	0.3(0.61)
Method of median adding					
Block 1	29.9(0.44)	24.3(2.92)	13.0(2.56)	9.4(2.13)	8.9(2.08)
Block 2	29.9(0.44)	24.2(2.87)	12.8(2.51)	9.3(2.12)	8.8(2.08)
Block 3	24.4(4.07)	1.6(2.19)	0.4(0.89)	0.3(0.70)	0.3(0.69)
Block 4	23.3(3.97)	1.2(1.76)	0.3(0.75)	0.3(0.65)	0.3(0.63)

Table 2.6: Comparison of RPE for different methods using uniform (-20, 20) censoring distribution for example 1

		α	
λ	0.1	0.5	0.9
Uncensored			
10th λ	46.02(4.02)	42.16(3.99)	40.79(3.98)
25th λ	25.75(3.38)	22.83(3.06)	22.53(3.05)
50th λ	8.47(1.48)	8.47(1.43)	9.02(1.51)
75th λ	3.30(0.68)	4.03(0.77)	4.55(0.82)
100th λ	1.90(0.41)	2.62(0.49)	2.98(0.53)
Method of weight 1			
10th λ	48.05(4.28)	44.65(4.28)	43.41(4.32)
25th λ	31.75(4.63)	28.82(4.70)	28.56(4.84)
50th λ	14.51(3.90)	14.42(3.70)	15.31(3.88)
75th λ	7.57(2.96)	8.64(2.80)	9.67(3.00)
100th λ	4.92(2.41)	6.23(2.33)	7.13(2.49)
Method of weight 0			
10th λ	51.98(6.31)	48.13(6.17)	46.63(6.12)
25th λ	35.44(6.36)	31.36(6.09)	30.68(6.11)
50th λ	15.67(4.69)	15.20(4.29)	15.84(4.35)
75th λ	7.51(2.94)	8.73(3.00)	9.70(3.17)
100th λ	4.49(2.00)	6.11(2.32)	7.01(2.51)
Method of weight average			
10th λ	48.95(4.67)	45.36(4.66)	44.02(4.69)
25th λ	32.40(4.88)	29.09(4.88)	28.70(6.11)
50th λ	14.47(3.86)	14.30(3.67)	15.09(3.83)
75th λ	7.40(2.83)	8.48(2.73)	9.46(2.94)
100th λ	4.79(2.28)	6.12(2.27)	6.97(2.46)
Method of weight depending			
10th λ	48.05(4.28)	44.65(4.28)	43.41(4.32)
25th λ	31.75(4.63)	28.82(4.70)	28.56(4.83)
50th λ	14.32(3.79)	14.19(3.58)	15.05(3.79)
75th λ	6.93(2.49)	8.15(2.52)	9.17(2.75)
100th λ	4.14(1.69)	5.69(1.90)	6.61(2.13)
Method of constant adding			
10th λ	48.05(4.28)	44.65(4.28)	43.40(4.32)
25th λ	31.75(4.63)	28.82(4.70)	28.56(4.84)
50th λ	14.31(3.80)	14.29(3.65)	15.15(3.81)
75th λ	6.79(2.39)	8.16(2.50)	9.16(2.71)
100th λ	3.86(1.57)	5.53(1.81)	6.55(2.10)
Method of mean adding			
10th λ	48.05(4.42)	44.50(4.41)	43.15(4.48)
25th λ	30.84(4.73)	27.71(4.80)	27.39(4.87)
50th λ	13.36(3.64)	13.35(3.46)	14.25(3.58)
75th λ	6.90(2.60)	8.02(2.51)	9.07(2.68)
100th λ	4.52(2.04)	5.84(2.03)	6.74(2.17)
Method of median adding			
10th λ	48.05(4.28)	44.65(4.28)	43.40(4.32)
25th λ	31.68(4.61)	28.69(4.66)	28.40(4.79)
50th λ	14.22(3.72)	14.10(3.52)	14.95(3.66)
75th λ	7.28(2.70)	8.39(2.61)	9.41(2.81)
100th λ	4.69(2.13)	6.03(2.14)	6.93(2.29)

Table 2.7: Comparison of RPE for different methods using extreme value censoring distribution for example 1

		α	
λ	0.1	0.5	0.9
Uncensored			
10th λ	45.85(4.06)	41.98(4.02)	40.63(3.98)
25th λ	25.65(3.48)	22.70(3.19)	22.40(3.16)
50th λ	8.42(1.50)	8.43(1.47)	8.97(1.57)
75th λ	3.28(0.67)	4.01(0.76)	4.53(0.85)
100th λ	1.90(0.41)	2.61(0.50)	2.97(0.55)
Method of weight 1			
10th λ	48.01(4.40)	44.63(4.30)	43.39(4.28)
25th λ	31.53(4.47)	28.61(4.41)	28.39(4.53)
50th λ	14.11(3.65)	14.13(3.49)	15.03(3.63)
75th λ	7.22(2.71)	8.43(2.67)	9.48(2.83)
100th λ	4.65(2.15)	6.08(2.23)	7.01(2.39)
Method of weight 0			
10th λ	51.76(6.15)	47.93(6.03)	46.44(5.99)
25th λ	35.10(6.12)	31.07(5.79)	30.42(5.73)
50th λ	15.33(4.49)	14.99(4.11)	15.68(4.17)
75th λ	7.30(2.89)	8.63(2.92)	9.58(3.02)
100th λ	4.37(2.04)	6.06(2.32)	6.97(2.47)
Method of weight average			
10th λ	51.76(4.61)	45.35(4.55)	44.01(4.54)
25th λ	32.20(4.66)	28.91(4.57)	28.51(4.64)
50th λ	14.11(3.58)	14.04(3.46)	14.84(3.62)
75th λ	7.08(2.58)	8.30(2.60)	9.29(2.77)
100th λ	4.54(2.05)	6.00(2.18)	6.89(2.35)
Method of weight depending			
10th λ	48.00(4.40)	44.63(4.30)	43.39(4.28)
25th λ	31.53(4.47)	28.60(4.41)	28.37(4.53)
50th λ	13.99(3.62)	13.97(3.43)	14.86(3.58)
75th λ	6.70(2.34)	8.06(2.47)	9.11(2.67)
100th λ	4.00(1.64)	5.63(1.93)	6.58(2.16)
Method of constant adding			
10th λ	48.01(4.40)	44.63(4.30)	43.38(4.28)
25th λ	31.52(4.47)	28.60(4.41)	28.37(4.53)
50th λ	13.91(3.57)	14.00(3.42)	14.88(3.57)
75th λ	6.55(2.33)	8.01(2.48)	9.07(2.67)
100th λ	3.73(1.59)	5.47(1.85)	6.46(2.07)
Method of mean adding			
10th λ	47.92(4.44)	44.39(4.38)	43.03(4.39)
25th λ	30.72(4.52)	27.67(4.42)	27.38(4.56)
50th λ	13.12(3.46)	13.15(3.26)	14.09(3.36)
75th λ	6.63(2.46)	7.88(2.46)	8.94(2.58)
100th λ	4.30(1.91)	5.75(2.01)	6.68(2.16)
Method of median adding			
10th λ	48.01(4.40)	44.63(4.30)	43.38(4.28)
25th λ	31.48(4.45)	28.53(4.36)	28.29(4.48)
50th λ	13.90(3.48)	13.85(3.28)	14.70(3.40)
75th λ	7.00(2.53)	8.21(2.53)	9.25(2.69)
100th λ	4.47(2.00)	5.92(2.10)	6.87(2.27)

Table 2.8: Comparison of RPE for different methods using uniform (-190, 190) censoring distribution for example 2

		α	
λ	0.1	0.5	0.9
Uncensored			
10th λ	4.17(0.34)	3.73(0.34)	3.57(0.34)
25th λ	3.29(0.30)	2.40(0.29)	2.23(0.29)
50th λ	2.04(0.22)	1.39(0.19)	1.32(0.19)
75th λ	1.38(0.18)	1.13(0.16)	1.11(0.16)
100th λ	1.13(0.16)	1.07(0.15)	1.07(0.15)
Method of weight 1			
10th λ	4.25(0.34)	3.89(0.35)	3.74(0.35)
25th λ	3.54(0.34)	2.78(0.38)	2.63(0.39)
50th λ	2.32(0.31)	1.69(0.31)	1.64(0.32)
75th λ	1.59(0.27)	1.33(0.25)	1.32(0.26)
100th λ	1.29(0.23)	1.24(0.23)	1.24(0.23)
Method of weight 0			
10th λ	4.41(0.40)	4.06(0.41)	3.91(0.41)
25th λ	3.73(0.40)	2.93(0.43)	2.75(0.44)
50th λ	2.49(0.37)	1.77(0.34)	1.69(0.34)
75th λ	1.70(0.30)	1.37(0.27)	1.35(0.26)
100th λ	1.35(0.25)	1.25(0.23)	1.25(0.23)
Method of weight average			
10th λ	4.29(0.36)	3.93(0.36)	3.78(0.37)
25th λ	3.58(0.35)	2.80(0.38)	2.63(0.39)
50th λ	2.35(0.32)	1.69(0.31)	1.62(0.31)
75th λ	1.61(0.27)	1.33(0.25)	1.32(0.25)
100th λ	1.30(0.23)	1.23(0.22)	1.23(0.23)
Method of weight depending			
10th λ	4.41(0.40)	4.06(0.41)	3.91(0.41)
25th λ	3.73(0.40)	2.93(0.43)	2.75(0.44)
50th λ	2.48(0.36)	1.76(0.34)	1.68(0.33)
75th λ	1.68(0.30)	1.36(0.26)	1.34(0.26)
100th λ	1.34(0.25)	1.25(0.23)	1.25(0.23)
Method of constant adding			
10th λ	4.25(0.34)	3.89(0.35)	3.74(0.35)
25th λ	3.54(0.34)	2.78(0.38)	2.63(0.39)
50th λ	2.32(0.31)	1.68(0.31)	1.62(0.31)
75th λ	1.58(0.26)	1.31(0.24)	1.30(0.24)
100th λ	1.27(0.22)	1.21(0.21)	1.22(0.22)
Method of mean adding			
10th λ	4.26(0.35)	3.88(0.35)	3.72(0.36)
25th λ	3.52(0.35)	2.72(0.38)	2.57(0.40)
50th λ	2.25(0.31)	1.63(0.30)	1.59(0.31)
75th λ	1.53(0.26)	1.30(0.25)	1.30(0.25)
100th λ	1.25(0.22)	1.22(0.23)	1.23(0.24)
Method of median adding			
10th λ	4.25(0.34)	3.89(0.35)	3.74(0.35)
25th λ	3.54(0.34)	2.77(0.38)	2.62(0.39)
50th λ	2.31(0.31)	1.66(0.30)	1.61(0.31)
75th λ	1.57(0.26)	1.30(0.24)	1.30(0.24)
100th λ	1.26(0.22)	1.21(0.22)	1.22(0.22)

Table 2.9: Comparison of RPE for different methods using extreme value censoring distribution for example 2

		α	
λ	0.1	0.5	0.9
Uncensored			
10th λ	4.18(0.33)	3.75(0.33)	3.59(0.33)
25th λ	3.30(0.29)	2.41(0.29)	2.24(0.29)
50th λ	2.05(0.22)	1.39(0.20)	1.33(0.20)
75th λ	1.38(0.18)	1.13(0.17)	1.11(0.17)
100th λ	1.13(0.16)	1.06(0.16)	1.06(0.16)
Method of weight 1			
10th λ	4.27(0.33)	3.91(0.34)	3.77(0.35)
25th λ	3.57(0.34)	2.83(0.39)	2.68(0.41)
50th λ	2.37(0.32)	1.74(0.32)	1.69(0.33)
75th λ	1.63(0.27)	1.36(0.26)	1.35(0.26)
100th λ	1.32(0.24)	1.25(0.24)	1.26(0.24)
Method of weight 0			
10th λ	4.45(0.41)	4.11(0.41)	3.97(0.42)
25th λ	3.79(0.41)	3.01(0.45)	2.83(0.46)
50th λ	2.56(0.38)	1.83(0.37)	1.75(0.37)
75th λ	1.75(0.32)	1.41(0.29)	1.39(0.29)
100th λ	1.39(0.27)	1.28(0.25)	1.28(0.25)
Method of weight average			
10th λ	4.32(0.35)	3.96(0.36)	3.82(0.37)
25th λ	3.62(0.35)	2.86(0.40)	2.70(0.41)
50th λ	2.41(0.32)	1.74(0.32)	1.68(0.32)
75th λ	1.65(0.27)	1.36(0.25)	1.35(0.26)
100th λ	1.33(0.23)	1.25(0.23)	1.25(0.23)
Method of weight depending			
10th λ	4.45(0.41)	4.11(0.41)	3.97(0.42)
25th λ	3.79(0.41)	3.01(0.45)	2.82(0.46)
50th λ	2.56(0.38)	1.82(0.36)	1.74(0.36)
75th λ	1.74(0.31)	1.40(0.28)	1.38(0.28)
100th λ	1.37(0.26)	1.27(0.23)	1.27(0.25)
Method of constant adding			
10th λ	4.27(0.33)	3.91(0.34)	3.77(0.35)
25th λ	3.57(0.34)	2.83(0.39)	2.68(0.41)
50th λ	2.37(0.32)	1.72(0.32)	1.67(0.33)
75th λ	1.62(0.27)	1.33(0.25)	1.33(0.25)
100th λ	1.29(0.22)	1.23(0.22)	1.23(0.23)
Method of mean adding			
10th λ	4.27(0.33)	3.91(0.35)	3.75(0.35)
25th λ	3.55(0.35)	2.76(0.40)	2.61(0.41)
50th λ	2.29(0.32)	1.67(0.31)	1.63(0.32)
75th λ	1.56(0.26)	1.32(0.24)	1.32(0.25)
100th λ	1.27(0.22)	1.23(0.23)	1.24(0.24)
Method of median adding			
10th λ	4.27(0.33)	3.91(0.34)	3.77(0.35)
25th λ	3.57(0.34)	2.82(0.39)	2.67(0.40)
50th λ	2.35(0.32)	1.70(0.31)	1.65(0.32)
75th λ	1.60(0.26)	1.32(0.24)	1.32(0.25)
100th λ	1.28(0.22)	1.23(0.22)	1.23(0.23)

Figure 2.1: Geometry of elastic net

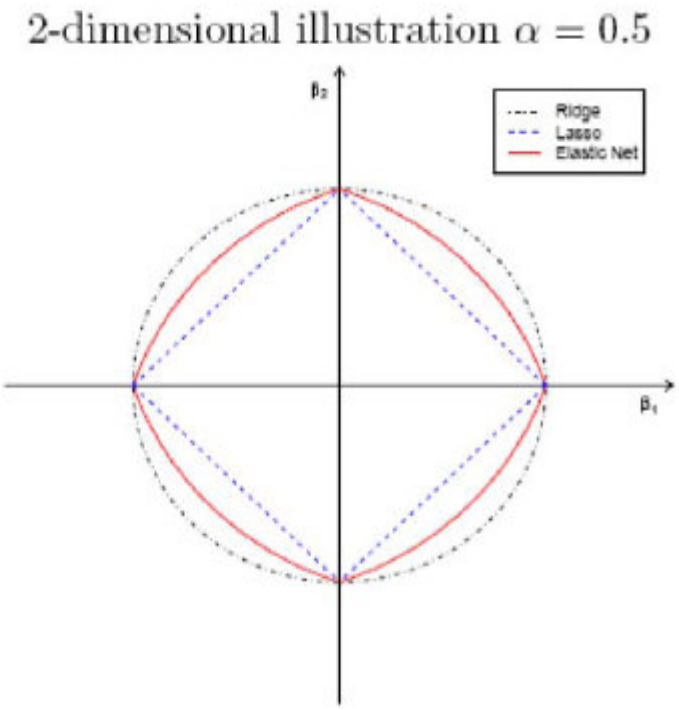


Figure 2.2: Diagram of redistribution to right. X denotes failures and ↓ denotes withdraws (censors)

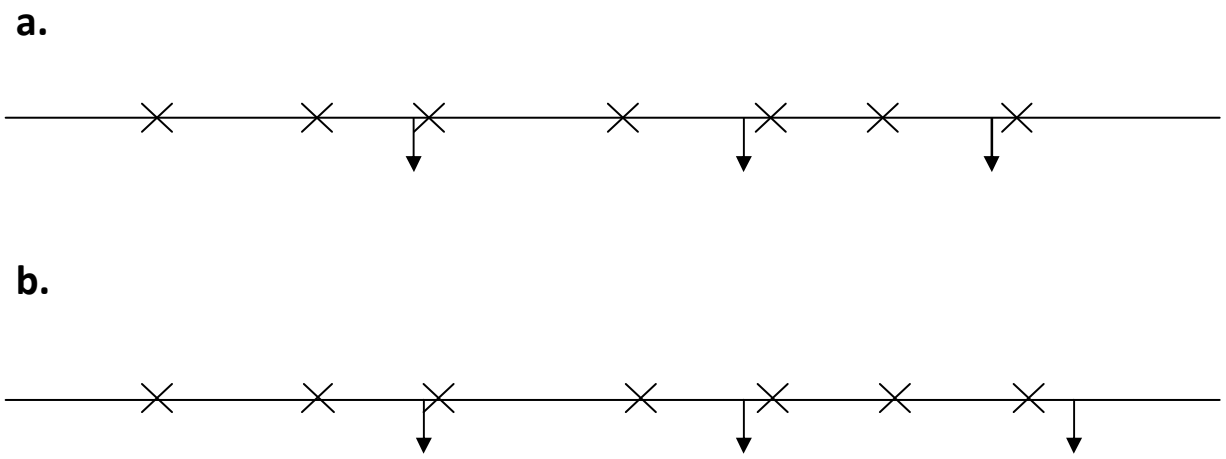
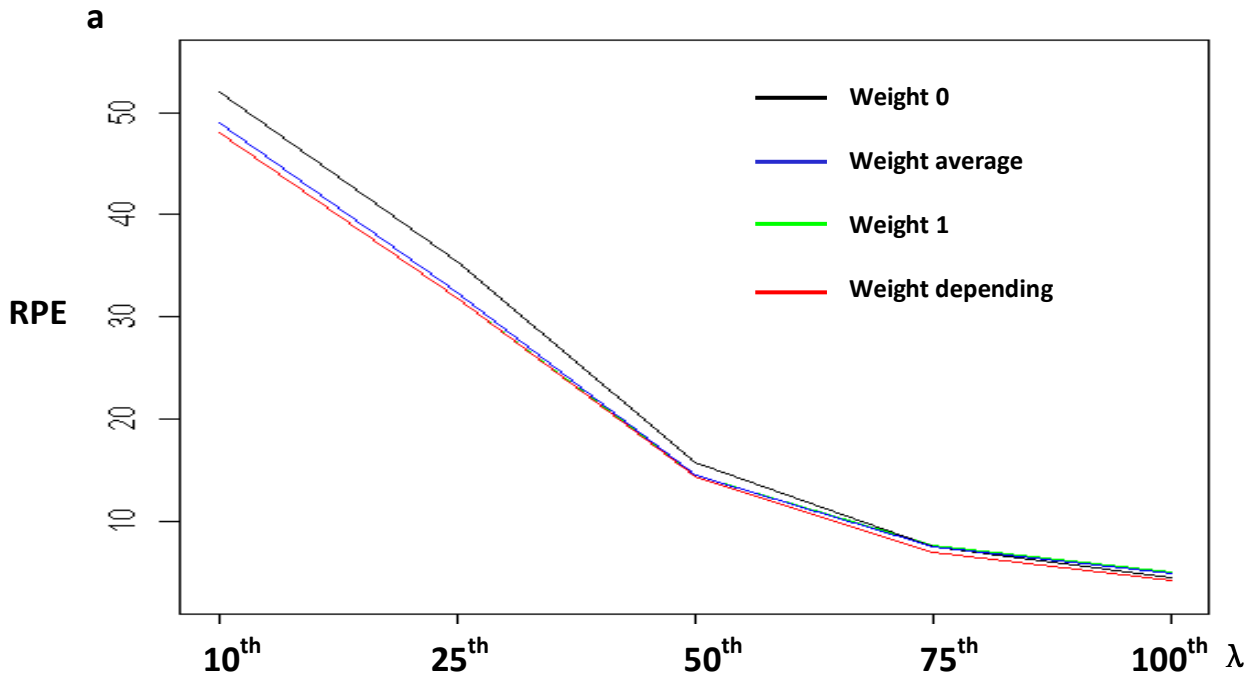


Figure 2.3: Comparison of methods at $\alpha = 0.1$. a) methods modifying $\delta_{(n)}$ and b) methods modifying $Y_{(n)}$



