STATISTICAL APPLICATIONS OF LINEAR PROGRAMMING FOR FEATURE SELECTION VIA REGULARIZATION METHODS

DISSERTATION

Presented in Partial Fulfillment of the Requirements for

the Degree Doctor of Philosophy in the

Graduate School of The Ohio State University

By

Yonggang Yao, Ph.D.

* * * * *

The Ohio State University

2008

Dissertation Committee:

Professor Yoonkyung Lee, Adviser

Professor Prem K. Goel

Professor Tao Shi

Approved by

_____

Adviser

Graduate Program in Statistics

ABSTRACT

We consider statistical procedures for feature selection defined by a family of regularization problems with convex piecewise linear loss functions and penalties of l_1 or l_∞ nature. For example, quantile regression and support vector machines with l_1 norm penalty fall into the category. Computationally, the regularization problems are linear programming (LP) problems indexed by a single parameter, which are known as 'parametric cost LP' or 'parametric right-hand-side LP' in the optimization theory. Their solution paths can be generated with certain simplex algorithms. This work exploits the connection between the family of regularization methods and the parametric LP theory and lays out a general simplex algorithm and its variant for generating regularized solution paths for the feature selection problems. The significance of such algorithms is that they allow a complete exploration of the model space along the paths and provide a broad view of persistent features in the data. The implications of the general path-finding algorithms are outlined for various statistical procedures, and they are illustrated with numerical examples.

This is dedicated to my parents, my wife Youlan Rao, and my daughter Rebecca R. Yao.

ACKNOWLEDGMENTS

I would like to express my appreciation to my adviser, Dr. Yoonkyung Lee, for her inspiring advice, her encouragement, her patience, and her consistently positive attitude. I thank Dr. Prem K. Goel and Dr. Tao Shi for their academic suggestions and willingness to serve as my committee members. I also thank Dr. Douglas A. Wolfe, Dr. Joseph S. Verducci, Dr. Elizabeth A. Stasny, and Dr. Noel A. Cressie for their long-term support and help.

VITA

1996	B.S. Electrical Engineering, Wuhan University
1998	M.S. Electrical Engineering, Wuhan University
2002	Ph.D. Space Physics, Wuhan University
2007	M.S. Statistics, The Ohio State University.
2003-2008	Graduate Research/Teaching Associate, The Ohio State University.

FIELDS OF STUDY

Major Field: Statistics

Studies in:

Topic 1 Statistical Computing
Topic 2 Statistical Learning

TABLE OF CONTENTS

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

Regularization methods cover a wide range of statistical procedures for estimation and prediction, and they have been used in many modern applications. To name a few, examples are ridge regression (Hoerl and Kennard; 1970), the LASSO regression (Tibshirani; 1996), smoothing splines (Wahba; 1990), and support vector machines (SVM) (Vapnik; 1998).

Given a training data set, $\{(y_i, \mathbf{x}_i) : \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}; i = 1, \cdots, n\}$ and a feature space $\mathcal{F}$, many statistical problems can be phrased as the problem of finding a functional relationship between the covariates, $\mathbf{x} \in \mathcal{X}$, and the response $y \in \mathcal{Y}$ within $\mathcal{F}$ based on the observed pairs. For example, a regularization method for prediction looks for a model $f(\mathbf{x}; \boldsymbol{\beta}) \in \mathcal{F}$ with unknown parameters $\boldsymbol{\beta}$ that minimizes a prediction error over the training data while controlling its model complexity. To be precise, let $\mathcal{L}(y, f(\mathbf{x}; \boldsymbol{\beta}))$ be a convex loss function for the prediction error and $J(f(\mathbf{x}; \boldsymbol{\beta}))$ be a convex penalty functional that measures the model complexity. The training error with respect to $\mathcal{L}$ is defined by $L(\boldsymbol{Y}, f(\mathbf{X}; \boldsymbol{\beta})) := \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y_i, f(\mathbf{x}_i; \boldsymbol{\beta}))$, where $\boldsymbol{Y} := (y_1, \cdots, y_n)'$ and $\mathbf{X} := (\mathbf{x}_1', \cdots, \mathbf{x}_n')'$. Formally, the solution to a regularization problem is defined to be f with the model parameters $\hat{\boldsymbol{\beta}}$ that minimize:

$$L(\boldsymbol{Y}, f) + \lambda \cdot J(f), \tag{1.1}$$

where $\lambda \geq 0$ is a pre-specified regularization parameter. The λ determines the trade-off between the prediction error and the model complexity, and thus the quality of the solution highly depends on the choice of λ. Identification of a proper value of the regularization parameter for model selection or a proper range for model averaging is a critical statistical problem. Note that $\hat{\boldsymbol{\beta}}(\lambda)$ is a function of λ. As in (1.1), each regularization method defines a continuum of optimization problems indexed by a tuning parameter. In most cases, the solution as a function of the tuning parameter is expected to change continuously with λ. This allows for the possibility of complete exploration of the model space as λ varies, and computational savings if (1.1) is to be optimized for multiple values of λ.

Alternatively, the regularization problem in (1.1) can be formulated to bound the model complexity or the penalty. In this complexity-bounded formulation, the optimal parameters are sought by minimizing:

$$L(\boldsymbol{Y}, f) \text{ s.t. } J(f) \leq s, \tag{1.2}$$

where s is an upper bound of the complexity.

For a certain combination of the loss $\mathcal{L}$ and the complexity measure J, it is feasible to generate the entire solution path of the regularization problem. Here, the path refers to the entire set of solutions to the regularization problem, for instance, $\hat{\boldsymbol{\beta}}(\lambda)$ in (1.1) as a function of λ (or $\hat{\boldsymbol{\beta}}(s)$ in (1.2) as a function of s). Some pairs of the loss and the complexity are known to allow such fast and efficient path finding algorithms; for instance, LARS (Efron et al.; 2004), the standard binary SVM (Hastie et al.; 2004), the multi-category SVM (Lee and Cui; 2006), and the l_1-norm quantile regression (Li and Zhu; 2008). Rosset and Zhu (2007) study general conditions for the combination of $\mathcal{L}$ and J such that solutions indexed

by a regularization parameter are piecewise linear and thus can be sequentially characterized. They provide generic path-finding algorithms under some appropriate assumptions on $\mathcal{L}$ and J.

In this thesis, we focus on an array of regularization methods aimed for feature selection with penalties of l_1 nature and piecewise linear loss functions. Many existing procedures are subsumed under this category. Examples include the l_1-norm SVM (Zhu et al.; 2004) and its extension to the multi-class case (Wang and Shen; 2006), l_1-norm quantile regression (Li and Zhu; 2008), Sup-norm multi-category SVM (Zhang et al.; 2006), the functional component selection step (called "θ-step") for structured multi-category SVM (Lee et al.; 2006), and the Dantzig selector (Candes and Tao; 2007). We also note that the ϵ-insensitive loss in the SVM regression (Vapnik; 1998) and relative absolute loss (also called relative absolute error in Narula and Wellington (1977)) fit into the category of a piecewise linear loss, and the sup norm gives rise to a linear penalty just as the l_1 norm in general.

There is a great commonality among these methods. That is, computationally the associated optimization problems are all linear programming (LP) problems indexed by a single regularization parameter. This family of LP problems are known as the *parametric cost* linear programming and have long been studied in the optimization theory. Furthermore, there already exist efficient algorithms for the solution paths. Despite the commonality, so far, only case-by-case treatments of some of the problems are available as in Zhu et al. (2004); Li and Zhu (2008) and Wang and Shen (2006). Although Wang and Shen (2006) notice that those solution path algorithms have fundamental connections with the *parametric right-hand-side* LP (see (2.6) for the definition), such connections have not been adequately

explored for other problems with generality. As noted, Rosset and Zhu (2007) have a comprehensive take on the computational properties of regularized solutions, however they did not tap into the LP theory for general treatments of the problems of our focus.

The goal of this thesis is to make the link between the parametric LP and a family of computational problems arising in statistics for feature selection via regularization more explicit and put those feature selection problems in perspectives. To this end, we pull together results from the linear programming literature and summarize them in an accessible and self-contained fashion.

Chapter 2 begins with an overview of the standard LP and parametric LP problems, gives a brief account of the optimality conditions for their solutions, and then introduces the simplex algorithm and the tableau-simplex algorithm for finding the entire solution paths of the parametric LP problems. Chapter 3 describes various examples of LP for feature selection, paraphrasing their computational elements in the LP terms. More computational issues including tableau simplification and a detailed comparison of the simplex algorithm with the existing algorithm for the l_1-norm SVM (Zhu et al.; 2004) are given in Chapter 4 highlighting the generality of the proposed approach. Numerical examples and data application of the algorithm follow in Chapter 5 for illustration. Technical proofs except for the key theorems and a description of the R package developed for generating LP solution paths, `lpRegPath`, are collected into Appendices.

CHAPTER 2

LINEAR PROGRAMMING

Linear programming (LP) is one of the cornerstones of the optimization theory. Since the publication of the simplex algorithm by Dantzig in 1947, there has been a wide range of applications of LP in operation research, microeconomics, business management, and many other engineering fields. For statistical applications, Wagner (1959) pointed out that LP can be used to solve the least absolute deviation problem (also known as median regression) and the least maximum deviation problem. Fisher (1961) further described the mathematical LP formula for least absolute deviation method and suggested several ways to add extra systematic constraints to the method.

We give an overview of LP here and describe the optimality conditions of the LP solution pertinent to our discussion of path-finding algorithms. The conditions are well known in the optimization literature, but we include them and their proofs for completeness. Our treatment of LP closely follows that in standard references such as Bertsimas and Tsitsiklis (1997) and Murty (1983). The readers are referred to them and references therein for more complete discussions.

2.1 Standard Linear Programs

A standard form of LP is

$$\begin{cases} \min\limits_{\boldsymbol{z} \in \mathcal{R}^N} & \mathbf{c}'\boldsymbol{z} \\ \text{s.t.} & \mathbf{A}\boldsymbol{z} = \mathbf{b} \\ & \boldsymbol{z} \geq \mathbf{0}, \end{cases} \tag{2.1}$$

where $\boldsymbol{z}$ is an N-vector of variables, $\mathbf{c}$ is a fixed N-vector, $\mathbf{b}$ is a fixed M-vector, and $\mathbf{A}$ is an $M \times N$ fixed matrix. Without loss of generality, it is assumed that $M \leq N$ and $\mathbf{A}$ is of full row rank. Standard techniques for solving LP include simplex method, dual simplex method, tableau method, and interior point methods.

Geometrically speaking, the standard LP problem in (2.1) searches the minimum of a linear function over a polyhedron whose edges are defined by hyperplanes. Therefore, if there exists a fixed solution for the LP problem, at least one of the intersection points (formally called basic solutions) of the hyperplanes should attain the minimum. For formal discussion of the optimality, a brief review of some terminologies in LP is provided. Let $\mathcal{N}$ denote the index set $\{1, \cdots, N\}$ of the unknowns, $\boldsymbol{z}$, in the LP problem in (2.1).

Definition 1 *A set $\mathcal{B}^* := \{B_1^*, \cdots, B_M^*\} \subset \mathcal{N}$ is called a basic index set, if $\mathbf{A}_{\mathcal{B}^*} := [\mathbf{A}_{B_1^*}, \cdots, \mathbf{A}_{B_M^*}]$ is invertible, where $\mathbf{A}_{B_i^*}$ is the B_i^*th column vector of $\mathbf{A}$ for $i \in \mathcal{B}^*$. $\mathbf{A}_{\mathcal{B}^*}$ is called the basic matrix associated with $\mathcal{B}^*$. Correspondingly, a vector $\boldsymbol{z}^* \in \mathcal{R}^N$ is called the basic solution associated with $\mathcal{B}^*$, if $\boldsymbol{z}^*$ satisfies*

$$\begin{cases} \boldsymbol{z}^*_{\mathcal{B}^*} := (z^*_{B_1^*}, \cdots, z^*_{B_M^*})' = \mathbf{A}^{-1}_{\mathcal{B}^*}\mathbf{b} \\ z^*_j = 0 \, \textit{for} \, j \in \mathcal{N} \setminus \mathcal{B}^*. \end{cases}$$

Definition 2 *Let $\boldsymbol{z}^*$ be the basic solution associated with $\mathcal{B}^*$.*

- *$\boldsymbol{z}^*$ is called a basic feasible solution if $\boldsymbol{z}^*_{\mathcal{B}^*} \geq \mathbf{0}$;*
- *$\boldsymbol{z}^*$ is called a non-degenerate basic feasible solution if $\boldsymbol{z}^*_{\mathcal{B}^*} > \mathbf{0}$;*

- $\boldsymbol{z}^*$ *is called a degenerate basic feasible solution if* $\boldsymbol{z}^*_{\mathcal{B}^*} \geq \mathbf{0}$ *and* $z^*_{B_i^*} = 0$ *for some* $i \in \mathcal{M} := \{1, \cdots, M\}$*;*

- $\boldsymbol{z}^*$ *is called an optimal basic solution if* $\boldsymbol{z}^*$ *is a solution of the LP problem.*

Since each basic solution is associated with its basic index set, the optimal basic solution can be identified with the optimal basic index set as defined below.

Definition 3 *A basic index set* $\mathcal{B}^*$ *is called a feasible basic index set if* $\mathbf{A}^{-1}_{\mathcal{B}^*}\mathbf{b} \geq \mathbf{0}$*. A feasible basic index set* $\mathcal{B}^*$ *is also called an optimal basic index set if*

$$\left[\mathbf{c} - \mathbf{A}' \left(\mathbf{A}^{-1}_{\mathcal{B}^*}\right)' \mathbf{c}_{\mathcal{B}^*}\right] \geq \mathbf{0}.$$

The following theorem indicates that the standard LP problem can be solved by finding the optimal basic index set (see Bertsimas and Tsitsiklis (1997) Theorem 3.1 for a more complete version).

Theorem 4 *For the LP problem in (2.1), let* $\boldsymbol{z}^*$ *be the basic solution associated with* $\mathcal{B}^*$*, an optimal basic index set. Then* $\boldsymbol{z}^*$ *is an optimal basic solution.*

Proof We need to show $\mathbf{c}'\boldsymbol{z} \geq \mathbf{c}'\boldsymbol{z}^*$ or $\mathbf{c}'(\boldsymbol{z} - \boldsymbol{z}^*) \geq \mathbf{0}$ for any feasible vector $\boldsymbol{z} \in \mathcal{R}^N$ with $\mathbf{A}\boldsymbol{z} = \mathbf{b}$ and $\boldsymbol{z} \geq 0$. Denote $\boldsymbol{d} := (d_1, \cdots, d_N) := (\boldsymbol{z} - \boldsymbol{z}^*)$. From

$$\mathbf{A}\boldsymbol{d} = \mathbf{A}_{\mathcal{B}^*}\boldsymbol{d}_{\mathcal{B}^*} + \sum_{i \in \mathcal{N} \setminus \mathcal{B}^*} \mathbf{A}_i d_i = \mathbf{0},$$

we have

$$\boldsymbol{d}_{\mathcal{B}^*} = -\sum_{i \in \mathcal{N} \setminus \mathcal{B}^*} \mathbf{A}^{-1}_{\mathcal{B}^*}\mathbf{A}_i d_i.$$

Then,

$$\begin{aligned} \mathbf{c}'(\boldsymbol{z} - \boldsymbol{z}^*) &= \mathbf{c}'\boldsymbol{d} = \mathbf{c}'_{\mathcal{B}^*}\boldsymbol{d}_{\mathcal{B}^*} + \sum_{i \in \mathcal{N} \setminus \mathcal{B}^*} \mathrm{c}_i d_i \\ &= \sum_{i \in \mathcal{N} \setminus \mathcal{B}^*} (\mathrm{c}_i - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}^{-1}_{\mathcal{B}^*}\mathbf{A}_i) d_i. \end{aligned}$$

Recall that for $i \in \mathcal{N} \setminus \mathcal{B}^*$, $z_i^* = 0$, which implies $d_i := (z_i - z_i^*) \geq 0$. Together with $\left[\mathbf{c} - \mathbf{A}' \left(\mathbf{A}_{\mathcal{B}^*}^{-1}\right)' \mathbf{c}_{\mathcal{B}^*}\right] \geq \mathbf{0}$, it ensures $(\mathrm{c}_i - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}_i)d_i \geq 0$. Thus, we have $\mathbf{c}'\boldsymbol{d} \geq 0$.

2.2 Parametric Linear Programs

In practical applications, the cost coefficients $\mathbf{c}$ or the constraint constants $\mathbf{b}$ in (C.2) are often partially known or controllable so that they may be modeled linearly as $(\mathbf{c} + \lambda\mathbf{a})$ or $(\mathbf{b} + \omega\mathbf{b}^*)$ with some parameters λ and $\omega \in \mathcal{R}$. A family of regularization methods for feature selection to be discussed share this characteristic. Although every parameter value creates a new LP problem in the setting, it is feasible to generate solutions for all values of the parameter via sequential updates. The new LP problems indexed by the parameters are called the parametric-cost LP and parametric right-hand-side LP, respectively. For reference, see Bertsimas and Tsitsiklis (1997), p. 217-221, and Murty (1983), p. 278-293.

The standard form of a parametric-cost LP is defined as

$$\begin{cases} \min\limits_{\boldsymbol{z} \in \mathcal{R}^N} & (\mathbf{c} + \lambda\mathbf{a})'\boldsymbol{z} \\ \text{s.t.} & \mathbf{A}\boldsymbol{z} = \mathbf{b} \\ & \boldsymbol{z} \geq \mathbf{0}. \end{cases} \tag{2.2}$$

Since the basic index sets of the parametric-cost LP do not depend on the parameter λ, an optimal basic index set $\mathcal{B}^*$ for some fixed value of λ would remain optimal for a range of λ values, say, $[\underline{\lambda}, \overline{\lambda}]$, which is called the optimality interval of $\mathcal{B}^*$. The following corollary originally proposed by Saaty and Gass (1954) describes an approach for finding the optimality interval.

Corollary 5 *For a fixed $\lambda^* \geq 0$, let $\mathcal{B}^*$ be an optimal basic index set of the problem in (2.2) at $\lambda = \lambda^*$. Define*

$$\underline{\lambda} := \max_{\{j\,:\,\check{a}_j^* > 0;\, j \in \mathcal{N} \setminus \mathcal{B}^*\}} \left(-\frac{\check{c}_j^*}{\check{a}_j^*}\right) \tag{2.3}$$
$$\text{and} \quad \overline{\lambda} := \min_{\{j\,:\,\check{a}_j^* < 0;\, j \in \mathcal{N} \setminus \mathcal{B}^*\}} \left(-\frac{\check{c}_j^*}{\check{a}_j^*}\right),$$

where $\check{a}_j^* := a_j - \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}_j$ and $\check{c}_j^* := c_j - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}_j$ for $j \in \mathcal{N}$. Then, $\mathcal{B}^*$ is an optimal basic index set of (2.2) for $\lambda \in [\underline{\lambda}, \overline{\lambda}]$, which includes λ^*.

Proof From the optimality of $\mathcal{B}^*$ for $\lambda = \lambda^*$, we have $\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b} \geq \mathbf{0}$ and

$$\left[\mathbf{c} - \mathbf{A}'\left(\mathbf{A}_{\mathcal{B}^*}^{-1}\right)'\mathbf{c}_{\mathcal{B}^*}\right] + \lambda^*\left[\mathbf{a} - \mathbf{A}'\left(\mathbf{A}_{\mathcal{B}^*}^{-1}\right)'\mathbf{a}_{\mathcal{B}^*}\right] \geq \mathbf{0},$$

which implies that $\check{\mathrm{c}}_j^* + \lambda^*\check{\mathrm{a}}_j^* \geq \mathbf{0}$ for $j \in \mathcal{N}$. To find the optimality interval $[\underline{\lambda}, \overline{\lambda}]$ of $\mathcal{B}^*$, by Theorem 4, we need to investigate the following inequality for each $j \in \mathcal{N}$:

$$\check{\mathrm{c}}_j^* + \lambda\check{\mathrm{a}}_j^* \geq 0. \tag{2.4}$$

It is easy to see that $\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}_{B_i^*} = \mathbf{e}_i$ for $i \in \mathcal{M}$ since $\mathbf{A}_{B_i^*}$ is the ith column of $\mathbf{A}_{\mathcal{B}^*}$. Consequently, the jth entries of $(\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A})$ and $(\mathbf{a}' - \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A})$ are both 0 for $j \in \mathcal{B}^*$, and $\check{\mathrm{c}}_j^* + \lambda\check{\mathrm{a}}_j^* = 0$ for any λ. So, the inequality holds for any $\lambda \in \mathcal{R}$ and $j \in \mathcal{B}^*$. When $\check{a}_j^* > 0$ (or $\check{a}_j^* < 0$) for $j \in (\mathcal{N} \setminus \mathcal{B}^*)$, (2.4) holds if and only if $\lambda \geq -\check{\mathrm{c}}_j^*/\check{\mathrm{a}}_j^*$ (or $\lambda \leq -\check{\mathrm{c}}_j^*/\check{\mathrm{a}}_j^*$). Thus, the lower bound and the upper bound of the optimality interval of $\mathcal{B}^*$ are given by the $\underline{\lambda}$ and $\overline{\lambda}$ in (2.3).

Note that $\check{c}_j^*$ and $\check{a}_j^*$ define the relative cost coefficient of z_j. Since the number of basic index sets is finite for fixed $\mathbf{A}$, there exist only a finite number of optimal basic index sets of the problem in (2.2). Corollary 5 implies that a version of the solution path of the problem as a function of λ, $\boldsymbol{z}(\lambda)$, is a step function.

On the other hand, if the parametric cost LP in (2.2) is recast in the form of (1.2), then the stepwise constant property of the solution path changes. The alternative complexity-bounded formulation of (2.2) is given by

$$\begin{cases} \min\limits_{\boldsymbol{z} \in \mathcal{R}^N, \delta \in \mathcal{R}} & \mathbf{c}'\boldsymbol{z} \\ \text{s.t.} & \mathbf{A}\boldsymbol{z} = \mathbf{b} \\ & \mathbf{a}'\boldsymbol{z} + \delta = s \\ & \boldsymbol{z} \geq \mathbf{0}, \delta \geq 0. \end{cases} \tag{2.5}$$

It can be transformed into a standard parametric right-hand-side LP problem:

$$\begin{cases} \min\limits_{\mathbb{Z} \in \mathcal{R}^{N+1}} & \mathbb{C}'\mathbb{Z} \\ \text{s.t.} & \mathbb{A}\mathbb{Z} = \mathbb{b} + \omega\mathbb{b}^* \\ & \mathbb{Z} \geq \mathbf{0} \end{cases} \tag{2.6}$$

by setting $\omega = s$, $\mathbb{Z} = \begin{bmatrix} \boldsymbol{z} \\ \delta \end{bmatrix}$, $\mathbb{C} = \begin{bmatrix} \mathbf{c} \\ 0 \end{bmatrix}$, $\mathbb{b} = \begin{bmatrix} \mathbf{b} \\ 0 \end{bmatrix}$, $\mathbb{b}^* = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}$, and $\mathbb{A} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{a}' & 1 \end{bmatrix}$. Note that when $\mathbf{A}$ in (2.6) is of full rank, so is $\mathbb{A}$. Let $\mathfrak{B}^*$ be an optimal basic index set of (2.6) at $\omega = \omega^*$. Similarly, we can show that $\mathfrak{B}^*$ is optimal for any ω satisfying $\mathbb{Z}_{\mathfrak{B}^*} = \mathbb{A}_{\mathfrak{B}^*}^{-1}(\mathbb{b} + \omega\mathbb{b}^*) \geq \mathbf{0}$, and there exist $\underline{\omega}$ and $\overline{\omega}$ such that $\mathfrak{B}^*$ is optimal for $\omega \in [\underline{\omega}, \overline{\omega}]$. This implies that a version of the solution path of (2.6) is a piecewise linear function.

2.3 Generating the Solution Path

Based on the basic concepts and the optimality condition of LP introduced in Section 2.1, we describe algorithms to generate the solution path for (2.2), namely, the simplex and tableau-simplex algorithms. The algorithms were originally proposed by Saaty and Gass (1954); Gass and Saaty (1955a,b) for solving the parametric cost LP problem only. However, we realized that the same algorithms also produce the solution path for (2.5) as stated in Theorem 7, and the proof could be new in the LP literature. Since the examples of the LP problem in Chapter 3 for feature selection involve non-negative $\mathbf{a}$, λ, and s only, we assume that they are non-negative in the following algorithms and take $s = 0$ (equivalently $\lambda = \infty$) as a starting value.

2.3.1 Simplex Algorithm

Initialization

Let $\boldsymbol{z}^0 := (z_1^0, \cdots, z_N^0)'$ denote the initial solution of (2.5) at $s = 0$. $\mathbf{a}'\boldsymbol{z}^0 = 0$ implies $z_j^0 = 0$ for all $j \notin \mathcal{I}_{\mathbf{a}} := \{i : \mathrm{a}_i = 0, i \in \mathcal{N}\}$. Thus, by extracting the coordinates of $\mathbf{c}$, $\boldsymbol{z}$, and the columns in $\mathbf{A}$ corresponding to $\mathcal{I}_{\mathbf{a}}$, we can simplify the initial LP problem of (2.2) and (2.5) to

$$\begin{cases} \min\limits_{\boldsymbol{z}_{\mathcal{I}_{\mathbf{a}}} \in \mathcal{R}^{|\mathcal{I}_{\mathbf{a}}|}} & \mathbf{c}_{\mathcal{I}_{\mathbf{a}}}{}'\boldsymbol{z}_{\mathcal{I}_{\mathbf{a}}} \\ \text{s.t.} & \mathbf{A}_{\mathcal{I}_{\mathbf{a}}}\boldsymbol{z}_{\mathcal{I}_{\mathbf{a}}} = \mathbf{b} \\ & \boldsymbol{z}_{\mathcal{I}_{\mathbf{a}}} \geq \mathbf{0} \end{cases}, \tag{2.7}$$

where $|\mathcal{I}_{\mathbf{a}}|$ is the cardinality of $\mathcal{I}_{\mathbf{a}}$. Accordingly, any initial optimal basic index set, $\mathcal{B}^0$ of (2.2) and (2.5) contains that of the reduced problem (2.7) and determines the initial solution $\boldsymbol{z}^0$.

Main Algorithm

For simplicity, we describe the algorithm for the solution path of the parametric-cost LP problem in (2.2) first, and then discuss how it also solves the complexity-bounded LP problem in (2.5).

Let $\mathcal{B}^l$ be the lth optimal basic index set at $\lambda = \lambda_{l-1}$. For convenience, define $\lambda_{-1} := \infty$, the starting value of the regularization parameter for the solution path of (2.2). Given $\mathcal{B}^l$, let $\boldsymbol{z}^l$ be the lth joint solution, which is given by $\boldsymbol{z}^l_{\mathcal{B}^l} = \mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{b}$ and $\boldsymbol{z}^l_j = 0$ for $j \in \mathcal{N} \backslash \mathcal{B}^l$. Since the optimal LP solution is identified by the optimal basic index set as in Theorem 4, it suffices to describe how to update the optimal basic index set as λ decreases. By the invertibility of the basic matrix associated with the index set, updating amounts to finding a new index that enters and the other that exits the current basic index set.