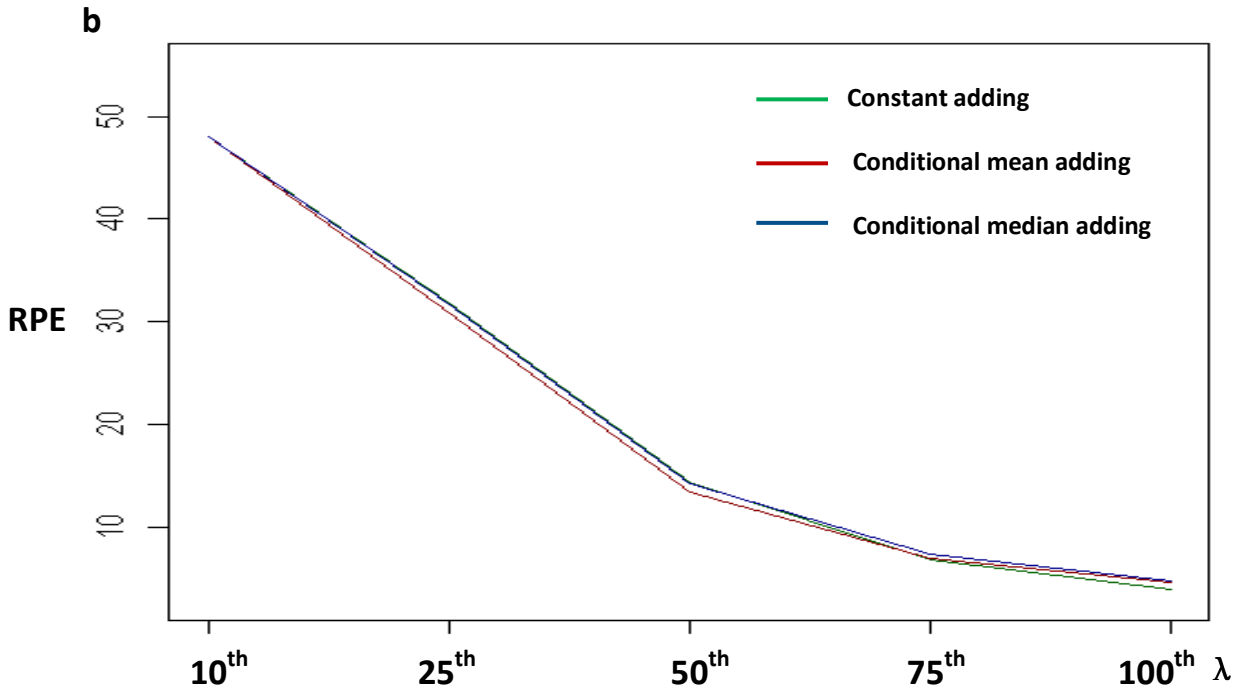
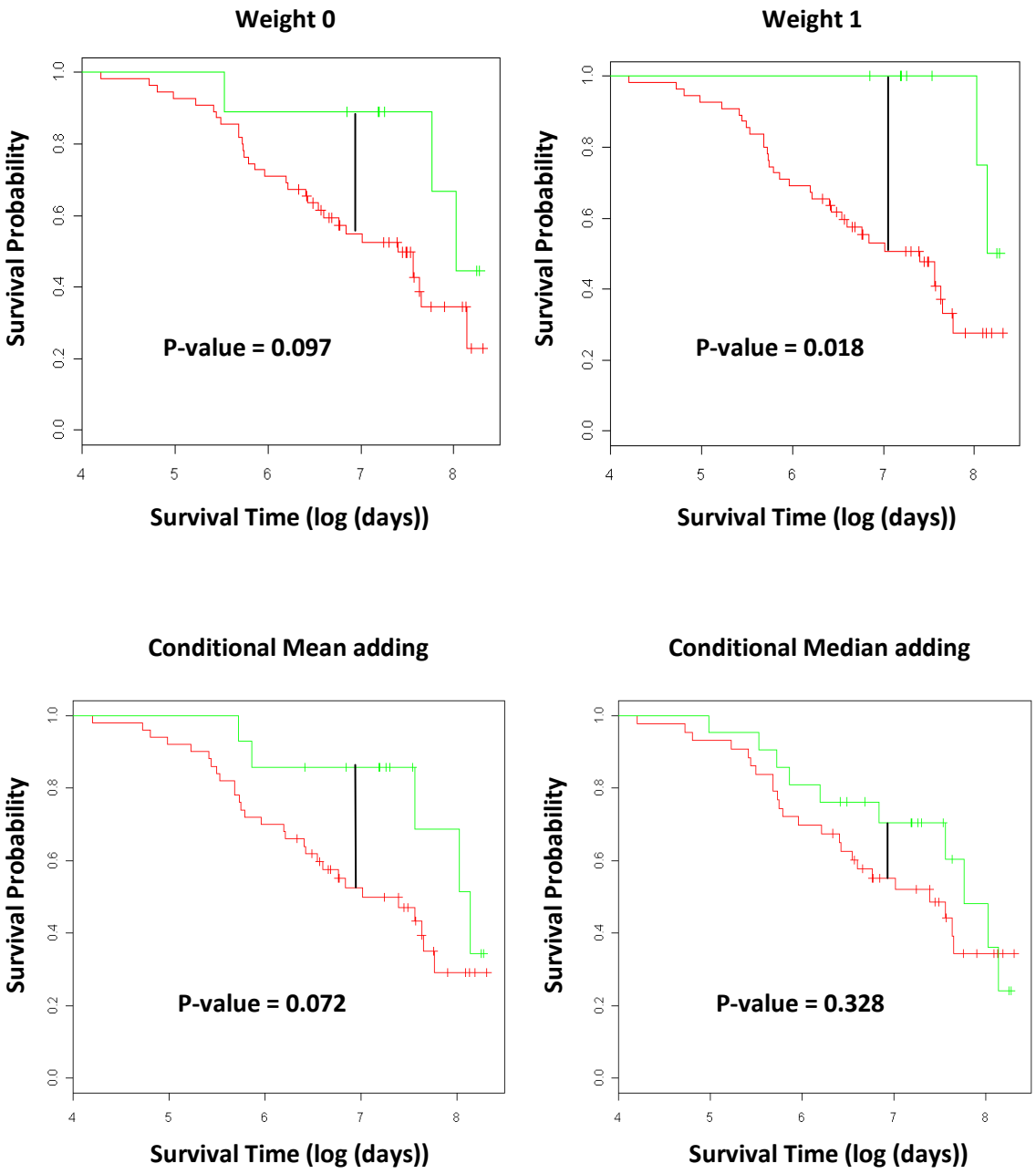


Figure 2.4: Survival curves for test data set using different methods



Chapter 3 Linear Quantile Regression to Identify Equine Cartilage Biomarkers

3.1 Background

DNA microarrays provide information about expression levels for thousands of genes simultaneously at the transcriptional level. It is being applied to determine how global (cell type, tissue, or organismal) differential transcription may affect biological systems. The development of microarray technology has motivated interest in their use for disease research and diagnosis. Many studies have attempted to find disease-specific biomarkers, a small subset of genes that distinguish normal tissue from diseased tissue. A wide variety of statistical methods have been applied to biomarker identification, including sparse logistic regression (SLogReg) (Shevade et al., 2003), receiver operating characteristic (ROC) curve approach (Pepe, 2003; Pepe et al., 2005) and Gaussian process models (Chu et al., 2005). However, most of these focus on disease classification, while far fewer studies have been done to identify tissue biomarkers or genes with a tissue-restricted pattern of expression. Genes with a high level of expression in one tissue compared to other tissue types in the body are likely to have corresponding tissue-restricted functional annotation. Further, loss of the functional product encoded by these genes will frequently be associated with tissue pathology. In general, the identification of tissue-specific biomarkers or genes with a tissue-restricted pattern of expression can provide important new insight into the biology of that tissue or the etiology/pathogenesis of diseases that impact that tissue.

Quantiles are measures of relative standing. For example, a student scoring at the τth quantile on a standardized test means that he/she performs better than a proportion τ and worse than a proportion $(1-\tau)$ of the reference group of students. For any $0 < \tau < 1$, $F^{-1}(\tau) = inf\{x : F(x) \leq \tau\}$ is called the τ quantile of the

distribution F (Koenker, 2005). Quantile regression as introduced by Koenker and Bassett (1978) extends this idea to the estimation of conditional quantile functions modeling quantiles of the conditional distribution of the response variable as functions of observed covariates. An ordinary least squares (OLS) regression models the relationship between covariates X and the conditional mean of the response variable Y given $X = x$. However, covariates X often influence the whole distribution of Y, not only the mean, thereby severely weakening OLS (Koenker and Bassett, 1978). For example, a change in covariates may have an opposite effect on the high and low percentiles of Y. Unlike OLS, quantile regression methods offer a mechanism for estimating models across the full range of conditional quantile functions given $X = x$. Two models of quantile regression can be distinguished, depending on whether or not independent identically distributed (iid) error terms are assumed. We will call the model without assumption of iid error terms the non iid error model. In linear quantile regression, if the slopes of the regression lines are different for different quantiles, then the non iid error model is more appropriate (Koenker, 2005). Recently, Wang and He proposed a rank score test (Wang and He, 2007 and 2008) for detecting differential gene expression by modeling and analyzing the quantiles of gene intensity distributions through probe-level measurements. Though also based on the quantile regression idea, Wang and He's method is otherwise not related to the approach presented here.

Fold change has been widely used in microarray experiments to identify genes with different expression levels between two types of samples (e.g., diseased versus normal tissue). A cut off of 2-fold up or down regulation has been chosen to define differential expression in most published studies (Schena et al., 1996; Vaishnav et al., 2008). However, the commonly used 2-fold change criterion does not take into account the magnitude of gene expression.

In this study, we propose an intensity-dependent linear quantile regression, us-

ing statistical and biological information to identify tissue-restricted patterns of gene expression. We demonstrate our methods on the analysis of cDNA microarray data to compare articular cartilage with ten different body tissues to identify genes with a cartilage-restricted pattern of expression representing potentially novel cartilage biomarkers. Chondrocytes are the only cell type in cartilage and they synthesize several proteins that are expressed in a highly tissue-restricted pattern, including type II procollagen and aggrecan core protein. Screening for novel genes that have a cartilage-restricted pattern of expression can expand our understanding of chondrocyte function and potentially improve our understanding of important diseases that involve cartilage, such as arthritis.

3.2 Results and Discussion

Implementation

A MA plot was used to remove intensity dependent dye bias and array-specific effects where $M = log_2(R/G)$ and $A = log_2\sqrt{RG}$ with R representing the gene expression level in cartilage and G representing the gene expression level in one of the other 10 tissues in our study. After scanning, the median intensities adjusted for background intensities of each pair of spots were Lowess (LOcally WEighted polynomial regreSSion) normalized for each individual slide. Two MA plots in Figure 3.1 represented the twenty cDNA microarray slides used for this study. The first plot illustrated unnormalized data and the second plot was the same data after Lowess. The MA plots in Figure 3.1 showed that the intensity dependent bias had been removed after lowess normalization (Yang et al, 2002; Dudoit et al., 2002). Because of some bad-flagged spots, the number of probesets available for analysis ranged from 9333 in the cartilage/lung comparison to 9411 in the cartilage/cerebellum comparison. For each comparison, a piecewise nonparametric approach was used to reveal the relationship between percentiles of M and A. The range of A was divided into 10

regions with a minimum of 900 probe sets and a maximum of 1000 probe sets in each region. The corresponding 1st, 5th, 10th, 20th, 50th, 80th, 90th, 95th, 99th percentiles of M were calculated for each region of A. Scatter plots of the mean of A for each region and quantiles of M in the corresponding region were plotted. For the cartilage versus bladder comparison (Figure 3.2a), the scatter plot showed an approximate linear relationship between A and each of the considered conditional quantiles of M given A, with slight deviations from a linear relationship at the high intensities. Similar patterns were also observed in the other 9 tissue comparisons (data not shown). Since the scatter plots for different quantiles were not parallel, the non iid error quantile regression model is more reasonable. Hence for each comparison, linear quantile regression (containing intercept and a linear term) under the non iid error model (He and Wei, 2005; Chen) (Figure 3.2b) was fitted to the data. Generally, the fit was good, except for small deviations at extreme high intensities (Figure 3.2c). The corresponding nine conditional percentiles (1st, 5th, 10th, 20th, 50th, 80th, 90th, 95th, 99th) of M were estimated for each observed A. Observed M was compared to the estimated nine conditional percentiles of M, and a cartilage specific Z-score was calculated according to Table 3.1. The average Z score and standard deviation were also calculated. Genes were considered potential cartilage biomarkers if the observed values for M were above the estimated 95th conditional percentile of M in all 10 of the cartilage/tissue comparisons analyzed (all Z-scores ≥ 1.96).

Cartilage Biomarkers Identified

Two red straight lines in Figure 3.3 corresponded to estimated 95th percentile and 5th percentile of $log_2(R/G)$ in cartilage versus bladder comparison, respectively. Thirty-seven probe sets (cyan spots in Figure 3.3) were identified that exhibit expression above the 95th conditional quantile in all 10 of the cartilage/tissue comparisons analyzed (Z-scores ≥ 1.96). Of these, 13 (35%) have existing annotation associ-

ated with cartilage including several well-established cartilage biomarkers (Table 3.2). BLAST hits for the remaining 24 probe sets (65%) in which the cartilage-specificity score was at least 1.96 in all 10 tissue comparisons have no reported sequence annotation associated with established functional roles in cartilage. From Table 3.2, we can also see that the means of the Z scores for these probe sets were high, with small standard deviations. In contrast, six probe sets (blue spots in Figure 3.3) exhibited expression levels below the 5th conditional quantile in all 10 of the cartilage/tissue comparisons analyzed (Z-scores $<$ -1.96). These 6 probe sets represent the ones on this microarray (cDNAs from an equine articular cartilage library) with consistently low relative gene expression in cartilage compared to the other tissue types studied. With microarrays that contain probe sets for all genes in the genome of an organism, an analysis of the lowest quantiles should be useful in identifying genes with a near absence of expression in the reference tissue of interest.

Probe sets with high fold change but very low intensities should be excluded. For example, a probe set might be reported with an intensity of 2 in bladder but 20 in cartilage, thus the fold change was 10. However, if the intensity reading in cartilage is low, then we cannot reliably identify this kind of probe set as one that is exhibiting cartilage-specific expression. For each chip, we calculate the 10th percentile of averaged log_2 intensities, denoted as A*. If a probe set's A value (averaged log_2 intensity) was less than A*, we excluded it from the candidate list even when the M value (log_2 fold change) for this probe set was very large. In other words, all 37 probe sets were selected from probe sets with values of A larger than A*. For one of the ten comparisons (cartilage vs. bladder), Figure 3.3 illustrates that A for all 37 probe sets was larger than 6 and M was larger than 1 which implies that the intensity reading in cartilage was at least greater than 64 after lowess normalization. Similar ranges of A values for these 37 probe sets were found in the other 9 cartilage/tissue comparisons. Taking COMP as an example in Table 3.3, we see that the intensity readings in

cartilage were high and the relative expression differences between cartilage and each of the ten other tissues (fold change) were large. Similar ranges of intensity and relative expression differences were found with the other 36 probe sets. Therefore, data for these thirty-seven probe sets were interpreted as consistent with a cartilage-restricted pattern of expression.

In this study, cartilage-specific scores were used in place of percentiles of M (Table 3.1). We have compared cartilage with ten other body tissues and have identified 37 probe sets with expression all above the 95th percentile of M. However, with a larger number of tissue comparisons, the criterion of above the 95th percentile of M in all tissue comparisons may be too stringent to identify a cartilage-restricted expression pattern. The idea of transforming percentiles of M into Z-scores and then choosing probe sets with a high average Z-score and low standard deviation makes the criterion more feasible to identify probe sets of interest. One of the advantages of the standardized Z-scores is that it is relatively simple to make adjustments that take the number of comparisons into account. The appropriate cutoff for average Z-score and standard deviation deserves further investigation.

Due to the fact that genes were classified as cartilage-specific only when they showed high relative expression in all 10 tissue comparisons, the probability of falsely identifying a chance outlier as a cartilage-specific gene is rather low. Loguinov et al. (2004) distinguish five different circumstances represented by 'outliers': a gene with higher individual variability than the majority of genes; an outlier by chance; a sporadic technical or biological outlier; a systematic technical outlier (due to, for example, heteroscedasticity); or a systematic biological outlier due to differential expression. Our result is based on limited biological replicates, so it is important to distinguish between differentially-expressed probe sets and the other four types of outliers. We define genes as potential cartilage biomarkers if the observed values for M were above the estimated 95th conditional percentile of M in all 10 of the

cartilage/tissue comparisons analyzed (all Z-scores $\geq$ 1.96). The probability that anyone of the thirty-seven probe sets identified would be due to the other four types of outliers in all 10 of cartilage/tissue comparisons is very small. For example, if we assume that the probability of probe set being one of the other four types of outliers is 20% in one cartilage/tissue comparison, then the probability of this probe set being such an outlier in all 10 cartilage/tissue comparisons is 0.2^{10}, which is 1.024e-07, a rather small value.

Feasibility and Appropriateness of Linear Quantile Regression

Volcano plots, which consider both statistical tests of differences between sample types (P-value) and biological effects (fold change) are commonly used in microarray experiments to identify genes with different expression levels between two experimental groups. With microarray experiments in which the design requires comparisons between many experimental groups, the number of biological replicates can be constrained by logistical variables. For example, with the sample set analyzed in this study, the articular cartilage and eight of the comparative tissues were collected from a single donor while placental villous and testis samples were each collected from other donors. The absence of biological replicates made statistical inference (e.g., t-test) of expression differences between cartilage and the other 10 tissues impossible. In addition, a 2-fold change criterion does not take into account the varied magnitude of gene expression. Hence, quantile regression was used to determine quantiles of M conditional on A. Microarray data consists of thousands of probe sets. Dividing the range of A into several regions still makes each region have enough probe sets (corresponding to spots in the graph) to calculate the quantiles of M. Thus, the piecewise nonparametric method is feasible and appropriate to reveal the relationship between A and percentiles of M.

In this study, scatter plots showed percentiles above the 50th percentile of M

(99th, 95th, 90th, 80th) linearly increasing with A while percentiles below 50th percentiles of M (1st , 5th , 10th , 20th) were linearly decreasing with A. Hence, linear quantile regression with a linear term was fitted to the data. Expression levels above the 95th percentile were defined as cartilage-restricted expression. Thirty-seven probe sets were identified as exhibiting a cartilage-restricted pattern of expression. Within this group are widely recognized cartilage biomarkers, including genes encoding type II procollagen and aggrecan core protein. The presence of genes encoding these established cartilage biomarkers validate the linear quantile regression approach. Hence we recognize that the expression pattern for the remaining genes that currently lack established functional annotation linked to cartilage needs to be confirmed with additional studies.

Simple linear regression (mean regression) should not be applied to these data since different quantiles of M behave differently (Figure 3.2d) and the iid error assumption (implying equal variances) which is used in simple linear regression is obviously violated. In Figure 3.2d, at medium and high intensities, the 95th linear quantile regression line (red) was above the 95% confidence interval upper bound of the simple linear regression line (purple). As a result, the approach of fitting a linear regression and then calculating a 95% confidence interval of individual predicted values of M conditional on each A would lead to the false positive identification of cartilage-specific probe sets at medium and high intensities.

Based on the M-A plots, one of the reviewers has suggested the following iterated logarithm approach for normalization. Let $log_2(log_2 R) - log_2(log_2 G)$ be M and $(log_2(log_2 R) + log_2(log_2 G))/2$ be A to perform Lowess normalization. After normalization, for each comparison, linear quantile regression of M on A was fitted to the data. 39 probe sets were above the estimated 95% conditional percentile of M in all 10 tissue comparisons. In contrast, 37 probe sets were above the estimated 95% conditional percentile of M in all 10 tissue comparisons using the originally proposed log_2 transfor-

mation method. There were 32 probe sets common in both approaches. However, the iterated logarithm approach failed to identify 3 well-established cartilage biomarkers, which could be identified by the single log_2 transformation approach. One possible reason is that the iterated logarithm may not remove intensity-dependent bias as well as the single logarithm.

3.3 Conclusions

Quantile regression is appropriate for the analysis of two color array experiments, especially for studies with only one replicate and hence highly limited quantifiable sources of experimental error. We used a nonparametric approach to reveal the relationship between A and quantiles of M and then applied the appropriate quantile regression (in this study, it is linear quantile regression with intercept and a linear term) to select genes with a high level of expression in specific tissue or tissue biomarkers.