By Corollary 5, we can compute the lower bound of the optimality interval of $\mathcal{B}^l$ denoted by λ_l and identify the entry index associated with it. Let

$$j^l := \underset{\{j\,:\,\check{\mathrm{a}}^l_j > 0; j \in (\mathcal{N} \setminus \mathcal{B}^l)\}}{\arg\max} \left(-\frac{\check{\mathrm{c}}^l_j}{\check{\mathrm{a}}^l_j} \right), \tag{2.8}$$

where $\check{\mathrm{a}}^l_j := (\mathrm{a}_j - \mathbf{a}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}_j)$ and $\check{\mathrm{c}}^l_j := (\mathrm{c}_j - \mathbf{c}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}_j)$. Then, the lower bound is given by $\lambda_l := -\check{\mathrm{c}}^l_{j^l}/\check{\mathrm{a}}^l_{j^l}$, and $\mathcal{B}^l$ is optimal for $\lambda \in [\lambda_l, \lambda_{l-1}]$.

To determine the index exiting $\mathcal{B}^l$, consider the moving direction from $\boldsymbol{z}^l$ to the next joint solution. Define $\boldsymbol{d}^l := (d^l_1, \cdots, d^l_N)$ as

$$\begin{aligned} &\boldsymbol{d}^l_{\mathcal{B}^l} = -\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}_{j^l}, d^l_{j^l} = 1, \text{ and} \\ &d^l_i = 0 \text{ for } i \in \mathcal{N} \setminus (\mathcal{B}^l \cup \{j^l\}). \end{aligned} \tag{2.9}$$

Lemma 12 in Appendix shows that $\boldsymbol{d}^l$ is the moving direction at $\lambda = \lambda_l$ in the sense that $\boldsymbol{z}^{l+1} = \boldsymbol{z}^l + \tau \boldsymbol{d}^l$ for some $\tau \geq 0$. For the feasibility of $\boldsymbol{z}^{l+1} \geq 0$, the step size τ can not exceed the minimum of $-z^l_i/d^l_i$ for $i \in \mathcal{B}^l$ with $d^l_i < 0$, and the index attaining the minimum is to leave $\mathcal{B}^l$. Denote the the exit index by

$$i^l := \underset{i \in \{j:\, d^l_j < 0,\ j \in \mathcal{B}^l\}}{\arg\min} \left(-\frac{z^l_i}{d^l_i} \right). \tag{2.10}$$

Therefore, the optimal basic index set at $\lambda = \lambda_l$ is given by $\mathcal{B}^{l+1} := \mathcal{B}^l \cup \{j^l\} \setminus \{i^l\}$. More precisely, we can verify the optimality of $\mathcal{B}^{l+1}$ at $\lambda = \lambda_l$ by showing that

$$\begin{aligned} &(\mathbf{c} + \lambda_l \mathbf{a}) - \mathbf{A}' \left(\mathbf{A}^{-1}_{\mathcal{B}^{l+1}} \right)' (\mathbf{c}_{\mathcal{B}^{l+1}} + \lambda_l \mathbf{a}_{\mathcal{B}^{l+1}}) \\ = \ &(\mathbf{c} + \lambda_l \mathbf{a}) - \mathbf{A}' \left(\mathbf{A}^{-1}_{\mathcal{B}^{l}} \right)' (\mathbf{c}_{\mathcal{B}^{l}} + \lambda_l \mathbf{a}_{\mathcal{B}^{l}}). \end{aligned} \tag{2.11}$$

The proof is given in Appendix A.2. Then the fact that $\mathcal{B}^l$ is optimal at $\lambda = \lambda_l$ implies that $\mathcal{B}^{l+1}$ is also optimal at $\lambda = \lambda_l$. As a result, the updating procedure can be repeated with

$\mathcal{B}^{l+1}$ and λ_l successively until $\lambda_l < 0$ or equivalently $\check{\mathrm{c}}^l_{j^l} \geq 0$. The algorithm for updating the optimal basic index sets is summarized as follows.

---

1. Initialize the optimal basic index set at $\lambda_{-1} = \infty$ with $\mathcal{B}^0$.

2. Given $\mathcal{B}^l$, the lth optimal basic index set at $\lambda = \lambda_{l-1}$, determine the solution $\boldsymbol{z}^l$ by $\boldsymbol{z}^l_{\mathcal{B}^l} = \mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{b}$ and $\boldsymbol{z}^l_j = 0$ for $j \in \mathcal{N} \setminus \mathcal{B}^l$.

3. Find the entry index
$$j^l = \underset{j\,:\,\check{\mathrm{a}}^l_j > 0;\, j \in \mathcal{N}\setminus\mathcal{B}^l}{\arg\max} \left(-\frac{\check{\mathrm{c}}^l_j}{\check{\mathrm{a}}^l_j}\right).$$

4. Find the exit index
$$i^l = \underset{i\in\{j:\, d^l_j<0,\; j\in\mathcal{B}^l\}}{\arg\min} \left(-\frac{z^l_i}{d^l_i}\right).$$
If there are multiple indices, choose one of them.

5. Update the optimal basic index set to $\mathcal{B}^{l+1} = \mathcal{B}^l \cup \{j^l\} \setminus \{i^l\}$.

6. Terminate the algorithm if $\check{\mathrm{c}}^l_{j^l} \geq 0$ or equivalently $\lambda_l \leq 0$. Otherwise, repeat 2 – 5.

If $-z^l_{i^l}/d^l_{i^l} = 0$, then $\boldsymbol{z}^l = \boldsymbol{z}^{l+1}$, which may result in the problem of cycling among several basic index sets with the same solution. We defer the description of the tableau-simplex algorithm which can avoid the cycling problem to Section 2.3.2. For brevity, we just assume that $\boldsymbol{z}^l + \tau\boldsymbol{d}^l \geq \mathbf{0}$ for some $\tau > 0$ so that $\boldsymbol{z}^l \neq \boldsymbol{z}^{l+1}$ for each l and call this *non-degeneracy* assumption. Under this assumption, suppose the simplex algorithm terminates after J iterations with $\{(\boldsymbol{z}^l, \lambda_l) : l = 0, 1, \cdots, J\}$. Then the entire solution path is obtained as described below.

Theorem 6 *The solution path of (2.2) is*

$$\begin{cases} \boldsymbol{z}^0 & \text{for } \lambda > \lambda_0 \\ \boldsymbol{z}^l & \text{for } \lambda_l < \lambda < \lambda_{l-1},\ \ l = 1, \cdots, J \\ \tau \boldsymbol{z}^l + (1-\tau)\boldsymbol{z}^{l+1} & \text{for } \lambda = \lambda_l \text{ and } \tau \in [0,1],\ l = 0, \cdots, J-1. \end{cases} \tag{2.12}$$

Likewise, the solutions to the alternative formulation of (2.5) with the complexity bound can be obtained as a function of s. By the correspondence of the two formulations, the lth joint of the piecewise linear solution is given by $s_l = \mathbf{a}'\boldsymbol{z}^l$, and the solution between the joints is a linear combination of $\boldsymbol{z}^l$ and $\boldsymbol{z}^{l+1}$ as described in Theorem 7 below. Its proof is in Appendix A.3. To the best of our knowledge, such a connection between parametric-cost LP and parametric right-hand-side LP has not been proved in linear algebraic language. It may be our contribution to the LP literature.

Theorem 7 *For $s \geq 0$, the solution path for (2.5) can be expressed as*

$$\begin{cases} \frac{s_{l+1} - s}{s_{l+1} - s_l}\boldsymbol{z}^l + \frac{s - s_l}{s_{l+1} - s_l}\boldsymbol{z}^{l+1} & \text{if } s_l \leq s < s_{l+1} \text{ and } l = 0, \cdots, J-1 \\ \boldsymbol{z}^J & \text{if } s \geq s_J. \end{cases}$$

2.3.2 Tableau-Simplex Algorithm

The non-degeneracy assumption in the simplex method that any two consecutive joint solutions are different may not hold in practice for many problems. When some columns of a basic matrix are discrete, the assumption may fail at some degenerate joint solutions. To deal with more general settings where the cycling problem may occur in generating the LP solution path by the simplex method, we discuss the *tableau-simplex* algorithm.

A tableau is a big matrix which contains all the information about the LP. It consists of the relevant terms in LP associated with a basic matrix such as the basic solution and the cost.

Definition 8 *For a basic index set $\mathcal{B}^*$, its tableau is defined in Table 2.1.*

	zeroth column	pivot columns
cost row	$-\mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}$
penalty row	$-\mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$\mathbf{a}' - \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}$
pivot rows	$\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}$

Table 2.1: Tableau for linear programming

We follow the convention for the names of the columns and rows in the tableau. For reference, see Murty (1983) and Bertsimas and Tsitsiklis (1997). Note that the zeroth column contains $\boldsymbol{z}^*_{\mathcal{B}^*} := \mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$, the non-zero part of the basic solution, $-\mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b} = -\mathbf{c}'\boldsymbol{z}^*$, the negative cost, and $-\mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b} = -\mathbf{a}'\boldsymbol{z}^*$, the negative penalty of $\boldsymbol{z}^*$ associated with $\mathcal{B}^*$, and the pivot columns contain $\check{\mathrm{c}}^*_j$'s and $\check{\mathrm{a}}^*_j$'s. The algorithm to be discussed updates the basic index sets by using the tableau, in particular, by ordering some rows of the tableau. To describe the algorithm, we introduce the lexicographic order of vectors first.

Definition 9 *For $\boldsymbol{v}$ and $\boldsymbol{w} \in \mathcal{R}^n$, we say that $\boldsymbol{v}$ is lexicographically greater than $\boldsymbol{w}$ (denoted by $\boldsymbol{v} \overset{L}{>} \boldsymbol{w}$) if the first non-zero entry of $\boldsymbol{v} - \boldsymbol{w}$ is strictly positive. We say that $\boldsymbol{v}$ is lexicographically positive if $\boldsymbol{v} \overset{L}{>} 0$.*

Consider the parametric-cost LP in (2.2).

Initial Tableau

With the index set $\mathcal{B}^0$, initialize the tableau. Since $\boldsymbol{z}^0_{\mathcal{B}^0} = \mathbf{A}_{\mathcal{B}^0}^{-1}\mathbf{b} \geq 0$ and the columns of $\mathbf{A}$ can be rearranged such that the sub-matrix with the first M columns of $\mathbf{A}_{\mathcal{B}^0}^{-1}\mathbf{A}$ is $\mathbf{I}$, we assume that the pivot rows, $[\mathbf{A}_{\mathcal{B}^0}^{-1}\mathbf{b} \;\; \mathbf{A}_{\mathcal{B}^0}^{-1}\mathbf{A}]$, of the initial tableau are lexicographically positive. In other words, there is a permutation $\pi : \mathcal{N} \to \mathcal{N}$ which maps $\mathcal{B}^0$ to $\mathcal{M} := \{1, \cdots, M\}$, and we can replace the problem with the π-permuted version (e.g., $\boldsymbol{z}_{\pi(\mathcal{N})}$ and

$\mathbf{A}_{\pi(\mathcal{N})}$). How to find the initial basic index set is another issue that will be addressed in Chapter 4 with examples.

Updating Tableau

Given the current optimal basic index set $\mathcal{B}^l$, the current tableau is

	zeroth column	pivot columns
cost row	$-\mathbf{c}'_{\mathcal{B}^l}\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{b}$	$\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^l}\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}$
penalty row	$-\mathbf{a}'_{\mathcal{B}^l}\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{b}$	$\mathbf{a}' - \mathbf{a}'_{\mathcal{B}^l}\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}$
pivot rows	$\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{b}$	$\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}$

Suppose all the pivot rows of the current tableau are lexicographically positive. The tableau-simplex algorithm differs from the simplex algorithm only in the way the exit index is determined. The following procedure is generalization of Step 4 in the simplex algorithm for finding the exit index.

Step 4. Let $\boldsymbol{u}^l := (u_1^l, \cdots, u_M^l)' := \mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l}$. For each $i \in \mathcal{M}$ with $u_i^l > 0$, divide the ith pivot row (including the entry in the zeroth column) by u_i^l. And, among those rows, find the index, i_*^l, of the lexicographically smallest row. Then, $i^l := B_{i_*^l}^l$ is the exit index.

Remark Since $\boldsymbol{u}^l = -\boldsymbol{d}_{\mathcal{B}^l}^l$, if i^l in (2.10) is unique with $z_{i^l}^l > 0$, then it is the same as the lexicographically smallest row that the tableau-simplex algorithm seeks. Hence the two algorithms coincide. The simplex algorithm determines the exit index based only on the zeroth column in the tableau while the lexicographic ordering involves the pivot columns additionally. The optimality of $\mathcal{B}^l$ for $\lambda \in [\lambda_l, \lambda_{l-1}]$ immediately follows by the same step 3, and (2.11) remains to hold true for the exit index i^l of the tableau-simplex algorithm, which implies the optimality of $\mathcal{B}^{l+1}$ at $\lambda = \lambda_l$.

Some characteristics of the updated tableau associated with $\mathcal{B}^{l+1}$ are described in the next theorem and its corollary. The proof is adapted from that for the lexicographic pivoting rule in Bertsimas and Tsitsiklis (1997) p. 108–111. See Appendix A.4 for details.

Theorem 10 *For the updated basic index set $\mathcal{B}^{l+1}$ by the tableau-simplex algorithm,*

i) all the pivot rows of the updated tableau are still lexicographically positive, and

ii) the updated cost row is lexicographically greater than that for $\mathcal{B}^l$.

Since $\mathbf{A}_{\mathcal{B}^{l+1}}^{-1}\mathbf{b}$ is the 'zeroth column' of the pivot rows, i) says that the basic solution for $\mathcal{B}^{l+1}$ is feasible, i.e., $\boldsymbol{z}^{l+1} \geq \mathbf{0}$. Moreover, it implies that the updating procedure can be repeated with $\mathcal{B}^{l+1}$ and the new tableau.

It is not hard to see that $\boldsymbol{z}^{l+1} = \boldsymbol{z}^l$ if and only if $z_{i^l}^l = 0$ (see the proof of Theorem 10 in the Appendix for more details). When $z_{i^l}^l = 0$, $\boldsymbol{z}^{l+1} = \boldsymbol{z}^l$, however the tableau-simplex algorithm uniquely updates $\mathcal{B}^{l+1}$ such that the previous optimal basic index sets $\mathcal{B}^l$'s never reappear in the process. This anti-cycling property is guaranteed by ii). By ii), we can strictly order the optimal basic index sets $\mathcal{B}^l$ based on their cost rows. Because of this and the fact that all possible basic index sets are finite, the total number of iterations must be finite. This proves the following.

Corollary 11 *The tableau updating procedure terminates after a finite number of iterations.*

Suppose that the tableau-simplex algorithm stops after J iterations with $\lambda_J \leq 0$. In parallel to the simplex algorithm, the tableau-simplex algorithm outputs the sequence $\{(\boldsymbol{z}^l, s_l, \lambda_l) : l = 0, \cdots, J\}$, and the solution paths for (2.2) and (2.5) admit the same forms as in Theorem 6 and Theorem 7 except for any duplicate joints λ_l and s_l.

CHAPTER 3

STATISTICAL APPLICATIONS

The connection between LP and regularization methods that arise in statistics for feature selection has been noted on a case-by-case basis (e.g., multi-category SVM in Wang and Shen (2006)). In this chapter, more concrete examples (including known examples and new examples) are given, and, for each example, its elements in the parametric LP form are identified. Through the examples, we systematically study the applications of LP in a certain family of regularization methods, and provide a unified LP perspective on understanding and solving the associated problems. Arguably, the general treatment with the LP algorithm may be at odds with computational efficiency in dealing with each of particular settings. And this issue will be discussed in Chapter 4.

3.1 Parametric Procedures

3.1.1 l_1-Norm Quantile Regression

Quantile Regression (QR) is a regression technique, introduced by Koenker and Bassett (1978), intended to estimate the conditional quantile functions. It is obtained by replacing the squared error loss of the classical linear regression for the conditional mean function with a piecewise linear loss called the check function. For a general introduction to QR, see Koenker and Hallock (2001).

For simplicity, assume that the conditional quantiles are linear in the predictors. Given a data set, $\{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in \mathcal{R}^p, y_i \in \mathcal{R}, i = 1, \cdots, n\}$, the τth conditional quantile function is estimated by

$$\min_{\beta_0 \in \mathcal{R}, \boldsymbol{\beta} \in \mathcal{R}^p} \quad \frac{1}{n} \sum_{i=1}^{n} \rho_\tau \left(y_i - \beta_0 - \mathbf{x}_i \boldsymbol{\beta} \right), \tag{3.1}$$

where $\boldsymbol{\beta}_0$ and $\boldsymbol{\beta} := (\beta_1, \ldots, \beta_p)'$ are the quantile regression coefficients for $\tau \in (0, 1)$, and $\rho_\tau(\cdot)$ is the check function defined as

$$\rho_\tau(t) := \begin{cases} \tau \cdot t & \text{for } t > 0 \\ -(1-\tau) \cdot t & \text{for } t \leq 0. \end{cases}$$

When $\tau = 1/2$, QR amounts to median regression, least absolute error (LAE) , or least absolute deviation (LAD) (Bloomfield and Steiger; 1980). The standard QR problem in (3.1) can be cast as an LP problem itself, and enumeration of the entire range of quantile functions parametrized by τ is feasible as noted in Koenker (2005b), p.185. Since it is somewhat different from an array of statistical optimization problems for feature selection that we intend to address in this thesis, we leave an adequate treatment of this topic elsewhere and turn to a regularized QR.

Aiming at estimating the conditional quantile function simultaneously with selecting relevant predictors, Li and Zhu (2008) propose the l_1-norm QR. It is defined by the following constrained optimization problem:

$$\begin{cases} \min\limits_{\beta_0 \in \mathcal{R}, \boldsymbol{\beta} \in \mathcal{R}^p} & \frac{1}{n} \sum_{i=1}^{n} \rho_\tau \left(y_i - \beta_0 - \mathbf{x}_i \boldsymbol{\beta} \right) \\ \text{s.t.} & \|\boldsymbol{\beta}\|_1 \leq s, \end{cases}$$

where $s > 0$ is a regularization parameter. Equivalently, with another tuning parameter λ , the l_1-norm QR can be recast as

$$\begin{cases} \min\limits_{\beta_0 \in \mathcal{R}, \boldsymbol{\beta} \in \mathcal{R}^p, \boldsymbol{\zeta} \in \mathcal{R}^n} & \frac{1}{n} \sum_{i=1}^{n} [\tau(\zeta_i)_+ + (1-\tau)(\zeta_i)_-] + \lambda \|\boldsymbol{\beta}\|_1 \\ \text{s.t.} & \beta_0 + \mathbf{x}_i \boldsymbol{\beta} + \zeta_i = y_i \text{ for } i = 1, \cdots, n, \end{cases} \tag{3.2}$$

where $(x)_+ = \max(x, 0)$ and $(x)_- = \max(-x, 0)$. The optimization problem in (3.2) can be formulated as an LP parametrized by λ, which is a common feature of the examples discussed. For the non-negativity constraint in the standard form of LP, consider both positive and negative parts of each variable and denote, for example, $((\beta_1)_+, \ldots, (\beta_p)_+)'$ by $\boldsymbol{\beta}^+$ and $((\beta_1)_-, \ldots, (\beta_p)_-)'$ by $\boldsymbol{\beta}^-$. Note that $\boldsymbol{\beta} = \boldsymbol{\beta}^+ - \boldsymbol{\beta}^-$ and the l_1-norm $\|\boldsymbol{\beta}\|_1 := \sum_{i=1}^p |\beta_i|$ is given by $\mathbf{1}'(\boldsymbol{\beta}^+ + \boldsymbol{\beta}^-)$ with $\mathbf{1} := (1, \cdots, 1)'$ of appropriate length. Let $\boldsymbol{Y} := (y_1, \cdots, y_n)'$, $\mathbf{X} := (\mathbf{x}_1', \cdots, \mathbf{x}_n')'$, $\boldsymbol{\zeta} := (\zeta_1, \cdots, \zeta_n)'$, and $\mathbf{0} := (0, \cdots, 0)'$ of appropriate length. Then the following elements define the l_1-norm quantile regression in the standard form of a parametric-cost LP in (2.2)

$$
\begin{array}{rccccccc}
\boldsymbol{z} := & (& \beta_0^+ & \beta_0^- & (\boldsymbol{\beta}^+)' & (\boldsymbol{\beta}^-)' & (\boldsymbol{\zeta}^+)' & (\boldsymbol{\zeta}^-)' &)' \\
\mathbf{c} := & (& 0 & 0 & \mathbf{0}' & \mathbf{0}' & \tau\mathbf{1}'/n & (1-\tau)\mathbf{1}'/n &)' \\
\mathbf{a} := & (& 0 & 0 & \mathbf{1}' & \mathbf{1}' & \mathbf{0}' & \mathbf{0}' &)' \\
\mathbf{A} := & [& \mathbf{1} & -\mathbf{1} & \mathbf{X} & -\mathbf{X} & \mathbf{I} & -\mathbf{I} &] \\
\mathbf{b} := & \boldsymbol{Y} & & & & & & &
\end{array}
$$

with a total of $N = 2(1 + p + n)$ variables and $M = n$ equality constraints.

Any joint solution $\boldsymbol{\beta}(\lambda_l)$ (including the intercept) in the solution path of an l_1-norm QR depends on the dataset only through a subset of the observation pairs with $\zeta_i = 0$. The cardinality of the subset equals the number of β's whose indices are in the associated basic index set. In other words, if the lth basic index set contains indices for p_l covariates (including the intercept), the expression of $\boldsymbol{\beta}(\lambda_l)$ depends on the dataset only through p_l observation pairs. It is analogous to the well known fact that a sample median depends on the entire sample only through the middle values. And this property is consistent with the conclusions by Koenker (2005a) (page 34) for QR without penalty. As a consequence, when n is small but p is large (i.e., small n large p), the number of selected variables can not be larger than n. The proof is straightforward from the structure of the associated basic matrix, and is omitted in this thesis.

3.1.2 Quantile Regression with Grouped Variables

In real applications, the covariates are often grouped in nature where group selection may be more interesting than individual variable selection. In this section, we explore an LP algorithm that can do group selection for QR. Since QR can be viewed as an extension of median regression, we first give the LP application to median regression with grouped variables, and then the QR version follows.

Consider a linear model with J groups of variables:

$$\boldsymbol{Y} = \sum_{j=1}^{J} \mathbf{X}_j \boldsymbol{\beta}_j + \boldsymbol{\epsilon},$$

where $\boldsymbol{Y}$ and $\boldsymbol{\epsilon}$ are n-vectors, $\mathbf{X}_j$ is an $n \times p_j$ matrix associated with the jth group of variables, and $\boldsymbol{\beta}_j := (\beta_{1j}, \cdots, \beta_{p_j j})'$ is a coefficient vector of size p_j for $j = 1, \cdots, J$.

Let $\boldsymbol{\beta} := (\boldsymbol{\beta}_1', \cdots, \boldsymbol{\beta}_J')'$ and $\mathbf{X} := (\mathbf{X}_1, \cdots, \mathbf{X}_J)$. We are interested in selecting important variable groups and estimating the corresponding $\boldsymbol{\beta}$. To achieve this goal, Yuan and Lin (2006) proposed a convex grouped lasso penalty defined as

$$\|\boldsymbol{\beta}\|_{glasso} := \sum_{j=1}^{J} \|\boldsymbol{\beta}_j\|_2,$$

where $\|\boldsymbol{\beta}_j\|_2$ is the l_2-norm of $\boldsymbol{\beta}_j$. Grouped lasso estimates $\boldsymbol{\beta}$ by minimizing

$$\frac{1}{n} \left\| \boldsymbol{Y} - \sum_{j=1}^{J} \mathbf{X}_j \boldsymbol{\beta}_j \right\|_2^2 + \lambda \|\boldsymbol{\beta}\|_{glasso}.$$

Unlike the lasso penalty (Tibshirani; 1996), the grouped lasso penalty is not a piecewise linear function of $\boldsymbol{\beta}$. Its solution path in general is not piecewise linear, and thus has to be calculated at each λ.

For piecewise linearity, one may consider another penalty for grouped variable selection (tentatively called G-penalty) given as

$$\|\boldsymbol{\beta}\|_g := \sum_{j=1}^{J} \|\boldsymbol{\beta}_j\|_\infty,$$

which is suggested by Robert Cohen at SAS Institute through personal communication.

This section addresses median regression with G-penalty first, and a toy example of the method can be found in Section 5.1.3.

Here, $\boldsymbol{\beta}$ is estimated by minimizing

$$\frac{1}{n}\left\|\boldsymbol{Y}-\beta_0-\sum_{j=1}^{J}\mathbf{X}_j\boldsymbol{\beta}_j\right\|_1+\lambda\|\boldsymbol{\beta}\|_g. \tag{3.3}$$

In order to reduce bias, the intercept term β_0 is not penalized with G-penalty. The optimization problem in (3.3) can be transformed into a parametric LP problem. To explicitly describe the transformation, we introduce the slack variables $\boldsymbol{\xi}$, $\boldsymbol{\rho}^+ := (\rho_1^+, \cdots, \rho_J^+)$, and $\boldsymbol{\eta}^+ := (\boldsymbol{\eta}_1^{+\prime}, \cdots, \boldsymbol{\eta}_J^{+\prime})'$ that satisfy

$$\begin{aligned}
&\boldsymbol{\xi} := \boldsymbol{\xi}^+ - \boldsymbol{\xi}^- := \boldsymbol{Y} - \beta_0 - \sum_{j=1}^{J}\mathbf{X}_j\boldsymbol{\beta}_j,\\
&\rho_j^+\mathbf{1} = \boldsymbol{\beta}_j^+ + \boldsymbol{\beta}_j^- + \boldsymbol{\eta}_j^+ \text{ with } \boldsymbol{\beta}_j = \boldsymbol{\beta}_j^+ - \boldsymbol{\beta}_j^-; j = 1, \cdots, J,\\
&\boldsymbol{\xi}^+, \boldsymbol{\xi}^-, \boldsymbol{\rho}^+, \boldsymbol{\eta}^+, \boldsymbol{\beta}^+, \boldsymbol{\beta}^- \geq \mathbf{0}.
\end{aligned}$$

Then the median regression with G-penalty in (3.3) can be recast as

$$\begin{aligned}
\min \quad & \frac{1}{n}\mathbf{1}'(\boldsymbol{\xi}^+ + \boldsymbol{\xi}^-), \\
\text{s.t.} \quad & \boldsymbol{\xi}^+ - \boldsymbol{\xi}^- := \boldsymbol{Y} - (\beta_0^+ - \beta_0^-) - \sum_{j=1}^{J}\mathbf{X}_j(\boldsymbol{\beta}_j^+ - \boldsymbol{\beta}_j^-) \\
& (\rho_1^+\mathbf{1}_{p_1}', \cdots, \rho_J^+\mathbf{1}_{p_J}')' = \boldsymbol{\beta}^+ + \boldsymbol{\beta}^- + \boldsymbol{\eta}^+ \\
\text{w.r.t.} \quad & \boldsymbol{\xi}^+, \boldsymbol{\xi}^-, \boldsymbol{\rho}^+, \boldsymbol{\eta}^+, \boldsymbol{\beta}^+, \boldsymbol{\beta}^- \geq \mathbf{0}.
\end{aligned} \tag{3.4}$$

By matching the elements in (3.4) and (2.2), we have

$$
\begin{array}{rl}
\boldsymbol{z} := & (\ (\boldsymbol{\xi}^+)' \quad (\boldsymbol{\eta}^+)' \quad (\boldsymbol{\xi}^-)' \quad \beta_0^+ \quad \beta_0^- \quad (\boldsymbol{\beta}^+)' \quad (\boldsymbol{\beta}^-)' \quad (\boldsymbol{\rho}^+)'\ )' \\
\mathbf{c} := & (\ \mathbf{1}'/n \quad \mathbf{0}' \quad \mathbf{1}'/n \quad 0 \quad 0 \quad \mathbf{0}' \quad \mathbf{0}' \quad \mathbf{0}'\ )' \\
\mathbf{a} := & (\ \mathbf{0}' \quad \mathbf{0}' \quad \mathbf{0}' \quad 0 \quad 0 \quad \mathbf{0}' \quad \mathbf{0}' \quad \mathbf{1}'\ )' \\
\mathbf{A} := & \begin{bmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{I} & \mathbf{1} & -\mathbf{1} & \mathbf{X} & -\mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{I} & -\mathbf{G} \end{bmatrix} \\
\mathbf{b} := & \begin{pmatrix} \boldsymbol{Y} \\ \mathbf{0} \end{pmatrix}
\end{array}
$$

where $\mathbf{G} := \begin{pmatrix} \mathbf{1}_{p_1} & \mathbf{0}_{p_1} & \cdots & \mathbf{0}_{p_1} \\ \mathbf{0}_{p_2} & \mathbf{1}_{p_2} & \cdots & \mathbf{0}_{p_2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{p_J} & \mathbf{0}_{p_J} & \cdots & \mathbf{1}_{p_J} \end{pmatrix}$.

For QR in general, we find β_0 and $\boldsymbol{\beta}$ that minimize

$$
\frac{\tau}{n}\left(\boldsymbol{Y} - \beta_0 - \sum_{j=1}^{J}\mathbf{X}_j\boldsymbol{\beta}_j\right)_+ + \frac{1-\tau}{n}\left(\boldsymbol{Y} - \beta_0 - \sum_{j=1}^{J}\mathbf{X}_j\boldsymbol{\beta}_j\right)_- + \lambda\|\boldsymbol{\beta}\|_g.
$$

The standard form of LP for the QR with G-penalty is the same as that for median regression except

$$
\mathbf{c} := (\tau\mathbf{1}'/n, \mathbf{0}', (1-\tau)\mathbf{1}'/n, 0, 0, \mathbf{0}', \mathbf{0}', \mathbf{0}')'.
$$

3.1.3 Dantzig Selector

In this section, we adopt the notation used in Section 3.1.1. For variable selection with large p small n data, Candes and Tao (2007) proposed Dantzig selector. The method achieves model selection for linear models via the following regularization

$$
\min_{\boldsymbol{\beta}}\|\boldsymbol{\beta}\|_1 \text{ subject to} \|\mathbf{X}'(\boldsymbol{Y} - \beta_0\mathbf{1} - \mathbf{X}\boldsymbol{\beta})\|_\infty \le \lambda_D,
$$

where λ_D is a tuning parameter. It has been noted that there exist many connections between LASSO and Dantzig selector (Meinshausen et al.; 2007; James et al.; 2008).

The Dantzig selector can be equivalently rewritten as

$$
\min_{\beta_0, \boldsymbol{\beta}}\|\mathbf{X}'(\boldsymbol{Y} - \beta_0\mathbf{1} - \mathbf{X}\boldsymbol{\beta})\|_\infty + \lambda\|\boldsymbol{\beta}\|_1.
$$

To cast Dantzig selector in the form of a parametric LP, we introduce the following slack variables:

$$\boldsymbol{\xi} := \mathbf{X}'\boldsymbol{Y} - \mathbf{X}'\mathbf{1}\beta_0 - \mathbf{X}'\mathbf{X}\boldsymbol{\beta},$$

$$\boldsymbol{\eta}^+ \text{ and } \rho^+ \text{ satisfying } \boldsymbol{\xi}^+ + \boldsymbol{\xi}^- + \boldsymbol{\eta}^+ = \rho^+\mathbf{1}.$$

Then, the associated parametric LP is defined by setting

$$\begin{array}{rcl}
\boldsymbol{z} := & ((\boldsymbol{\xi}^+)' & (\boldsymbol{\eta}^+)' \quad (\boldsymbol{\xi}^-)' \quad \beta_0^+ \quad \beta_0^- \quad (\boldsymbol{\beta}^+)' \quad (\boldsymbol{\beta}^-)' \quad \rho^+)' \\
\mathbf{c} := & (\mathbf{0}' & \mathbf{0}' \quad \mathbf{0}' \quad 0 \quad 0 \quad \mathbf{0}' \quad \mathbf{0}' \quad 1 \ )' \\
\mathbf{a} := & (\mathbf{0}' & \mathbf{0}' \quad \mathbf{0}' \quad 0 \quad 0 \quad \mathbf{1}' \quad \mathbf{1}' \quad 0 \ )' \\
\mathbf{A} := & \left[\begin{array}{cccccccc} \mathbf{I} & \mathbf{0} & -\mathbf{I} & \mathbf{X}'\mathbf{1} & -\mathbf{X}'\mathbf{1} & \mathbf{X}'\mathbf{X} & -\mathbf{X}'\mathbf{X} & \mathbf{0} \\ \mathbf{I} & \mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{1} \end{array} \right] \\
\mathbf{b} := & \left(\begin{array}{c} \mathbf{X}'\boldsymbol{Y} \\ \mathbf{0} \end{array} \right).
\end{array}$$

3.1.4 l_1-Norm Support Vector Machine

Consider a binary classification problem where $y_i \in \{-1, 1\}, i = 1, \cdots, n$ denote the class labels. The Support Vector Machine (SVM) introduced by Cortes and Vapnik (1995) is a classification method that finds the optimal hyperplane maximizing the margin between the classes. It is another example of a regularization method with a margin based hinge loss and the ridge regression type l_2 norm penalty. The optimal hyperplane ($\beta_0 + \mathbf{x}\boldsymbol{\beta} = 0$) in the standard SVM is determined by the solution to the problem:

$$\min_{\beta_0 \in \mathcal{R}, \boldsymbol{\beta} \in \mathcal{R}^p} \frac{1}{n} \sum_{i=1}^{n} \{1 - y_i (\beta_0 + \mathbf{x}_i\boldsymbol{\beta})\}_+ + \lambda \|\boldsymbol{\beta}\|_2^2.$$

Replacing the l_2 norm with the l_1 norm for selection of variables, Bradley and Mangasarian (1998) and Zhu et al. (2004) arrive at a variant of the soft-margin SVM:

$$\left\{ \begin{array}{cl} \min\limits_{\beta_0 \in \mathcal{R}, \boldsymbol{\beta} \in \mathcal{R}^p, \boldsymbol{\zeta} \in \mathcal{R}^n} & \frac{1}{n} \sum_{i=1}^{n} (\zeta_i)_+ + \lambda \|\boldsymbol{\beta}\|_1 \\ \text{s.t.} & y_i(\beta_0 + \mathbf{x}_i\boldsymbol{\beta}) + \zeta_i = 1 \text{ for } i = 1, \cdots, n. \end{array} \right. \tag{3.5}$$

Both SVM and l_1-norm SVM can be viewed as regularized versions of the Minimization of the Sum of Deviations (MSD) method (Freed and Glover; 1981a,b) which in fact minimizes the empirical risk with respect to the hinge loss.

Similarly to MSD, this l_1-norm SVM can be formulated as a parametric cost LP with the following elements

$$\begin{array}{rccccccccc}
\boldsymbol{z} := & (& \beta_0^+ & \beta_0^- & (\boldsymbol{\beta}^+)' & (\boldsymbol{\beta}^-)' & (\boldsymbol{\zeta}^+)' & (\boldsymbol{\zeta}^-)' &)' \\
\mathbf{c} := & (& 0 & 0 & \mathbf{0}' & \mathbf{0}' & \mathbf{1}'/n & \mathbf{0}' &)' \\
\mathbf{a} := & (& 0 & 0 & \mathbf{1}' & \mathbf{1}' & \mathbf{0}' & \mathbf{0}' &)' \\
\mathbf{A} := & [& \boldsymbol{Y} & -\boldsymbol{Y} & \text{diag}(\boldsymbol{Y})\mathbf{X} & -\text{diag}(\boldsymbol{Y})\mathbf{X} & \mathbf{I} & -\mathbf{I} &] \\
\mathbf{b} := & \mathbf{1}. & & & & & & &
\end{array}$$

This example will be revisited in great detail in Section 4.3.

Extending the definition of the support vector for the standard SVM, we call an observation pair $(y_i, \mathbf{x}_i)$ a support vector if $y_i(\beta_0 + \mathbf{x}_i\boldsymbol{\beta}) \leq 1$. Referring to (3.5), we can see that $(y_i, \mathbf{x}_i)$ is a support vector if and only if $\zeta_i \geq 0$. The following remark describes the relationship between the size of dataset, the number of support vectors, and the number of selected variables for l_1-norm SVM.

Remark The expression of any joint solution $\boldsymbol{\beta}(\lambda_l)$ (including the intercept) in the solution path of an l_1-norm support vector machine depends on the dataset only through the set of support vectors whose indices are not in the associated basic index set. The cardinality of such a set of support vectors equals the number of β's whose indices are in the associated basic index set. Consequently, for a small n large p problem, the number of selected variables can not be larger than the size of dataset.

3.1.5 Multi-category Support Vector Machine

For generality, consider a k-category problem with potentially different misclassification costs. A class label of y is also coded by a k-vector; $\mathrm{y} = (y^1, \ldots, y^k)'$ with $y^j = 1$

and $-1/(k-1)$ elsewhere if $y = j$. $L(y_i) = (L^1_{y_i}, \ldots, L^k_{y_i})$ is a misclassification cost vector, where $L^j_{y_i}$ is the cost of misclassifying the class index of y_i as j. A multi-category classification method aims to find $f = (f^1, \ldots, f^k)'$ closely matching an appropriate class code y given $\mathbf{x}$ which induces a classifier $\phi(\mathbf{x}) = \arg\max_{j=1,\ldots,k} f^j(\mathbf{x})$. Lee et al. (2004) extended the binary SVM to Multi-category SVM (MSVM) with a multi-category hinge loss

$$L(y)\{f(\mathbf{x}) - \mathbb{y}\}_+.$$

A standard form of $L(y)$ has $L^j_y = \begin{cases} 0 & \text{if } y = j \\ 1 & \text{if } y \neq j \end{cases}$. Using this loss, the MSVM method finds f by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} I(y_i \neq j)\left\{f^j(\mathbf{x}_i) + \frac{1}{k-1}\right\}_+ \text{ or, equivalently, } \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} I(y_i \neq j)\left\{f^j(\mathbf{x}_i) + 1\right\}_+.$$

Consider the linear classifier with $f^j(\mathbf{x}) = \beta^j_0 + \mathbf{x}\boldsymbol{\beta}^j$. Let $\boldsymbol{\beta} := [\boldsymbol{\beta}^1, \cdots, \boldsymbol{\beta}^k] := [\boldsymbol{\beta}'_1, \cdots, \boldsymbol{\beta}'_p]'$, where $\boldsymbol{\beta}^j$ is the jth column vector of $\boldsymbol{\beta}$ and $\boldsymbol{\beta}_t$ is the tth row vector of $\boldsymbol{\beta}$ for $j = 1, \cdots, k$ and $t = 1, \cdots, p$. We denote a family of matrix norms as follows:

$$\|\boldsymbol{\beta}\|_{c(r)} := \left\|(\|\boldsymbol{\beta}_1\|_r, \cdots, \|\boldsymbol{\beta}_p\|_r)'\right\|_c; c, r \geq 1.$$

For example, the Frobenius norm is $\|\boldsymbol{\beta}\|_{2(2)}$, $\|\boldsymbol{\beta}\|_{1(1)} = \sum_{t=1}^{p}\|\boldsymbol{\beta}_t\|_1 = \sum_{j=1}^{k}\|\boldsymbol{\beta}^j\|_1$, and $\|\boldsymbol{\beta}\|_{1(\infty)} = \sum_{t=1}^{p}\|\boldsymbol{\beta}_t\|_\infty$.

One way to achieve variable selection is to penalize the sum of $\|\boldsymbol{\beta}^j\|_1$'s as in Wang and Shen (2006, 2007). Its associated optimization problem is

$$\begin{aligned} \min_{\beta_0, \boldsymbol{\beta}} \quad & \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} I(y_i \neq j)\left\{\beta^j_0 + \mathbf{x}_i\boldsymbol{\beta}^j + 1\right\}_+ + \lambda\|\boldsymbol{\beta}\|_{1(1)} \\ \text{s.t.} \quad & \boldsymbol{\beta}\mathbf{1} = \mathbf{0} \text{ and } \sum_{j=1}^{k}\beta^j_0 = 0. \end{aligned}$$

Zhang et al. (2008) suggest to select important variables via sup-norm for MSVM. The regularization formulation for their method is:

$$\begin{aligned} \min_{\beta_0, \boldsymbol{\beta}} \quad & \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} I(y_i \neq j) \left\{ \beta_0^j + \mathbf{x}_i \boldsymbol{\beta}^j + 1 \right\}_+ + \lambda \|\boldsymbol{\beta}\|_{1(\infty)} \\ \text{s.t.} \quad & \boldsymbol{\beta} \mathbf{1} = 0 \text{ and } \sum_{j=1}^{k} \beta_0^j = 0. \end{aligned}$$

Again, both of the two MSVM methods can be cast as parametric LP problems. The corresponding elements of LP for the two MSVM methods can be derived in a similar way as in the previous sections, so will not be listed here.

3.2 Nonparametric Procedures

So far, parametric regularization procedures have been discussed. Despite the simplicity and their broad applications, they may be limited in some situations that call for more flexible model/feature spaces. This section regards nonparametric regularization procedures, which consider a potentially infinite dimensional model/feature space.

When a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ is chosen as the feature space $\mathcal{F}$, one may take the squared functional norm of a model f in $\mathcal{H}$ (i.e., $J(f) := ||f||^2_{\mathcal{H}}$) or its projection in a subspace as a measure of the model complexity. Such a choice of the feature space and the model complexity provides a wide range of nonparametric procedures for statistical estimation and prediction. To name a few, examples are smoothing splines (Wahba; 1990), support vector machines (SVM) (Vapnik; 1998), nonparametric QR (Koenker; 2005a).

The popularity of the nonparametric regularization methods is partially ascribed to their versatility and competitive prediction accuracy as demonstrated in many applications. Their main flexibility is achieved by embedding of attributes or variables into a

high-dimensional feature space via kernel mapping, where complex models can make the empirical risk arbitrarily close to zero. Such embedding does not need to be explicit for prediction as the fitted model is expressed as a linear combination of the data representers determined by the pre-specified reproducing kernel. For an implicit mapping, the solution is given as a "black box" function, which may not provide clear explanation of the importance of each variable involved in the solution. This drawback can be remedied with the scheme of functional Analysis of Variance (ANOVA) decomposition (Wahba; 1990; Gu; 2002).

In functional ANOVA, a model f is a multivariate function of p covariates $\mathbf{x} := (x_1, \cdots, x_p)'$ defined on $\mathcal{X} := \mathcal{X}_1 \times \cdots \times \mathcal{X}_p$, where $x_\alpha \in \mathcal{X}_\alpha$ for $\alpha = 1, \cdots, p$. Consider an RKHS $\mathcal{H}$ that is constructed as a tensor product of functional subspace $\mathcal{H}_\alpha$, an RKHS of functions on $\mathcal{X}_\alpha$. Further decompose $\mathcal{H}_\alpha$ as $\{1\} \oplus \bar{\mathcal{H}}_\alpha$, where $\bar{\mathcal{H}}_\alpha$ is the subspace of $\mathcal{H}_\alpha$ orthogonal to $\{1\}$. The overall feature space is then given by

$$\mathcal{H} := \{1\} \oplus_{\alpha=1}^{p} \bar{\mathcal{H}}_\alpha \oplus_{\alpha<\beta} \{\bar{\mathcal{H}}_\alpha \oplus \bar{\mathcal{H}}_\beta\} \oplus \cdots .$$

And an ANOVA-like decomposition of $f \in \mathcal{H}$ is in the form of

$$f(\mathbf{x}) = f_0 + \sum_{\alpha=1}^{p} f_\alpha(x_\alpha) + \sum_{\alpha<\beta} f_{\alpha\beta}(x_\alpha, x_\beta) + \cdots , \tag{3.6}$$

where $f_0 \in \{1\}$, $f_\alpha \in \bar{\mathcal{H}}_\alpha$, $f_{\alpha\beta} \in \{\bar{\mathcal{H}}_\alpha \oplus \bar{\mathcal{H}}_\beta\}$, and so on. Truncating a set of subspaces in $\mathcal{H}$ for higher-order interactions results in the corresponding simplification of $\mathcal{H}$. Relabel the remaining subspaces as $\mathcal{F}_\nu$ for $\nu \in \{1, \cdots, d\}$ after truncation, and let the resulting RKHS be

$$\mathcal{F} := \{1\} \oplus \bar{\mathcal{F}} \text{ with } \bar{\mathcal{F}} := \oplus_{\nu=1}^{d} \mathcal{F}_\nu. \tag{3.7}$$

Setting the model penalty to be the sum of squared norms of all the terms in (3.6) commonly used for the ANOVA-like decomposition (e.g., smoothing spline ANOVA (Wahba;

1990)), we define a regularization procedure that minimizes

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, b+h(\mathbf{x}_i))+\lambda\|h\|^2_{\bar{\mathcal{F}}} := \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, f(\mathbf{x}_i))+\lambda\sum_{\nu=1}^{d}\|P^\nu f\|^2_{\mathcal{F}}, \tag{3.8}$$

where $f(\cdot) := b + h(\cdot)$ with b being a constant and $h \in \bar{\mathcal{F}}$, $\|\cdot\|_{\bar{\mathcal{F}}}$ and $\|\cdot\|_{\mathcal{F}}$ are respectively the norms defined on the RKHS $\bar{\mathcal{F}}$ and $\mathcal{F}$, P^ν is the orthogonal projection operator from $\mathcal{F}$ onto $\mathcal{F}_\nu$. $P^\nu f$ is called the functional component of f on $\mathcal{F}_\nu$. As is well known, an RKHS can be characterized by its associated reproducing kernel. Let K_ν denote the reproducing kernel for $\mathcal{F}_\nu$. Then the kernel for $\bar{\mathcal{F}}$ in (3.7) is given by $\sum_{\nu=1}^{d} K_\nu$.

Analogous to the parametric cases, an important issue in nonparametric modeling and prediction is to determine which covariates or functional components should be included in the model f. To address the problem, we parameterize the feature space $\bar{\mathcal{F}}$ as $\bar{\mathcal{F}}_{\boldsymbol{\theta}}$ with re-scaling parameter vector $\boldsymbol{\theta} := (\theta_1, \ldots, \theta_d)' \geq \mathbf{0}$. The non-negative weights θ_ν's are introduced for re-calibration of the functional components f_ν. For fixed $\boldsymbol{\theta}$, the feature space is the RKHS $\bar{\mathcal{F}}_{\boldsymbol{\theta}}$ characterized by the kernel $K = \sum_{\nu=1}^{d}\theta_\nu K_\nu$. Accordingly, the objective function in (3.8) becomes

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, b+h(\mathbf{x}_i))+\lambda\|h\|^2_{\bar{\mathcal{F}}_{\boldsymbol{\theta}}} := \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, f(\mathbf{x}_i))+\lambda\sum_{\nu=1}^{d}\theta_\nu^{-1}\|P^\nu f\|^2_{\mathcal{F}}. \tag{3.9}$$

It is easy to see that, if $\theta_\nu = 0$ for some ν, we have $\theta_\nu^{-1}\|P^\nu f\|^2_{\mathcal{F}} = \infty$ for any f satisfying $\|P^\nu f\|_{\mathcal{F}} \neq 0$, thus any reasonable optimal solution for the regularization problem should have the corresponding functional component $f_\nu := P^\nu f$ equal to zero.

Identification of a model in the parametrized feature space requires estimation of $\boldsymbol{\theta}$ as well as $f \in \mathcal{F}$. For simultaneous model fitting and functional component selection, we consider another layer of regularization of the solution with a penalty on $\boldsymbol{\theta}$, which leads to a general procedure for functional component pursuit akin to basis pursuit (Chen et al.; 1999). The procedure under consideration can be formulated as the problem of finding

$f \in \mathcal{F}$ that minimizes

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, b + h(\mathbf{x}_i)) + \lambda_\circ \|h\|^2_{\mathcal{F}_{\boldsymbol{\theta}}} + \lambda_\theta \mathcal{P}(\boldsymbol{\theta}) \text{ subject to } \boldsymbol{\theta} \geq \mathbf{0}, \tag{3.10}$$

where $\lambda_\circ$ and λ_θ are positive tuning parameters, and $\mathcal{P}(\boldsymbol{\theta})$ is a measure of kernel complexity different from the model complexity, $\|h\|^2_{\mathcal{F}_{\boldsymbol{\theta}}}$. For example, the l_1 penalty $\mathcal{P}(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_1 = \sum_{\nu=1}^{d}\theta_\nu$ has been used in COSSO (Lin and Zhang; 2006) and structured support vector machines (Lee et al.; 2006) for measuring kernel complexity.

By the representer theorem (Wahba; 1990), the solution of the problem in (3.9) is of the form

$$\hat{f}(\mathbf{x}) := \hat{b} + \hat{h}(\mathbf{x}) \text{ with } \hat{h}(\mathbf{x}) := \sum_{i=1}^{n} c_i K(\mathbf{x}_i, \mathbf{x}) = \sum_{\nu=1}^{d}\theta_\nu \sum_{i=1}^{n} c_i K_\nu(\mathbf{x}_i, \mathbf{x}) = \sum_{\nu=1}^{d}\theta_\nu \hat{h}_\nu(\mathbf{x}),$$

where $\hat{h}_\nu(\mathbf{x}) := \sum_{i=1}^{n} c_i K_\nu(\mathbf{x}_i, \mathbf{x})$. Therefore, the regularization problem can be reduced to the minimization of

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, b + \sum_{j=1}^{n}\sum_{\nu=1}^{d} K_\nu(\mathbf{x}_i, \mathbf{x}_j)c_j) + \lambda_\circ \sum_{\nu=1}^{d}\theta_\nu \boldsymbol{c}'\mathcal{K}_\nu \boldsymbol{c} + \lambda_\theta \mathcal{P}(\boldsymbol{\theta}) \tag{3.11}$$
$$\text{with respect to } \boldsymbol{c} \in \mathcal{R}^n \text{ and } \boldsymbol{\theta} \geq \mathbf{0},$$

where $\boldsymbol{c} := (c_1, \cdots, c_n)'$, and $\mathcal{K}_\nu := [K_\nu(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^{n}$. Note that, by utilizing the kernel structure, the connection between $\hat{f}$ and each of the functional components in (3.6) can be established through individual K_ν's, and then the estimated $\hat{f}(\mathbf{x})$ has an ANOVA-like interpretation in terms of the corresponding functional components.

An efficient way to solve the problem in (3.11) is to alternately update $\boldsymbol{c}$ and $\boldsymbol{\theta}$, termed as the $\boldsymbol{c}$-step and $\boldsymbol{\theta}$-step, respectively, which has been applied in Lin and Zhang (2006) and Lee et al. (2006). Detailed discussions of the idea can be found in Lin and Zhang (2006); Gunn and Kandola (2002); Zhang (2006); Lee et al. (2006). More generally, Micchelli and Pontil (2005) treat it as a regularization procedure for optimal kernel combination.

This section discusses the application of linear programming (LP) algorithms for solving the nonparametric regularization problems with structured kernels. If $\mathcal{L}$ and $\mathcal{P}$ in (3.10) are both convex linear functions with respect to the kernel coefficients, the corresponding $\boldsymbol{\theta}$-step of the nonparametric regularization problem can be rephrased as a parametric LP problem, and then its solution path can be computed by simplex algorithm. As mentioned before, our motivation comes from the interpretability of the nonparametric models and the fact that the $\boldsymbol{\theta}$-step in (3.11) can be treated as a regularization problem in its own right.

3.2.1 Structured Multi-category Support Vector Machine

Consider the "θ-step" of the Structured Multi-category SVM (SMSVM) in Lee et al. (2006), which yields another parametric cost LP problem. Following the notation defined in Section 3.1.5, suppose that each f^j is of the form $c_0^j + h^j(\mathbf{x}) := c_0^j + \sum_{i=1}^n c_i^j \sum_{\nu=1}^d \theta_\nu K_\nu(\mathbf{x}_i, \mathbf{x})$. By the reproducing property of $\bar{\mathcal{F}}_{\boldsymbol{\theta}}$, $\|h^j\|_K^2 = (\boldsymbol{c}^j)' \left(\sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu\right) \boldsymbol{c}^j$, where $\boldsymbol{c}^j := (c_1^j, \ldots, c_n^j)'$ is the jth coefficient vector, and $\mathcal{K}_\nu$ is the n by n kernel matrix associated with K_ν. With the extended hinge loss $\mathcal{L}\{y_i, f(\mathbf{x}_i)\} := L(y_i)\{f(\mathbf{x}_i) - \mathbb{y}_i\}_+$, the SMSVM finds f with $\boldsymbol{c}$ and $\boldsymbol{\theta}$ minimizing

$$\frac{1}{n}\sum_{i=1}^n L(y_i)[f(\mathbf{x}_i) - \mathbb{y}_i]_+ + \lambda_\circ \sum_{j=1}^k \|h^j\|_K^2 + \lambda_\theta \sum_{\nu=1}^d \theta_\nu \tag{3.12}$$

subject to $\theta_\nu \geq 0$ for $\nu = 1, \ldots, d$. By alternating estimation of $\boldsymbol{c}$ and $\boldsymbol{\theta}$, we attempt to find the optimal kernel configuration (a linear combination of pre-specified kernels) and the coefficients associated with the optimal kernel. The θ-step refers to optimization of the functional component weights $\boldsymbol{\theta}$ given $\boldsymbol{c}$. More specifically, treating $\boldsymbol{c}$ as fixed, the weights of the features are chosen to minimize

$$\frac{1}{n}\sum_{j=1}^k (\mathbf{L}^j)' \left(c_0^j \mathbf{1} + \sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu \boldsymbol{c}^j - \boldsymbol{y}^j\right)_+ + \lambda_\circ \sum_{j=1}^k (\boldsymbol{c}^j)' \left(\sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu\right) \boldsymbol{c}^j + \lambda_\theta \sum_{\nu=1}^d \theta_\nu,$$

where $\mathbf{L}^j := (L^j_{y_1}, \ldots, L^j_{y_n})'$ and $\boldsymbol{y}^j = (y^j_1, \ldots, y^j_n)'$.