3.4 Methods

Microarray Experiments

Articular cartilage and eight comparative tissues (kidney, lung, lymph node, cerebellum, spleen, bladder, liver, and muscle) were collected from a two-year old donor horse. Placental villous and testis samples were obtained independently from other donor horses. Total RNA was isolated from all of these eleven tissues by a traditional guanidinium isothiocyanate and phenol/chloroform separation protocol for total RNA isolation. Dye-coupled probes from the articular cartilage and each of the 10 tissues individually (cartilage/kidney, cartilage/lung, cartilage/lymph node, cartilage/placental villus, cartilage/cerebellum, cartilage/spleen, cartilage/bladder, cartilage/testis, cartilage/liver, and cartilage/muscle) were then hybridized to a 9852

element equine-specific cDNA microarray. All hybridizations were performed in duplicate with a dye swap to eliminate possible dye bias. After the post-hybridization washes, each slide was then immediately scanned using a GenePix 4100A scanner and spot intensities were computed using GENEPIX 6.0 image analysis software (Axon Instruments/Molecular Devices). Following background correction, the median intensities of each pair of spots were Lowess normalized for each individual slide. The bad-flagged spots on each slide were removed from the analyses.

Algorithm and Analysis

The statistical model in this study is that the τth conditional quantile of Y_i is $X_i\beta(\tau)$ where Y_i is the observed M, $X_i = (1, x_i)$ and x_i are the A, $\beta(\tau) = (\beta_0(\tau), \beta_1(\tau))^T$. We have employed τ values 1, 5, 10, 20, 50, 80, 95 and 99%. Another way of writing this model is $Y_i = X_i\beta(\tau) + \varepsilon_i(\tau)$ with $\varepsilon_i(\tau)$ quantile zero. The parameter $\beta(\tau)$ can be estimated by solving the minimizing problem: $\hat{\beta}(\tau) = \arg\min_{\beta \in R^2} \sum_{i=1}^{n} \rho_\tau(Y_i - X_i\beta), 0 < \tau < 1$ where $\rho_\tau(z) = z(\tau - I(z < 0))$ and I(.) is the indicator function. Based on the estimated $\hat{\beta}(\tau)$, the predicted τth quantile of Y given covariate value x_i is $X_i\hat{\beta}(\tau)$.

Table 3.1: Transforming quantiles of $log_2(R/G)$ to Z score

Quantile of $log_2(R/G)$	Z-score
observed $log_2(R/G)$ $\geq$99th estimated quantile	2.57
99th estimated quantile $>$ observed $log_2(R/G) \geq$ 95th estimated quantile	1.96
95th estimated quantile $>$ observed $log_2(R/G) \geq$ 90th estimated quantile	1.44
90th estimated quantile $>$ observed $log_2(R/G) \geq$ 80th estimated quantile	1.04
80th estimated quantile $>$ observed $log_2(R/G) \geq$ 50th estimated quantile	0.39
50th estimated quantile $>$ observed $log_2(R/G) \geq$ 20th estimated quantile	-0.39
20th estimated quantile $>$ observed $log_2(R/G) \geq$ 10th estimated quantile	-1.04
10th estimated quantile $>$ observed $log_2(R/G) \geq$ 20th estimated quantile	-1.44
5th estimated quantile $>$ observed $log_2(R/G) \geq$ 10th estimated quantile	-1.96
observed $log_2(R/G) <$ 1st estimated quantile	-2.57

Table 3.2: Cartilage specific score for genes with functional annotation linked to cartilage

Gene Symbol	Gene Name	Cartilage-Specificity Score				
		Mean	S.D.	Low	High	Median
Hapln1	Hyaluronan and proteoglycan link protein 1	2.57	0	2.57	2.57	2.57
COMP	Cartilage oligomeric matrix protein	2.51	0.19	1.96	2.57	2.57
COL11A1	Collagen, type XI, alpha 1	2.51	0.19	1.96	2.57	2.57
AGC1	Aggrecan core protein	2.51	0.19	1.96	2.57	2.57
COL2A1	Collagen, type II, alpha 1	2.39	0.29	1.96	2.57	2.57
TNC	Tenascin C	2.33	0.32	1.96	2.57	2.57
PRG4	Proteoglycan 4	2.33	0.32	1.96	2.57	2.57
SOX9	SRY-box 9	2.20	0.32	1.96	2.57	1.96
ITGA10	Integrin, $\alpha 10$	2.20	0.32	1.96	2.57	1.96

Table 3.3: Intensities of COMP expression in all ten cartilage/tissue omparisons

Tissue Comparison	Original Intensity		After Lowess Normalization		
	Cartilage	Tissue	A	M	Fold Change
Cartilage/Bladder	4797.70	214.07	9.17	7.63	194.01
Cartilage/Cerebellum	7860.70	947.79	10.36	5.35	40.79
Cartilage/Kidney	3249.25	264.16	8.69	5.89	59.30
Cartilage/Liver	5685.64	420.13	9.58	5.91	60.13
Cartilage/Lung	4494.31	166.70	8.77	5.83	56.89
Cartilage/Lymph node	10382.70	706.12	10.43	7.25	152.22
Cartilage/Muscle	6191.20	621.54	9.68	5.66	50.56
Cartilage/Placental villus	4256.93	238.23	8.83	7.27	154.34
Cartilage/Spleen	11358.35	806.09	10.52	7.18	145.01
Cartilage/Testis	8075.92	1774.16	11.04	4.60	24.25

Figure 3.1: MA plot to remove intensity dependent bias

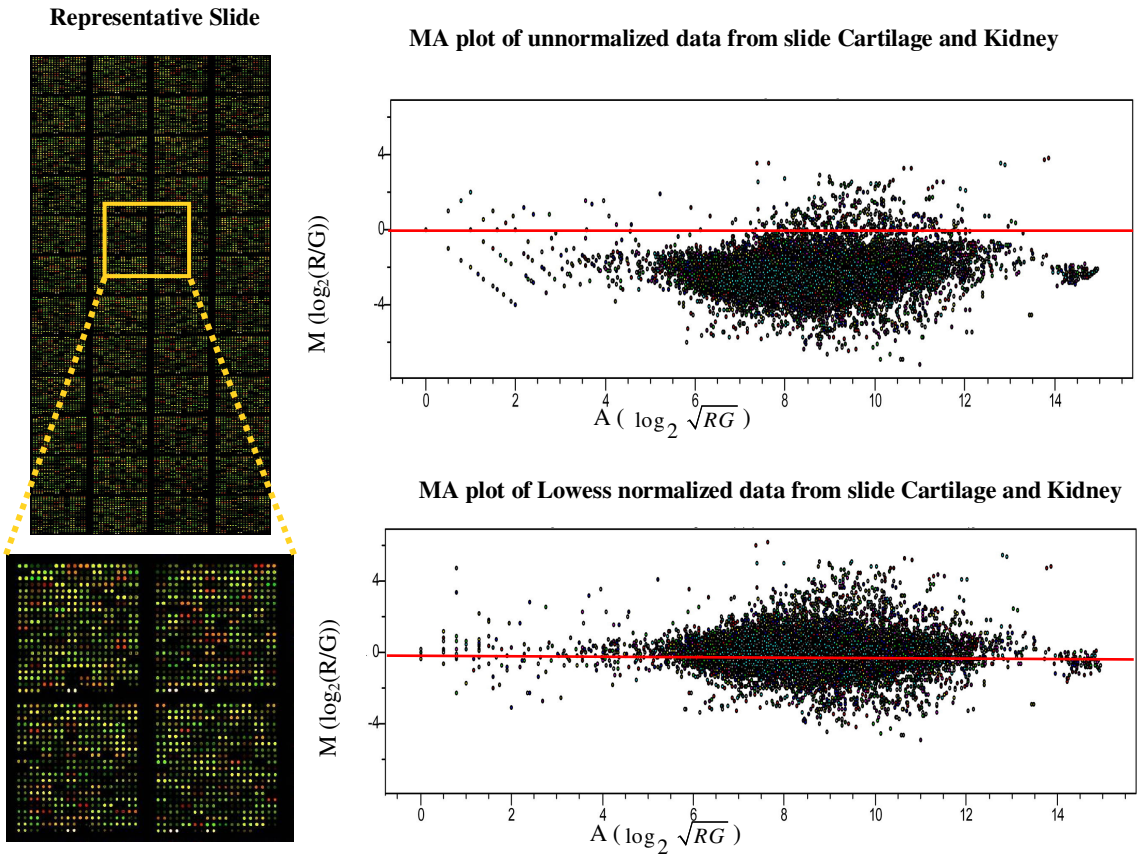


Figure 3.2: Nonparametric approach to reveal the relationship between A and quantiles of M and linear quantile regression fitting

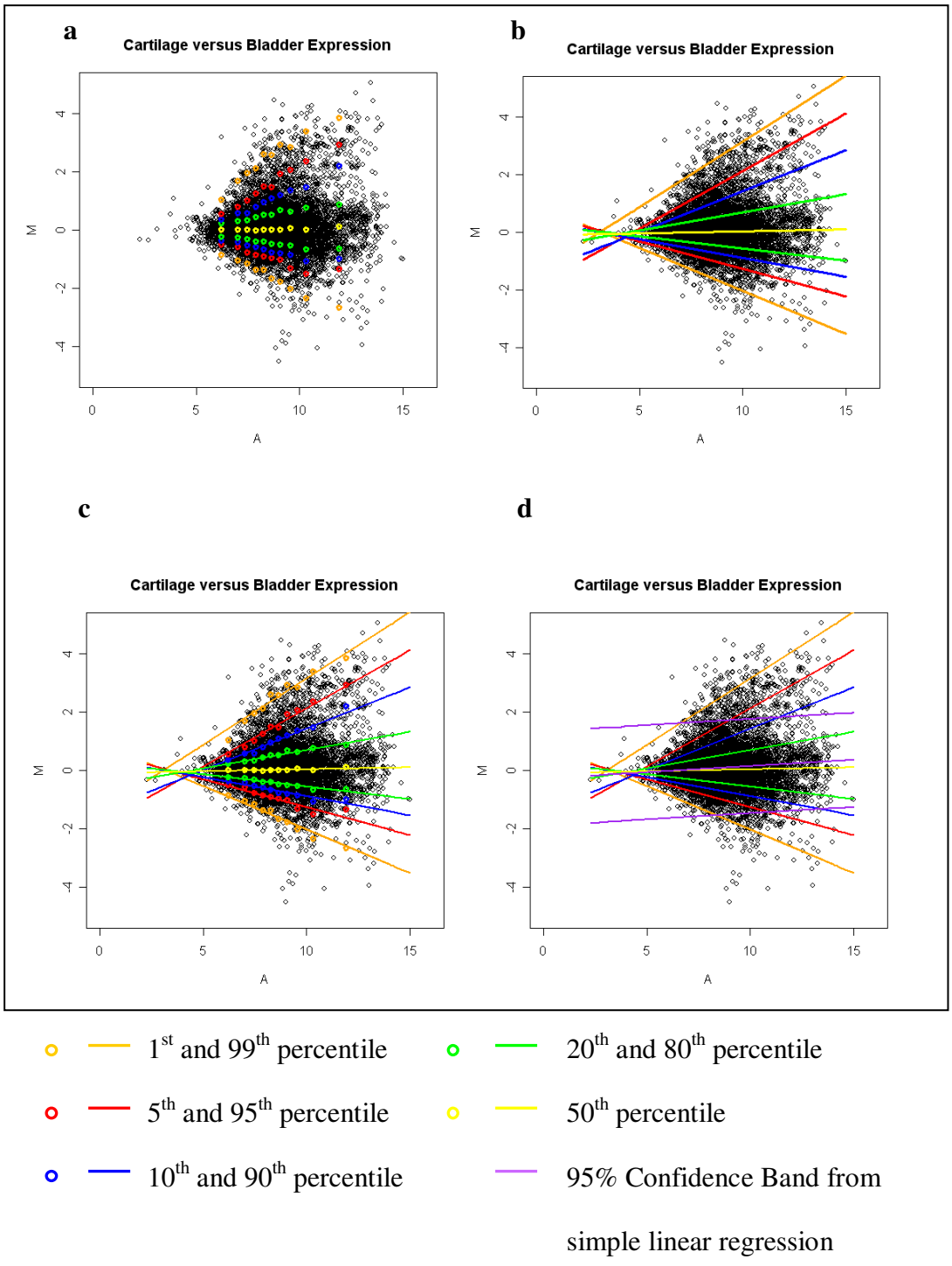
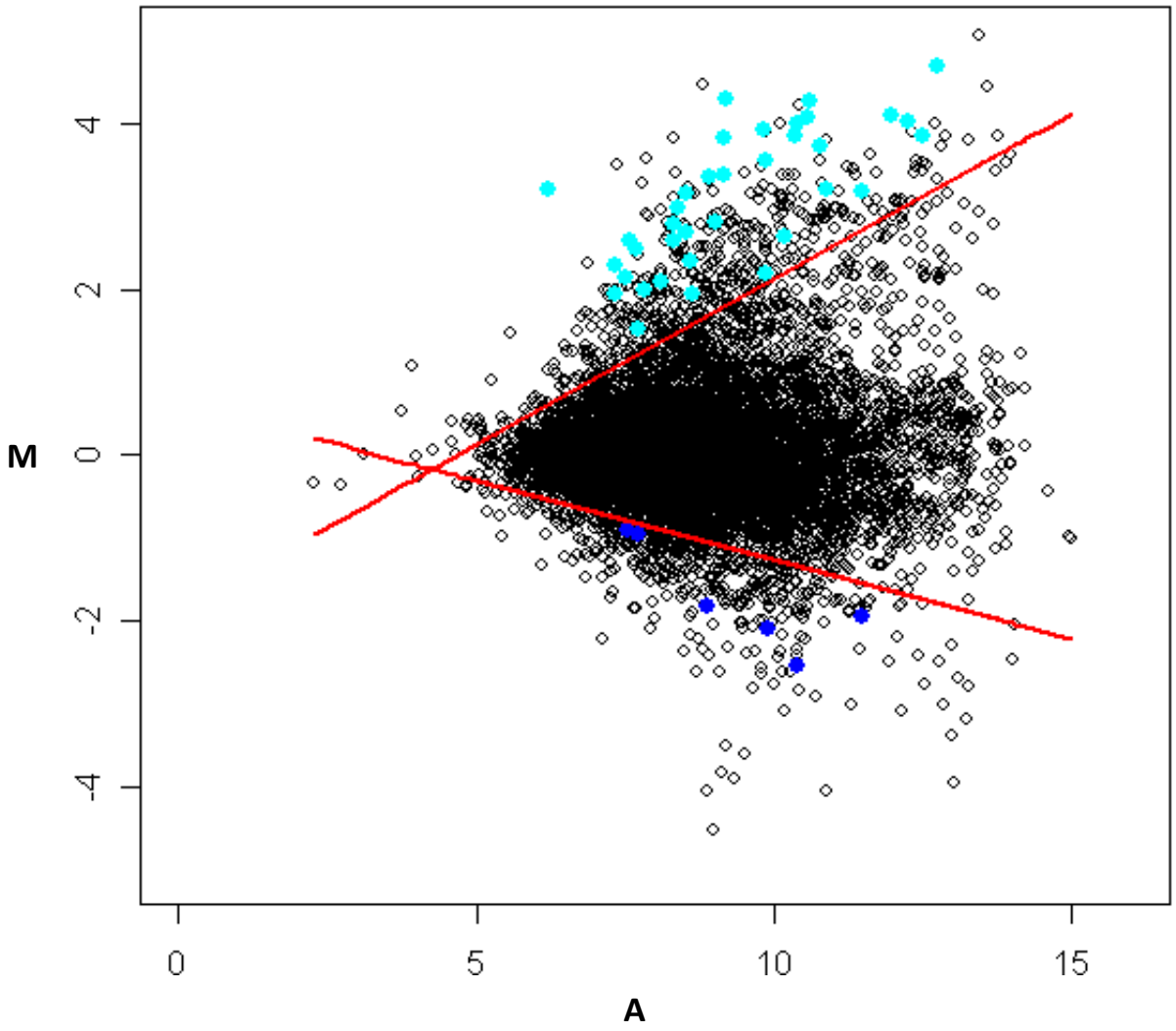


Figure 3.3: 37 probe sets identified in cartilage/bladder comparison



Chapter 4 Normalization and Analysis of cDNA Microarray Using Linear Combinations

4.1 Background

Two-color cDNA microarray technology has evolved into a routine laboratory procedure. A key purpose of a cDNA microarray experiment is to quantify the relative gene expression between experimental samples for thousands of probe sets simultaneously. In a cDNA microarray experiment, two mRNA samples are linearly amplified, reverse-transcribed into cDNA, labelled with green (AlexaFluor555) and red (AlexaFluor647) fluorescent dyes, and mixed in equal proportions and hybridized with the immobilized probe sequence on the microarrays. The pixel intensities from green and red wavelength images are measured for the amount of hybridization between the mRNA samples and the probe sequence on the microarrays.

A key concern in the analysis of cDNA microarray data is normalization, the purpose of which is to adjust for effects arising from variation in the microarray technology rather than from biological differences between the RNA samples. Systematic variation in microarray experiments may arise from imbalances between the red and green dyes (dye bias), uneven hybridizations, differences in print quality, etc. Dye bias, the most common source systematic variation, can be obviously seen in an experiment where two identical mRNA samples are labeled with different dyes and subsequently hybridized to the same slide. In this situation, it is rare to have the dye intensities equal on average and often the intensities are higher for the green dye. Dye bias may stem from physical properties of the dyes, differential efficiency of dye incorporation, and scanner settings at the data collection step. A proper normalization procedure ensures that red/green ratios are unbiased and thus representative of relative gene expression levels.

In recent years, several methods have been proposed for normalization. Let $log_2 R$

and log_2G be the green and red background corrected intensities, then M and A are defined by $M = log_2R/G$ and $A = log_2\sqrt{RG}$. Yang et al. (2001) discussed several normalization methods, including global normalization, intensity dependent linear normalization and intensity dependent nonlinear normalization. Global method assumes that the red and green intensities are related by a constant factor and normalization model can be expressed as $M = \beta_0 + \epsilon$, where β_0 is estimated by the median of M and ϵ is a vector of random error. For linear normalization, the model is $M = \beta_0 + \beta_1 A + \epsilon$, here β_0 and β_1 are obtained by least squares estimation. The model for nonlinear normalization is in the form of $M = c(A) + \epsilon$, where $c(A)$ is estimated using Lowess smoothing. All three normalization methods are based on regression models of M in terms of A. The analysis of variance (ANOVA) method was presented by Kerr et al. (2000) and Wolfinger et al. (2001) extended the ANOVA approach, including random effects to model the dependence among observations relative to the same spots or arrays and allowing for gene-specific variance components. A two-step approach was proposed by Wolfinger et al. (2001). First stage is referred to as 'global normalization'. From the global normalization, the estimated residuals are saved. In the second stage, the mixed model performs independent analyses for each gene using estimated residuals from the first step as response variable.

Lowess normalization is one of the most widely used normalization methods. However, this method is appropriate when at least one of the two biological assumptions is satisfied: (a) the majority of genes are not differentially regulated, or (b) there is symmetry in the expression levels between up-regulated and down-regulated genes (Zien et al., 2001). For experiments that these two assumptions are violated, Tseng et al. (2001) suggested using a rank-based procedure, first selecting a set of genes with constant expression levels, then carrying out lowess normalization based on this set of genes. Huang et al. (2005) presented a two-way semilinear model (TW-SLM) that does not make the assumptions underlying the lowess normalization method and

does not require preselection of invariant genes in an array.

In this study, we propose using well-planned linear combinations to remove dye and array effects and further to identify differentially expressed genes. We demonstrate our method in two datasets: (1) a comparison of normal cartilage to repair tissue to profile genes playing a key role in the process of cartilage repair, and (2) a dataset with a 2x2 factorial design with two main effects: oxygen concentration (hypoxia versus normoxia) and cell culture methods (monolayer versus aggregate).

4.2 Results and Discussion

Implementation

After scanning, log_2 transformation was applied to the median intensities adjusted for background intensities of each pair of spots. If the median intensities adjusted for background was less or equal to 0, then 1 was used to replace it. There were 36 (0.38%) probe sets in first data set and 34 (0.36%) probe sets in the second data set had such replacement. Only a few probe sets had such replacement in both data sets, so statistical analysis based on data after replacing was reliable. Our statistical models were based on intensities after log_2 transformation.