This optimization problem can be rephrased as

$$\begin{cases} \min\limits_{\boldsymbol{\zeta} \in \mathcal{R}^{nk}, \boldsymbol{\theta} \in \mathcal{R}^d} & \frac{1}{n}\sum_{j=1}^k (\mathbf{L}^j)'(\boldsymbol{\zeta}^j)_+ + \lambda_\circ \sum_{\nu=1}^d \theta_\nu \left(\sum_{j=1}^k (\boldsymbol{c}^j)' \mathcal{K}_\nu \boldsymbol{c}^j\right) + \lambda_\theta \sum_{\nu=1}^d \theta_\nu \\ \text{s.t.} & \sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu \boldsymbol{c}^j - \boldsymbol{\zeta}^j = \boldsymbol{y}^j - c_0^j \mathbf{1} \text{ for } j = 1, \ldots, k \\ & \theta_\nu \geq 0 \text{ for } \nu = 1, \ldots, d. \end{cases}$$

Let

$$\begin{aligned} &\mathbf{g} := (\mathrm{g}_1, \cdots, \mathrm{g}_d)' \text{ with } \mathrm{g}_\nu := \lambda_\circ \textstyle\sum_{j=1}^k (\boldsymbol{c}^j)' \mathcal{K}_\nu \boldsymbol{c}^j \\ &\mathbf{L} := \left((\mathbf{L}^1)', \cdots, (\mathbf{L}^k)'\right)', \ \boldsymbol{\zeta} := \left((\boldsymbol{\zeta}^1)', \cdots, (\boldsymbol{\zeta}^k)\right)' \\ &\mathbf{X} := \begin{bmatrix} \mathcal{K}_1 \boldsymbol{c}^1 & \cdots & \mathcal{K}_d \boldsymbol{c}^1 \\ \vdots & \ddots & \vdots \\ \mathcal{K}_1 \boldsymbol{c}^k & \cdots & \mathcal{K}_d \boldsymbol{c}^k \end{bmatrix} \\ &\boldsymbol{Y} := ((\boldsymbol{y}^1)', \cdots, (\boldsymbol{y}^k)')' \\ &\boldsymbol{c}_0 := (c_0^1, \cdots, c_0^k)', \end{aligned} \tag{3.13}$$

then the following elements define the θ-step as a parametric cost LP indexed by λ_θ with $N = d + 2nk$ variables and $M = nk$ equality constraints

$$\begin{aligned} \boldsymbol{z} &:= \begin{pmatrix} \boldsymbol{\theta}' & (\boldsymbol{\zeta}^+)' & (\boldsymbol{\zeta}^-)' \end{pmatrix}' \\ \mathbf{c} &:= \begin{pmatrix} \mathbf{g}' & \mathbf{L}'/n & \mathbf{0}' \end{pmatrix}' \\ \mathbf{a} &:= \begin{pmatrix} \mathbf{1}' & \mathbf{0}' & \mathbf{0}' \end{pmatrix}' \\ \mathbf{A} &:= \begin{bmatrix} \mathbf{X} & -\mathbf{I} & \mathbf{I} \end{bmatrix} \\ \mathbf{b} &:= (\boldsymbol{Y} - \boldsymbol{c}_0 \otimes \mathbf{1}_n). \end{aligned}$$

The SMSVM with l_1-norm penalty often selects a small portion of the available functional components, and the number of selected components can not exceed the sample size n. Such a property may be a limitation for many applications, especially for small n large p data. Intuitively, if one component is selected in the classifier, SMSVM with l_1 penalty intends to exclude other components that are numerically similar to the selected components. To overcome this drawback, one can additionally set an upper bound u on the coefficients of $\boldsymbol{\theta}$ or penalize $\boldsymbol{\theta}$ according to the ranks of its elements.

The former can be formulated as the $\boldsymbol{\theta}$-step with a combination of l_1-norm and l_∞-norm of $\boldsymbol{\theta}$ ($l_1 - l_\infty$ penalty) as the kernel complexity, which is given by one of the two following

equivalent forms

$$\begin{cases} \min\limits_{\boldsymbol{\zeta} \in \mathcal{R}^{nk}, \boldsymbol{\theta} \in \mathcal{R}^d} & \frac{1}{n}\sum_{j=1}^k (\mathbf{L}^j)'(\boldsymbol{\zeta}^j)^+ + \lambda_\circ \sum_{\nu=1}^d \theta_\nu \left(\sum_{j=1}^k (\boldsymbol{c}^j)' \mathcal{K}_\nu \boldsymbol{c}^j\right) + \lambda_\theta \sum_{\nu=1}^d \theta_\nu \\ \text{s.t.} & \sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu \boldsymbol{c}^j - \boldsymbol{\zeta}^j = \boldsymbol{y}^j - c_0^j \mathbf{1} \text{ for } j = 1, \ldots, k \\ & \boldsymbol{\theta} \geq \mathbf{0} \text{ and } \|\boldsymbol{\theta}\|_\infty \leq u. \end{cases} \tag{3.14}$$

and

$$\begin{cases} \min\limits_{\boldsymbol{\zeta} \in \mathcal{R}^{nk}, \boldsymbol{\theta} \in \mathcal{R}^d} & \frac{1}{n}\sum_{j=1}^k (\mathbf{L}^j)'(\boldsymbol{\zeta}^j)^+ + \lambda_\circ \sum_{\nu=1}^d \theta_\nu \left(\sum_{j=1}^k (\boldsymbol{c}^j)' \mathcal{K}_\nu \boldsymbol{c}^j\right) \\ & + \lambda_\theta \sum_{\nu=1}^d \theta_\nu + \lambda_\infty \|\boldsymbol{\theta}\|_\infty \\ \text{s.t.} & \sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu \boldsymbol{c}^j - \boldsymbol{\zeta}^j = \boldsymbol{y}^j - c_0^j \mathbf{1} \text{ for } j = 1, \ldots, k \\ & \boldsymbol{\theta} \geq \mathbf{0}. \end{cases},$$

where $\|\boldsymbol{\theta}\|_\infty := \max\{\theta_\nu : \nu = 1, \cdots, d\}$, u is the upper bound of θ's, and λ_∞ is a tuning parameter.

An example of the kernel complexity measure for the latter is OSCAR (Octagonal Shrinkage and Clustering Algorithm for Regression) penalty proposed by Bondell and Reich (2008). OSCAR is proposed to simultaneously achieve model fitting and variable selection while grouping variables into clusters. The clustering property may further enhance the interpretability of the fitted model.

Recall that $\boldsymbol{\theta}$ re-scales the functional components in the ANOVA decomposition with $\boldsymbol{\theta} = \mathbf{1}$ corresponding to the original scale. For shrinkage, one may restrict each $\theta_\nu \leq 1$ or $\|\boldsymbol{\theta}\|_\infty \leq 1$. In general, we treat the upper bound of θ, u, as yet another tuning parameter. The inequality $\|\boldsymbol{\theta}\|_\infty \leq u$ can be represented as a constraint $\theta_\nu + \eta_\nu = u$ with a slack variable η_ν subject to $\theta_\nu \geq 0$ and $\eta_\nu \geq 0$ for $\nu = 1, \cdots, d$. Then, with the notation in (3.13), the SMSVM with $l_1 - l_\infty$ penalty can be recast as

$$\begin{cases} \min\limits_{\boldsymbol{\zeta} \in \mathcal{R}^{nk}, \boldsymbol{\theta} \in \mathcal{R}^d} & \frac{1}{n}\sum_{j=1}^k (\mathbf{L}^j)'(\boldsymbol{\zeta}^j)_+ + \lambda_\circ \sum_{\nu=1}^d \theta_\nu \left(\sum_{j=1}^k (\boldsymbol{c}^j)' \mathcal{K}_\nu \boldsymbol{c}^j\right) + \lambda_\theta \|\boldsymbol{\theta}\|_1 \\ \text{s.t.} & \sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu \boldsymbol{c}^j - \boldsymbol{\zeta}^j = \boldsymbol{y}^j - c_0^j \mathbf{1} \text{ for } j = 1, \ldots, k \\ & \boldsymbol{\theta} + \boldsymbol{\eta} = u\mathbf{1} \\ & \boldsymbol{\theta} \geq \mathbf{0}, \boldsymbol{\eta} \geq \mathbf{0}. \end{cases}$$

Similarly to the θ-step for the SMSVM in the previous section, the θ-step for the new SMSVM has

$$
\begin{array}{rcccccc}
\boldsymbol{z} := & (& \boldsymbol{\theta}' & \boldsymbol{\eta}' & (\boldsymbol{\zeta}^+)' & (\boldsymbol{\zeta}^-)' &)' \\
\mathbf{c} := & (& \mathbf{g}' & \mathbf{0}' & \mathbf{L}'/n & \mathbf{0}' &)' \\
\mathbf{a} := & (& \mathbf{1}' & \mathbf{0}' & \mathbf{0}' & \mathbf{0}' &)' \\
\mathbf{A} := & \multicolumn{5}{c}{\begin{bmatrix} \mathbf{X} & \mathbf{0} & -\mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix}} \\
\mathbf{b} := & \multicolumn{5}{c}{\begin{pmatrix} \boldsymbol{Y} - \boldsymbol{c}_0 \otimes \mathbf{1}_n \\ u\mathbf{1} \end{pmatrix}}
\end{array}
$$

as its elements for LP. Note that the rank of $\mathbf{A}$ for this LP problem is $(n+d)$, which is larger than the number of kernel components d. Thus, the number of selected kernel components is not limited by the sample size n. By controlling the tuning parameter λ_θ and upper bound u, flexible functional component selection can be attained for large p small n applications.

With OSCAR penalty, the SMSVM is formulated as

$$
\left\{
\begin{array}{ll}
\min\limits_{\boldsymbol{\zeta} \in \mathcal{R}^{nk},\, \boldsymbol{\theta} \in \mathcal{R}^d} & \frac{1}{n}\sum_{j=1}^k (\mathbf{L}^j)'(\boldsymbol{\zeta}^j)_+ + \lambda_\circ \sum_{\nu=1}^d \theta_\nu \left(\sum_{j=1}^k (\boldsymbol{c}^j)' \mathcal{K}_\nu \boldsymbol{c}^j \right) \\
& +\lambda_\theta \sum\limits_{1 \le \nu \le \omega \le d} \max(\theta_\nu, \theta_\omega) \\
\text{s.t.} & \sum_{\nu=1}^d \theta_\nu \mathcal{K}_\nu \boldsymbol{c}^j - \boldsymbol{\zeta}^j = \boldsymbol{y}^j - c_0^j \mathbf{1} \text{ for } j = 1, \ldots, k \\
& \boldsymbol{\theta} \ge \mathbf{0}.
\end{array}
\right. \tag{3.15}
$$

Let $\boldsymbol{e}_i$ be the vector with its ith element equal to 1 and other elements being 0. And let $\boldsymbol{\Delta}$ denote a $d(d+1)/2 \times d$ matrix whose row vectors are in the form of $(\boldsymbol{e}_i - \boldsymbol{e}_j)$ for $1 \le i < j \le d$. Adopting the notation defined in (3.13), the θ-step SMSVM with OSCAR penalty can be rephrased as a parametric LP problem with

$$
\begin{array}{rcccccc}
\boldsymbol{z} := & (& \boldsymbol{\theta}' & (\boldsymbol{\eta}^+)' & (\boldsymbol{\eta}^-)' & (\boldsymbol{\zeta}^+)' & (\boldsymbol{\zeta}^-)' &)' \\
\mathbf{c} := & (& \mathbf{g}' & \mathbf{0}' & \mathbf{0}' & \mathbf{L}'/n & \mathbf{0}' &)' \\
\mathbf{a} := & (& d(d-1)\mathbf{1}'/2 & \mathbf{1}' & \mathbf{1}' & \mathbf{0}' & \mathbf{0}' &)' \\
\mathbf{A} := & \multicolumn{6}{c}{\begin{bmatrix} \mathbf{X} & \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{I} \\ \boldsymbol{\Delta} & \mathbf{I} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix}} \\
\mathbf{b} := & \multicolumn{6}{c}{\begin{pmatrix} \boldsymbol{Y} - \boldsymbol{c}_0 \otimes \mathbf{1}_n \\ \mathbf{0} \end{pmatrix}.}
\end{array}
$$

3.2.2 Structured Nonparametric Quantile Regression

Extending the parametric QR with l_1 penalty, Park et al. (2006) applied the idea of structured kernels to nonparametric QR. Similarly as the SMSVM in (3.12), the Structured Nonparametric QR (SNQR) finds $c_0 \in \mathcal{R}$, $\boldsymbol{c} := (c_1, \cdots, c_n)' \in \mathcal{R}^n$, and $\theta_\nu \geq 0$ for $\nu = 1, \ldots, d$ that minimize

$$\frac{\tau}{n}\mathbf{1}'\left(\mathbf{Y} - c_0\mathbf{1} - \sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\boldsymbol{c}\right)_+ + \frac{1-\tau}{n}\mathbf{1}'\left(\mathbf{Y} - c_0\mathbf{1} - \sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\boldsymbol{c}\right)_- + \lambda_\circ\boldsymbol{c}'\left(\sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\right)\boldsymbol{c} + \lambda_\theta\sum_{\nu=1}^{d}\theta_\nu.$$

Define

$$\begin{cases} \mathbf{g} := (\mathrm{g}_1, \cdots, \mathrm{g}_d)' \text{ with } \mathrm{g}_\nu := \lambda_\circ\boldsymbol{c}'\mathcal{K}_\nu\boldsymbol{c} \\ \boldsymbol{\zeta} := (\zeta_1, \cdots, \zeta_n)' \text{ with } \zeta_i = y_i - \left(c_0 + \sum_{\nu=1}^{d}\theta_\nu\boldsymbol{e}_i'\mathcal{K}_\nu\boldsymbol{c}\right) \\ \mathbf{X} := \left[\ \mathcal{K}_1\boldsymbol{c},\ \cdots,\ \mathcal{K}_d\boldsymbol{c}\ \right] \\ \boldsymbol{Y} := (y_1, \cdots, y_n)', \end{cases} \tag{3.16}$$

Thus, its θ-step can be rephrased as a parametric LP problem with

$$\begin{array}{rcccccc} \boldsymbol{z} := & (& \boldsymbol{\theta}' & (\boldsymbol{\zeta}^+)' & (\boldsymbol{\zeta}^-)' &)' \\ \mathbf{c} := & (& \mathbf{g}' & \tau\mathbf{1}'/n & (1-\tau)\mathbf{1}'/n &)' \\ \mathbf{a} := & (& \mathbf{1}' & \mathbf{0}' & \mathbf{0}' &)' \\ \mathbf{A} := & [& \mathbf{X} & \mathbf{I} & -\mathbf{I} &] \\ \mathbf{b} := & & (\boldsymbol{Y} - c_0\mathbf{1}). & & & \end{array}$$

Similarly, other variants of SNQR with $l_1 - l_\infty$ penalty or OSCAR penalty can be useful. SNQR with $l_1 - l_\infty$ penalty finds $c_0 \in \mathcal{R}$, $\boldsymbol{c} := (c_1, \cdots, c_n)' \in \mathcal{R}^n$, and $0 \leq \theta_\nu \leq u$ for $\nu = 1, \ldots, d$ that minimize

$$\frac{\tau\mathbf{1}'}{n}\left(\mathbf{Y} - c_0\mathbf{1} - \sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\boldsymbol{c}\right)_+ + \frac{(1-\tau)\mathbf{1}'}{n}\left(\mathbf{Y} - c_0\mathbf{1} - \sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\boldsymbol{c}\right)_- + \lambda_\circ\boldsymbol{c}'\left(\sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\right)\boldsymbol{c} + \lambda_\theta\sum_{\nu=1}^{d}\theta_\nu.$$

Using the terms defined in (3.16), the θ-step for SNQR with $l_1 - l_\infty$ penalty can be written as a parametric LP problem with

$$
\begin{array}{rcccccc}
\boldsymbol{z} := & (& \boldsymbol{\theta}' & \boldsymbol{\eta}' & (\boldsymbol{\zeta}^{+})' & (\boldsymbol{\zeta}^{-})' &)' \\
\mathbf{c} := & (& \mathbf{g}' & \mathbf{0}' & \tau\mathbf{1}'/n & (1-\tau)\mathbf{1}'/n &)' \\
\mathbf{a} := & (& \mathbf{1}' & \mathbf{0}' & \mathbf{0}' & \mathbf{0}' &)' \\
\mathbf{A} := & \multicolumn{5}{l}{\left[\begin{array}{ccccc} \mathbf{X} & \mathbf{0} & \mathbf{I} & -\mathbf{I} \\ \mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{array}\right]} \\
\mathbf{b} := & \multicolumn{5}{l}{\left(\begin{array}{c} \boldsymbol{Y} - c_0\mathbf{1} \\ u\mathbf{1} \end{array}\right).}
\end{array}
$$

With OSCAR penalty, the SNQR method selects and clusters the functional components by minimizing

$$
\begin{aligned}
& \frac{\tau}{n}\mathbf{1}'\left(\mathbf{Y} - c_0\mathbf{1} - \sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\boldsymbol{c}\right)_+ + \frac{(1-\tau)}{n}\mathbf{1}'\left(\mathbf{Y} - c_0\mathbf{1} - \sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\boldsymbol{c}\right)_- \\
+ \quad & \lambda_\circ\boldsymbol{c}'\left(\sum_{\nu=1}^{d}\theta_\nu\mathcal{K}_\nu\right)\boldsymbol{c} + \lambda_\theta\sum_{1\le\nu\le\omega\le d}\max(\theta_\nu,\theta_\omega).
\end{aligned}
$$

The θ-step for SNQR with OSCAR penalty is now a parametric LP problem with

$$
\begin{array}{rcccccc}
\boldsymbol{z} := & (& \boldsymbol{\theta}' & (\boldsymbol{\eta}^{+})' & (\boldsymbol{\eta}^{-})' & (\boldsymbol{\zeta}^{+})' & (\boldsymbol{\zeta}^{-})' &)' \\
\mathbf{c} := & (& \mathbf{g}' & \mathbf{0}' & \mathbf{0}' & \tau\mathbf{1}'/n & 1-\tau\mathbf{1}'/n &)' \\
\mathbf{a} := & & (d(d-1)\mathbf{1}'/2 & \mathbf{1}' & \mathbf{1}' & \mathbf{0}' & \mathbf{0}' &)' \\
\mathbf{A} := & \multicolumn{6}{l}{\left[\begin{array}{ccccc} \mathbf{X} & \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{I} \\ \boldsymbol{\Delta} & -\mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{array}\right]} \\
\mathbf{b} := & \multicolumn{6}{l}{\left(\begin{array}{c} \boldsymbol{Y} - c_0\mathbf{1} \\ \mathbf{0} \end{array}\right).}
\end{array}
$$

CHAPTER 4

COMPUTATIONAL ISSUES

As mentioned in Chapter 3, a general LP algorithm for solving a family of regularization methods is usually inefficient in terms of computation complexity. Now, we discuss the techniques that could be applied to accelerate the general tableau-simplex algorithm (see Section 2.3.2). By making the most of the structural characteristics of the associated statistical regularization methods, the simplification techniques aim to reduce the memory size that is necessary to store a tableau and therefore achieve fast computation. We also show that the current l_1-norm SVM solution-path algorithm (Zhu et al.; 2004) is a simplified simplex algorithm.

4.1 Finding Initial Basic Index Sets

The tableau simplex algorithm discussed in Chapter 2 begins with an initial basic index set $\mathcal{B}^*$ whose tableau has lexicographically positive pivot rows. This section mainly addresses how to identify proper initial basic index sets for some of the examples described in Chapter 3.

The LP problems for the four foregoing examples (l_1-norm SVM, l_1-norm QR, and $\boldsymbol{\theta}$-step for SMSVM and SQR with l_1 penalty) share a similar structure that can be exploited in computation. First of all, each of the $\mathbf{A}$ matrices has both $\mathbf{I}$ and $-\mathbf{I}$ as its sub-matrices,

and the entries of the penalty coefficient vector $\mathbf{a}$ corresponding to $\mathbf{I}$ and $-\mathbf{I}$ in $\mathbf{A}$ are zero. Thus, the ranks of $\mathbf{A}$ and $\mathbf{A}_{\mathcal{I}_\mathbf{a}}$ are M, and the initial optimal solution exists and can be easily identified. Due to the special structure of $\mathbf{A}_{\mathcal{I}_\mathbf{a}}$, it is easy to find a basic index set $\mathcal{B}^* \subset \mathcal{I}_\mathbf{a}$ for the initial LP problem in (2.7) which gives a feasible solution. For instance, a feasible basic solution can be obtained by constructing a basic index set $\mathcal{B}^*$ such that for $b_j \geq 0$, we choose the jth index from those for $\mathbf{I}$, and otherwise from the indices for $-\mathbf{I}$.

After some modification, the same idea can be applied to find initial basic index sets for QR with grouped variables, SMSVM and SQR with $l_1 - l_\infty$ penalty or OSCAR penalty, which do not have the aforementioned form of $\mathbf{A}$ originally. For modification, we augment $\mathbf{A}$ with additional columns and $\boldsymbol{z}$ with extra slack variables. Take QR with grouped variables as an example. Its modified $\boldsymbol{z}$ and $\mathbf{A}$ are

$$\begin{array}{rl} \boldsymbol{z} := & \left(\begin{array}{ccccccccc} (\boldsymbol{\xi}^+)' & (\boldsymbol{\eta}^+)' & (\boldsymbol{\xi}^-)' & (\boldsymbol{\eta}^-)' & \beta_0^+ & \beta_0^- & (\boldsymbol{\beta}^+)' & (\boldsymbol{\beta}^-)' & (\boldsymbol{\rho}^+)' \end{array} \right)' \\ \mathbf{A} := & \left[\begin{array}{ccccccccc} \mathbf{I} & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{1} & -\mathbf{1} & \mathbf{X} & -\mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{I} & -\mathbf{G} \end{array} \right]. \end{array}$$

Compared with its counterpart in Section 3.1.2, the modified LP contains extra slack variables, $\boldsymbol{\eta}^-$, and its $\mathbf{A}$ is now of the form $[\mathbf{I}, -\mathbf{I}, \mathbf{A}^*]$. Correspondingly, we augment $\mathbf{c}$ and $\mathbf{a}$ with NA entries for symbolic extension. They are given as

$$\begin{array}{rl} \mathbf{a} := & \left(\begin{array}{cccccccccc} \mathbf{0}' & \mathbf{0}' & \mathbf{0}' & \text{NA} & 0 & 0 & \mathbf{0}' & \mathbf{0}' & \mathbf{1}' \end{array} \right)' \\ \mathbf{c} := & \left(\begin{array}{cccccccccc} \mathbf{1}'/n & \mathbf{0}' & \mathbf{1}'/n & \text{NA} & 0 & 0 & \mathbf{0}' & \mathbf{0}' & \mathbf{0}' \end{array} \right)' \quad . \end{array}$$

Then, redefine $\mathcal{I}_\mathbf{a} := \{i : \mathrm{a}_i = 0 \text{ or NA}, i \in \mathcal{N}\}$, where $\mathcal{N}$ is for the extended $\boldsymbol{z}$. We require that any $i \in \mathcal{N}$ with $\mathrm{c}_i = \text{NA}$ or $\mathrm{a}_i = \text{NA}$ should not be selected as an element in the initial basic index set nor in the candidate set of entry indices for update of the optimal index set later. After augmentation of the LP elements with NA, the basic index set obtained by the initialization step would have no index corresponding to $\boldsymbol{\eta}^-$. In other words, $\boldsymbol{\eta}^-$ is forced to be $\mathbf{0}$ at the initial stage. Furthermore, as the NA labels prevent the elements in $\boldsymbol{\eta}^-$

from being selected in the following stages, the modification does not change the original problem.

It may be a concern that such a modification would substantially increase the computational complexity or consumption of computer memory. This concern can be resolved through tableau simplification discussed in next section.

To update the initial basic index set using the simplex algorithm, we need to generalize the vector operations involving **c** and **a** as follows:

$$\mathbf{c}'\boldsymbol{z} := \sum_{\{i:\mathrm{c}_i \neq \mathrm{NA}\}} \mathrm{c}_i z_i \text{ and } \mathbf{a}'\boldsymbol{z} := \sum_{\{i:\mathrm{a}_i \neq \mathrm{NA}\}} \mathrm{a}_i z_i.$$

The same principle applies wherever the operations involve a vector containing NA entries.

In practice, the initial basic index set is often derivable by setting the model parameters to zero. We next illustrate this idea with Dantzig selector (see Section 3.1.3), where the augmentation trick is not directly applicable.

With $\beta_0 = 0$ and $\boldsymbol{\beta} = \mathbf{0}$, the expression of the largest absolute covariance between x_j and residuals, $\|\mathbf{X}'(\boldsymbol{Y} - \beta_0\mathbf{1} - \mathbf{X}\boldsymbol{\beta})\|_\infty = \|\mathbf{X}'\boldsymbol{Y}\|_\infty$ is simplified via a permutation $\pi : \mathcal{N}_1 \to \mathcal{N}_1$ which sorts the absolution values of the elements in $\mathbf{X}'\boldsymbol{Y}$ in an ascending order, where $\mathcal{N}_1 := \{1, \ldots, p\}$.