First dataset: Table 4.1 represents the model equation for one array set in a microarray experiment with dye swap design. The model can be written as $y_{ijkl(m)} = \mu_i + \tau_j + \lambda_{jk} + \gamma_{jkl} + \eta_{jkm(i,l)} + \epsilon_{ijkl(m)}$ with $i = 1, 2$; $j = 1, \ldots, 4$; $k = 1, 2$; $l = 1, 2$; $m = 1, 2$. $\mathbf{y_{ijk(m)}} = (y_{1jk1(1)}, y_{1jk2(2)}, y_{2jk1(2)}, y_{2jk2(1)})$ are gene expression levels (intensities) for jth horse kth leg; $\mu = (\mu_1, \mu_2)$ are mean response with μ_1 corresponding to mean response for repair tissue and μ_2 representing mean response for normal cartilage; τ_j is jth horse effect; $\lambda_{\mathbf{j}} = (\lambda_{j1}, \lambda_{j2})$ are right and left leg effects for jth horse, respectively; $\gamma_{\mathbf{jk}} = (\gamma_{jk1}, \gamma_{jk2})$ are array 1 and array 2 effects for jth horse kth leg. Since dye effect is confounded in treatment and array effect, we defined $m(i, l)$ in this way: $m(1, 1) = 1$, $m(2, 1) = 2$, $m(1, 2) = 2$ and $m(2, 2) = 1$ so that different pairs

of (i, l) determine different dye effects. Therefore $\eta_{\mathbf{jk}} = (\eta_{jk1}, \eta_{jk2})$ are red and green dye effects for jth horse kth leg respectively. Under this definition, dye effects differ across the horse and leg. $\epsilon_{\mathbf{ijk}} = (\epsilon_{1jk1(1)}, \epsilon_{1jk2(2)}, \epsilon_{2jk1(2)}, \epsilon_{2jk2(1)})$ is random effect for jth horse kth leg. The sum of the first row minus the sum of second row, cancels out the dye, array, horse and leg effects thus leaving the difference of mean response between repair cartilage and normal cartilage plus a random error term. Cancelling out horse and leg effects results in eight observations instead of four observations per probe set. Assuming that the error vectors are i.i.d. multivariate normal with mean vector zero, a one sample t-test is applied to these 8 observations (differences of row sum) to select genes with differential expression level between repair tissue and normal cartilage. Benjamini and Hochberg (1995) procedure of controlling FDR at 2% is used for comparing P-values of the t tests.

Second dataset: M and A are used to represent two chondrocyte culture methods: monolayer and aggregate; N and H are used to represent two oxygen tensions: normoxia and hypoxia. MH, MN, AH and AN represent 4 treatment groups in this 2x2 factorial design. Table 4.2 displays the assumed statistical model. μ_{kj} is the mean response for treatment group with k = 1 (M), 2 (A) and j = 1 (H), 2 (N). $\eta_{\mathbf{i}} = (\eta_{1i}, \eta_{2i})$ are red and green dye effects, respectively; $\gamma_{\mathbf{i}} = (\gamma_{1i}, \gamma_{2i})$ are array 1 and array 2 effects; $\epsilon_{\mathbf{i}} = (\epsilon_{11i}, \epsilon_{12i}, \epsilon_{21i}, \epsilon_{22i})$ is random effect for the ith array set, $i = 1, 2, 3$. The difference is obtained by subtracting sum of second row from sum of first row. These differences are free of extraneous dye and array effects. Based on these differences, linear combinations have been constructed as shown in Table 4.3 to estimate interactions, main effects, and pairwise comparisons. The test statistic is

$$F = \frac{3 * (\sum_{k=1}^{4} c_k \bar{D}_{k.})^2}{s_p^2 \sum_{k=1}^{4} c_k^2} \tag{4.1}$$

where $\bar{D}_{k.}$ is the average difference for the kth block type as shown in Table 4.2. Let s_k^2 be the estimated variance of D_{ki} , $i = 1, \ldots, 3$, and s_p^2 the pooled variance

estimate. c_k is the corresponding coefficient given in Table 4.3. The test statistic is $F(1, 8)$ distributed.

First Data Set

Differentially Expressed Probe Sets Identified with Linear Combinations

The equine cDNA array contains 9413 probe sets. For the first data set, linear comparisons identified 4269 probe sets with differential expression levels between repair tissue and normal cartilage (P-value < 0.01). After Benjamini Hochberg adjustment at 1%, 3327 probe sets remained. Of these probe sets, 1454 demonstrated greater transcript abundance in repair tissue relative to normal articular cartilage, and 1873 demonstrated greater transcript abundance in normal articular cartilage relative to repair tissue. Assessment of probe set annotation by BLAST queries identified 2688 significant probe sets representing 2101 unique gene symbols. Of these, 858 gene symbols were present at higher levels in repair cartilage while 1243 of the gene symbols were at higher levels in normal cartilage.

Evaluation of Other Normalization Methods

Lowess normalization method cannot be applied to this data set since a high proportion of differentially expressed genes between repair tissue and normal cartilage would be expected according to biological knowledge. Also, there is no evidence to show the symmetric expression levels between up-regulated and down-regulated genes. Alternatively, the approach of Lowess normalization using probe sets with constant expression levels can be used with this data set. The bottom row of each array contains a constant set of positive and negative controls. These probe sets are buffer and some house keep genes, which are considered to express constantly across the arrays. Figure 4.1 used the array representing right leg of horse 1 as an example to show this method. A is $log_2\sqrt{RG}$ and M is $log_2 R/G$ with R denoting

repair cartilage and G being normal cartilage. After Lowess normalization, the M values of control probe sets were approximately symmetric around $M = 0$ (Figure 4.1b). Then the normalization curve (red curve) obtained from control probe sets was applied to all probe sets on the same array (Figure 4.1c) to remove the intensity-dependent dye bias. Figure 4.1d depicted the result of all probe sets on this array after Lowess normalization. Other arrays used same procedure to eliminate bias (data not shown). After Lowess normalization, one sample t-test was performed on the average of $log_2 R/G$ obtained from left and right legs of one horse.

The comparisons of two methods (linear combinations versus Lowess using probe sets with constant expression levels) were summarized in Table 4.4. It is obvious that linear combinations identified much more differentially expressed probe sets. Using P-value < 0.01 and FDR adjustment at 0.02 as a cut off, linear combinations identified 4156 probe sets and Lowess using probe sets with constant expression levels identified 1791 probe sets. Both methods shared identification of 1529 probe sets. Fold change has been widely used in microarray experiments to identify genes with different expression levels between two types of samples (e.g., diseased versus normal tissue). A cut off of 2-fold up or down regulation has been chosen to define differential expression in most published studies (Schena et al., 1996; Vaishnav et al., 2008). Hence in this study, we used 2 fold up or down as an additional criterion to further select probe sets with differential expression levels between repair tissue and normal cartilage. Table 4.4 showed that taking P-value, FDR and fold change into accounts, linear combinations identified not only much more probe sets, but also all the probe sets identified by Lowess procedure (last two rows in Table 4.4).

Quantitative Polymerase Chain Reaction Validation of Data

Real time qPCR was further performed on probe sets with biological interests to validate the statistical approaches. For each probe set, means of gene expression

levels of left and right legs obtained from real time qPCR were used to do paired t-test. In Table 4.5, Lowess is Lowess using probe sets with constant expression levels. FC is fold change, mean gene expression levels in repair cartilage/ mean gene expression levels in normal cartilage. It showed that real time qPCR identified DPT, COMP and COL2A1 to be differentially expressed between repair and normal cartilage (P-values < 0.05) (Table 4.5). If we choose the FDR-pvalue cut off at 0.01, linear combinations identified DPT and COMP and failed to find COL2A1. However, Lowess identified none of them. If we relax the cut off of FDR-pvalue to be 0.05, linear combinations successfully identified probe sets corresponding to these three genes. In contrast, Lowess using probe sets with constant expression levels only identified DPT and COMP and failed to identify the majority of probe sets representing COL2A1 (P-values were greater than 0.05 and highlighted). One possible reason is that the leg and horse effects cannot be canceled out by Lowess procedure. The average of $log_2 R/G$ from right and left legs of each horse was considered as one 'replicate'. Hence the effective sample size using Lowess method was 4. In contrast, using linear combinations, the chip, dye, horse and leg effects were removed by calculating the described differences in Table 4.1. Since there were 4 horses with 2 knees per horse, eight such differences were obtained which thus resulted in increased statistical power because it allowed for an increased of the effective sample size to eight. For fold change, real time qPCR showed that DPT transcript abundance was up-regulated in repair tissue while COMP and COL2A1 transcripts were down-regulated in repair tissue. Both linear combinations and Lowess using probe sets with constant expression levels showed the same directions (up or down) of fold change as real time qPCR did. However, compared to Lowess method, the magnitudes of fold change identified by linear combinations were closer to results from real time qPCR. Lowess method tried to draw the M value which is log_2 (fold change) towards 0 during normalization, therefore possibly reducing the fold change measured. It may explain

why Lowess identifies the smaller magnitudes of fold change in either direction (up or down). Note that in normalizing intensities regionally, there were not enough control probe sets to normalize the entire chip (Figure 4.1a). So using probe sets with constant expression levels may not provide a good fit for non-linear normalization curves (Tseng et al., 2001) and therefore may influence both P-values and fold change identified.

Second Data Set

Differentially Expressed Probe Sets Identified with Linear Combinations

For the second data set, linear comparisons identified 87 probe sets with interaction effects (P-value < 0.01). Among 9326 probe sets without interaction effect, 1211 probe sets have culture method effect (P-value < 0.01) and 620 probe sets have oxygen tension effect (P-value < 0.01). Partial pressure of oxygen has been shown to be an important regulator of cell function and gene expression (Henrotin et al., 2005). The aggregate culture method mimics hypoxic condition. Hence for four combinations of oxygen tension and culture methods (AH, AN, MH and MN), aggregate and hypoxia (AH) creates the lowest oxygen condition and monolayer and normoxia (MN) gives the highest oxygen condition. Table 4.6 showed results of pairwise comparisons. These six probe sets in Table 4.6 did not have interaction effect but had both oxygen tension and culture method effects. There is a significant difference between two extreme oxygen conditions (AH vs MN) and the significant difference disappears in two medium oxygen conditions comparison (MH vs AN). When oxygen tension is fixed and we compare two culture methods, we can detect the significant difference between aggregate and monolayer under normoxic oxygen tension and fail to find this difference under hypoxic oxygen tension. One possible reason is that hypoxia weakens the difference of oxygen condition between aggregate and monolayer. Similarly, when culture method is fixed and two oxygen tensions are compared, the

significant difference can be observed only under monolayer culture method. Table 4.6 also showed that PGK-1 expression levels increase under low oxygen condition, which is consistent with findings from Pfander et al. (2003). When we take into account fold change in addition to P-values, the upper three probe sets in Table 4.6 have same gene expression pattern and lower three probe sets have an opposite gene expression pattern. Therefore, the upper two probe sets may have same function as PGK-1 while the lower three probe sets may inhibit this function. This needs to be confirmed with additional biological studies.

Evaluation of Other Normalization Methods

The mixed model approach, which performs independent analyses for each gene, one at a time, is often used to analyze experiments with factorial designs. The drawback of this method is that, it does not suffice to include biological replication in the experiment and to include random effects into the model (Rosa et al., 2005). Many factors may be included into the model, depending on the treatment structure (e.g., factorial experiments) and design settings (e.g., patch effects, spot effects and interactions involving random factors). Steibel et al. (2009) discussed the choice of fixed and random effects in microarray experiment. The choice of fixed and random effects may influence the final results. In contrast with the mixed model approach, the linear combinations procedure avoids estimation of the random effects. Because of the balanced design, dye and chip effects are eliminated by the process of taking difference of row sums in Table 4.2.

4.3 Conclusions

Planned linear combinations first eliminate dye and chip effects by negation and then construct a sequence of comparisons to identify probe sets with differential expression levels. This method is not based on strong biological assumptions and cancels

out the dye and chip effects instead of modeling them. Hence, under the appropriate experimental design, planned linear comparisons can be a powerful alternative to Lowess and mixed model approaches for two-color microarray data analysis.

4.4 Microarray Experiments

First Experiment

One-cm^2 full-thickness articular cartilage lesions were arthroscopically made bilaterally with subseqent microfracture into the subchondral bone of the medial femoral condyles for four adult Quarterhorses (2-3 years) as previously described by Frisbie et al. (1999, 2003). Four months after treatment, repair tissue from the lesions and full-thickness grossly normal articular cartilage from within the same joint were collected from each femorotibial joint (knee). For each horse, two-color cDNA chips were used with each treatment applied on each chip , and the dye combinations swapped on the second chip.

Second Experiment

Chondrocytes were isolated from full-thickness articular cartilage shavings of a one year old horse, split into 12 units and randomly allocated to four treatment groups; 2x2 factorial combination of culture method and oxygen tension. Two chondrocyte culture methods were used, 1) adherent monolayer (M) and 2) non-adherent aggregate (A); and two oxygen tensions were applied, 1) normoxia (20% oxygen concentration) (N) and 2) hypoxia (2% oxygen concentration) (H). Treatments were allocated in an incomplete block arrangement with two treatments per block and three replications of each block. Two chips were used per block as in the first experiment.

For these two experiments, total RNA was isolated from tissue or cell samples, linearly amplified, and hybridized to cDNA microarrays. After the post-hybridization washes, each slide was then scanned using a GenePix 4100A scanner and spot intensi-

ties were computed using GENEPIX 6.0 image analysis software (Axon Instruments / Molecular Devices). After scanning, the median intensities adjusted for background of each pair of spots were used for statistical analysis.

Table 4.1: Model equation for repair tissue versus normal cartilage for kth leg in jth horse, $j = 1, \ldots, 4$ and $k = 1, 2$

Tissue type	Array 1	Array 2
Repair cartilage	$y_{1jk1(1)} = \mu_1 + \tau_j + \lambda_{jk} + \gamma_{jk1} + \eta_{jk1} + \epsilon_{1jk1(1)}(R)$	$y_{1jk2(2)} = \mu_1 + \tau_j + \lambda_{jk} + \gamma_{jk2} + \eta_{jk2} + \epsilon_{1jk2(2)}(G)$
Normal cartilage	$y_{2jk1(2)} = \mu_2 + \tau_j + \lambda_{jk} + \gamma_{jk1} + \eta_{jk2} + \epsilon_{2jk1(2)}(G)$	$y_{2jk2(1)} = \mu_2 + \tau_j + \lambda_{jk} + \gamma_{jk2} + \eta_{jk1} + \epsilon_{2jk2(1)}(R)$
Difference of row sums	$\mu_1 - \mu_2 + \epsilon_{1jk1(1)} + \epsilon_{1jk2(2)} - \epsilon_{2jk1(2)} - \epsilon_{2jk2(1)},$	$i = 1, 2, 3, 4$ and $k = 1, 2.$

Table 4.2: Model equations for one replicate of the incomplete block design

Treatment	Array 1	Array 2
MH	$\mu_{11} + \eta_{1i} + \gamma_{1i} + \epsilon_{11i}, (Red)$	$\mu_{11} + \eta_{2i} + \gamma_{2i} + \epsilon_{22i}, (Green)$
AH	$\mu_{21} + \eta_{2i} + \gamma_{1i} + \epsilon_{21i}, (Green)$	$\mu_{21} + \eta_{1i} + \gamma_{2i} + \epsilon_{12i}, (Red)$
Difference of row sums (D_{1i})	$\mu_{11} - \mu_{21} + \epsilon_{11i} + \epsilon_{22i} - \epsilon_{21i} - \epsilon_{12i},$	$i = 1, 2, 3$
MN	$\mu_{12} + \eta_{1i} + \gamma_{1i} + \epsilon_{11i}, (Red)$	$\mu_{12} + \eta_{2i} + \gamma_{2i} + \epsilon_{22i}, (Green)$
AN	$\mu_{22} + \eta_{2i} + \gamma_{1i} + \epsilon_{21i}, (Green)$	$\mu_{22} + \eta_{1i} + \gamma_{2i} + \epsilon_{12i}, (Red)$
Difference of row sums (D_{2i})	$\mu_{12} - \mu_{22} + \epsilon_{11i} + \epsilon_{22i} - \epsilon_{21i} - \epsilon_{12i},$	$i = 4, 5, 6$
MH	$\mu_{11} + \eta_{1i} + \gamma_{1i} + \epsilon_{11i}, (Red)$	$\mu_{11} + \eta_{2i} + \gamma_{2i} + \epsilon_{22i}, (Green)$
MN	$\mu_{12} + \eta_{2i} + \gamma_{1i} + \epsilon_{21i}, (Green)$	$\mu_{12} + \eta_{1i} + \gamma_{2i} + \epsilon_{12i}, (Red)$
Difference of row sums (D_{3i})	$\mu_{11} - \mu_{12} + \epsilon_{11i} + \epsilon_{22i} - \epsilon_{21i} - \epsilon_{12i},$	$i = 7, 8, 9$
AH	$\mu_{21} + \eta_{1i} + \gamma_{1i} + \epsilon_{11i}, (Red)$	$\mu_{21} + \eta_{2i} + \gamma_{2i} + \epsilon_{22i}, (Green)$
AN	$\mu_{22} + \eta_{2i} + \gamma_{1i} + \epsilon_{21i}, (Green)$	$\mu_{22} + \eta_{1i} + \gamma_{2i} + \epsilon_{12i}, (Red)$
Difference of row sums (D_{4i})	$\mu_{21} - \mu_{22} + \epsilon_{11i} + \epsilon_{22i} - \epsilon_{21i} - \epsilon_{12i},$	$i = 10, 11, 12$

Table 4.3: Linear combinations to identify interaction, main effects and pairwise comparisons

Effects	$\mu_{11} - \mu_{21}$	$\mu_{12} - \mu_{22}$	$\mu_{11} - \mu_{12}$	$\mu_{21} - \mu_{22}$
		Linear Combinations		
Interaction	0.5	-0.5	0.5	-0.5
Culture method	0.5	0.5	0	0
Oxygen tension	0	0	0.5	0.5
AH vs MN	-0.5	-0.5	0.5	0.5
MH vs AN	0.5	0.5	0.5	0.5
MH vs AH	0.5	0.5	0.5	-0.5
MN vs AN	0.5	0.5	-0.5	0.5
MH vs MN	0.5	-0.5	0.5	0.5
AH vs AN	-0.5	0.5	0.5	0.5

Table 4.4: Comparisons of linear combinations with Lowess using probe sets with constant expression levels

Criteria	Linear Combinations	Lowess	In Common
P-value<0.01	4269	2891	2356
P-value<0.01 and FDR adjustment at 0.02	4156	1791	1529
P-value< 0.01, FDR adjustment at 0.02 and 2 fold up in repair cartilage	1134	278	278
P-value<0.01, FDR adjustment at 0.02 and 2 fold down in repair cartilage	1148	186	186

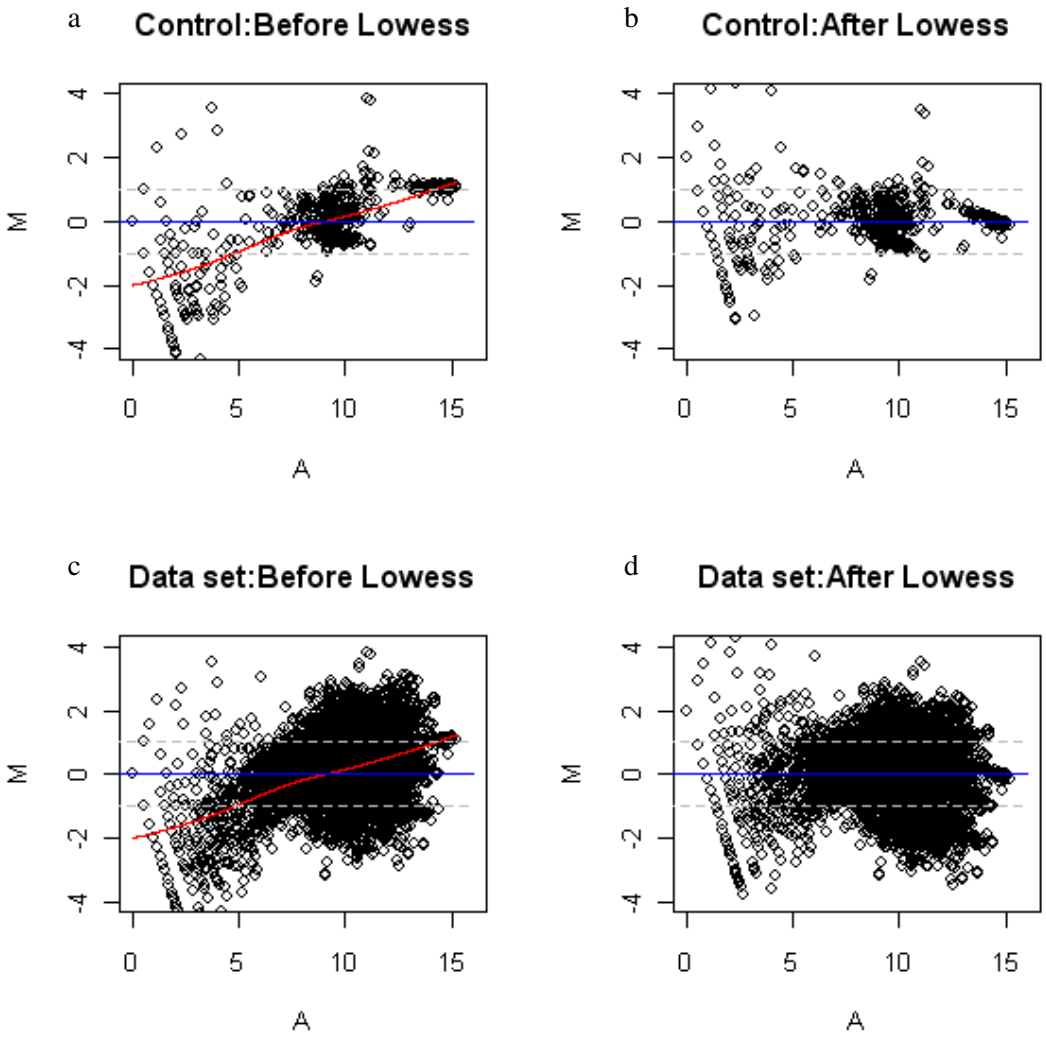
Table 4.5: Using real time qPCR to verify approach of linear combinations