Correspondingly, define

$$\begin{aligned}
\boldsymbol{Y}^* &:= \{\mathrm{diag}[\mathrm{sign}(\mathbf{X}'\boldsymbol{Y})]\mathbf{X}'\boldsymbol{Y}\}_{\pi(\mathcal{N}_1)}, \\
\mathbf{X}^* &:= \{\mathrm{diag}[\mathrm{sign}(\mathbf{X}'\boldsymbol{Y})]\mathbf{X}'\mathbf{X}\}_{\pi(\mathcal{N}_1)}, \\
\mathbf{X}_0^* &:= \{\mathrm{diag}[\mathrm{sign}(\mathbf{X}'\boldsymbol{Y})]\mathbf{X}'\mathbf{1}\}_{\pi(\mathcal{N}_1)}.
\end{aligned}$$

Then, the elements in $\boldsymbol{Y}^*$ are all nonnegative and ascendingly ordered, and the last (or pth) element of $\boldsymbol{Y}^*$ equals $\|\mathbf{X}'\boldsymbol{Y}\|_\infty$. Recast the LP problem with

$$
\begin{array}{rl}
\boldsymbol{z} := & (\ (\boldsymbol{\xi}^+)' \ (\boldsymbol{\eta}^+)' \ (\boldsymbol{\xi}^-)' \ \beta_0^+ \ \beta_0^- \ (\boldsymbol{\beta}^+)' \ (\boldsymbol{\beta}^-)' \ \rho^+)' \\
\mathbf{c} := & (\ \mathbf{0}' \ \mathbf{0}' \ \mathbf{0}' \ 0 \ 0 \ \mathbf{0}' \ \mathbf{0}' \ 1\ )' \\
\mathbf{a} := & (\ \mathbf{0}' \ \mathbf{0}' \ \mathbf{0}' \ 0 \ 0 \ \mathbf{1}' \ \mathbf{1}' \ 0\ )' \\
\mathbf{A} := & \begin{bmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{I} & \mathbf{X}_0^* & -\mathbf{X}_0^* & \mathbf{X}^* & -\mathbf{X}^* & \mathbf{0} \\ \mathbf{I} & \mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{1} \end{bmatrix} \\
\mathbf{b} := & \begin{pmatrix} \boldsymbol{Y}^* \\ \mathbf{0} \end{pmatrix}
\end{array}
$$

For $\beta_0 = 0$ and $\boldsymbol{\beta} = \mathbf{0}$, we have $\boldsymbol{\xi}^+ = \boldsymbol{Y}^*$ and $\boldsymbol{\xi}^- = \mathbf{0}$. Since $\boldsymbol{\xi}^+ + \boldsymbol{\xi}^- + \boldsymbol{\eta}^+ = \rho^+ \mathbf{1}$ leads to $\rho^+ = \xi_p^+ \geq 0$ and $\eta_i^+ \geq 0$ for $i = 1, \cdots, (p-1)$, a feasible initial basic index set, $\mathcal{B}^*$, can be specified by collecting all possible positive variables $\{\boldsymbol{\xi}^+, \eta_1^+, \cdots, \eta_{p-1}^+, \rho^+\}$, which guarantees $\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b} \geq \mathbf{0}$.

For some examples (e.g., the θ-step of SMSVM or SQR with l_1, $l_1 - l_\infty$, or OSCAR penalty), the initial basic index set $\mathcal{B}^*$ is already optimal with respect to the initial tuning parameter, and it gives a trivial initial solution. However, for many other examples (e.g., the l_1-norm SVM, l_1-norm QR, and Dantzig selector), $\mathcal{B}^*$ defined above may not be optimal. In the case, the initial optimal basic index set can be obtained by extra runs of simplex algorithm starting from $\mathcal{B}^*$. In general, the tableau-simplex algorithm in Section 2.3 can be used to find the optimal basic index set of a standard LP problem. For the l_1-norm SVM, l_1-norm QR, and Dantzig selector, the necessary modification of the algorithm for standard LP problems is that, at Step 3, the entry index $j^l \in \mathcal{N}$ could be any index j satisfying $\check{\mathrm{c}}_j^l < 0$ and $\mathrm{a}_j = 0$. For $\mathcal{B}^*$, all but the indices j for β_0^+ and β_0^- satisfy $\check{\mathrm{c}}_j^l \geq 0$. Therefore, one of the indices for β_0 will move into the basic index set first by the algorithm, and it may take some iterations to get the initial optimal index set for the regularization problems.

4.2 Computational Complexity

A tableau contains all the information on the current LP solution and the terms necessary for the next update. To discuss the computational complexity of the tableau updating

algorithm in Section 2.3.2, let $\mathbf{T}^l$ denote the tableau, an $(N+1) \times (M+2)$ matrix associated with the current optimal basic index set $\mathcal{B}^l$. For a compact statement of the updating formula, assume that the tableau is rearranged such that the pivot columns and the pivot rows precede the zeroth column and the cost row and the penalty row, respectively. For the entry index j^l and exit index i^l defined in the algorithm, $\mathbf{T}^l_{j^l}$ denotes its j^lth column vector, $\mathbf{T}^{l\prime}_{i^l_*}$ the i^l_*th row vector of $\mathbf{T}^l$, and $T_{i^l_* j^l}$ the $i^l_* j^l$th entry of $\mathbf{T}^l$. The proof of Theorem 10 in Appendix A.4 implies the following updating formula:

$$\mathbf{T}^{l+1} = \mathbf{T}^l - \frac{1}{T_{i^l_* j^l}} \left(\mathbf{T}^l_{j^l} - \boldsymbol{e}_{i^l_*} \right) \mathbf{T}^{l\prime}_{i^l_*}. \tag{4.1}$$

Therefore, the computational complexity of the tableau updating is approximately $O(MN)$ for each iteration in general.

For some examples in Chapter 3, tableau update can be further streamlined. Exploiting the structure of $\mathbf{A}$ with paired columns and fixed elements in the tableau associated with $\mathcal{B}^l$, we can compress each tableau, retaining the information about the current tableau, and update the reduced tableau instead.

The following sections discuss common structural properties of LP for the regularization problems of interest in detail, which allow substantial savings in the computational complexity of the proposed algorithm and its implementation.

4.2.1 Symmetry in A

Suppose the $\mathbf{A}$ matrix of a parametric-cost LP is of the form $[\mathbf{A}^0, -\mathbf{A}^0]$, where $\mathbf{A}^0$ is of full row rank. Correspondingly, we can split $\boldsymbol{z}$ into $[\boldsymbol{z}_\oplus, \boldsymbol{z}_\ominus]$, the cost vector $\mathbf{c}$ into $[\mathbf{c}_\oplus, \mathbf{c}_\ominus]$, and the penalty vector $\mathbf{a}$ into $[\mathbf{a}_\oplus, \mathbf{a}_\ominus]$, such that $\mathbf{A}'\boldsymbol{z} = \mathbf{A}^0\boldsymbol{z}_\oplus - \mathbf{A}^0\boldsymbol{z}_\ominus$, $\mathbf{c}'\boldsymbol{z} = \mathbf{c}'_\oplus\boldsymbol{z}_\oplus + \mathbf{c}'_\ominus\boldsymbol{z}_\ominus$, and $\mathbf{a}'\boldsymbol{z} = \mathbf{a}'_\oplus\boldsymbol{z}_\oplus + \mathbf{a}'_\ominus\boldsymbol{z}_\ominus$. Then the tableau is decomposed as

	zeroth column	pivot columns
cost row $\oplus$	$-\mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$\mathbf{c}'_{\oplus} - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0$
cost row $\ominus$	$\mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$-(\mathbf{c}'_{\ominus} + \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0)$
penalty row $\oplus$	$-\mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$\mathbf{a}'_{\oplus} - \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0$
penalty row $\ominus$	$\mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$-(\mathbf{a}'_{\ominus} + \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0)$
pivot rows	$\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{b}$	$\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0$

Since, by definition, we have $\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A} = [\mathbf{c}'_{\oplus} - \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0, \mathbf{c}'_{\ominus} + \mathbf{c}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0]$ and $\mathbf{a}' - \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A} = [\mathbf{a}'_{\oplus} - \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0, \mathbf{a}'_{\ominus} + \mathbf{a}'_{\mathcal{B}^*}\mathbf{A}_{\mathcal{B}^*}^{-1}\mathbf{A}^0]$, the step for finding the entry index is the same as the one described in Section 2.3.2. If the entry index $j^l > \frac{N}{2}$, replace the Step 4 in Section 2.3.2 with

---

Step 4^*. Let $\boldsymbol{u}^l := (u_1^l, \cdots, u_M^l)' := \mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}^0_{j^l - \frac{N}{2}}$. For each $i \in \mathcal{M}$ with $u_i^l < 0$, divide the ith pivot row (including the entry in the zeroth column) by u_i^l. And, among those rows, find the index, i_*^l, of the lexicographically largest row. Then, $i^l := B_{i_*^l}^l$ is the exit index.

Otherwise, follow the original Step 4 in Section 2.3.2. This simplification results in an algorithm whose computational complexity is half of that for the original tableau-simplex algorithm. By allowing $\mathbf{c}$ and $\mathbf{a}$ to contain NA entries, we can equivalently formulate each LP problem to have the symmetry in $\mathbf{A}$ of the form $[\mathbf{A}^0, -\mathbf{A}^0]$. Therefore, such a tableau simplification is useful even if the $\mathbf{A}$ for the original LP is partially symmetric or asymmetric.

4.2.2 Redundancy

Another feature of the tableau that enables a compact coding of its representation is that the term $\mathbf{A}_{\mathcal{B}}^{-1}\mathbf{A}$ in the tableau always contains an identity matrix as its sub-matrix, since $\mathbf{A}_{\mathcal{B}}$ is a permuted sub-matrix of $\mathbf{A}$ given $\mathcal{B}$. For the same reason, a cost row $\mathbf{c}' - \mathbf{c}^*_{\mathcal{B}}\mathbf{A}_{\mathcal{B}}^{-1}\mathbf{A}$

(or a penalty row $\mathbf{a}' - \mathbf{a}^*_{\mathcal{B}}\mathbf{A}^{-1}_{\mathcal{B}}\mathbf{A}$) contains a zero sub-vector whose elements are indexed by $\mathcal{B}$. Therefore, for given $\mathcal{B}$, a tableau (or a symmetrically simplified tableau) can be further reduced in size by removing those columns that are pre-determined.

Here, we only consider a tableau with $\mathbf{A} := [\mathbf{A}^0, -\mathbf{A}^0]$ and $\mathbf{A}^0$ being an $M \times N/2$ matrix. For simplicity, relabel the elements B_i in a basic index set $\mathcal{B}$ that are greater than $N/2$, as $N/2 - B_i$ for $i = 1, \cdots, M$, and denote the new basic index set by $\mathcal{B}^0 := \{B^0_1, \cdots, B^0_M\}$ and $\mathbf{A}^0_{\mathcal{B}^0} := [\text{sign}(B^0_i)\mathbf{A}^0_{|B^0_i|}]^M_{i=1} = \mathbf{A}_{\mathcal{B}}$. Then, $\left[\mathbf{A}^0_{\mathcal{B}^0}\right]^{-1}\mathbf{A}_0$ retains all the information in $\mathbf{A}^{-1}_{\mathcal{B}}\mathbf{A}$, but its size is $M \times (N/2 - M)$ and much smaller than the size of $\mathbf{A}^{-1}_{\mathcal{B}}\mathbf{A}$, $M \times N$.

Suppose the tableau algorithm stops in J iterations. And let p be the number of variables, d denote the number of kernel functions, and k be the number of categories. As a result of the previous simplification, the computational complexity of both l_1-norm SVM and l_1-norm QR as a whole is $O(pnJ)$ since their $M = n$ and $N/2 = n + p + 1$. The complexity for the $\boldsymbol{\theta}$-step of SMSVM with l_1 penalty is roughly $O(dnkJ)$, that with $l_1 - l_\infty$ penalty is roughly $O(2dnkJ)$, and with OSCAR penalty is roughly $O(d(nk+d(d-1)/2)J)$. The complexity for other methods can be derived in the similar way.

4.2.3 Structural Simplification

For certain problems, their associated tableau can be further simplified by utilizing the structural traits of the problems.

Here, we sketch out an example for $\boldsymbol{\theta}$-step of SMSVM with $l_1 - l_\infty$ penalty. Other examples can be similarly dealt with on a case-by-case basis. Consider the tableau with matrix $\mathbf{A}$ defined in (3.2.1):

$$\mathbf{A} := \begin{pmatrix} \mathbf{X} & \mathbf{0} & -\mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{pmatrix}.$$

Assume an $\mathbf{X}$ is a $n \times p$ matrix with $p >> n$. Its associated $\mathbf{A}$ may be too large to store. However, if we can store $\mathbf{X}$, all the information about $\mathbf{A}$ would be known. Therefore, the key to simplification of the tableau of this kind is to sufficiently utilize its structural traits to obtain the tableau based only on $\mathbf{X}$.

Given a basic index set $\mathcal{B}$, with an appropriate column and row permutation $\mathbf{A}_{\mathcal{B}}$ can be re-arranged as

$$\mathbf{A}_{\mathcal{B}} := \begin{pmatrix} \mathbf{X}_1^* & \mathbf{0} & \mathbf{0} & \mathbf{I}_1^* \\ \mathbf{X}_2^* & \mathbf{0} & \mathbf{I}^* & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{I}_2 \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{pmatrix},$$

where $\mathbf{I}^*$, $\mathbf{I}_1^*$, and $\mathbf{I}_2$ are all diagonal matrices satisfying $\mathbf{I}^*\mathbf{I}^* = \mathbf{I}$, $\mathbf{I}_1^*\mathbf{I}_2 = \mathbf{0}$, and $(\mathbf{I}_1^*\mathbf{I}_1^* + \mathbf{I}_2) = \mathbf{I}$. And $[\mathbf{X}_1^{*\prime}, \mathbf{X}_2^{*\prime}]'$ is a permuted sub-matrix of $\mathbf{X}$.

The inverse of $\mathbf{A}_{\mathcal{B}}$ is of the form

$$\mathbf{A}_{\mathcal{B}}^{-1} := \begin{pmatrix} -\mathbf{I}_2(\mathbf{I}_1^* - \mathbf{X}_1^*\mathbf{I}_2)^{-1} & \mathbf{0} & \mathbf{I} + \mathbf{I}_2(\mathbf{I}_1^* - \mathbf{X}_1^*\mathbf{I}_2)^{-1}\mathbf{X}_1^* & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{I}^*\mathbf{X}_2^*\mathbf{I}_2(\mathbf{I}_1^* - \mathbf{X}_1^*\mathbf{I}_2)^{-1} & \mathbf{I}^* & -\mathbf{I}^*\mathbf{X}_2^*\mathbf{I}_2(\mathbf{I}_1^* - \mathbf{X}_1^*\mathbf{I}_2)^{-1} & \mathbf{0} \\ (\mathbf{I}_1^* - \mathbf{X}_1^*\mathbf{I}_2)^{-1} & \mathbf{0} & -(\mathbf{I}_1^* - \mathbf{X}_1^*\mathbf{I}_2)^{-1}\mathbf{X}_1^* & \mathbf{0} \end{pmatrix}.$$

Through the simplification, technically, the entire tableau can be stored with a matrix whose size is about two times the size of $\mathbf{X}$ (about $2(nk)d$). So, the size of the matrix we actually handle can be much smaller than $(nk)d + d^2$ resulting from the simplification in Section 4.2.1 and 4.2.2.

4.3 A Closer Look at the l_1-Norm Support Vector Machine

Taking the l_1-norm SVM as a case in point, we describe the implications of the tableau-simplex algorithm for generating the solution path. Zhu et al. (2004) provide a specific path-finding algorithm for the l_1-norm SVM in the complexity-bounded formulation of (2.5) and give a careful treatment of this particular problem. We discuss the correspondence and generality of the tableau-simplex algorithm in comparison with their algorithm.

4.3.1 Status Sets

For the SVM problem with the complexity bound s, i.e. $\|\boldsymbol{\beta}\|_1 \leq s$, let $\beta_0(s)$ and $\boldsymbol{\beta}(s) := (\beta_1(s), \cdots, \beta_p(s))$ be the optimal solution at s. Zhu et al. (2004) categorize the variables and cases that are involved in the regularized LP problem as follows:

- Active set: $\mathcal{A}(s) := \{j : \beta_j(s) \neq 0,\ j = 0, 1, \ldots, p\}$
- Elbow set: $\mathcal{E}(s) := \{i : y_i\{\beta_0(s) + \mathbf{x}_i\boldsymbol{\beta}(s)\} = 1,\ i = 1, \ldots, n\}$
- Left set: $\mathcal{L}(s) := \{i : y_i\{\beta_0(s) + \mathbf{x}_i\boldsymbol{\beta}(s)\} < 1,\ i = 1, \ldots, n\}$
- Right set: $\mathcal{R}(s) := \{i : y_i\{\beta_0(s) + \mathbf{x}_i\boldsymbol{\beta}(s)\} > 1,\ i = 1, \ldots, n\}$.

Now, consider the solution $\boldsymbol{z}(s)$ given by the tableau-simplex algorithm as defined in Section 3.1.4 and the equality constraints of $\mathbf{A}\boldsymbol{z}(s) = \boldsymbol{b}$, that is,

$$\mathbf{A}\boldsymbol{z}(s) := \beta_0(s)\boldsymbol{Y} + \text{diag}(\boldsymbol{Y})\mathbf{X}\boldsymbol{\beta}(s) + \boldsymbol{\zeta}(s) = \mathbf{1}.$$

It is easy to see that for any solution $\boldsymbol{z}(s)$, its non-zero elements must be one of the following types, and hence associated with $\mathcal{A}(s)$, $\mathcal{L}(s)$, and $\mathcal{R}(s)$:

- $\beta_j^+(s) > 0$ or $\beta_j^-(s) > 0$ (but not both) $\Rightarrow j \in \mathcal{A}(s)$;
- $\zeta_i^+(s) > 0$ and $\zeta_i^-(s) = 0 \ \Rightarrow i \in \mathcal{L}(s)$;
- $\zeta_i^+(s) = 0$ and $\zeta_i^-(s) > 0 \ \Rightarrow i \in \mathcal{R}(s)$.

On the other hand, if $\zeta_i^+(s) = 0$ and $\zeta_i^-(s) = 0$, then $i \in \mathcal{E}(s)$, the elbow set.

From the perspectives of the simplex method, the algorithm in Zhu et al. (2004) can be explained as follows. Consider a non-degenerate joint solution $\boldsymbol{z}^l$ associated with its optimal basic index set $\mathcal{B}^l$. With some abuse of notation, $\boldsymbol{z}^l_{\mathcal{B}^l}$ can be expressed as

$$\text{abs}\left(\boldsymbol{\beta}_{\mathcal{A}(s^l)}(s^l), \boldsymbol{\zeta}^+_{\mathcal{U}(s^l)}(s^l), -\boldsymbol{\zeta}^-_{\mathcal{L}(s^l)}(s^l)\right)',$$

where abs[$\cdot$] is the absolute-value function, $\boldsymbol{\beta}_{\mathcal{A}(s^l)}$ consists of β_i's for $i \in \mathcal{A}(s^l)$, and $\boldsymbol{\zeta}^+_{\mathcal{U}(s^l)}$ and $\boldsymbol{\zeta}^-_{\mathcal{L}(s^l)}$ can be interpreted similarly. Because each column vector in $\mathbf{A}_{\mathcal{B}^l}$ associated with nonzero ζ's has only one nonzero entry and the entry equals either 1 or -1 (see Table 3.1.4), the invertibility of $\mathbf{A}_{\mathcal{B}^l}$ ensures that there exists a pair of permutations to order indices of the row and column vectors for $\mathbf{A}_{\mathcal{B}^l}$ such that the structure of the permuted $\mathbf{A}_{\mathcal{B}^l}$ is of the form:

$$\begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{Q} & \mathbf{I}^* \end{bmatrix}.$$

Here $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{I}^*$ are all matrices, and $\mathbf{I}^*$ is a diagonal matrix with nonzero entries being either 1 or -1.

Tentatively assume that $j \in \mathcal{N} \setminus \mathcal{B}^l$ is the entering index at $s = s^l$, such that we can find $\boldsymbol{d}^l$ by solving $\mathbf{A}_{\mathcal{B}^l}\boldsymbol{d}^l_{\mathcal{B}^l} = -\mathbf{A}_j$. Let $\mathbf{A}_j^P$ and $\mathbf{A}_j^Q$ denote the sub-vectors of $\mathbf{A}_j$ corresponding to the sub-matrices $\mathbf{P}$ and $\mathbf{Q}$ in $\mathbf{A}_{\mathcal{B}^l}$. Based on $\boldsymbol{z}^{l+1} = \boldsymbol{z}^l + \tau\boldsymbol{d}^l$, define $\Delta\boldsymbol{\beta}_{\mathcal{A}(s^l)}$ as

$$\Delta\boldsymbol{\beta}_{\mathcal{A}(s^l)} = -\mathbf{P}^{-1}\mathbf{A}_j^P,$$

then,

$$\Delta\boldsymbol{\beta}_{\mathcal{A}(s^l)} \propto \text{abs}\left(\boldsymbol{\beta}_{\mathcal{A}(s^l)}(s^{l+1})\right) - \text{abs}\left(\boldsymbol{\beta}_{\mathcal{A}(s^l)}(s^l)\right)$$

and the corresponding permuted $-\boldsymbol{d}^l_{\mathcal{B}^l} = \mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}_j$ equals

$$\begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{Q} & \mathbf{I}^* \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{A}_j^P \\ \mathbf{A}_j^Q \end{bmatrix} = \begin{bmatrix} -\Delta\boldsymbol{\beta}_{\mathcal{A}(s^l)} \\ \mathbf{I}^*(\mathbf{Q}\Delta\boldsymbol{\beta}_{\mathcal{A}(s^l)} + \mathbf{A}_j^Q) \end{bmatrix}.$$

Consequently, we obtain an alternative procedure to compute $\check{\mathrm{a}}^l_j := (\mathrm{a}_j - \mathbf{a}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}_j)$, $\check{\mathrm{c}}^l_j := (\mathrm{c}_j - \mathbf{c}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}_j)$, and the ratio $-\frac{\check{\mathrm{c}}^l_j}{\check{\mathrm{a}}^l_j}$ through $\Delta\boldsymbol{\beta}_{\mathcal{A}(s^l)}$, which can be plugged into the step 2 of the simplex method. In principle, the algorithm in Zhu et al. (2004) is the same as the procedure just being discussed.

Remark The assumption for the simplex algorithm is slightly different from the one used in Zhu et al. (2004). It requires non-degenerate $\boldsymbol{z}^l$ (i.e., $\boldsymbol{z}^l > \mathbf{0}$) at each joint solution where

their updating algorithm applies, while the simplex algorithm assumes $\boldsymbol{z}^{l+1} \neq \boldsymbol{z}^{l}$. Since $\boldsymbol{z}^{l} > \mathbf{0}$ implies $\boldsymbol{z}^{l+1} \neq \boldsymbol{z}^{l}$, the non-degeneracy assumption is more restrictive than the one for the simplex algorithm. It is appealing that the tableau-simplex algorithm needs neither of them.

4.3.2 Assumption

Suppose that the lth joint solution at $s = s^l$ is non-degenerate. Then $z_j(s^l) > 0$ if and only if $j \in \mathcal{B}^l$. This gives

$$|\mathcal{A}(s^l)| + |\mathcal{L}(s^l)| + |\mathcal{R}(s^l)| = n.$$

Since $\mathcal{E}(s) \cup \mathcal{L}(s) \cup \mathcal{R}(s) = \{1, \ldots, n\}$ for any s, the relationship that $|\mathcal{A}(s^l)| = |\mathcal{E}(s^l)|$ must hold for all the joint solutions. In fact, the equality of the cardinality of the active set and the elbow set is stated as an assumption for uniqueness of the solution in the algorithm of Zhu et al. (2004). The implicit assumption of $\boldsymbol{z}^l_{\mathcal{B}^l} > \mathbf{0}$ at each joint implies $\boldsymbol{z}^{l+1} \neq \boldsymbol{z}^{l}$, the non-degeneracy assumption for the simplex algorithm. Thus the simplex algorithm is less restrictive. In practice, the assumption that joint solutions are non-degenerate may not hold, especially when important predictors are discrete or coded categorical variables such as gender. For instance, the initial solution of the l_1-norm SVM violates the assumption in most cases, requiring a separate treatment for finding the next joint solution after initialization. In general, there could be more than one degenerate joint solutions along the solution path. This would make the tableau-simplex algorithm appealing as it does not rely on any restrictive assumption.

4.3.3 Duality in Algorithm

To move from one joint solution to the next, the simplex algorithm finds the entry index j^l. For the l_1-norm SVM, each index is associated with either β_j or ζ_i. Under the non-degeneracy assumption, the variable associated with j^l must change from zero to non-zero after the joint ($s > s^l$). Therefore, only one of the following "events" as defined in Zhu et al. (2004) can happen immediately after a joint solution:

- $\beta_j(s^l) = 0$ becomes $\beta_j(s) \neq 0$, i.e., an inactive variable becomes active;
- $\zeta_i(s^l) = 0$ becomes $\zeta_i(s) \neq 0$, i.e., an element leaves the elbow set and joins either the left set or the right set.

In conjunction with the entry index, the simplex algorithm determines the leaving index, which accompanies one of the reverse events.

The algorithm in Zhu et al. (2004), driven by the Karush-Kuhn-Tucker optimality conditions, seeks the event with the smallest "$\Delta loss/\Delta s$," in other words, the one that decreases the cost with the fastest rate. The simplex algorithm is consistent with this existing algorithm. As in (2.8), recall that the entry index j^l is chosen to minimize $(\check{\mathrm{c}}_j^l/\check{\mathrm{a}}_j^l)$ among $j \in \mathcal{N} \setminus \mathcal{B}^l$ with $\check{\mathrm{a}}_j^l > 0$. $\mathcal{N} \setminus \mathcal{B}^l$ contains those indices corresponding to $j \notin \mathcal{A}(s^l)$ or $i \in \mathcal{E}(s^l)$. Analogous to the optimal moving direction $\boldsymbol{d}^l$ in (2.9), define $\boldsymbol{v}^j = (v_1^j, \ldots, v_N^j)'$ such that

$$\boldsymbol{v}_{\mathcal{B}^l}^j = -\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_j, v_j^j = 1, \text{ and } v_i^j = 0 \text{ for } i \in \mathcal{N} \setminus (\mathcal{B}^l \cup \{j\}).$$

Then $\check{\mathrm{a}}_j^l := (\mathrm{a}_j - \mathbf{a}_{\mathcal{B}^l}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_j) = \mathbf{a}'\boldsymbol{v}^j \propto \Delta s_j$ and $\check{\mathrm{c}}_j^l := (\mathrm{c}_j - \mathbf{c}_{\mathcal{B}^l}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_j) = \mathbf{c}'\boldsymbol{v}^j \propto \Delta loss_j$. Thus, the index chosen by the simplex algorithm in (2.8) maximizes the rate of reduction in the cost, $\Delta loss/\Delta s$.

The existing l_1-norm SVM path algorithm needs to solve roughly p groups of $|\mathcal{E}|$-variate linear equation systems for each iteration. Its computational complexity can be $O(p|\mathcal{E}|^2 + p|\mathcal{L}|)$ if Sherman-Morrison updating formula is used. On the other hand, the computational complexity of the tableau-simplex algorithm is $O(pn)$ for each iteration as mentioned in Section 4.2.2. Therefore, the former could be faster if n/p is large; otherwise, the tableau-simplex algorithm is faster.

Most of the arguments in this section also apply for the comparison of the simplex algorithms with the extended solution path algorithm for the l_1-norm multi-class SVM by Wang and Shen (2006).

CHAPTER 5

NUMERICAL STUDIES

We illustrate the use of the tableau-simplex algorithm for the parametric LP in statistical applications with simulated examples and analysis of real data sets, and discuss model selection or variable selection problems therein. The computation for generating the entire solution paths was carried out with R package `lpRegPath`. Its brief descrption is given in Appendix C.

5.1 Simulation

5.1.1 l_1-norm Support Vector Machine

Consider a simple binary classification problem with linear classifiers. In this simulation, 10-dimensional independent covariates are generated from the standard normal distribution, $\mathbf{x} := (x_1, \ldots, x_{10}) \sim N(\mathbf{0}, \mathbf{I})$, and the response variable is generated via the following probit model:

$$Y = \text{sign}(\beta_0 + \mathbf{x}\boldsymbol{\beta} + \epsilon), \tag{5.1}$$

where $\epsilon \sim N(0, \sigma^2)$, and $\mathbf{x}$ and ϵ are assumed to be mutually independent. Let $\phi(\mathbf{x}) = \text{sign}(\hat{\beta}_0 + \mathbf{x}\hat{\boldsymbol{\beta}})$ denote a linear classifier with $\hat{\beta}_0$ and $\hat{\boldsymbol{\beta}}$ estimated from data. Under the probit model, the theoretical error rate of $\phi(\mathbf{x})$ can be analytically obtained as follows.

Given β_0 and $\boldsymbol{\beta}$,

$$\begin{aligned} & \Pr\left\{Y \neq \text{sign}(\hat{\beta}_0 + \mathbf{X}\hat{\boldsymbol{\beta}})\right\} \\ = \; & \Phi\left(-\hat{u}_0\right) + \mathrm{E}\left\{\Phi\left(-\frac{u_0 + \hat{\boldsymbol{u}}'\boldsymbol{u}Z}{\sqrt{(1 + 1/\text{SNR}) - (\hat{\boldsymbol{u}}'\boldsymbol{u})^2}}\right) \text{sign}(Z + \hat{u}_0)\right\}, \end{aligned}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution, Z is a standard normal random variable, $u_0 := \beta_0/\|\boldsymbol{\beta}\|_2$, $\boldsymbol{u} := \boldsymbol{\beta}/\|\boldsymbol{\beta}\|_2$, $\hat{u}_0 := \hat{\beta}_0/\|\hat{\boldsymbol{\beta}}\|_2$, and $\hat{\boldsymbol{u}} := \hat{\boldsymbol{\beta}}/\|\hat{\boldsymbol{\beta}}\|_2$. The SNR refers to the signal-to-noise ratio defined as $\text{var}(\mathbf{X}\boldsymbol{\beta})/\sigma^2$ in this case. Note that the error rate is invariant to scaling of $(\hat{\beta}_0, \hat{\boldsymbol{\beta}})$. Setting $\sigma^2 = 50$, $\beta_0 = 0$, and $\boldsymbol{\beta} = (2, 0, 2, 0, 2, 0, 0, 0, 0, 2)'$, we have the SNR of 0.32. Then, for the Bayes decision rule, in particular, we have the error rate of

$$\Pr\{Y \neq \text{sign}(\beta_0 + \mathbf{X}\boldsymbol{\beta})\} = \frac{1}{2} - \frac{1}{\pi}\arctan\sqrt{\text{SNR}} \approx 0.336, \tag{5.2}$$

which is the minimum possible value under the probit model (5.1).

Figure 5.1 shows the coefficient paths of the l_1-norm SVM indexed by $-\log(\lambda)$ (piecewise constant) and s (piecewise linear) for a simulated data set of size 400 from the model. Clearly, as $1/\lambda$ or s increases, those estimated coefficients corresponding to the non-zero β_j's ($j = 1, 3, 5$, and 10) grow large very quickly. The error rate associated with the solution at each point of the paths is theoretically available for this example, and thus the optimal value of the regularization parameter can be defined. However, in practice, λ (or s) needs to be chosen data-dependently, and this gives rise to an important class of model selection problems in general. For the feasibility of data-dependent choice of λ, we carried out cross validation and made comparison with the theoretically optimal values. The dashed lines in Figure 5.1 indicate the optimal values of λ (or s) chosen by five-fold cross validation with 0-1 loss (blue) and hinge loss (red), respectively. The discontinuity of the

0-1 loss tends to give jagged cross validation curves, which have an adverse effect on identification of the optimal value of the tuning parameter. To increase the stability, one may smooth out individual cross validated error rate curves by averaging them over different splits of the data. To that effect, cross validation was repeated 50 times with respect to the 0-1 loss and the hinge loss for averaging.

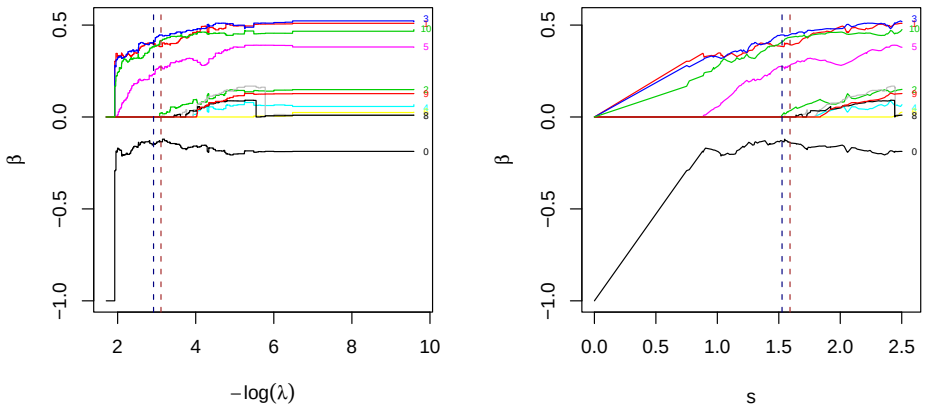

Figure 5.1: The solution paths of the l_1-norm SVM for simulated data. The numbers at the end of the paths are the indices of β's. The values of $-\log(\lambda)$ (or s) with the minimum five-fold cross validated error rate and hinge loss are indicated by the blue and red dashed lines, respectively.

Figure 5.2 displays the path of average misclassification rates from five-fold cross validation over the training data and the true error rate path for the l_1-norm SVM under the probit model. The true error rates were approximated by numerical integration up to the precision of 10^{-4}. Selection of s by cross validation with the 0-1 loss and hinge loss gave very similar results. The smallest error rate achieved by the l_1-norm SVM for this particular training data set is approximately 0.34, which is fairly close to the Bayes error rate.

We observe that the linear classifiers at both of the chosen values include the four relevant predictors.

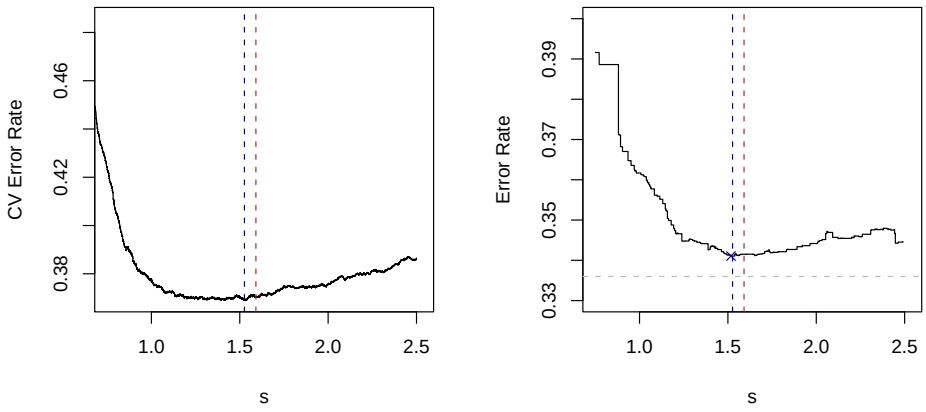

Figure 5.2: Error rate paths. The left panel shows the path of average misclassification rates from five-fold cross validation of the training data repeated 50 times, and the right panel shows the true error rate path for the l_1-norm SVM under the probit model. The vertical dashed lines are the same as in Figure 5.1. The cross on the path in the right panel pinpoints the value of s with the minimum error rate, and the gray horizontal dashed line indicates the Bayes error rate.

5.1.2 l_1-norm Quantile Regression

For another example, consider a QR problem where covariates are simulated by the same setting as in the previous example, but a continuous response variable is defined by $Y = \beta_0 + \mathbf{x}\boldsymbol{\beta} + \epsilon$. Under the assumption that $\epsilon \sim N(0, \sigma^2)$, the theoretical τth conditional quantile function is given by $m_\tau(\mathbf{x}) = \sigma\Phi^{-1}(\tau) + \beta_0 + \mathbf{x}\boldsymbol{\beta}$. Restricting to linear functions only, suppose that an estimated τth conditional quantile function is $f(\mathbf{x}) = \hat{\beta}_0 + \mathbf{x}\hat{\boldsymbol{\beta}}$. With

respect to the check function as a loss, one can calculate the risk of f, which is defined by

$$\begin{aligned} R(f;\beta_0,\boldsymbol{\beta}) &:= \mathrm{E}\left\{\tau(Y-\hat{\beta}_0-\mathbf{X}\hat{\boldsymbol{\beta}})_+ + (1-\tau)(Y-\hat{\beta}_0-\mathbf{X}\hat{\boldsymbol{\beta}})_-\right\} \\ &= \left\{\tau - \Phi\left(\frac{\hat{\beta}_0-\beta_0}{\sqrt{\sigma^2+\|\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}\|_2^2}}\right)\right\}(\beta_0-\hat{\beta}_0) \\ &\quad + \sqrt{\frac{\sigma^2+\|\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}\|_2^2}{2\pi}}\exp\left\{-\frac{(\hat{\beta}_0-\beta_0)^2}{2(\sigma^2+\|\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}\|_2^2)}\right\}. \end{aligned}$$

For each τ, the true risk of $m_\tau(\mathbf{x})$ is $(\sigma/\sqrt{2\pi})\exp\{-\Phi^{-1}(\tau)^2\}$, which represents the minimal achievable risk. Note that the maximum of the minimal risks occurs when $\tau = 0.5$ in this case, i.e., for the median, and the true conditional median function is $m_{0.5}(\mathbf{x}) = \beta_0 + \mathbf{x}\boldsymbol{\beta}$ with the risk of $\sigma/\sqrt{2\pi} \approx 2.821$.

Figure 5.3 shows the coefficient paths of the l_1-norm median regression applied to simulated data of size 400. Similarly, Figure 5.4 shows the corresponding path of the averaged 10-fold cross validated risk with respect to the check loss from 10 repetitions and its corresponding theoretical risk path. At the chosen value of λ by cross validation, the four correct predictors and one extra predictor have non-zero coefficients, and the theoretical risk of the selected model is not far from the minimal risk denoted by the horizontal reference line. We note that the complete risk path levels off roughly after $-\log\lambda = 5$, implying that moderately regularized models are almost as good as the full model of the unconstrained solution. In terms of the risk, the realized benefit of penalization appears little compared to the previous classification example.

5.1.3 Median Regression with Grouped Variables

The median regression with G-penalty is designed to implement simultaneous group selection and model estimation. It is illustrated with a toy example. Data are simulated

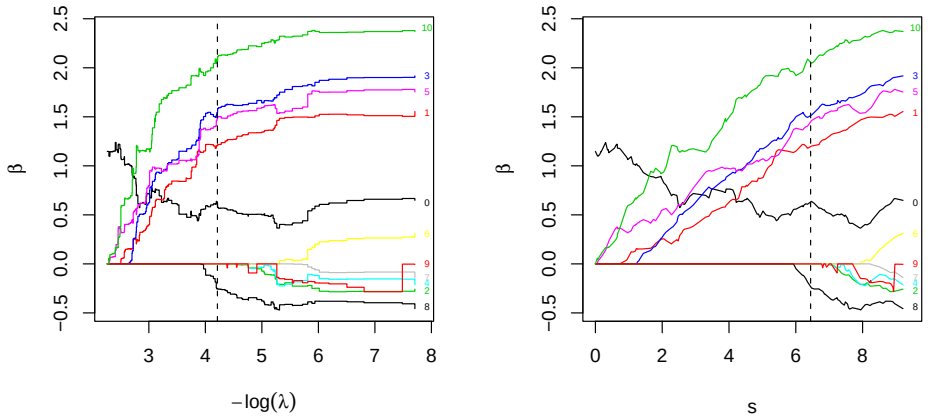

Figure 5.3: The solution paths of the l_1-norm median regression for simulated data. The dashed lines specify the value of the regularization parameter with the minimum of 10-fold cross validated risk with respect to the check loss over the training data.

from a linear model:

$$y_t = \sum_{j=1}^{J} \sum_{i}^{p_j} x_{tij}\beta_{ij} + \epsilon_t \text{ for } t = 1, \cdots, 1000,$$

where $x_{tij} \overset{i.i.d.}{\sim} N(0,1)$ and $\epsilon_t \overset{i.i.d.}{\sim} N(0,50)$ for $j = 1, \cdots, J$ and $i = 1, \cdots, p_j$.

With the number of groups $J = 3$, the true $\boldsymbol{\beta}$ is set to be $((2,3,2,0),(0,0,0),(-3,2,-2))'$ and its elements are indexed by ((11,12,13,14), (21,22,23), (31,32,33)). And the intercept β_0 is set to 0.

Figure 5.5 shows solution paths for median regression on the simulated dataset. The computation took 19.50 seconds including user time 17.54 seconds and system time 1.80 seconds on a Pentium M 2GHz personal computer. From the figure, we can see that variable group 1 and variable group 3 stand out at the early stage of the solution path as expected. To make it clearer, the plot in the right panel shows the same solution path but in terms of the absolute values of the coefficients. It illustrates the general characteristic of group

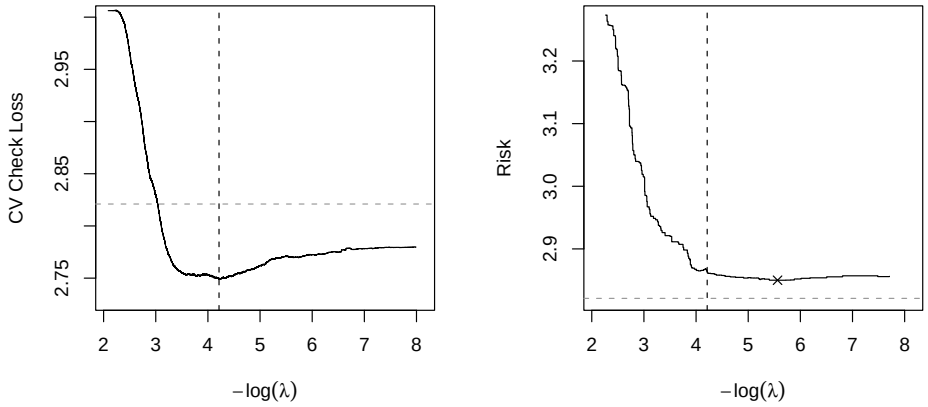

Figure 5.4: Estimated risk path and its theoretical counterpart. The left panel shows the averaged risk of the l_1-norm median regression function from 10-fold cross validation over the training data repeated 10 times. The right panel shows the theoretical risks of the regression functions corresponding to the coefficient paths with the minimum indicated by the cross. The vertical dashed lines locate the value of $-\log(\lambda)$ with the minimum cross validated risk, and the horizontal dashed lines indicate the minimal theoretical risk.

selection that the coefficients in each group form a stem in the begining and then branch out later for a better fit to the data. With only 9 variables, overfitting is not an issue for this simulation study. We can also see that, at the end of the solution path, all the estimated coefficients are fairly close to their respective true values.

5.1.4 Structured Support Vector Machine

Consider binary classification problems where the optimal classification rules are non-linear in covariates. Data are simulated by the following scheme:

$$Y = \text{sign}\left[\sum_{i=1}^{p_1}\left(e^{-X_i} - e^{X_i} - .5\sin(4\pi X_i)\right) + \sum_{j=p_1+1}^{p_1+p_2}\left(e^{2X_j} - e^{-2X_j}\right) + \epsilon\right],$$

$$\text{where} \quad X_1, \cdots, X_p \overset{i.i.d.}{\sim} Uniform[-.5, .5] \text{ and } \epsilon \sim N(0,1). \tag{5.3}$$

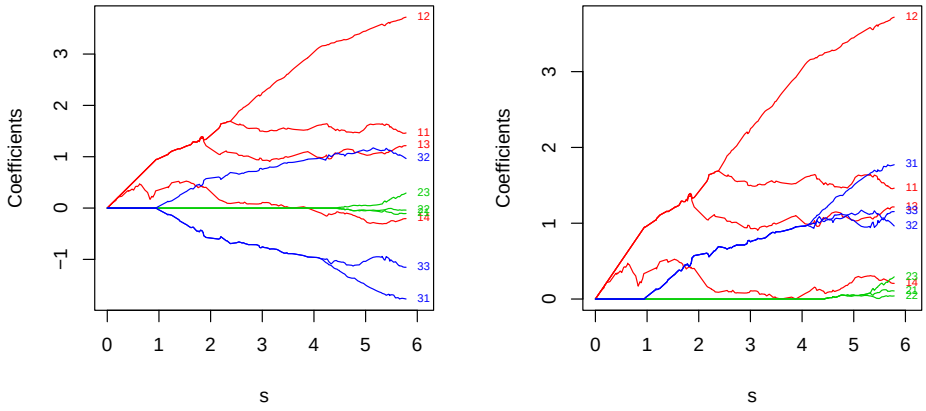

Figure 5.5: Solution paths of coefficients (left) and absolute coefficients (right) for median regression with G-penalty.

Under this setting, the variance of $\left(e^{-X_i} - e^{X_i} - .5\sin(4\pi X_i)\right)$ is about .3838, and that for $\left(e^{2X_j} - e^{-2X_j}\right)$ is about 3.9228.

Among the p covariates, the class labels depend on two groups of covariates only (p_1 variables in the first group and p_2 variables in the second group). Each group shares the specified functional form. The Structured SVM (SSVM) was applied to the simulated data. Our R program for SSVM is based on the software, SMSVM v1.2.1, that can be downloaded at `http://www.stat.osu.edu/~yklee/software.html`. By treating all the covariates equally, the initial kernel was set to be

$$K(\mathbf{x}, \mathbf{x}^*) = \sum_{\nu=1}^{p} \theta_\nu K_\nu(x_\nu, x_\nu^*),$$

where $\mathbf{x}^* := (x_1^*, \cdots, x_p^*)'$, $K_\nu(\cdot)$ is the univariate spline kernel function, and $\theta_\nu = 1$ for $\nu = 1, \cdots, p$. The ordinary SVM classifier was calculated first. In $\boldsymbol{c}$-step, the tuning parameter $\lambda_\circ$ was chosen by five-fold cross validation (CV) with four repetitions using the

hinge loss. Then, with the updated $\boldsymbol{c}$, $\boldsymbol{\theta}$-step was carried out to choose the optimal $\boldsymbol{\theta}$ also by five-fold CV with four repetitions. The entire procedure of data simulation and functional component pursuit was repeated 100 times.

Small p Large n Case

With $p_1 = 1, p_2 = 1, p = 6$, and $n = 200$, the Bayes error rate for this simulation setting is about 0.1803. Note that the class label in (5.3) depends on the functional components of the first two covariates only. So θ_1 and θ_2 are expected to be selected much more frequently than other θ_ν's. Figure 5.6 illustrates a set of simulated data with the first two covariates, class labels, and the boundary of the Bayes decision rule.

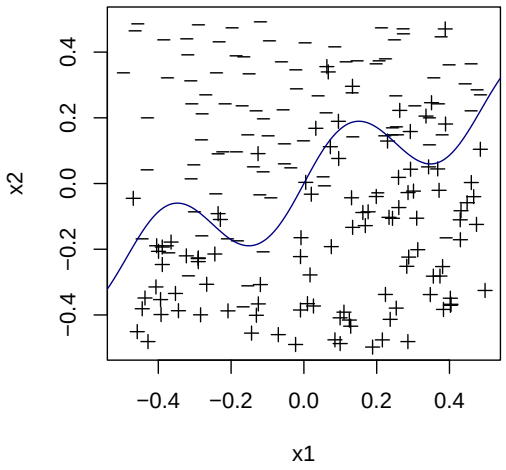

Figure 5.6: Simulated data and the Bayes classification boundary.

Table 5.1 summarizes the error rates of the structured SVM for the simulation study. Table 5.2 shows the relative selection frequency of the functional components. Here, the

	Mean	S.D.	Median
$\boldsymbol{c}$-step	.1983	.0012	.1967
$\boldsymbol{\theta}$-step (l_1 penalty)	.2042	.0030	.1967
$\boldsymbol{\theta}$-step ($l_1 - l_\infty$ penalty)	.2032	.0029	.1967
$\boldsymbol{\theta}$-step (OSCAR penalty)	.2031	.0029	.1967

Table 5.1: Mean, standard deviation, and median of error rates

Method	X_1	X_2	X_3, X_4, X_5, X_6
SSVM with l_1 penalty	.9620	.9873	.2405
SSVM with $l_1 - l_\infty$ penalty	.9620	.9873	.2088
SSVM with OSCAR penalty	.9620	.9873	.3323

Table 5.2: Relative selection frequencies of functional components

relative selection frequency is defined as the proportion of selection of each functional component. Since X_3 through X_6 are irrelevant to Y, their selection frequencies are aggregated into a single value. From the tables, we can see that the $\boldsymbol{\theta}$-step has the effect of keeping similar test error rates as $\boldsymbol{c}$-step, but correctly identifying the important components and excluding the majority of the irrelevant components. This result confirms the common observation that even as a black-box like prediction algorithm, the regular SVM attains high prediction accuracy, and the main utility of the $\boldsymbol{\theta}$-step is to improve its interpretability.

Figure 5.7 displays typical solution paths of $\boldsymbol{\theta}$ and their associated risk curves. Figure 5.8 shows the boxplots of optimal θ's for SSVM with three different kinds of kernel penalty. The solution paths with l_1 and OSCAR penalty are similar. Especially at the early stage, they are almost identical. As pointed out by Bondell and Reich (2008), the OSCAR penalty often yields merged pieces in its solution paths. For example, θ_3 (in blue) and θ_4 (in sky

blue) follow the same trajectory at their early stage, while their paths are separate with quite large l_1 penalty. l_1 penalty is sometimes too greedy in selection. When two or more components have a similar kernel matrix derived from training data, the $\boldsymbol{\theta}$-step with l_1 penalty tends to select only one of them. The solution path with the additional upper bound of 3 on θ has selected the correct functional components as well as the SSVM with l_1 penalty only. However, a too small upper bound could cause faulty selection of functional components. The risk curves in Figure 5.7 also suggest that the hinge loss may be more reliable than the 0-1 loss for functional component pursuit, since its risk curves are usually smoother than those for 0-1 loss.

Large p Small n Case To assess the performance of structured SVM on large p small n data, this simulation sets $p_1 = 10$, $p_2 = 10$, $p = 100$, and $n = 50$. The selection frequency associated with the first group of variables, $X_1, \cdots, X_{10}$, and the second group of variables, $X_{11}, \cdots, X_{20}$, is expected to be significantly larger than that for the rest of the variables irrelevant to Y (call them Group Three). For large p case, the structured SVM with OSCAR penalty is not applicable due to its large computational complexity, so is not pursued here. Figure 5.9 shows typical solution paths of $\boldsymbol{\theta}$, the number of selected components, and the risk curves. We can see that the number of selected components is roughly linearly increasing in the l_1 penalty. For SSVM with l_1 penalty, the number of selected components can not exceed the observation size, 50; while, for SSVM with l_1 penalty and an upper bound 4, the observation size is not any more a hard limitation on the number of selected components.

Table 5.1.4 shows that functional components in Group Two have the highest selection frequency (about .3810), and the selection frequency for the components in Group Two is much higher than that for the components in Group Three. However, due to the small

Method	$X_1, \cdots, X_{10}$	$X_{11}, \cdots, X_{20}$	$X_{21}, \cdots, X_{100}$
SSVM with l_1 penalty	.2010	.3810	.1165
SSVM with $l_1 - l_\infty$ penalty	.3080	.5290	.2526

Table 5.3: Relative selection frequencies of functional components

sample size, the contrast between the relevant variables and the rest in terms of the selection frequency is not as strong as in the small p and large n case. Table 5.1.4 also shows that setting upper bound for the recalibration parameter of functional components (i.e. $l_1 - l_\infty$ penalty) significantly increases the selection frequencies for all the functional components.

In summary, this simulation study exemplifies that the proposed algorithms are useful for functional component pursuit with large p small n data.

5.2 Real Data Analysis

5.2.1 Income Data Analysis

For a real application, we take the income data in Hastie et al. (2001), which are extracted from a marketing database for a survey conducted in the Bay area (1987). The data set is available at `http://www-stat.stanford.edu/~tibs/ElemStatLearn/`. It consists of 14 demographic attributes with a mixture of categorical and continuous variables, which include age, gender, education, occupation, marital status, householder status (own home/rent/other), and annual income among others. The main goal of the analysis is to predict the annual income of the household (or personal income if single) from the other 13 demographics attributes.

The original response of the annual income takes one of the following income brackets: < 10, [10, 15), [15, 20), [20, 25), [25, 30), [30, 40), [40, 50), [50, 75), and ≥ 75 in the unit

of \$1,000. For simplification, we created a proxy numerical response by converting each bracket into its middle value except the first and the last ones, which were mapped to some reasonable values albeit arbitrary. Removing the records with missing values yields a total of 6,876 records. Because of the granularity in the response, the normal-theory regression would not be appropriate. As an alternative, we considered median regression, in particular, the l_1 norm median regression for simultaneous variable selection and prediction. In the analysis, each categorical variable with k categories was coded by (k-1) 0-1 dummy variables with the majority category treated as the baseline. Some genuinely numerical but bracketed predictors such as age were also coded similarly as the response. As a result, 35 variables were generated from the 13 original variables.