Gene Symbol	Microarray ID (Probe set)	Methods							
		Linear Combinations			Lowess			Real time qPCR	
		P-value	FDR-Pvalue	FC	P-value	FDR-value	FC	P-value	FC
DPT	CT02036A2G09	4.93E-07	8.28E-05	29.51	0.0013	0.0145	5.05	0.0246	77.08
COMP	CT020002B2B06	6.19E-06	2.35E-04	0.07	7.27E-04	1.35E-02	0.30		
	CT020003B1G12	4.92E-05	7.22E-04	0.16	1.92E-03	1.55E-02	0.49		
	CT020004B1E11	5.72E-06	2.30E-04	0.04	5.79E-04	1.30E-02	0.27	0.0020	0.11
	CT020019A2F12	3.66E-06	1.90E-04	0.05	3.49E-04	1.22E-02	0.26		
	CT020027B1G12	4.24E-06	2.04E-04	0.08	1.47E-03	1.47E-02	0.29		
	CT020028B2C08	5.05E-06	2.21E-04	0.07	3.89E-04	1.22E-02	0.31		
COL2A1	CT020023A1F09	8.00E-04	3.47E-03	0.42	2.14E-04	1.16E-02	0.59		
	CT02029A1F01	4.84E-03	1.26E-02	0.32	**7.43E-02**	**1.22E-01**	0.66		
	CT02037A2H05	2.33E-02	4.36E-02	0.43	**2.57E-01**	**3.26E-01**	0.81	0.0182	0.31
	CT02040B2A05	2.25E-02	4.24E-02	0.49	**2.17E-01**	**2.84E-01**	0.82		
	CT02042B1F04	2.64E-02	4.83E-02	0.47	**5.19E-02**	**9.42E-02**	0.70		

Table 4.6: List of probe sets having the same or opposite gene expression pattern as PGK-1

Probeset	Gene Symbol	Interaction -pvalue	Oxygen -pvalue	Culture -pvalue	AH vs MN		MH vs AN		AH vs MH		AN vs MN		MH vs MN		AH vs AN	
					Pvalue	FC	Pvalue	FC	Pvalue	FC	Pvalue	FC	Pvalue	FC	Pvalue	FC
CT02038B1B05	PGK-1	0.0254	0.0023	0.0009	0.0001	9.99	0.6231	0.84	0.0647	2.22	0.0009	5.26	0.0024	4.81	0.0999	1.74
CT02035B1A02	GREM1	0.0205	0.0011	0.0090	0.0004	7.11	0.3151	1.43	0.4563	1.47	0.0036	3.33	0.0015	5.48	0.0542	1.85
CT02029A2F01	PGAM1	0.0288	0.0075	0.0047	0.0008	4.02	0.8202	0.94	0.1962	1.45	0.0050	2.94	0.0050	2.77	0.2697	1.37
CT020015B2D02	RTN4	0.0678	0.0013	0.0028	0.0002	0.18	0.6719	0.89	0.0698	0.62	0.0045	0.33	0.0017	0.32	0.0561	0.51
CT020018A2G06	CTNNB1	0.0363	0.0012	0.0040	0.0002	0.26	0.5518	0.87	0.0794	0.78	0.0066	0.37	0.0009	0.4	0.1193	0.56
CT020023B1F06	CTNNB1	0.0181	0.0036	0.0058	0.0006	0.27	0.8194	0.95	0.1815	0.81	0.0051	0.35	0.0016	0.38	0.3075	0.67

Figure 4.1: Lowess normalization based on probe sets with constant expression levels in one array representing right leg of horse 1



Appendix: R or SAS Source Codes

R Codes for First Topic

Count Frequency - High Dimensional Case

```
library(mvtnorm) library(Matrix) library(glmnet) library(lars)
library(quantreg) library(emplik)

c11<-matrix(0.7,nrow=30,ncol=30) diag(c11)<-1
c12<-matrix(0,nrow=90,ncol=30) c21<-matrix(0,nrow=30,ncol=30)
c22<-matrix(0.7,nrow=30,ncol=30) diag(c22)<-1
c23<-matrix(0.2,nrow=30,ncol=30) c24<-matrix(0,nrow=30,ncol=30)
c31<-matrix(0,30,30) c32<-matrix(0.2,30,30) c33<-matrix(0.7,30,30)
diag(c33)<-1 c34<-matrix(0,30,30) c41<-matrix(0,90,30)
c42<-matrix(0.7,30,30) diag(c42)<-1 c1<-rbind(c11,c12)
c2<-rbind(c21,c22,c23,c24) c3<-rbind(c31,c32,c33,c34)
c4<-rbind(c41,c42) cov<-cbind(c1,c2,c3,c4)

# out function returns non-zero betas out<-function(betaini){
tempnames <- row.names(betaini) index<-!(betaini[ ] == 0 )
row.names(index) <- NULL tempnames[as.vector( index ) ]
r0<-as.matrix(cbind(tempnames[as.vector(index)],betaini[betaini!=0]))
return(r0) }

# count function counts number of non-zero betas
count<-function(bjr){ a<-out(bjr)[,1] nr<-length(a)
b<-as.numeric(out(bjr)[,2]) bnoint<-b[2:nr]
an<-as.numeric(substr(a,2,100)) ann<-an[!is.na(an)]
in1<-ann[ann<=30] in2<-ann[ann>30 & ann<=60]
in3<-ann[ann>60&ann<=90] in4<-ann[ann>90 & ann<=120] f1<-length(in1)
f2<-length(in2) f3<-length(in3) f4<-length(in4)
return(c(f1,f2,f3,f4)) }

#addindC is function to add conditional mean to last observed
#survival time

addingC <- function (x, y, delta, beta) {

#### input must be ordered according to Y and last Y being censored
#### this code is to impute the last Y value.
    N <- length(delta)
    u <- cbind(1,x) %*% beta
```

```
    res <- y - u
    #### kk <- rank(res)[N]
    niceorder <- order(as.vector(res), -delta)
    kk <- which( niceorder == N ) #### this should work with tie
    resorder <- res[niceorder]
    dorder <- delta[niceorder]
    dorder[N] <- 1
    uorder <- u[niceorder]
    ystar <- y[niceorder]
    xorder <- as.matrix(x[niceorder, ])
    temp <- WKM(x = resorder, d = dorder, zc = 1:N)
    jifen <- rev(cumsum(rev(resorder * temp$jump)))
    Sresorder <- temp$surv
    if( Sresorder[kk] >0 ) {
              ystar[kk] <- uorder[kk] + jifen[kk]/(Sresorder[kk])
    }
    return( ystar[kk] )
}

#addingM is a function to add conditional median to observed
#survival time addingM <- function (x, y, delta, beta) {

#### input must be ordered according to (y, -delta). Typically the
#### last y is censored (so needs to be imputed). This code impute
#### the last y value by median and return this value.
    N <- length(delta)
    u <- cbind(1,x) %*% beta
    res <- y - u
    delta[N] <- 1  ### do we need this?
    niceorder <- order(as.vector(res), -delta)
    kk <- which( niceorder == N )
    resorder <- res[niceorder]
    dorder <- delta[niceorder]
    dorder[N] <- 1
    uorder <- u[niceorder]
    ystar <- y[niceorder]
    xorder <- as.matrix(x[niceorder, ])
    temp <- WKM(x = resorder, d = dorder, zc = 1:N)
    Sresorder <- temp$surv
    if( Sresorder[kk] >0 ) {
          Ptheta <- Sresorder[kk]/2
          pivec <- temp$jump
          pivec[1:(kk-1)] <- 0
          posi <- sum( cumsum(pivec) < Ptheta )
          theta <- temp$times[ posi +1]
```

```
          ystar[kk] <- uorder[kk] + theta
    }
    return( ystar[kk] )
}

simcen<-function(cov){ repeat{
x<-rmvnorm(50,mean=rep(0,120),sigma=cov)
a<-x%*%as.matrix(betafix)
e<-rnorm(50) y<-a+15*e c<-140*log(rexp(50,1))+59.7
#c<-runif(50,-190,190) z<-rep(NA,50) d<-rep(NA,50) for (i in 1:50){
if (y[i]<c[i]) {
            z[i]<-y[i]
            d[i]<-1}
    else    {z[i]<-c[i]
            d[i]<-0}
} sorted <- order(z) sz<- as.double(z[sorted])
sy<-as.double(y[sorted]) sstat <- as.integer(d[sorted])
sx<-x[sorted,] me<-mean(sstat[41:50]) if (sstat[50]==0) break sz<-sz
sstat<-sstat sx<-sx sy<-sy }


w0<-rep(1/50,50) w1<-WKM(x=sz,d=sstat)$jump

###w1 treats the last obs to be dead. w2<-w1*sstat

###w2 treats the last obs to be what it originally is (=0).
w<-w1[50] w3<-c(w1[1:49],0.5*w1[50]) w5<-c(w1[1:49],0.25*w1[50])

object30a<-glmnet(sx,sy,weight=w0,alpha=0)
beta30a<-predict(object30a,s=object30a$lambda[100],type="coefficients")
object30b<-glmnet(sx,sy,weight=w0,alpha=0.1)
beta30b<-predict(object30b,s=object30b$lambda[100],type="coefficients")
object30c<-glmnet(sx,sy,weight=w0,alpha=0.5)
beta30c<-predict(object30c,s=object30c$lambda[100],type="coefficients")
object30d<-glmnet(sx,sy,weight=w0,alpha=0.9)
beta30d<-predict(object30d,s=object30d$lambda[100],type="coefficients")
object30e<-glmnet(sx,sy,weight=w0,alpha=1)
beta30e<-predict(object30e,s=object30e$lambda[100],type="coefficients")

object31a<-glmnet(sx,sz,weight=w1,alpha=0)
beta31a<-predict(object31a,s=object31a$lambda[100],type="coefficients")
object31b<-glmnet(sx,sz,weight=w1,alpha=0.1)
beta31b<-predict(object31b,s=object31b$lambda[100],type="coefficients")
object31c<-glmnet(sx,sz,weight=w1,alpha=0.5)
beta31c<-predict(object31c,s=object31c$lambda[100],type="coefficients")
```

```
object31d<-glmnet(sx,sz,weight=w1,alpha=0.9)
beta31d<-predict(object31d,s=object31d$lambda[100],type="coefficients")
object31e<-glmnet(sx,sz,weight=w1,alpha=1)
beta31e<-predict(object31e,s=object31e$lambda[100],type="coefficients")

object32a<-glmnet(sx,sz,weight=w2,alpha=0)
beta32a<-predict(object32a,s=object32a$lambda[100],type="coefficients")
object32b<-glmnet(sx,sz,weight=w2,alpha=0.1)
beta32b<-predict(object32b,s=object32b$lambda[100],type="coefficients")
object32c<-glmnet(sx,sz,weight=w2,alpha=0.5)
beta32c<-predict(object32c,s=object32c$lambda[100],type="coefficients")
object32d<-glmnet(sx,sz,weight=w2,alpha=0.9)
beta32d<-predict(object32d,s=object32d$lambda[100],type="coefficients")
object32e<-glmnet(sx,sz,weight=w2,alpha=1)
beta32e<-predict(object32e,s=object32e$lambda[100],type="coefficients")

object33a<-glmnet(sx,sz,weight=w3,alpha=0)
beta33a<-predict(object33a,s=object33a$lambda[100],type="coefficients")
object33b<-glmnet(sx,sz,weight=w3,alpha=0.1)
beta33b<-predict(object33b,s=object33b$lambda[100],type="coefficients")
object33c<-glmnet(sx,sz,weight=w3,alpha=0.5)
beta33c<-predict(object33c,s=object33c$lambda[100],type="coefficients")
object33d<-glmnet(sx,sz,weight=w3,alpha=0.9)
beta33d<-predict(object33d,s=object33d$lambda[100],type="coefficients")
object33e<-glmnet(sx,sz,weight=w3,alpha=1)
beta33e<-predict(object33e,s=object33e$lambda[100],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a)>0)==1 )
w4a<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a)<0)==1) w4a<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a)<0)==1 )
w4a<-w5

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b)>0)==1 )
w4b<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b)<0)==1) w4b<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b)<0)==1 )
w4b<-w5

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c)>0)==1 )
```

```
w4c<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c)<0)==1) w4c<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c)<0)==1 )
w4c<-w5

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32d)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31d)>0)==1 )
w4d<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32d)<0)==1) w4d<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32d)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31d)<0)==1 )
w4d<-w5

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32e)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31e)>0)==1 )
w4e<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32e)<0)==1) w4e<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32e)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31e)<0)==1 )
w4e<-w5

object33a<-glmnet(sx,sz,weight=w3,alpha=0)
beta33a<-predict(object33a,s=object33a$lambda[100],type="coefficients")
object33b<-glmnet(sx,sz,weight=w3,alpha=0.1)
beta33b<-predict(object33b,s=object33b$lambda[100],type="coefficients")
object33c<-glmnet(sx,sz,weight=w3,alpha=0.5)
beta33c<-predict(object33c,s=object33c$lambda[100],type="coefficients")
object33d<-glmnet(sx,sz,weight=w3,alpha=0.9)
beta33d<-predict(object33d,s=object33d$lambda[100],type="coefficients")
object33e<-glmnet(sx,sz,weight=w3,alpha=1)
beta33e<-predict(object33e,s=object33e$lambda[100],type="coefficients")

object34a<-glmnet(sx,sz,weight=w4a,alpha=0)
beta34a<-predict(object34a,s=object34a$lambda[100],type="coefficients")
object34b<-glmnet(sx,sz,weight=w4b,alpha=0.1)
beta34b<-predict(object34b,s=object34b$lambda[100],type="coefficients")
object34c<-glmnet(sx,sz,weight=w4c,alpha=0.5)
beta34c<-predict(object34c,s=object34c$lambda[100],type="coefficients")
object34d<-glmnet(sx,sz,weight=w4d,alpha=0.9)
beta34d<-predict(object34d,s=object34d$lambda[100],type="coefficients")
object34e<-glmnet(sx,sz,weight=w4e,alpha=1)
beta34e<-predict(object34e,s=object34e$lambda[100],type="coefficients")

n<-50 c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
```

```
usz[50]<-sz[50]+c[i] object35a1<-glmnet(sx,usz,weight=w1,alpha=0)
beta35a1<-predict(object35a1,s=object35a1$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35a1)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c1-1 nsz<-sz
nsz[n]<-sz[n]+cu object35a<-glmnet(sx,nsz,weight=w1,alpha=0)
beta35a<-predict(object35a,s=object35a$lambda[100],type="coefficients")

c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[50]<-sz[50]+c[i] object35b1<-glmnet(sx,usz,weight=w1,alpha=0.1)
beta35b1<-predict(object35b1,s=object35b1$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35b1)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c1-1 nsz<-sz
nsz[n]<-sz[n]+cu object35b<-glmnet(sx,nsz,weight=w1,alpha=0.1)
beta35b<-predict(object35b,s=object35b$lambda[100],type="coefficients")

c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[50]<-sz[50]+c[i] object35c1<-glmnet(sx,usz,weight=w1,alpha=0.5)
beta35c1<-predict(object35c1,s=object35c1$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35c1)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c1-1 nsz<-sz
nsz[n]<-sz[n]+cu object35c<-glmnet(sx,nsz,weight=w1,alpha=0.5)
beta35c<-predict(object35c,s=object35c$lambda[100],type="coefficients")

c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[50]<-sz[50]+c[i] object35d1<-glmnet(sx,usz,weight=w1,alpha=0.9)
beta35d1<-predict(object35d1,s=object35d1$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35d1)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c1-1 nsz<-sz
nsz[n]<-sz[n]+cu object35d<-glmnet(sx,nsz,weight=w1,alpha=0.9)
beta35d<-predict(object35d,s=object35d$lambda[100],type="coefficients")

c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[50]<-sz[50]+c[i] object35e1<-glmnet(sx,usz,weight=w1,alpha=1)
beta35e1<-predict(object35e1,s=object35e1$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35e1)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c1-1 nsz<-sz
nsz[n]<-sz[n]+cu object35e<-glmnet(sx,nsz,weight=w1,alpha=1)
beta35e<-predict(object35e,s=object35e$lambda[100],type="coefficients")

####add conditional mean sstat1<-sstat sstat1[n]<-1 szz<-sz
szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31a)
object31aa<-glmnet(sx,szz,weight=w1,alpha=0)
beta31aa<-predict(object31aa,s=object31aa$lambda[100],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31b)
```

```
object31ba<-glmnet(sx,szz,weight=w1,alpha=0.1)
beta31ba<-predict(object31ba,s=object31ba$lambda[100],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31c)
object31ca<-glmnet(sx,szz,weight=w1,alpha=0.5)
beta31ca<-predict(object31ca,s=object31ca$lambda[100],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31d)
object31da<-glmnet(sx,szz,weight=w1,alpha=0.9)
beta31da<-predict(object31da,s=object31da$lambda[100],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31e)
object31ea<-glmnet(sx,szz,weight=w1,alpha=1)
beta31ea<-predict(object31ea,s=object31ea$lambda[100],type="coefficients")

####add conditional median order7<-order(sz, -sstat) ssz<-sz[order7]
ssx<-sx[order7,] ssd<-sstat[order7] sszz<-ssz
sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31a)
object31aaa<-glmnet(ssx,sszz,weight=w1,alpha=0)
beta31aaa<-predict(object31aaa,s=object31aaa$lambda[100],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31b)
object31baa<-glmnet(ssx,sszz,weight=w1,alpha=0.1)
beta31baa<-predict(object31baa,s=object31baa$lambda[100],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31c)
object31caa<-glmnet(ssx,sszz,weight=w1,alpha=0.5)
beta31caa<-predict(object31caa,s=object31caa$lambda[100],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31d)
object31daa<-glmnet(ssx,sszz,weight=w1,alpha=0.9)
beta31daa<-predict(object31daa,s=object31daa$lambda[100],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31e)
object31eaa<-glmnet(ssx,sszz,weight=w1,alpha=1)
beta31eaa<-predict(object31eaa,s=object31eaa$lambda[100],
type="coefficients")

r0a<-count(beta30a) r0b<-count(beta30b) r0c<-count(beta30c)
r0d<-count(beta30d) r0e<-count(beta30e) r1a<-count(beta31a)
r1b<-count(beta31b) r1c<-count(beta31c) r1d<-count(beta31d)
```

```
r1e<-count(beta31e) r2a<-count(beta32a) r2b<-count(beta32b)
r2c<-count(beta32c) r2d<-count(beta32d) r2e<-count(beta32e)
r3a<-count(beta33a) r3b<-count(beta33b) r3c<-count(beta33c)
r3d<-count(beta33d) r3e<-count(beta33e) r4a<-count(beta34a)
r4b<-count(beta34b) r4c<-count(beta34c) r4d<-count(beta34d)
r4e<-count(beta34e) r5a<-count(beta35a) r5b<-count(beta35b)
r5c<-count(beta35c) r5d<-count(beta35d) r5e<-count(beta35e)
r6a<-count(beta31aa) r6b<-count(beta31ba) r6c<-count(beta31ca)
r6d<-count(beta31da) r6e<-count(beta31ea) r7a<-count(beta31aaa)
r7b<-count(beta31baa) r7c<-count(beta31caa) r7d<-count(beta31daa)
r7e<-count(beta31eaa)
return(c(r0a,r0b,r0c,r0d,r0e,r1a,r1b,r1c,r1d,r1e,
         r2a,r2b,r2c,r2d,r2e,r3a,r3b,r3c,r3d,r3e,
         r4a,r4b,r4c,r4d,r4e,r5a,r5b,r5c,r5d,r5e,
         r6a,r6b,r6c,r6d,r6e,r7a,r7b,r7c,r7d,r7e))
}

set.seed(321) betafix1<-rnorm(60,3,0.5)
betafix<-c(betafix1,rep(0,60)) set.seed(123)
result1<-matrix(NA,1000,140) for (i in 1:1000)
result1[i,]<-simcen(cov)
```

Calculate RPE

```
betafix<-c(c(3,3,2,3,3,2),rep(0,114)) c11<-matrix(0.5,3,3)
diag(c11)<-1 c12<-matrix(0,3,3) c1<-rbind(c11,c12)
c2<-rbind(c12,c11) cc<-cbind(c1,c2) c3<-matrix(0,120,114)
c4<-matrix(0,114,6) cov1<-rbind(cc,c4) cov<-cbind(cov1,c3)



simcen<-function(m1,m2){

repeat{ x<-rmvnorm(100,mean=rep(0,120),sigma=cov)
a<-x%*%as.matrix(betafix)
e<-rnorm(100) y<-a+5*e #c<-runif(100,-k,k) c<-m1*log(rexp(100,1))+m2
z<-rep(NA,100) d<-rep(NA,100) for (i in 1:100){ if (y[i]<c[i]) {
            z[i]<-y[i]
            d[i]<-1}
    else    {z[i]<-c[i]
            d[i]<-0}
} sorted <- order(z) sz<- as.double(z[sorted])
sy<-as.double(y[sorted]) sstat <- as.integer(d[sorted])
sx<-x[sorted,] if (sstat[100]==0) break sz<-sz sstat<-sstat sx<-sx
sy<-sy }
```

```
w0<-rep(1/100,100) w1<-WKM(x=sz,d=sstat)$jump

###w1 treats the last obs to be dead. w2<-w1*sstat ###w2 treats the
last obs to be what it originally is (=0). w<-w1[100]
w3<-c(w1[1:99],0.5*w1[100]) w5<-c(w1[1:99],0.25*w1[100])

#noncensor object30a<-glmnet(sx,sy,weight=w0,alpha=0.1)
beta30a1<-predict(object30a,s=object30a$lambda[10],type="coefficients")
beta30a2<-predict(object30a,s=object30a$lambda[50],type="coefficients")
beta30a3<-predict(object30a,s=object30a$lambda[100],type="coefficients")

object30b<-glmnet(sx,sy,weight=w0,alpha=0.5)
beta30b1<-predict(object30b,s=object30b$lambda[10],type="coefficients")
beta30b2<-predict(object30b,s=object30b$lambda[50],type="coefficients")
beta30b3<-predict(object30b,s=object30b$lambda[100],type="coefficients")

object30c<-glmnet(sx,sy,weight=w0,alpha=0.9)
beta30c1<-predict(object30c,s=object30c$lambda[10],type="coefficients")
beta30c2<-predict(object30c,s=object30c$lambda[50],type="coefficients")
beta30c3<-predict(object30c,s=object30c$lambda[100],type="coefficients")

#weight 1 method object31a<-glmnet(sx,sz,weight=w1,alpha=0.1)
beta31a1<-predict(object31a,s=object31a$lambda[10],type="coefficients")
beta31a2<-predict(object31a,s=object31a$lambda[50],type="coefficients")
beta31a3<-predict(object31a,s=object31a$lambda[100],type="coefficients")

object31b<-glmnet(sx,sz,weight=w1,alpha=0.5)
beta31b1<-predict(object31b,s=object31b$lambda[10],type="coefficients")
beta31b2<-predict(object31b,s=object31b$lambda[50],type="coefficients")
beta31b3<-predict(object31b,s=object31b$lambda[100],type="coefficients")

object31c<-glmnet(sx,sz,weight=w1,alpha=0.9)
beta31c1<-predict(object31c,s=object31c$lambda[10],type="coefficients")
beta31c2<-predict(object31c,s=object31c$lambda[50],type="coefficients")
beta31c3<-predict(object31c,s=object31c$lambda[100],type="coefficients")

#weight 0 method object32a<-glmnet(sx,sz,weight=w2,alpha=0.1)
beta32a1<-predict(object32a,s=object32a$lambda[10],type="coefficients")
beta32a2<-predict(object32a,s=object32a$lambda[50],type="coefficients")
beta32a3<-predict(object32a,s=object32a$lambda[100],type="coefficients")

object32b<-glmnet(sx,sz,weight=w2,alpha=0.5)
beta32b1<-predict(object32b,s=object32b$lambda[10],type="coefficients")
beta32b2<-predict(object32b,s=object32b$lambda[50],type="coefficients")
beta32b3<-predict(object32b,s=object32b$lambda[100],type="coefficients")
```

```
object32c<-glmnet(sx,sz,weight=w2,alpha=0.9)
beta32c1<-predict(object32c,s=object32c$lambda[10],type="coefficients")
beta32c2<-predict(object32c,s=object32c$lambda[50],type="coefficients")
beta32c3<-predict(object32c,s=object32c$lambda[100],type="coefficients")

#weight average method object33a<-glmnet(sx,sz,weight=w3,alpha=0.1)
beta33a1<-predict(object33a,s=object33a$lambda[10],type="coefficients")
beta33a2<-predict(object33a,s=object33a$lambda[50],type="coefficients")
beta33a3<-predict(object33a,s=object33a$lambda[100],type="coefficients")

object33b<-glmnet(sx,sz,weight=w3,alpha=0.5)
beta33b1<-predict(object33b,s=object33b$lambda[10],type="coefficients")
beta33b2<-predict(object33b,s=object33b$lambda[50],type="coefficients")
beta33b3<-predict(object33b,s=object33b$lambda[100],type="coefficients")

object33c<-glmnet(sx,sz,weight=w3,alpha=0.9)
beta33c1<-predict(object33c,s=object33c$lambda[10],type="coefficients")
beta33c2<-predict(object33c,s=object33c$lambda[50],type="coefficients")
beta33c3<-predict(object33c,s=object33c$lambda[100],type="coefficients")

# constant adding n<-100
min1<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32a1)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.1)
beta35aa<-predict(object35aa,s=object35aa$lambda[10],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc1<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35a1<-glmnet(sx,nsz,weight=w1,alpha=0.1)
beta35a1<-predict(object35a1,s=object35a1$lambda[10],type="coefficients")

min2<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32a2)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.1)
beta35aa<-predict(object35aa,s=object35aa$lambda[50],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc2<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35a2<-glmnet(sx,nsz,weight=w1,alpha=0.1)
beta35a2<-predict(object35a2,s=object35a2$lambda[50],type="coefficients")

min3<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32a3)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.1)
```

```
beta35aa<-predict(object35aa,s=object35aa$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc3<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35a3<-glmnet(sx,nsz,weight=w1,alpha=0.1)
beta35a3<-predict(object35a3,s=object35a3$lambda[100],type="coefficients")

min4<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32b1)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.5)
beta35aa<-predict(object35aa,s=object35aa$lambda[10],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc4<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35b1<-glmnet(sx,nsz,weight=w1,alpha=0.5)
beta35b1<-predict(object35b1,s=object35b1$lambda[10],type="coefficients")

min5<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32b2)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.5)
beta35aa<-predict(object35aa,s=object35aa$lambda[50],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc5<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35b2<-glmnet(sx,nsz,weight=w1,alpha=0.5)
beta35b2<-predict(object35b2,s=object35b2$lambda[50],type="coefficients")

min6<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32b3)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.5)
beta35aa<-predict(object35aa,s=object35aa$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc6<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35b3<-glmnet(sx,nsz,weight=w1,alpha=0.5)
beta35b3<-predict(object35b3,s=object35b3$lambda[100],type="coefficients")

min7<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32c1)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.9)
beta35aa<-predict(object35aa,s=object35aa$lambda[10],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc7<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35c1<-glmnet(sx,nsz,weight=w1,alpha=0.9)
```

```
beta35c1<-predict(object35c1,s=object35c1$lambda[10],type="coefficients")

min8<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32c2)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.9)
beta35aa<-predict(object35aa,s=object35aa$lambda[50],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc8<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35c2<-glmnet(sx,nsz,weight=w1,alpha=0.9)
beta35c2<-predict(object35c2,s=object35c2$lambda[50],type="coefficients")

min9<-as.numeric(sz[n]-c(1,sx[n,])%*%beta32c3)
c<-seq(0,100,1) resi<-rep(0,101) for (i in 1:101){ usz<-sz
usz[n]<-sz[n]+c[i] object35aa<-glmnet(sx,usz,weight=w1,alpha=0.9)
beta35aa<-predict(object35aa,s=object35aa$lambda[100],type="coefficients")
resi[i]<-sum(w1*(usz-cbind(1,sx)%*%beta35aa)^2)
} c1<-which(resi == min(resi), arr.ind = TRUE) cu<-c[c1] cc9<-cu
nsz<-sz nsz[n]<-sz[n]+cu
object35c3<-glmnet(sx,nsz,weight=w1,alpha=0.9)
beta35c3<-predict(object35c3,s=object35c3$lambda[100],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a1)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a1)>0)==1 )
w4a1<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a1)<0)==1)
w4a1<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a1)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a1)<0)==1 )
w4a1<-w5 object34a1<-glmnet(sx,sz,weight=w4a1,alpha=0.1)
beta34a1<-predict(object34a1,s=object34a1$lambda[10],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a2)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a2)>0)==1 )
w4a2<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a2)<0)==1) w4a2<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a2)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a2)<0)==1 )
w4a2<-w5 object34a2<-glmnet(sx,sz,weight=w4a2,alpha=0.1)
beta34a2<-predict(object34a2,s=object34a2$lambda[50],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a3)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a3)>0)==1 )
w4a3<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a3)<0)==1)
```

```
w4a3<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32a3)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31a3)<0)==1 )
w4a3<-w5 object34a3<-glmnet(sx,sz,weight=w4a3,alpha=0.1)
beta34a3<-predict(object34a3,s=object34a3$lambda[100],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b1)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b1)>0)==1 )
w4b1<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b1)<0)==1)
w4b1<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b1)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b1)<0)==1 )
w4b1<-w5 object34b1<-glmnet(sx,sz,weight=w4b1,alpha=0.5)
beta34b1<-predict(object34b1,s=object34b1$lambda[10],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b2)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b2)>0)==1 )
w4b2<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b2)<0)==1)
w4b2<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b2)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b2)<0)==1 )
w4b2<-w5 object34b2<-glmnet(sx,sz,weight=w4b2,alpha=0.5)
beta34b2<-predict(object34b2,s=object34b2$lambda[50],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b3)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b3)>0)==1 )
w4b3<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b3)<0)==1)
w4b3<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32b3)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31b3)<0)==1 )
w4b3<-w5 object34b3<-glmnet(sx,sz,weight=w4b3,alpha=0.5)
beta34b3<-predict(object34b3,s=object34b3$lambda[100],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c1)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c1)>0)==1 )
w4c1<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c1)<0)==1)
w4c1<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c1)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c1)<0)==1 )
w4c1<-w5 object34c1<-glmnet(sx,sz,weight=w4c1,alpha=0.9)
beta34c1<-predict(object34c1,s=object34c1$lambda[10],type="coefficients")
```

```
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c2)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c2)>0)==1 )
w4c2<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c2)<0)==1)
w4c2<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c2)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c2)<0)==1 )
w4c2<-w5 object34c2<-glmnet(sx,sz,weight=w4c2,alpha=0.9)
beta34c2<-predict(object34c2,s=object34c2$lambda[50],type="coefficients")

if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c3)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c3)>0)==1 )
w4c3<-w1
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c3)<0)==1)
w4c3<-w2
if (as.numeric((sz[50]-c(1,sx[50,])%*%beta32c3)>0)==1 &
as.numeric((sz[50]-c(1,sx[50,])%*%beta31c3)<0)==1 )
w4c3<-w5 object34c3<-glmnet(sx,sz,weight=w4c3,alpha=0.9)
beta34c3<-predict(object34c3,s=object34c3$lambda[100],type="coefficients")

#add conditional mean sstat1<-sstat sstat1[n]<-1 szz<-sz
szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31a1)
object31a1a<-glmnet(sx,szz,weight=w1,alpha=0.1)
beta31a1a<-predict(object31a1a,s=object31a1a$lambda[10],
type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31a2)
object31a2a<-glmnet(sx,szz,weight=w1,alpha=0.1)
beta31a2a<-predict(object31a2a,s=object31a2a$lambda[50],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31a3)
object31a3a<-glmnet(sx,szz,weight=w1,alpha=0.1)
beta31a3a<-predict(object31a3a,s=object31a3a$lambda[100],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31b1)
object31b1a<-glmnet(sx,szz,weight=w1,alpha=0.5)
beta31b1a<-predict(object31b1a,s=object31b1a$lambda[10],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31b2)
object31b2a<-glmnet(sx,szz,weight=w1,alpha=0.5)
beta31b2a<-predict(object31b2a,s=object31b2a$lambda[50],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31b3)
object31b3a<-glmnet(sx,szz,weight=w1,alpha=0.5)
```

```
beta31b3a<-predict(object31b3a,s=object31b3a$lambda[100],
type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31c1)
object31c1a<-glmnet(sx,szz,weight=w1,alpha=0.9)
beta31c1a<-predict(object31c1a,s=object31c1a$lambda[10],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31c2)
object31c2a<-glmnet(sx,szz,weight=w1,alpha=0.9)
beta31c2a<-predict(object31c2a,s=object31c2a$lambda[50],type="coefficients")

szz<-sz szz[n]<-addingC(x=sx,y=sz,d=sstat1,beta=beta31c3)
object31c3a<-glmnet(sx,szz,weight=w1,alpha=0.9)
beta31c3a<-predict(object31c3a,s=object31c3a$lambda[100],
type="coefficients")

#adding conditional median order7<-order(sz, -sstat) ssz<-sz[order7]
ssx<-sx[order7,] ssd<-sstat[order7] sszz<-ssz
sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31a1)
object31a1aa<-glmnet(ssx,sszz,weight=w1,alpha=0.1)
beta31a1aa<-predict(object31a1aa,s=object31a1aa$lambda[10],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31a2)
object31a2aa<-glmnet(ssx,sszz,weight=w1,alpha=0.1)
beta31a2aa<-predict(object31a2aa,s=object31a2aa$lambda[50],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31a3)
object31a3aa<-glmnet(ssx,sszz,weight=w1,alpha=0.1)
beta31a3aa<-predict(object31a3aa,s=object31a3aa$lambda[100],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31b1)
object31b1aa<-glmnet(ssx,sszz,weight=w1,alpha=0.5)
beta31b1aa<-predict(object31b1aa,s=object31b1aa$lambda[10],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31b2)
object31b2aa<-glmnet(ssx,sszz,weight=w1,alpha=0.5)
beta31b2aa<-predict(object31b2aa,s=object31b2aa$lambda[50],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31b3)
object31b3aa<-glmnet(ssx,sszz,weight=w1,alpha=0.5)
```

```
beta31b3aa<-predict(object31b3aa,s=object31b3aa$lambda[100],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31c1)
object31c1aa<-glmnet(ssx,sszz,weight=w1,alpha=0.9)
beta31c1aa<-predict(object31c1aa,s=object31c1aa$lambda[10],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31c2)
object31c2aa<-glmnet(ssx,sszz,weight=w1,alpha=0.9)
beta31c2aa<-predict(object31c2aa,s=object31c2aa$lambda[50],
type="coefficients")

sszz<-ssz sszz[n]<-addingM(x=ssx, y=ssz, d=ssd, beta=beta31c3)
object31c3aa<-glmnet(ssx,sszz,weight=w1,alpha=0.9)
beta31c3aa<-predict(object31c3aa,s=object31c3aa$lambda[100],
type="coefficients")


xtest<-rmvnorm(400,mean=rep(0,120),sigma=cov) xtest1<-cbind(1,xtest)
ap<-xtest%*%as.matrix(betafix)
ep<-rnorm(400) yp<-ap+5*e

yesti0a1<-xtest1%*%beta30a1
yesti0a2<-xtest1%*%beta30a2
yesti0a3<-xtest1%*%beta30a3

yesti0b1<-xtest1%*%beta30b1
yesti0b2<-xtest1%*%beta30b2
yesti0b3<-xtest1%*%beta30b3

yesti0c1<-xtest1%*%beta30c1
yesti0c2<-xtest1%*%beta30c2
yesti0c3<-xtest1%*%beta30c3

yesti1a1<-xtest1%*%beta31a1
yesti1a2<-xtest1%*%beta31a2
yesti1a3<-xtest1%*%beta31a3

yesti1b1<-xtest1%*%beta31b1
yesti1b2<-xtest1%*%beta31b2
yesti1b3<-xtest1%*%beta31b3

yesti1c1<-xtest1%*%beta31c1
yesti1c2<-xtest1%*%beta31c2
```

```
yesti1c3<-xtest1%*%beta31c3

yesti2a1<-xtest1%*%beta32a1
yesti2a2<-xtest1%*%beta32a2
yesti2a3<-xtest1%*%beta32a3

yesti2b1<-xtest1%*%beta32b1
yesti2b2<-xtest1%*%beta32b2
yesti2b3<-xtest1%*%beta32b3

yesti2c1<-xtest1%*%beta32c1
yesti2c2<-xtest1%*%beta32c2
yesti2c3<-xtest1%*%beta32c3

yesti3a1<-xtest1%*%beta33a1
yesti3a2<-xtest1%*%beta33a2
yesti3a3<-xtest1%*%beta33a3

yesti3b1<-xtest1%*%beta33b1
yesti3b2<-xtest1%*%beta33b2
yesti3b3<-xtest1%*%beta33b3

yesti3c1<-xtest1%*%beta33c1
yesti3c2<-xtest1%*%beta33c2
yesti3c3<-xtest1%*%beta33c3

yesti4a1<-xtest1%*%beta34a1
yesti4a2<-xtest1%*%beta34a2
yesti4a3<-xtest1%*%beta34a3

yesti4b1<-xtest1%*%beta34b1
yesti4b2<-xtest1%*%beta34b2
yesti4b3<-xtest1%*%beta34b3

yesti4c1<-xtest1%*%beta34c1
yesti4c2<-xtest1%*%beta34c2
yesti4c3<-xtest1%*%beta34c3


yesti5a1<-xtest1%*%beta35a1
yesti5a2<-xtest1%*%beta35a2
yesti5a3<-xtest1%*%beta35a3

yesti5b1<-xtest1%*%beta35b1
yesti5b2<-xtest1%*%beta35b2
```

```
yesti5b3<-xtest1%*%beta35b3

yesti5c1<-xtest1%*%beta35c1
yesti5c2<-xtest1%*%beta35c2
yesti5c3<-xtest1%*%beta35c3


yesti6a1<-xtest1%*%beta31a1a
yesti6a2<-xtest1%*%beta31a2a
yesti6a3<-xtest1%*%beta31a3a

yesti6b1<-xtest1%*%beta31b1a
yesti6b2<-xtest1%*%beta31b2a
yesti6b3<-xtest1%*%beta31b3a

yesti6c1<-xtest1%*%beta31c1a
yesti6c2<-xtest1%*%beta31c2a
yesti6c3<-xtest1%*%beta31c3a

yesti7a1<-xtest1%*%beta31a1aa
yesti7a2<-xtest1%*%beta31a2aa
yesti7a3<-xtest1%*%beta31a3aa

yesti7b1<-xtest1%*%beta31b1aa
yesti7b2<-xtest1%*%beta31b2aa
yesti7b3<-xtest1%*%beta31b3aa

yesti7c1<-xtest1%*%beta31c1aa
yesti7c2<-xtest1%*%beta31c2aa
yesti7c3<-xtest1%*%beta31c3aa

p0a1<-mean((yp-yesti0a1)^2)/5/5 p0a2<-mean((yp-yesti0a2)^2)/5/5
p0a3<-mean((yp-yesti0a3)^2)/5/5

p0b1<-mean((yp-yesti0b1)^2)/5/5 p0b2<-mean((yp-yesti0b2)^2)/5/5
p0b3<-mean((yp-yesti0b3)^2)/5/5

p0c1<-mean((yp-yesti0c1)^2)/5/5 p0c2<-mean((yp-yesti0c2)^2)/5/5
p0c3<-mean((yp-yesti0c3)^2)/5/5

p1a1<-mean((yp-yesti1a1)^2)/5/5 p1a2<-mean((yp-yesti1a2)^2)/5/5
p1a3<-mean((yp-yesti1a3)^2)/5/5

p1b1<-mean((yp-yesti1b1)^2)/5/5 p1b2<-mean((yp-yesti1b2)^2)/5/5
p1b3<-mean((yp-yesti1b3)^2)/5/5
```

```
p1c1<-mean((yp-yesti1c1)^2)/5/5 p1c2<-mean((yp-yesti1c2)^2)/5/5
p1c3<-mean((yp-yesti1c3)^2)/5/5

p2a1<-mean((yp-yesti2a1)^2)/5/5 p2a2<-mean((yp-yesti2a2)^2)/5/5
p2a3<-mean((yp-yesti2a3)^2)/5/5

p2b1<-mean((yp-yesti2b1)^2)/5/5 p2b2<-mean((yp-yesti2b2)^2)/5/5
p2b3<-mean((yp-yesti2b3)^2)/5/5

p2c1<-mean((yp-yesti2c1)^2)/5/5 p2c2<-mean((yp-yesti2c2)^2)/5/5
p2c3<-mean((yp-yesti2c3)^2)/5/5

p3a1<-mean((yp-yesti3a1)^2)/5/5 p3a2<-mean((yp-yesti3a2)^2)/5/5
p3a3<-mean((yp-yesti3a3)^2)/5/5

p3b1<-mean((yp-yesti3b1)^2)/5/5 p3b2<-mean((yp-yesti3b2)^2)/5/5
p3b3<-mean((yp-yesti3b3)^2)/5/5

p3c1<-mean((yp-yesti3c1)^2)/5/5 p3c2<-mean((yp-yesti3c2)^2)/5/5
p3c3<-mean((yp-yesti3c3)^2)/5/5

p4a1<-mean((yp-yesti4a1)^2)/5/5 p4a2<-mean((yp-yesti4a2)^2)/5/5
p4a3<-mean((yp-yesti4a3)^2)/5/5

p4b1<-mean((yp-yesti4b1)^2)/5/5 p4b2<-mean((yp-yesti4b2)^2)/5/5
p4b3<-mean((yp-yesti4b3)^2)/5/5

p4c1<-mean((yp-yesti4c1)^2)/5/5 p4c2<-mean((yp-yesti4c2)^2)/5/5
p4c3<-mean((yp-yesti4c3)^2)/5/5

p5a1<-mean((yp-yesti5a1)^2)/5/5 p5a2<-mean((yp-yesti5a2)^2)/5/5
p5a3<-mean((yp-yesti5a3)^2)/5/5

p5b1<-mean((yp-yesti5b1)^2)/5/5 p5b2<-mean((yp-yesti5b2)^2)/5/5
p5b3<-mean((yp-yesti5b3)^2)/5/5

p5c1<-mean((yp-yesti5c1)^2)/5/5 p5c2<-mean((yp-yesti5c2)^2)/5/5
p5c3<-mean((yp-yesti5c3)^2)/5/5


p6a1<-mean((yp-yesti6a1)^2)/5/5 p6a2<-mean((yp-yesti6a2)^2)/5/5
p6a3<-mean((yp-yesti6a3)^2)/5/5

p6b1<-mean((yp-yesti6b1)^2)/5/5 p6b2<-mean((yp-yesti6b2)^2)/5/5
```

```
p6b3<-mean((yp-yesti6b3)^2)/5/5

p6c1<-mean((yp-yesti6c1)^2)/5/5 p6c2<-mean((yp-yesti6c2)^2)/5/5
p6c3<-mean((yp-yesti6c3)^2)/5/5

p7a1<-mean((yp-yesti7a1)^2)/5/5 p7a2<-mean((yp-yesti7a2)^2)/5/5
p7a3<-mean((yp-yesti7a3)^2)/5/5

p7b1<-mean((yp-yesti7b1)^2)/5/5 p7b2<-mean((yp-yesti7b2)^2)/5/5
p7b3<-mean((yp-yesti7b3)^2)/5/5

p7c1<-mean((yp-yesti7c1)^2)/5/5 p7c2<-mean((yp-yesti7c2)^2)/5/5
p7c3<-mean((yp-yesti7c3)^2)/5/5


cc<-c(p0a1,p1a1,p2a1,p3a1,p4a1,p5a1,p6a1,p7a1,
      p0a2,p1a2,p2a2,p3a2,p4a2,p5a2,p6a2,p7a2,
      p0a3,p1a3,p2a3,p3a3,p4a3,p5a3,p6a3,p7a3,
      p0b1,p1b1,p2b1,p3b1,p4b1,p5b1,p6b1,p7b1,
      p0b2,p1b2,p2b2,p3b2,p4b2,p5b2,p6b2,p7b2,
      p0b3,p1b3,p2b3,p3b3,p4b3,p5b3,p6b3,p7b3,
      p0c1,p1c1,p2c1,p3c1,p4c1,p5c1,p6c1,p7c1,
      p0c2,p1c2,p2c2,p3c2,p4c2,p5c2,p6c2,p7c2,
      p0c3,p1c3,p2c3,p3c3,p4c3,p5c3,p6c3,p7c3)
return(cc) }

set.seed(123) result1<-matrix(NA,1000,72) for (i in 1:1000)
#result1[i,]<-simcen(20) result1[i,]<-simcen(12,5)
```