The data set was split into a training set of 2,000 observations and a test set of 4,876 for evaluation. All the predictors were centered to zero and scaled to have the squared norm equal to the training sample size before fitting models. Inspection of the marginal associations of the original attributes with the response necessitated inclusion of a quadratic term for age. We then considered linear median regression with the main effect terms only (35 variables plus the quadratic term) and with two-way interaction terms as well as the main effects. There are potentially 531 two-way interaction terms by taking the product of each pair of the normalized main effect terms from different attributes. In an attempt to exclude nearly constant terms, we screened out any product with the relative frequency of its mode 90% or above. This resulted in addition of 69 two-way interactions to the main effects model. Note that the interaction terms were put in the partial two-way interaction model without further centering and normalization for the clarity of the model. Approximately three quarters of the interactions had their norms within 10% difference from that of the main effects. Figure 5.10 shows the coefficient paths of the main effects model in the left

panel and the partial two-way interaction model in the right panel for the training data set. The coefficients of the dummy variables grouped for each categorical variable are of the same color. In both models, several variables emerge at early stages as important predictors of the household income and remain important throughout the paths. Among those, the factors positively associated with household income are home ownership (in dark blue relative to renting), education (in brown), dual income due to marriage (in purple relative to 'not married'), age (in skyblue), and being male (in light green). Marital status and occupation are also strong predictors. As opposed to those positive factors, being single or divorced (in red relative to 'married') and being a student, clerical worker, retired or unemployed (in green relative to professionals/managers) are negatively associated with the income. So is the quadratic term of age in blue as expected. In general, it would be too simplistic to assume that the demographic factors in the data affect the household income additively. Truthful models would need to take into account some high order interactions, reflecting the socio-economic fabric of the household income structure. Some of the two-way interactions worthwhile to mention are 'dual income $*$ home ownership', 'home ownership $*$ education', and 'married but no dual income $*$ education' with positive coefficients, and 'single $*$ education' and 'home ownership $*$ age' with negative coefficients.

As in the QR simulation, we chose optimal values of s by cross validation with the absolute deviation loss. Five-fold cross validation was repeated 5 times for different splits of the training data, and the resulting risks were averaged. Figure 5.11 displays the paths of actual risks over the test set for the main effect models (left) and for the partial two-way interaction models (right). The dashed lines indicate the minimizers s of the averaged risks and the solid lines those of the actual risks over the test set. Cross validation seems to give a reasonable choice of s in terms of risk. Note that there is a range of optimal values with

about the same risk in both panels, which suggests that one may as well average the models in the range. A notable difference between the risk paths is the amount of regularization desired to attain the minimum risk in comparison with the full models. That is, regularization improves the two-way interaction models much more than the main effects models. Moreover, the selected two-way interaction model has a smaller risk over the test set than the main effect model in accordance with our understanding of the data. On the basis of evaluation over the test data, 95% confidence intervals of the true risk associated with the main effects and the two-way interaction models selected by the CV criteria are 7.799 ± 0.238 and 7.653 ± 0.236, respectively. In particular, a 95% confidence interval of the risk difference of the main effects model from the two-way model is given by 0.146 ± 0.0585, which indicates that the latter improves the former significantly in terms of the risk.

5.2.2 Breast Cancer Data Analysis

Biologically, it is speculated that a malignant disease can cause characteristic changes in certain gene expressions of patients' blood. Blood samples are much easier to collect than traditional clinical samples (diseased tissues and cells) because they do not rely on the prior knowledge of disease status. If the hypothesis is true, one may be able to design an inexpensive prognosis for breast cancer by testing peripheral blood cells of women. Aiming at the goal, Sharma et al. (2005) collected data to identify important genes that are numerically relevant to breast cancer. The dataset contains batch-adjusted expression levels (mRNA) of 1368 genes from 60 blood samples of 56 women. Among the 56 women, 24 women are breast cancer patients, and the rest were previously suspected to have breast cancer but no sign of the disease. A preliminary gene-by-gene examination indicates that there exist correlations between breast cancer and gene expressions. For example, the

scatter plots in Figure 5.12 suggest that the lower the expression levels of gene 56 or 801 there is the higher possibility of the breast cancer occurrence. Figure 5.13 shows that certain combinations of gene expressions may provide fairly accurate information about breast cancer status.

Sharma et al. (2005) applied the nearest-shrunken-centroid method (Tibshirani et al.; 2002) to the dataset and identified 37 genes. For comparison, we applied l_1-norm SVM and structured SVM as alternatives. In the original paper, the external leave-one-out CV (Ambroise and McLachlan; 2002) was used for unbiased evaluation of the error rate. It means that each time one observation was left out and the error was obtained on the observation for the nearest-shrunken-centroid method with its threshold chosen by internal leave-one-out CV. Instead, we used external 6-fold CV for all the methods, that is, we split 60 observations into a training set of 50 and a test set of 10 and applied internal 5-fold CV or its variant for choice of tuning parameters.

For structured SVM method, we specified the kernel function for each of the gene expressions to be a spline kernel with linear and smooth parts combined, and set the initial kernel of the overall feature space to be the sum of all the individual kernel functions. In $\boldsymbol{c}$-step, a regular SVM method was applied for a sequence of tuning parameter values of λ. The value with the minimum of 5-fold CV error rate averaged over 20 different splits of the training set was chosen and passed to the next $\boldsymbol{\theta}$-step. Then, in $\boldsymbol{\theta}$-step, we generated the paths of the kernel coefficients and selected the optimal λ_θ via 5-fold CV procedure with 20 different splits. For l_1-norm linear SVM, we considered two classifiers: one with standardized gene expressions as covariates and the other with additional square terms. The CV was done similarly as in the $\boldsymbol{\theta}$-step of structured SVM.

To illustrate the utility of solution paths, with all the 60 observations, we generated the coefficient paths of the l_1-norm SVM and structured SVM as given in Figure 5.14 and Figure 5.18 respectively. In addition, Figure 5.15 shows the corresponding 5-fold CV error rate curves for l_1-norm SVM; Figure 5.17 and Figure 5.19 are the counterparts for $\boldsymbol{c}$-step and $\boldsymbol{\theta}$-step of structured SVM. We notice that the estimated error rate curves in Figures 5.15, 5.17, and 5.19 all have a valley in the middle. Roughly speaking, to the left side of the valley over-penalization occurs, which causes insufficient utilization of the data. On the other hand, to the right side of the valley over-fitting occurs, which results in unsatisfactory prediction performance. Therefore, an advantage of having such error rate curves is that we can select models or classifiers balancing data fit and penalty.

For comparison of the four methods, we carried out external 6-fold CV and Table 5.4 lists the estimated error rates ($\hat{p}$) and their standard errors ($\sqrt{\hat{p}(1-\hat{p})/60}$). Among the methods, structured SVM gives the smallest error rate followed by the nearest-shrunken-centroid method. We also see that the error rate of l_1-norm SVM with linear terms only (about 0.197) is much smaller than that with additional square terms (about 0.279). Because the latter contains more covariates than the former, one may expect that the classifier with both linear and square terms would outperform the one with linear terms only. However, having more covariates also increases the chance to wrongly include some covariates, and as a consequence the prediction accuracy may drop. The results show that at least for the data, adding those square terms degrades the accuracy.

Other than error rates, it is of statistical interest to compare the variability of the estimated coefficients and selection frequencies of genes for each method. Here, we give a result for assessment of the variability of the l_1-norm SVM with linear terms only. Similar analysis can be done for other methods, which is to be carried out in the future.

To emulate the setting for external 6-fold CV in terms of the training set size yet to create more replicates, we randomly selected 50 observations out of the total 60 and applied the l_1-norm SVM to the subset. The optimal classifier was selected by using 5-fold CV with 10 different splits. And the entire procedure was repeated 20 times, yielding 20 replicates of l_1-norm SVM classifiers. Each classifier contained about 20 genes with nonzero coefficients. Figure 5.16 shows the boxplots of the coefficients for the genes selected at least 5 times. They are ordered by the absolute values of their median coefficients. The selection frequencies are also given on the left side. None of the boxplots contains both positive and negative coefficients. It means that for each gene in the figure, its estimated non-zero coefficients are consistent in the sign. In addition, we investigated consistency in ranks of the coefficients for each pair of the 20 replicated classifiers by computing the Kendall's τ (also called Kendall's correlation coefficient proposed by Kendall (1938)). As mentioned before, only 20 genes out of 1368 had nonzero coefficients on average, and this sparsity would lead to too many ties in the rank. To reduce the effect of the ties on the Kendall's τ, for each pair, we removed the genes with zero coefficients in both classifiers, and computed the Kendall's τ only based on the ranks of the rest. As it is often the case that the importance of a predictor is measured by the absolute value of its estimated coefficient, the Kendall's τ is also computed for the absolute values of the coefficients. Figure 5.20 shows the histogram of the Kendall's τ's for coefficients of all pairs of 20 replicates in the upper panel and that for absolute coefficients in the lower panel, respectively. The former is centered at .33 while the latter is roughly centered at 0. Expectedly, Kendall's τ' for coefficients is higher than that for the absolute values on average. However, the result that the correlation in the importance of the covariates (excluding those with zero coefficient) is nearly zero from replicate to replicate requires some explanation. It is conjectured that due

Method	Setting for internal CV	Error rate	SE
Nearest Shrunken Centroid	5-fold	0.186	.050
l_1-norm SVM with linear terms	5-fold with 20 different splits	0.197	.051
l_1-norm SVM with additional square term	5-fold with 20 different splits	0.279	.058
Structured SVM	5-fold with 20 different splits	0.170	.048

Table 5.4: Comparison of error rate for four classification methods

to the large p small n data structure, there usually exist a large number of covariates that are equally correlated with the responses, and this would make the Kendall's correlation for the absolute coefficients close to zero on average.

In summary, similarly to the results in Sharma et al. (2005), our data analysis also suggests that breast cancer even during early stages of disease development affects the expression pattern of certain genes. By identifying these genes and analyzing their expression pattern, it is possible to develop a blood-based gene expression test for early detection of breast cancer.

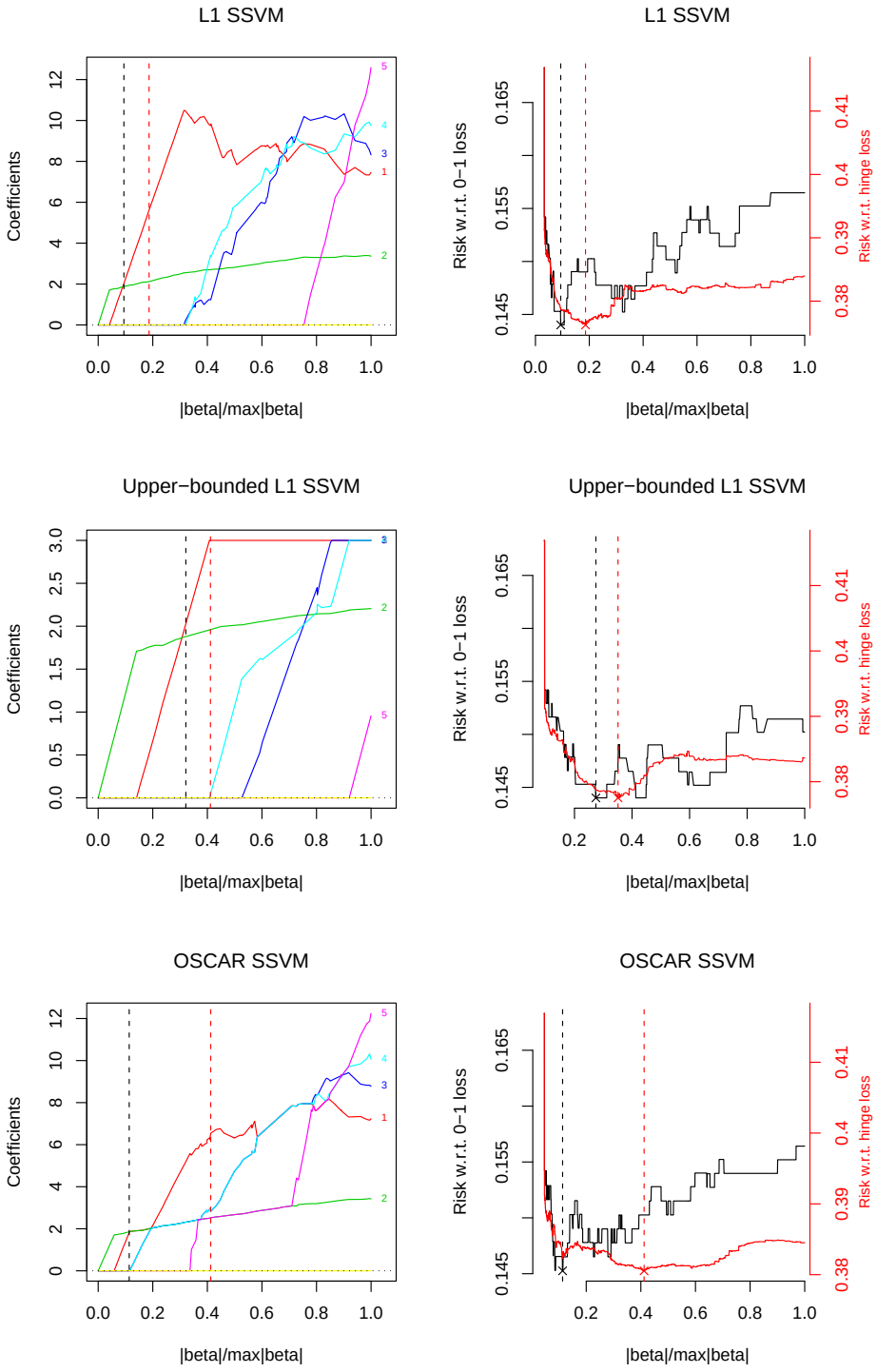

Figure 5.7: Solution paths of the recalibration parameter θ and empirical risk curves for Structured SVM. The red and black dashed lines locate $\boldsymbol{\theta}$'s of the minimum empirical risks with respect to the hinge and 0-1 loss functions.

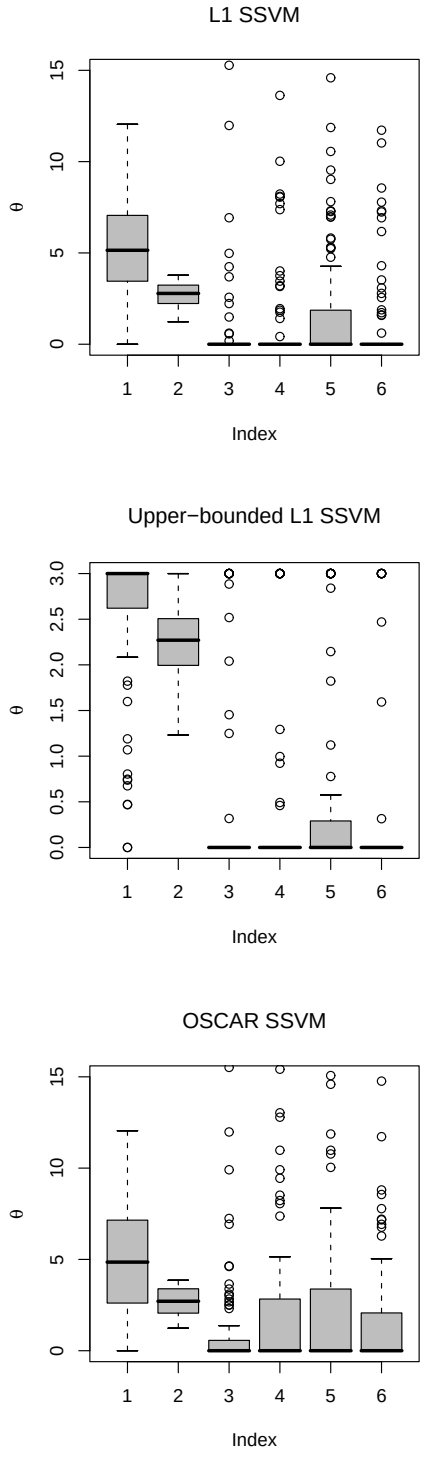

Figure 5.8: Box-plots of the estimated recalibration parameter θ for Structured SVM with three different kinds of penalty.

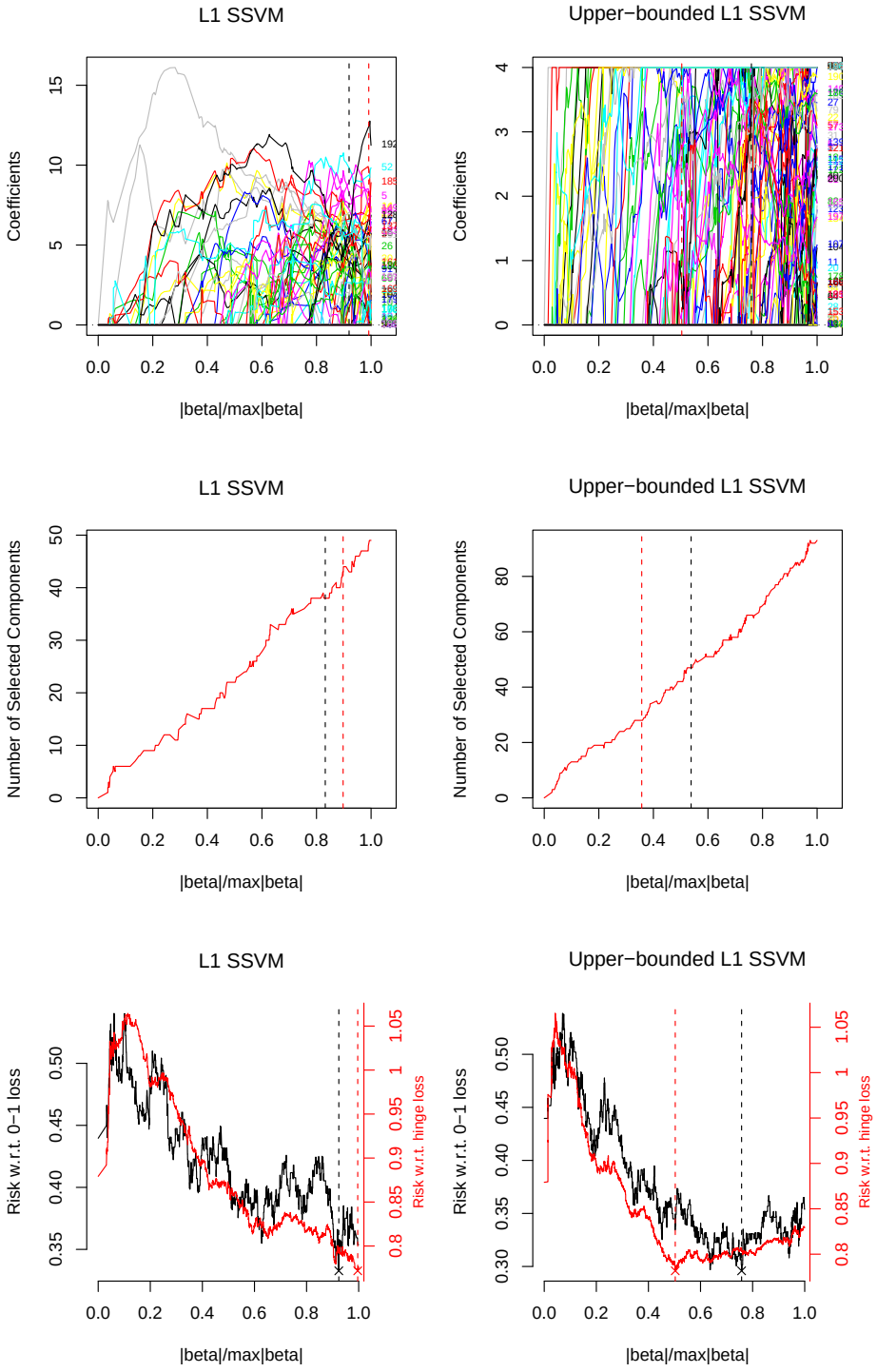

Figure 5.9: Solution paths of the recalibration parameter θ and empirical risk curves for Structured SVM on a large p small n dataset. The red and black dashed lines locate $\boldsymbol{\theta}$'s of the minimum empirical risks with respect to the hinge and 0-1 loss functions.

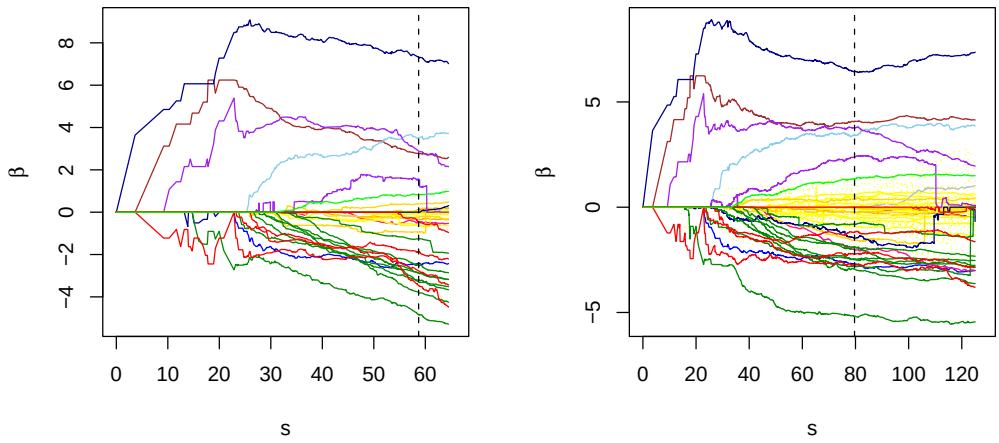

Figure 5.10: The coefficient paths of the main effects model (left) and the partial two-way interaction model (right) for the income data. For each categorical variable, the coefficients of the corresponding dummy variables are of the same color except for the two-way interactions that are all in yellow in the right panel. The dashed lines indicate the models chosen by five-fold cross validation with the absolute deviation loss.

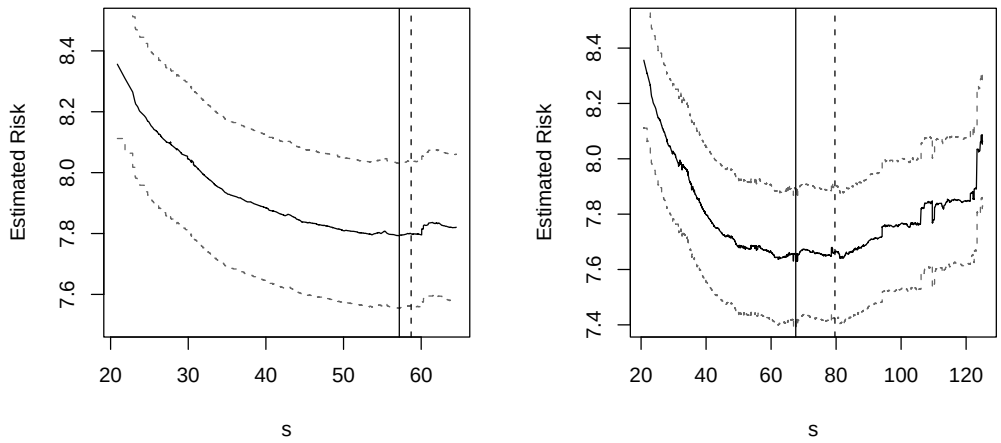

Figure 5.11: The paths of actual risks estimated over the test set and their 95% confidence intervals for the main effect models (left) and the partial two-way interaction models (right) with the minimum denoted by the solid lines. The dashed vertical lines indicate the values of s minimizing the averaged risks from five-fold cross validation repeated five times.

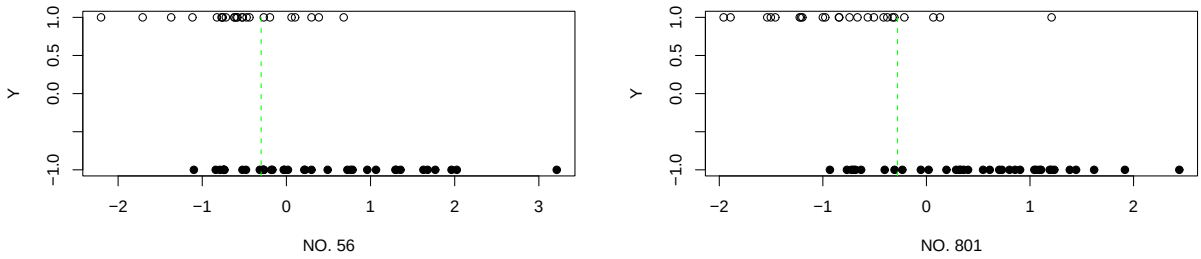

Figure 5.12: Scatter plots of gene expression and breast cancer. Open circles are for breast cancer patients, and solid circles are for normal subjects. The green dashed lines indicate the optimal separation thresholds with minimum error rates.

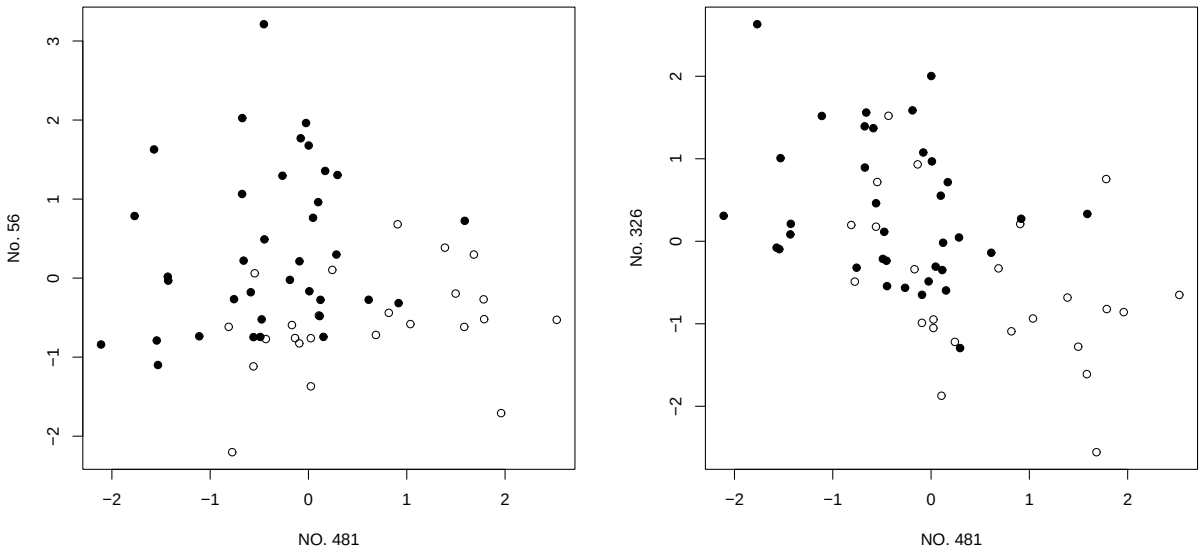

Figure 5.13: Scatter plots of expression levels of gene 56, 326, and 481. Open circles indicate breast cancer patients, and solid circles indicate normal subjects.