SAS Codes for Second Topic

```
/*read data*/
%macro impdata(exceldata, sasdata);
proc import out=&sasdata
            datafile="D:\RA\horsedata\wenying\data\&exceldata..xls"
            dbms=excel2000 replace;
        getnames=yes;
run; proc sort data=&sasdata; by id; run;
%mend;
%impdata(cartcerea,cartcerea)
%impdata(cartcerem,cartcerem)
%impdata(cartspleea,cartspleea)
%impdata(cartspleem,cartspleem)
%impdata(cartlyma,cartlyma)
%impdata(cartlymm,cartlymm)
```

```
%impdata(cartblada,cartblada)
%impdata(cartbladm,cartbladm)
%impdata(cartlivera,cartlivera)
%impdata(cartliverm,cartliverm)
%impdata(cartmusca,cartmusca)
%impdata(cartmuscm,cartmuscm)
%impdata(cartplacea,cartplacea)
%impdata(cartplacem,cartplacem)
%impdata(carttesta,carttesta)
%impdata(carttestm,carttestm)
%impdata(cartlunga,cartlunga)
%impdata(cartlungm,cartlungm)
%impdata(cartkida,cartkida)
%impdata(cartkidm,cartkidm)

%macro re(cartdata,cartdata1,cartdata2,a1,a2,m2,m1);
data &cartdata; merge &cartdata1 &cartdata2; by id;
log_a=(&a1+&a2)/2; log_m=/*(&m1-&m2)/2-for lung and kidney*/
(&m1+&m2)/2; keep id log_a log_m; run;
%mend;
%re(cartcere,cartcerea,cartcerem,cartcerea1,cartcerea2,cartcerem2,
cartcerem1)
%re(cartsplee,cartspleea,cartspleem,cartspleea1,cartspleea2,cartspleem2,
cartspleem1)
%re(cartlym,cartlyma,cartlymm,cartlyma1,cartlyma2,cartlymm2,
cartlymm1)
%re(cartblad,cartblada,cartbladm,cartblada1,cartblada2,cartbladm2,
cartbladm1)
%re(cartliver,cartlivera,cartliverm,cartlivera1,cartlivera2,cartliverm2,
cartliverm1)
%re(cartmusc,cartmusca,cartmuscm,cartmusca1,cartmusca2,cartmuscm2,
cartmuscm1)
%re(cartplace,cartplacea,cartplacem,cartplacea1,cartplacea2,cartplacem2,
cartplacem1)
%re(carttest,carttesta,carttestm,carttesta1,carttesta2,carttestm2,
carttestm1)
%re(cartlung,cartlunga,cartlungm,cartlunga1,cartlunga2,cartlungm2,
cartlungm1)
%re(cartkid,cartkida,cartkidm,cartkida1,cartkida2,cartkidm2,
cartkidm1)

%macro reg(cartout,out);
proc transreg design data=&cartout details; model
bspline(log_a/knots=  6 8 10 12 14 /*6 8 10 12 14 -for lung and
kidney*/); output out=&out(drop=_: int:); proc sort data=&cartout;by
```

```
log_a; proc sort data=&out;by log_a; run;
%mend;
%reg(cartcere,cere)
%reg(cartsplee,splee)
%reg(cartlym,lym)
%reg(cartblad,blad)
%reg(cartliver,liver)
%reg(cartmusc,musc)
%reg(cartplace,place)
%reg(carttest,test)
%reg(cartlung,lung)
%reg(cartkid,kid)

/*quantile regression*/ proc quantreg data=cere alpha=0.1
ci=resampling; model log_m=  log_a / quantile= .95
  seed=0;
output out=cere95 pred=predcere95; run;

proc sort data=cere95;by id;run;

proc quantreg data=cere alpha=0.1 ci=resampling; model log_m= log_a
/ quantile= .99
  seed=0;
output out=cere99 pred=predcere99; run;

proc sort data=cere99;by id;run;

proc quantreg data=cere alpha=0.1 ci=resampling; model log_m=  log_a
/ quantile= .80
  seed=0;
output out=cere80  pred=predcere80; run;

proc sort data=cere80;by id;run;

proc quantreg data=cere alpha=0.1 ci=resampling; model log_m= log_a
/ quantile= .50
  seed=0;
output out=cere50  pred=predcere50; run;

proc sort data=cere50;by id;run;

proc quantreg data=cere alpha=0.1 ci=resampling; model log_m=  log_a
/ quantile= .10
  seed=0;
output out=cere10  pred=predcere10; run;
```

```
proc sort data=cere10;by id;run;

proc quantreg data=cere alpha=0.1 ci=resampling; model log_m=log_a /
quantile= .05
  seed=0;
output out=cere05  pred=predcere05; run;

proc sort data=cere05;by id;run;

proc quantreg data=cere alpha=0.1 ci=resampling; model log_m=  log_a
/ quantile= .01
  seed=0;
output out=cere01  pred=predcere01; run;

proc sort data=cere01;by id;run;

proc quantreg data=cere alpha=0.1 ci=resampling; model log_m=  log_a
/ quantile= .20
  seed=0;
output out=cere20  pred=predcere20; run;

proc sort data=cere20;by id;run;

data all1; merge cere01 cere05 cere10 cere20 cere50 cere80 cere90
cere95 cere99;by id;run;
```

SAS Codes for Third Topic

```
%macro impdata(exceldata, sasdata);
proc import out=&sasdata
            datafile="C:\huangliping\RA\RA\horsedata\naoki\2oxigen2culture
            \data\data\&exceldata..xls"
            dbms=excel2000 replace;
        getnames=yes;run;
        data &sasdata;
        set &sasdata(rename=(F635_Median___B635=&sasdata._red
        F532_Median___B532=&sasdata._green));
        keep id &sasdata._red &sasdata._green Flags; run;
proc sort data=&sasdata; by id; run;
%mend;
%impdata(IA-AN7-1-AH7-1-1 092805,a1)
%impdata(IA-AN7-1-AH7-1-A 092805,a1a)
%impdata(IA-Re-AN7-2-AH7-2-1_032306,a2)
```

```
%impdata(IA-Re-AN7-2-AH7-2-A_032306,a2a)
%impdata(IA-AN7-3-AH7-3-1 093005,a3)
%impdata(IA-AN7-3-AH7-3-A 093005,a3a)

%impdata(IA-MN7-1-MH7-1-1 091605,b1)
%impdata(IA-MN7-1-MH7-1-A 091605,b1a)
%impdata(IA-MN7-2-MH7-2-1 092305,b2)
%impdata(IA-MN7-2-MH7-2-A 092305,b2a)
%impdata(IA-MN7-3-MH7-3-1 092305,b3)
%impdata(IA-MN7-3-MH7-3-A 092305,b3a)

%impdata(IA-AH7-1-MH7-1-1 101205,c1)
%impdata(IA-AH7-1-MH7-1-A 101205,c1a)
%impdata(IA-AH7-2-MH7-2-2 101205,c2)
%impdata(IA-AH7-2-MH7-2-A 101205,c2a)
%impdata(IA-AH7-3-MH7-3-1 101405,c3)
%impdata(IA-AH7-3-MH7-3-B 101405 ,c3a)

%impdata(IA-AN7-1-MN7-1-1 100705,d1)
%impdata(IA-AN7-1-MN7-1-A 100705,d1a)
%impdata(IA-AN7-2-MN7-2-1 100705,d2)
%impdata(IA-AN7-2-MN7-2-A 100705,d2a)
%impdata(IA-AN7-3-MN7-3-1 093005,d3)
%impdata(IA-AN7-3-MN7-3-A 093005,d3a)

%macro red(sasdata,rsasdata);
ods listing close; ods noresults; ods output
BasicMeasures=red_&sasdata; proc univariate data=&sasdata; by ID;
var &sasdata._red; run; ods listing; ods results; data &rsasdata;
set red_&sasdata(rename=(LocValue=&sasdata._rred)); where
LocMeasure='Median'; keep  id &sasdata._rred; run;
%mend;
%red(a1,ra1)
%red(a1a,ra1a)
%red(a2,ra2)
%red(a2a,ra2a)
%red(a3,ra3)
%red(a3a,ra3a)
%red(b1,rb1)
%red(b1a,rb1a)
%red(b2,rb2)
%red(b2a,rb2a)
%red(b3,rb3)
%red(b3a,rb3a)
%red(c1,rc1)
```

```
%red(c1a,rc1a)
%red(c2,rc2)
%red(c2a,rc2a)
%red(c3,rc3)
%red(c3a,rc3a)
%red(d1,rd1)
%red(d1a,rd1a)
%red(d2,rd2)
%red(d2a,rd2a)
%red(d3,rd3)
%red(d3a,rd3a)

%macro green(sasdata,gsasdata);
ods listing close; ods noresults; ods output
BasicMeasures=green_&sasdata; proc univariate data=&sasdata; by ID;
var &sasdata._green; run; ods listing; ods results; data &gsasdata;
set green_&sasdata(rename=(LocValue=&sasdata._ggreen));

where LocMeasure='Median'; keep  id &sasdata._ggreen; run;
%mend;
%green(a1,ga1)
%green(a1a,ga1a)
%green(a2,ga2)
%green(a2a,ga2a)
%green(a3,ga3)
%green(a3a,ga3a)
%green(b1,gb1)
%green(b1a,gb1a)
%green(b2,gb2)
%green(b2a,gb2a)
%green(b3,gb3)
%green(b3a,gb3a)
%green(c1,gc1)
%green(c1a,gc1a)
%green(c2,gc2)
%green(c2a,gc2a)
%green(c3,gc3)
%green(c3a,gc3a)
%green(d1,gd1)
%green(d1a,gd1a)
%green(d2,gd2)
%green(d2a,gd2a)
%green(d3,gd3)
%green(d3a,gd3a)
```

```
%macro revise(ra1,a1_rred,a1_red);
data &ra1; set &ra1; if &a1_rred<=0 then &a1_red=1; if &a1_rred>0
then &a1_red=&a1_rred;drop &a1_rred; proc sort data=&ra1; by id;run;
%mend;
%revise(ra1,a1_rred,a1_red)
%revise(ra1a,a1a_rred,a1a_red)
%revise(ra2,a2_rred,a2_red)
%revise(ra2a,a2a_rred,a2a_red)
%revise(ra3,a3_rred,a3_red)
%revise(ra3a,a3a_rred,a3a_red)

%revise(rb1,b1_rred,b1_red)
%revise(rb1a,b1a_rred,b1a_red)
%revise(rb2,b2_rred,b2_red)
%revise(rb2a,b2a_rred,b2a_red)
%revise(rb3,b3_rred,b3_red)
%revise(rb3a,b3a_rred,b3a_red)

%revise(rc1,c1_rred,c1_red)
%revise(rc1a,c1a_rred,c1a_red)
%revise(rc2,c2_rred,c2_red)
%revise(rc2a,c2a_rred,c2a_red)
%revise(rc3,c3_rred,c3_red)
%revise(rc3a,c3a_rred,c3a_red)

%revise(rd1,d1_rred,d1_red)
%revise(rd1a,d1a_rred,d1a_red)
%revise(rd2,d2_rred,d2_red)
%revise(rd2a,d2a_rred,d2a_red)
%revise(rd3,d3_rred,d3_red)
%revise(rd3a,d3a_rred,d3a_red)

%revise(ga1,a1_ggreen,a1_green)
%revise(ga1a,a1a_ggreen,a1a_green)
%revise(ga2,a2_ggreen,a2_green)
%revise(ga2a,a2a_ggreen,a2a_green)
%revise(ga3,a3_ggreen,a3_green)
%revise(ga3a,a3a_ggreen,a3a_green)

%revise(gb1,b1_ggreen,b1_green)
%revise(gb1a,b1a_ggreen,b1a_green)
%revise(gb2,b2_ggreen,b2_green)
%revise(gb2a,b2a_ggreen,b2a_green)
%revise(gb3,b3_ggreen,b3_green)
%revise(gb3a,b3a_ggreen,b3a_green)
```

```
%revise(gc1,c1_ggreen,c1_green)
%revise(gc1a,c1a_ggreen,c1a_green)
%revise(gc2,c2_ggreen,c2_green)
%revise(gc2a,c2a_ggreen,c2a_green)
%revise(gc3,c3_ggreen,c3_green)
%revise(gc3a,c3a_ggreen,c3a_green)

%revise(gd1,d1_ggreen,d1_green)
%revise(gd1a,d1a_ggreen,d1a_green)
%revise(gd2,d2_ggreen,d2_green)
%revise(gd2a,d2a_ggreen,d2a_green)
%revise(gd3,d3_ggreen,d3_green)
%revise(gd3a,d3a_ggreen,d3a_green)

data all;

merge ra1 ga1 ra1a ga1a ra2 ga2 ra2a ga2a ra3 ga3 ra3a ga3a
          rb1 gb1 rb1a gb1a rb2 gb2 rb2a gb2a rb3 gb3 rb3a gb3a
          rc1 gc1  rc1a gc1a rc2 gc2 rc2a gc2a rc3 gc3 rc3a gc3a
          rd1 gd1 rd1a gd1a rd2 gd2 rd2a gd2a rd3 gd3 rd3a gd3a ;
by id;run;

data all1; set all;
hmha1=log2(c1_green)+log2(c1a_red)-log2(c1_red)-log2(c1a_green);
hmha2=log2(c2_green)+log2(c2a_red)-log2(c2_red)-log2(c2a_green);
hmha3=log2(c3_green)+log2(c3a_red)-log2(c3_red)-log2(c3a_green);
nmna1=log2(d1_green)+log2(d1a_red)-log2(d1_red)-log2(d1a_green);
nmna2=log2(d2_green)+log2(d2a_red)-log2(d2_red)-log2(d2a_green);
nmna3=log2(d3_green)+log2(d3a_red)-log2(d3_red)-log2(d3a_green);
hmnm1=log2(b1_green)+log2(b1a_red)-log2(b1_red)-log2(b1a_green);
hmnm2=log2(b2_green)+log2(b2a_red)-log2(b2_red)-log2(b2a_green);
hmnm3=log2(b3_green)+log2(b3a_red)-log2(b3_red)-log2(b3a_green);
hana1=log2(a1_green)+log2(a1a_red)-log2(a1_red)-log2(a1a_green);
hana2=log2(a2_green)+log2(a2a_red)-log2(a2_red)-log2(a2a_green);
hana3=log2(a3_green)+log2(a3a_red)-log2(a3_red)-log2(a3a_green);
keep id hmha1 hmha2 hmha3 nmna1 nmna2 nmna3 hmnm1 hmnm2 hmnm3 hana1
hana2 hana3; run;

data array; set all1; array x(i) hmha1 hmha2 hmha3 nmna1 nmna2 nmna3
hmnm1 hmnm2 hmnm3 hana1 hana2 hana3; do i=1 to 12; signal=x; if i in
(1,2,3) then treat='hmha'; if i in (4,5,6) then treat='nmna'; if i
in (7,8,9) then treat='hmnm'; if i in (10,11,12) then treat='hana';
keep  id treat signal; output; end; run;
```

```
ods listing close; ods noresults; ods output
            Contrasts=contrast
            lsmeans=lsmeans
            PredictedValues=resi;

proc glm data=array; /*where id='CT020023B1E01';*/ by id;

class treat; model signal=treat/p; contrast 'inter' treat -0.5 0.5
0.5 -0.5; contrast 'hanm' treat 0.5 -0.5 0.5 -0.5; lsmeans treat;
run; quit; ods listing; ods results;

/*get pvalue for inter and hanm*/ data inter(keep=id probf)
hanm(keep=id  probf); set contrast; if source='inter' then output
inter; if source='hanm' then output hanm; run;

/*get means for 4 treatment group*/ data hmha nmna hmnm hana; set
lsmeans; if treat='hmha' then output hmha; if treat='nmna' then
output nmna; if treat='hmnm' then output hmnm; if treat='hana' then
output hana; run;

%macro m(ahan,ahanmean);
data &ahanmean; set &ahan(rename=(signalLSMean=&ahan._mean)); keep
id &ahan._mean; proc sort data=&ahan; by id; run;
%mend;
%m(hmha,hmha_mean)
%m(nmna,nmna_mean)
%m(hmnm,hmnm_mean)
%m(hana,hana_mean)
data mean; merge hmha_mean nmna_mean hmnm_mean hana_mean; by id;
run;

data array1; set array; if treat='hmha' then c1=1; if treat='nmna'
then c1=-1; if treat='hmnm' then c1=1; if treat='hana' then c1=-1;
time=signal*c1; run;

ods listing close; ods noresults; ods output
            Contrasts=contrast1;

proc glm data=array1; by id; /*where id='CT020023B1E01';*/ class
treat; model time=treat; contrast 'cult' treat 0 0.5 0 -0.5;
contrast 'oxy' treat -0.5 0 0.5 0; contrast 'hmna' treat -0.5 0.5
0.5 -0.5; run; quit; ods listing; ods results; data cult(keep=id
probf) oxy(keep=id  probf) hmna(keep=id  probf); set contrast1; if
source='cult' then output cult; if source='oxy' then output oxy; if
source='hmna' then output hmna; run;
```

```
%macro cont(a);
data &a; set &a(rename=(probf=&a._pvalue)); keep id &a._pvalue;

proc sort data=&a; by id;run;
%mend;
%cont(inter)
%cont(hanm)
%cont(cult)
%cont(oxy)
%cont(hmna)

data pvalue; merge inter hanm cult oxy hmna /*hmha nmna hmnm hana*/;
by id;run; proc sort data=pvalue; by id; run;

data result; merge pvalue mean; by id; run; data hasinter nointer;
set result; if inter_pvalue<0.01 then output hasinter;/*87*/ if
inter_pvalue>=0.01 then output nointer;/*9326*/ run; data cul ox;
set nointer; if cult_pvalue<0.01 then output cul;/*1211*/ if
oxy_pvalue<0.01 then output ox;run;/*620*/

%macro his(result,inter_pvalue,interaction);
proc capability data=&result; histogram &inter_pvalue/
vscale=percent endpoints=0 to 1 by 0.01 cfill=blue cframe=ligr;
title "&interaction"; run;
%mend;
%his(result,inter_pvalue,interaction_contrast)
%his(result,cult_pvalue,cult_contrast)
%his(result,oxy_pvalue,oxy_contrast)
%his(result,hanm_pvalue,hanm_contrast)
%his(result,hmna_pvalue,hmna_contrast)
```

Bibliography

[1] Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In *second international symposium on information theory*, B. N. Petrov and F. Csake (eds), 267-281. Budapest: Akademiai Kaido.

[2] Alizadeh, A. A., Eisen, M. B., Davis, R. E., Ma, C., Lossos, I. S., Rosenwald, A., Boldrick, J. C., Sabet, H., Tran, T., Yu, X., Powell, J. I., Yang, L., Marti, G. E., Moore, T., Hudson, J. J., Lu, L., Lewis, D. B., Tibshirani, R., Sherlock, G., Chan, W. C., Greiner, T. C., Weisenburger, D. D., Armitage, J. O., Warnke, R., Levy, R., Wilson, W., Grever, M. R., Byrd, J. C., Botstein, D., Brown, P. O., and Staudt, L. M. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature* **403,** 503-511.

[3] Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl Acad. Sci.* **96,** 6745-6750.

[4] Aubert, J., Bar-Hen, A., Daudin J. J., and Robin, S. (2004). Determination of the differentiall expressed genes in microaray experiments using local FDR. *BMC Bioinformatics* **5,** 125 - 133.

[5] Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. B* **57,** 289-300.

[6] Benjamini, Y. and Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *Ann. Stat* **29,** 1165C1188.

[7] Brown, P. O. and Botstein, D. (1999). Exploring the new world of the genome with DNA microarrays. *Nat Genet* **21,** (Suppl 1) 33-37.

[8] Buckley, J. and James, I. (1979). Linear regression with censored data. *Biometrika* **66,** 429-436.

[9] Chen, C. *An introduction to quantile regression and the quantreg procedure.* http://www2.sas.com/proceedings/sugi30/213-30.pdf.

[10] Choi, J. K., Yu, U., Kim, S., and Yoo, O. J. (2003). Combining multiple microarray studeis and modeling interstudy variation. *Bioinformactis* **19,** 84-90.

[11] Chu, W., Ghahramani, Z., Falciani, F. and Wild, D. L. (2005). Biomarker discovery in microarray gene expression data with Gaussian processes. *Bioinformatics* **21,** 3385-3393.

[12] Cleveland, W. S. (1979). Robust Locally Weighted Regression and Smoothing Scatter plots, *Journal of the American Statistical Association* **74,** 829-836.

[13] Cox, D.R. (1972). Regression models and life-tables. *J. R. Statist. Soc. B* **34,** 187-220.

[14] Datta, S., Le-Rademacher, J., and Datta, S. (2007). Predicting patitent survival from microarray data by accelerated failure time modeling using partial least squares and LASSO. *Biometrics* **63,** 259-271.

[15] Daubechies, I., Defrise, M., and De Mol, C. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics* **57,** 1413-1457.

[16] Donoho, D. L. and Johnstone, I. M. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika* **81,** 425-455.

[17] Dudoit, S., Yang, Y. H., Callow, M. J., and Speed, T. P. (2002). Statistical Methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica Sinica* **12,** 111-139.

[18] Efron, B. (1967). The two sample problem with censored data. *Proceedings of the fifth Berkeley Symposium* **4,** 831-853. University of California Press, Berkeley, CA.

[19] Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression (with discussions). *Annals of Statistics* **32,** 407-499.

[20] Engler, D. A. and Li, Y. (2007). *Survival analysis with large dimensional covariates: an application in microarray studies.* http://www.bepress.com/harvardbiostat/paper68/.

[21] Friedman, J., Hastie, T., Hoe ing, H., and Tibshirani, R. (2007). Pathwise coordinate optimization. *Annals of Applied Statistics* **2,** 302-332.

[22] Friedman, J., Hastie, T., and Tibshirani, R. (2008). *Regularization paths for generalized linear models via coordinate descent.* http://www-stat.stanford.edu/ hastie/Papers/glmnet.pdf.

[23] Friedman, J. and Popescu, B. (2004). *Gradient Directed Regularization.* Technical Report, Stanford University, Stanford, California.

[24] Frisbie, D. D., Oxford, J. T., Southwood, L., Trotter, G. W., Rodkey, W. G., Steadman, J. R., Goodnight, J. L., and McIlwraith, C. W. (2003). Early events in cartilage repair after subchondral bone microfracture. *Clin. Orthop. Relat. Res* **407,** 215-227.

[25] Frisbie, D. D., Trotter, G. W., Powers, B. E., Rodkey, W. G., Steadman, J. R., Howard, R. D., Park, R. D., and McIlwraith, C. W. (1999). Arthroscopic subchondral bone plate microfracture technique augments healing of large chondral defects in the radial carpal bone and medial femoral condyle of horses. *Vet Surg* **28,** 242-55.

[26] Fu, W. (1998). Penalized regressions: the bridge vs. the lasso. *Journal of Computational and Graphical Statistics* **7,** 397-416.

[27] Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* **286,** 531C537.

[28] Gui, J. and Li, H. (2005b). Penalized Cox regression analysis in the high-dimensional and low-sample size settings, with applications to microarray gene expression data. *Bioinformatics* **21,** 3001-3008.

[29] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. New York: Springer-Verlag.

[30] He, X. and Wei, Y. *Tutorial on quantile regression*. http://www.stat.uiuc.edu/ x-he/ENAR-Tutorial.pdf.

[31] Henrotin, Y., Kurz, B., and Aigner, T. (2005). Oxygen and reactive oxygen species in cartilage degradation: friends or foes? *Osteoarthritis Cartilage* **13,** 643-654.

[32] Hoerl, A. and Kennard, R. (1970). Ridge regression: Biased estimation for non-orthogonal problem. *Technometrics* **12,** 55C67.

[33] Huang, J. and Harrington, D. (2002). Penalized partial likelihood regression for right-censored data. *Biometrics* **58,** 781-791.

[34] Huang, J., Ma, S., and Xie, H. (2006). Regularized estimation in the accelerated failure time model with high dimensional covariates. *Biometrics* **62,** 813-820.

[35] Huang, J., Wang, D., and Zhang, C. (2005). A Two-Way Semilinear Model for Normalization and Analysis of cDNA Microarray Data. *Journal of the American Statistical Association* **100,** 814-829.

[36] Jiang, H., Deng, Y., Chen, H. S., Tao, L., Sha, Q., Chen, J., Tsai, C. J., and Zhang, S. (2004) Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes. *BMC Bioinformactis* **5,** 81.

[37] Jolliffe, I. (1986). *Principal Component Analysis*. New York: Springer- Verlag.

[38] Kalbfleisch, J. D. and Prentice, R. L. (1980). *The statistical analysis of failure time data*. New York: Wiely.

[39] Kerr, M. K., Martin, M., and Churchill, G. A. (2000). Analysis of Variance for Gene Expression Microarray Data. *Journal of Computational Biology* **7,** 819-837.

[40] Keselman, H. J., Cribbie, R., and Holland, B. (2002). Controlling the rate of type I error over a large set of statistical tests. *Br. J. Math. Stat. Psychol.* **55,** 27-39.

[41] Koenker, R. (2005). *Quantile Regression*. Cambridge University Press, New York.

[42] Koenker, R. and Bassett, G. (1978). Regression quantiles. *Econometrica* **46,** 33-50.

[43] Li, G. and Wang, Q. (2003). Empirical likelihood-based inference for a censored linear regression model. *Statistica Sinica* **13,** 51-68.

[44] Li, H. (2008). Censored data regression in high-dimension and low sample size settings for genomic applications. In *Statistical Advances in Biomedical Sciences: State of Art and Future Directions*, A. Biswas, S. Datta, J. Fine, and M. Segal (eds), 1st edition. Hoboken, New Jersey: Wiley.

[45] Li, H. and Luan, Y. (2003). Kernel Cox regression models for linking gene expression profiles to censored survival data. *Pacific Symposium of Biocomputing* **8,** 65-76.

[46] Lockhart, D. J. and Winzler, E. A. (2000). Genomics, gene expression and DNA array. *Nature* **405,** 827-836.

[47] Loguinov, A. V., Mian, I. S., and Vulpe, C. D. (2004). Exploratory differential gene expression analysis in microarray experiments with no or limited replication. *Genome Biol* **5,** 18.

[48] Martens, H. and Naes, T. (1989). *Multivariate Calibration*. Chichester: John Wiley & Sons Ltd.

[49] Nguyen, D. V. (2002). DNA microarray experiments: biological and technological aspects. *Biometrics* **58,** 701C717.

[50] Park, S. (1981). Collinearity and optimal restrictions on regression parameters for estimating responses. *Technometrics* **23,** 289C295.

[51] Pepe, M. S. (2003). *The Statistical Evaluation of Medical Tests for Classification and Prediction*. UK Oxford University Press.

[52] Pepe, M. S., Cai, T., and Longton, G. (2005). Combining predictors for classification using the area under the ROC curve. *Biometrics* **62,** 221-229.

[53] Pfander, D., Cramer, T., Schipani, E., and Johnson, R. S. (2003). HIF-1 alpha controls extracellular matrix synthesis by epiphyseal chondrocytes. *Journal of Cell Science* **116,** 1819-1826.

[54] Reiner, A., Yekutieli, D., and Benjamini, Y. (2003). Identifiying differentially expressed genes using false discovery rate controlling procedures. *Bioinformatics* **19,** 368-375.

[55] Rhodes, D. R., Barrette, T. R., Rubin, M. A., Ghosh, D., and Chinnaiyan, A. M. (2002). Meta-Analysis of microarrays: interstudy validation of gene expression profiles reveals pathway dysregulation in prostate cancer. *Cancer Res.* **62,** 4427-4433.

[56] Robins, J. and Finkelstein, D. (2000). Correcting for noncompliance and dependent censoring in an AIDS clinical trail with inverse probability of censoring weighted (IPCW) log-rank tests. *Biometrics* **56,** 779-788.

[57] Robins, J. M. and Rotnitzky, A. (1992). Recovery of information and adjustment for dependent censoring using surrogate markers. In *AIDS Epidemiology-Methodological Issues*, N. Jewell, K. Dietz, and V. Farewell (eds), 297-331. Boston: Birkhauser.

[58] Rosa, G. J. M., Steibel, J. P., and Tempelman, R. J. (2005). Reassessing design and analysis of two-color microarray experiments using mixed effects models. *Comparative and Functional Genomics* **6,** 123-131.

[59] Rotnitzky, A. and Robins, J. M. (2005). Inverse probability weighted estimation in survival analysis in: *The Encyclopedia of Biostatistics*, 2nd edition, Ed. P. Armitage and T. Coulton, New York: Wiely.

[60] Satten, G. A. and Datta, S. (2001). The Kaplan-Meier estimator as an inverse-probability-of-censoring weighted average. *American Statistician* **55,** 207-210.

[61] Schena, M., Shalon, D., Heller, R., Chai, A., Brown, P. O., and Davis, R. W. (1996). Parallel human genome analysis: microarray-based expression monitoring of 1000 genes. *Proc Natl Acad Sci* **93,** 10614-10619.

[62] Shen, R., Ghosh, D., and Chinnaiyan, A. M. (2004). Prognostic meta-signature of breast cancer developed by two-stage mixture modeling of microarray data. *BMC Genomics* **5,** 94.

[63] Shevade, S. K. and Keerthi, S. S. (2003). A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics* **19,** 2246-2253.

[64] Steibel, J. P., Tempelman, R. J., and Rosa, G. J. M. (2009). Power and sample size determinations for two color microarray experiments based on different levels of replication (submitted for publication).

[65] Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions (with discussion). *Journal of the royal statistical society Series B* **36,** 111-147.

[66] Storey, J. D. (2002). A direct approach to false discovery rate. *J. R. Stat. Soc. B.* **64,** 479C498.

[67] Storey, J. D. and Tibshirani, R. (2003). Statistical significance for genomwide studies. *Proc. Natl Acad. Sci.* **100,** 9440-9445.

[68] Stute, W. (1993). Consistent estimation under random censorship when covariables are available. *Journal of Multivariate Analysis* **45,** 89-103.

[69] Stute, W. (1996). Distributionla convergence under random censorship when covariables are present. *Scandinavia Journal of Statistics* **23,** 461-471.

[70] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B* **58,** 267-288.

[71] Tibshirani, R. (1997). The Lasso method for variable selection in the Cox model. *Statist. Med.* **16,** 385-395.

[72] Tseng, G. C., Oh, M. K., Rohlin, L., Liao, J. C., and Wong, W. H. (2001). Issues in cDNA Microarray Analysis: Quality Filtering, Channel Normalization, Models of Variation and Assessment of Gene Effects. *Nucleic Acids Research* **29,** 2549-2557.

[73] Vaishnav, R. A., Getchell, M. L., Huang, L., Hersh, M. A., Stromberg, A. J., and Getchell, T. V. (2008). Cellular and molecular characterization of oxidative stress in olfactory epithelium of Harlequin mutant mouse. *J Neurosci Res.* **86,** 165-182.

[74] Van der Kooij, A. (2007). *Prediction accuracy and stability of regrsssion with optimal sclaing transformations.* Technical report, Dept. of Data Theory, Leiden University.

[75] Van der Laan, M. and Robins, J. M. (2003). *Unified methods for censored longitudinal data and causality.* New York: Springer.

[76] Vapnik, V. (1995). *The Nature of Statistical Learning Theory.* New York: Springer.

[77] Verwij, P. and Van Houweingen, H. (1993). Cross validation in survival analysis. *Statis. Med.* **12,** 2305-2314.

[78] Wahba, G. (1990). Spline models for observational data. *CBMS-NSF regional conference series in applied mathematics* **59.** Philadelphia: SIAM.

[79] Wang, H. and He, X. (2007). Detecting differential expressions in GeneChip microarray studies: a quantile approach. *Journal of American Statistical Association* **102,** 104-112.

[80] Wang, H. and He, X. (2008). An enhanced quantile approach for assessing differential gene expressions. *Biometrics* **64,** 449-457.

[81] Wang, S., Nan, B., Zhu, J., and Beer, D. G. (2008). Doubly penalized Buckley-James method for survival data with high-dimensional covariates. *Biometrics* **6,** 132-140.

[82] Wei, L. (1992). The accelerated failure time model: A usefu l alternative to the Cox regression model in survival analysis. *Statistics in Medicine* **11,** 1871-1879.

[83] Wolfinger, R. D., Gibson, G., Wolfinger, E. D., Bennett, L., Hamadeh, H., Bushel, P., Afshari, C., and Paules, R. S. (2001). Assessing Gene Significance From cDNA Microarray Expression Data via Mixed Models. *Journal of Computational Biology* **8,** 625-637.

[84] Wu, T. and Lange, K. (2008a). Coordinate descent algorithms for lasso penalized regression. *Annals of Applied Statistics* **2,** 224-244.

[85] Yang, Y. H., Dudoit, S., Luu, P., Lin, D. M., Peng, V., Ngai, J., and Speed, T. P. (2002). Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acid Res* **30,** 15.

[86] Yang, Y. H., Dudoit, S., Luu, P., and Speed, T. P. (2001). Normalization for cDNA microarray data, in Microarrays: Optical Technologies and Informatics. *Proceedings of the SPIE, eds, M. L. Bittner, Y. Chen, A. N. Dorsel, and E. R. Dougherty, San Jose, CA: Society for Optical Engineering* **4266,** 141-152.

[87] Zien, A., Aigner, T., Zimmer, R., and Lengauer, T. (2001). Centralization: a new method for the normalization of gene expression data. *Bioinformatics* **17,** 323-331.

[88] Zhou, M. (1992b). M-estimation in censored linear models. *Biometrika* **79,** 837-841.

[89] Zou, H. and Hastie, T. (2005). Regularization and variable selectio via the elastic net. *Journal of the Royal Statistical Society Series B* **67,** 301-320.

Vita

Name: Liping Huang

Date of Birth: February 9, 1980

Place of Birth: Fujian, China

Education

- 2005-2008, Master of Science in Statistics, University of Kentucky.
- 1999-2004, Bachelor in Medicine, Peking University, China.

Employment

- 2005-present, Research Assistant, University of Kentucky.
- 2004-2005, Research Assistant, Graduate Center for Toxicology, University of Kentucky.

Publications

- **Huang, L.**, Zhu, W., Saunders, C. P., MacLeod, J. N., Zhou, M., Stromberg, A. J., and Bathke, A. C. (2008). A novel application of quantile regression for identification of cartilage biomarkers exemplified by equine cartilage microarray data. *BMC Bioinformatics*. **9,** 300.

- **Huang, L.**, Miura N., Mienaltowski M. J., MacLeod J. N., Stromberg A. J., Bathke A. C., Wood C. L. (2008). Rediscovering the power of well-planned comparisons: normalization and analysis of cDNA microarray using linear combinations. Jointing Statistical Meeting proceedings, Alexandria, VA: American Statistical Association.

- Mienaltowski, M. J., **Huang, L.**, Frisbie, D. D., McIlwraith, W., Stromberg, A. J., Bathke, A. C., and MacLeod, J. N. (2009). Transcriptional profiling differences for articular cartilage and repair tissue in equine joint surface lesions. *BMC Medical Genomics* In press.

• Mienaltowski, M. J., **Huang, L.**, Stromberg, A. J., and MacLeod, J. N. (2009). Differential gene expression associated with postnatal equine articular cartilage maturation. *BMC Musculoskeletal Disorders* **9,** 149.

• Athippozhy A. T., **Huang L.**, Zhao T., Wooton-Kee C. R., Jungsuwadee P., Stromberg A. J., Vore M. (2009). Utilizing a mixed model approach to compare the lactating rat transcriptome against age-matched control virgins. (meeting abstract) *BMC Bioinformatics*. **10,** 16.

• Vaishnav, R. A., Getchell, M. L., **Huang, L.**, Hersh, M. A., Stromberg, A. J., and Getchell, T. V. (2008). Cellular and molecular characterization of oxidative stress in olfactory epithelium of Harlequin mutant mouse. *J. Neurosci. Res.* **86,** 165-182.

• Zhang, J., **Huang, L.**, Kang, C. (2005). Survey on knowledge, attitudes and practice of childhood immunization in Qinghai province. *Chinese Primary Health Care* **19,** 52-53.