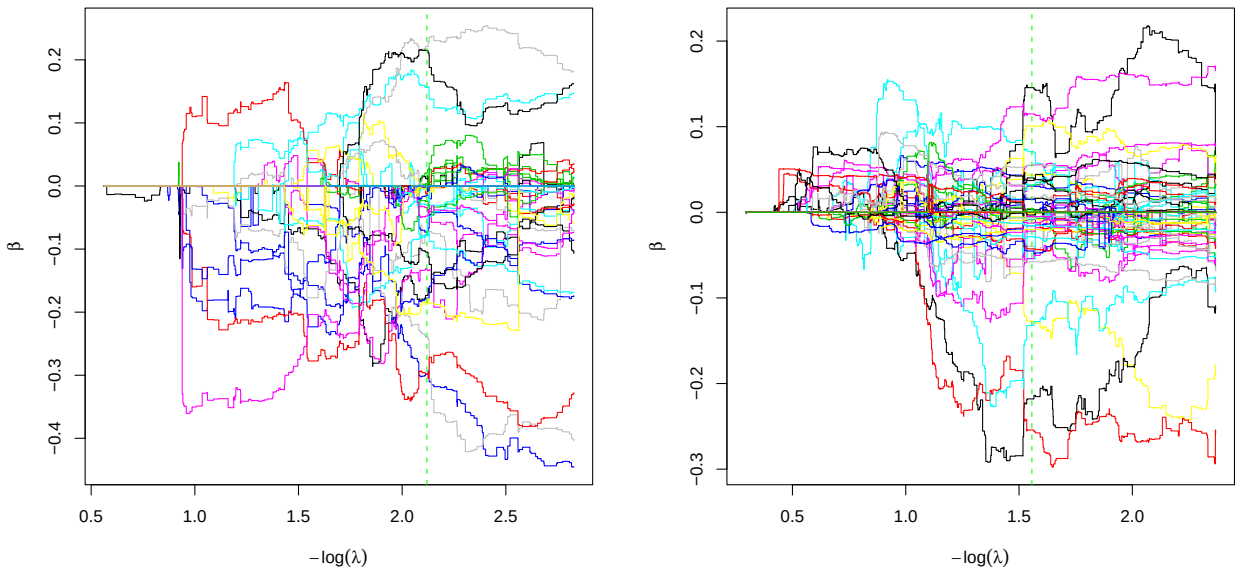

Figure 5.14: Coefficient paths for l_1-norm SVM. The left panel is for linear terms only, and the right panel is for additional square terms. The colored numbers are the associated gene indices. Each of the green dashed line indicates the classifier with with minimum 5-fold cross validation error rate.

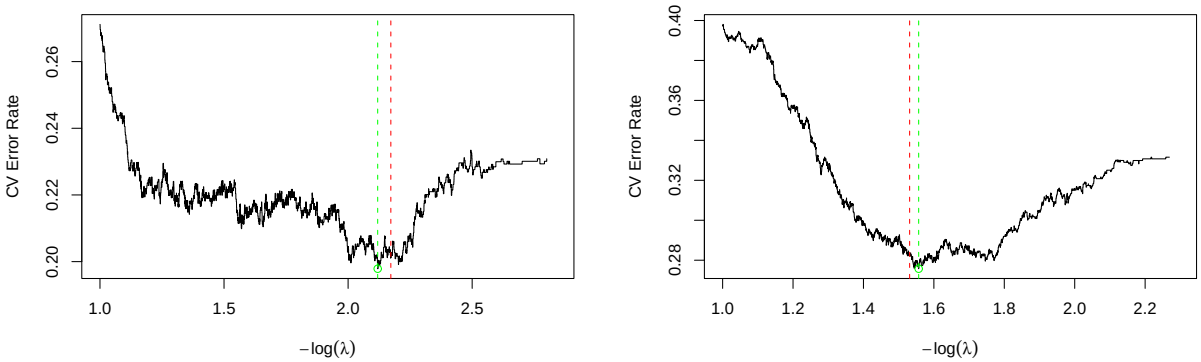

Figure 5.15: Error rate paths for l_1-norm SVM. The left panel is for linear terms only, and the right panel is for additional square terms. The error rates are computed by taking the average of 5-fold cross validated error rates over 20 different splits. Each of the green dashed line indicates the classifier with the minimum error rate along the path, and the red dashed line indicates that with the minimum risk in the hinge loss.

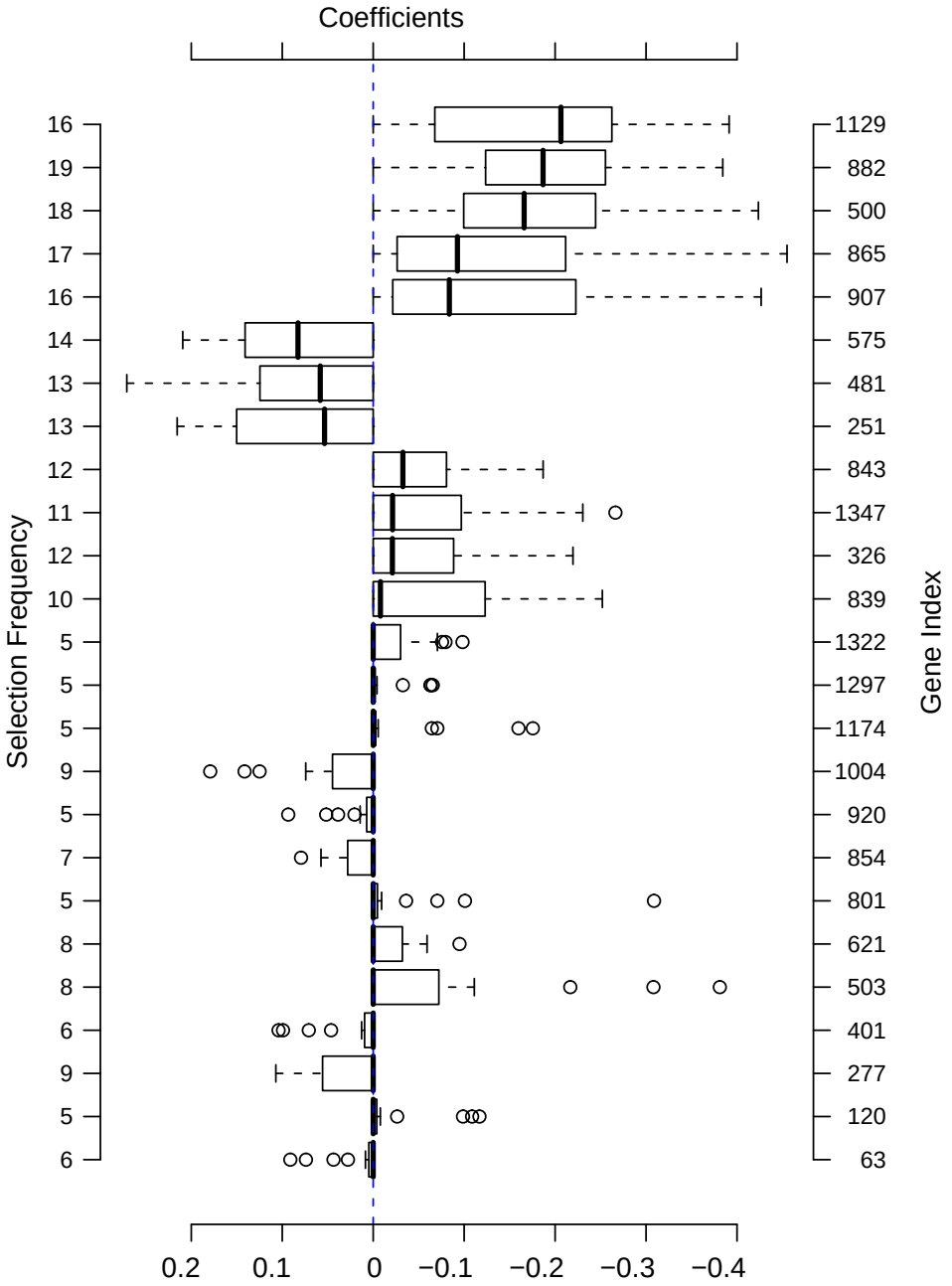

Figure 5.16: Boxplots of the coefficients of a subset of genes for l_1-norm linear SVM in 20 replicates. They are ordered by the absolute value of the median coefficient. The selection frequency of each gene is given in the left margin and the corresponding gene index in the right margin.

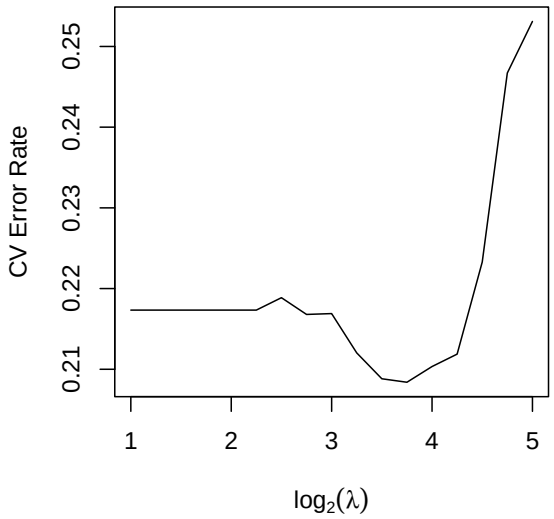

Figure 5.17: Path of 5-fold cross validated error rates for the first $\boldsymbol{c}$-step.

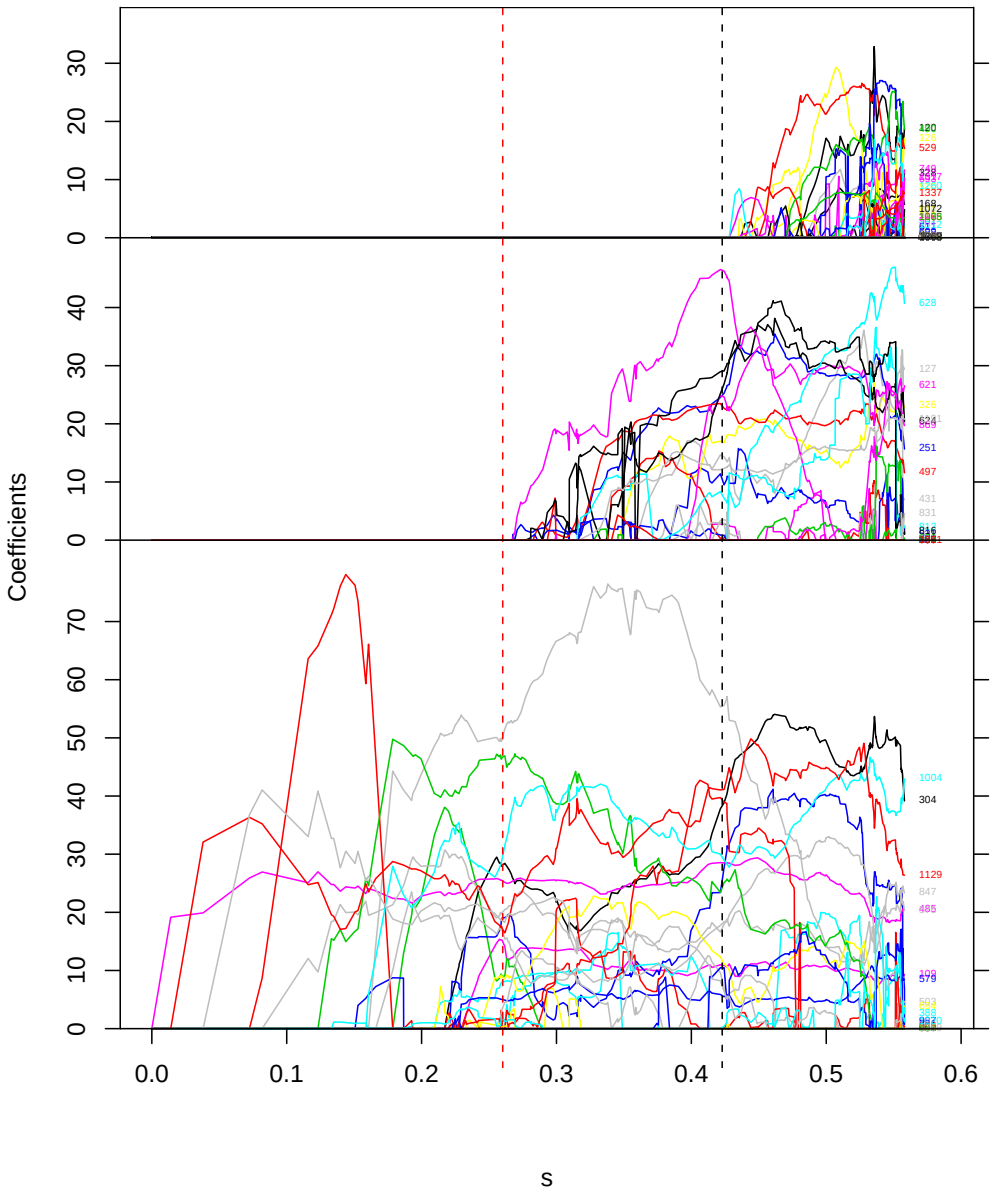

Figure 5.18: The path of recalibration parameters for structured SVM. The colored numbers are the associated gene indices. The black dashed line indicates the parameters with the minimum 5-fold cross validated error rate averaged over 20 different splits, and the red dashed line indicates that with the minimum risk in the hinge loss. The parameters are grouped into three panels depending on the first value of s at which they become nonzero.

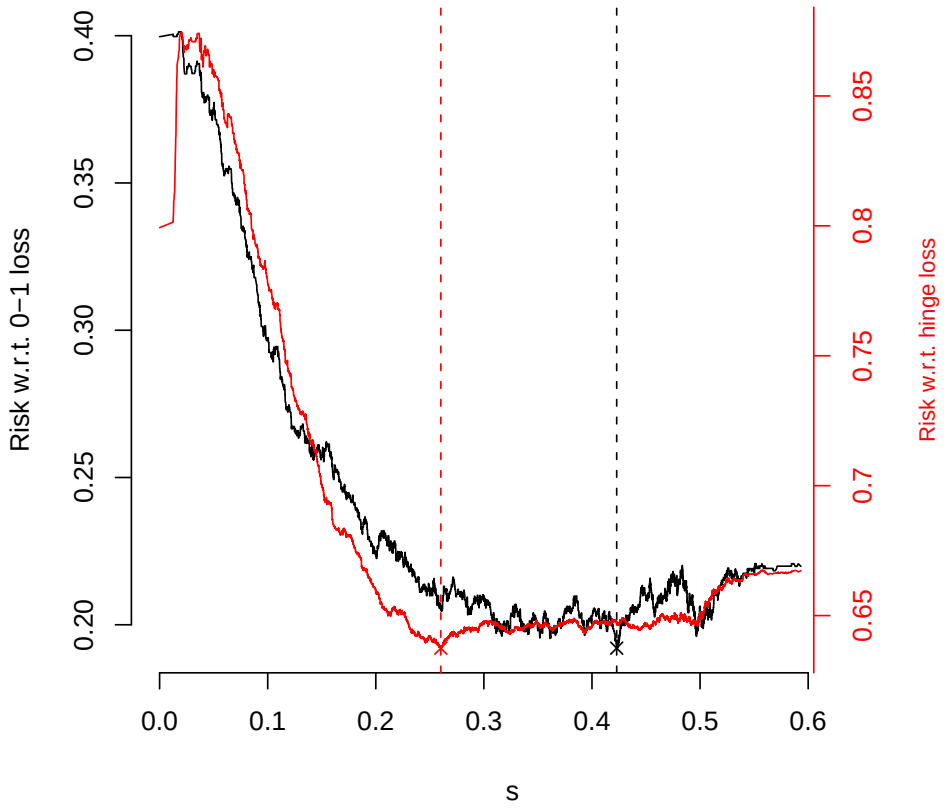

Figure 5.19: Error rate path for the θ-step of the structured SVM. The path is the average of 5-fold cross validated error rate curves for 20 different splits of the data. The black dashed line locates the minimum error rate along the curve.

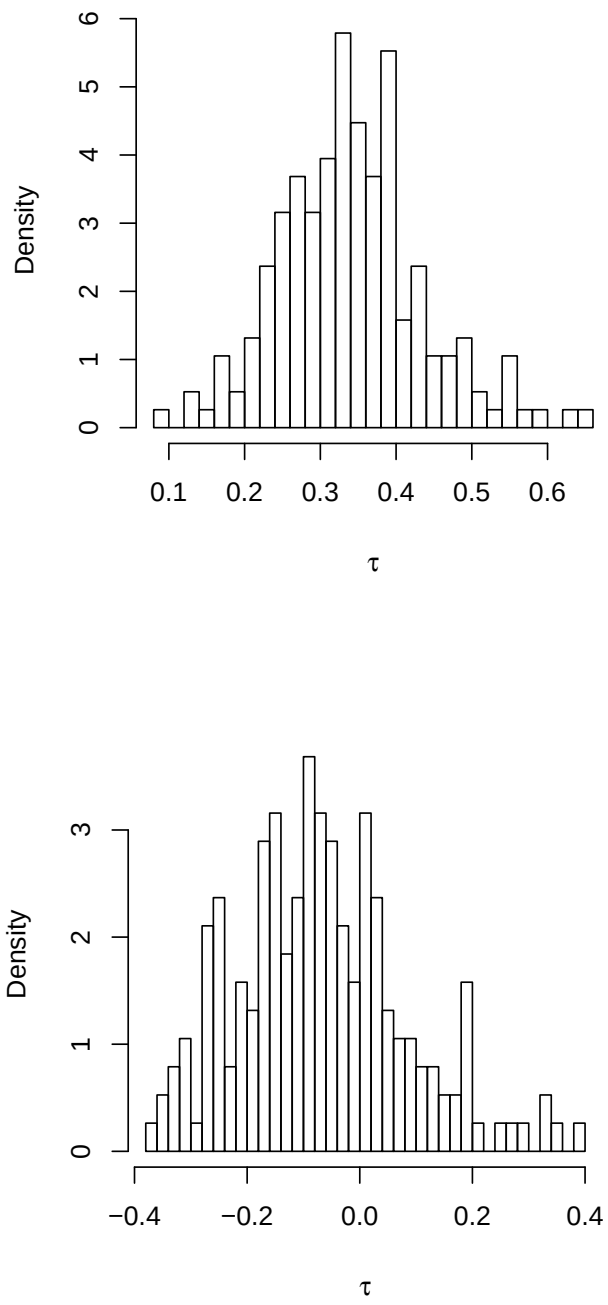

Figure 5.20: Histograms of Kendall's τ for all pairs of 20 linear classifiers produced by l_1-norm SVM. The upper panel is based on the coefficients, and the lower panel is based on their absolute values.

CHAPTER 6

CONCLUSION

This thesis has focused on elucidating the link between computational problems with linear constraints for feature selection in statistical modeling and the linear programming theory. Tapping into a rich theory of linear programming and its algorithmic developments, we have provided a broad and unified perspective on the properties of solutions to a wide family of regularization methods for feature selection. We have shown that the solutions can be characterized completely by using the parametric linear programming techniques. As for implementation, a single umbrella procedure can serve for all of the methods in the family in order to generate the entire set of regularized solutions. Efficiency can be gained further when the procedure is tailored to each individual method by utilizing the structure of the computational elements specific to the method. The connection does not only provide a useful computational tool but also helps understand the nature of the estimators given by the methods in consideration. For instance, the sparsity of the estimators in either variables or observations can be clearly apprehended via the tableau.

As illustrated, the solution paths offer rich information about how constrained models evolve with features. Especially, they make it easy to recognize persistent features in the data, which are of general interest in data analysis. In addition to facilitating computation and tuning, the path-finding algorithms for feature selection can equip the data analyst with

a useful tool for visualization of a model path. Combined with risk measures, such a path can portray a full spectrum of potentially good models for selection and averaging.

Moving beyond LP, we know that many statistical regularization methods can be cast as Parametric Quadratic Programming (PQP) problems. Examples are LASSO, SVM, non-parametric QR, etc. The PQP problem under consideration can be standardized as

$$\begin{aligned} \min \quad & \frac{1}{2}\boldsymbol{z}'\mathbf{C}\boldsymbol{z} + \lambda \mathbf{c}'\boldsymbol{z} \\ \text{s.t.} \quad & \mathbf{A}\boldsymbol{z} = \mathbf{b} \text{ and } \boldsymbol{z} \geq \mathbf{0}. \end{aligned}$$

General simplex-like methods for solving the problem have been proposed and developed by Wolfe (1959), Jagannathan (1966), Panne and Whinston (1969), Rusin (1971), and other researchers. Parallel to the approach presented in this thesis, it would be very interesting to develop a unified perspective and practical algorithms for solving such a class of statistical regularization problems.

Also, more flexible statistical regularization methods can be formulated with multiple tuning parameters instead of a single tuning parameter. For example, the $\boldsymbol{\theta}$-step of SQR with $l_1 - l_\infty$ penalty (see Section 3.2.2) can be viewed as a regularization method with two tuning parameters, λ_θ and an upper bound u. Multi-parametric linear or quadratic programming (Gal; 1979; Pistikopoulos et al.; 2007) would be relevant to handling multiple tuning parameters systematically.

A massive dataset often results in a large-scale optimization problem, which may impede many current algorithms whose computational complexities increase rapidly with the size of the dataset. Therefore, it would be necessary to develop alternative procedures that can quickly approximate the solution with reasonable accuracy. For such large scale problems, computing the entire solution path exactly would be unnecessary and perhaps

impractical. Instead, we may choose a relatively small set of values of the tuning parameter and calculate the corresponding optimal solutions with more efficient methods for a single value such as the interior-point method.

With the entire model path at hand, the data analyst is faced with the problem of model selection or averaging. Cross Validation (CV) is widely used to evaluate the performance of the models along the solution path. However, as shown in the numerical examples, the risk curve of potential models, now available in a very fine scale, could be quite jagged. For a stable mapping from the tuning parameter to the risk, smoothing techniques may be useful to reduce the variance of the risk curves and eventually lead to better model selection procedures.

APPENDIX A

PROOFS

A.1 Lemma 12

Lemma 12 *Suppose that* $\mathcal{B}^{l+1} := \mathcal{B}^l \cup \{j^l\} \setminus \{i^l\}$, *where* $i^l := B^l_{i^l_*}$. *Then*

$$\boldsymbol{z}^{l+1} = \boldsymbol{z}^l - \frac{z^l_{i^l}}{d^l_{i^l}} \boldsymbol{d}^l.$$

Proof First observe that

$$\boldsymbol{z}^{l+1}_{\mathcal{B}^{l+1}} = \mathbf{A}^{-1}_{\mathcal{B}^{l+1}} \mathbf{b} = \mathbf{A}^{-1}_{\mathcal{B}^{l+1}} \mathbf{A}_{\mathcal{B}^l} \mathbf{A}^{-1}_{\mathcal{B}^l} \mathbf{b} = [\mathbf{A}^{-1}_{\mathcal{B}^{l+1}} \mathbf{A}_{\mathcal{B}^l}] \boldsymbol{z}^l_{\mathcal{B}^l}.$$

Without loss of generality, the i^l_*th column vector $\mathbf{A}_{i^l}$ of $\mathbf{A}_{\mathcal{B}^l}$ is replaced with $\mathbf{A}_{j^l}$ to give $\mathbf{A}_{\mathcal{B}^{l+1}}$. For the $\mathbf{A}_{\mathcal{B}^{l+1}}$,

$$\begin{aligned}
[\mathbf{A}^{-1}_{\mathcal{B}^{l+1}} \mathbf{A}_{\mathcal{B}^l}]^{-1} &= \mathbf{A}^{-1}_{\mathcal{B}^l} \mathbf{A}_{\mathcal{B}^{l+1}} \qquad \text{(A.1)} \\
&= [\mathbf{e}_1, \cdots, \mathbf{e}_{i^l_*-1}, \boldsymbol{u}^l, \mathbf{e}_{i^l_*+1}, \cdots, \mathbf{e}_M] \\
&= \begin{bmatrix} 1 & & u^l_1 & & \\ & \ddots & \vdots & & \\ & & u^l_{i^l_*} & & \\ & & \vdots & \ddots & \\ & & u^l_M & & 1 \end{bmatrix},
\end{aligned}$$

where $\boldsymbol{u}^l := \mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l} = -\boldsymbol{d}_{\mathcal{B}^l}^l$. Thus, we have

$$\mathbf{A}_{\mathcal{B}^{l+1}}^{-1}\mathbf{A}_{\mathcal{B}^l} = \begin{bmatrix} 1 & & -\frac{u_1^l}{u_{i_*^l}^l} & & \\ & \ddots & \vdots & & \\ & & \frac{1}{u_{i_*^l}^l} & & \\ & & \vdots & \ddots & \\ & & -\frac{u_M^l}{u_{i_*^l}^l} & & 1 \end{bmatrix}. \tag{A.2}$$

Then it immediately follows that

$$\boldsymbol{z}_{\mathcal{B}^{l+1}}^{l+1} = \boldsymbol{z}_{\mathcal{B}^l}^l - \frac{z_{i^l}^l}{d_{i^l}^l}\boldsymbol{d}_{\mathcal{B}^l}^l - \frac{z_{i^l}^l}{d_{i^l}^l}\boldsymbol{e}_{i_*^l}.$$

Hence, $\boldsymbol{z}^{l+1} = \boldsymbol{z}^l - (z_{i^l}^l/d_{i^l}^l)\boldsymbol{d}^l$.

A.2 Proof of (2.11)

For $l = 0, \cdots, J-1$, consider the following difference

$$\begin{aligned} & \left[(\mathbf{c} + \lambda_l\mathbf{a}) - \mathbf{A}'\left(\mathbf{A}_{\mathcal{B}^{l+1}}^{-1}\right)'(\mathbf{c}_{\mathcal{B}^{l+1}} + \lambda_l\mathbf{a}_{\mathcal{B}^{l+1}})\right] - \left[(\mathbf{c} + \lambda_l\mathbf{a}) - \mathbf{A}'\left(\mathbf{A}_{\mathcal{B}^l}^{-1}\right)'(\mathbf{c}_{\mathcal{B}^l} + \lambda_l\mathbf{a}_{\mathcal{B}^l})\right] \\ = \; & -\mathbf{A}'\left(\mathbf{A}_{\mathcal{B}^l}^{-1}\right)'\left(\mathbf{A}_{\mathcal{B}^{l+1}}^{-1}\mathbf{A}_{\mathcal{B}^l}\right)'(\mathbf{c}_{\mathcal{B}^{l+1}} + \lambda_l\mathbf{a}_{\mathcal{B}^{l+1}}) + \mathbf{A}'\left(\mathbf{A}_{\mathcal{B}^l}^{-1}\right)'(\mathbf{c}_{\mathcal{B}^l} + \lambda_l\mathbf{a}_{\mathcal{B}^l}). \end{aligned}$$

By the intermediate calculation in Lemma 12, we can show that the difference is $\kappa_{\lambda_l}\mathbf{A}'\left(\mathbf{A}_{\mathcal{B}^l}^{-1}\right)'\mathbf{e}_{i_*^l}$, where

$$\begin{aligned} \kappa_{\lambda_l} &:= (\mathrm{c}_{B_{i_*^l}} + \lambda_l\mathrm{a}_{B_{i_*^l}}) - \frac{\mathrm{c}_{j^l} + \lambda_l\mathrm{a}_{j^l}}{u_{i_*^l}^l} + \sum_{i \in \left(\mathcal{B}^{l+1} \setminus \{j^l\}\right)} \frac{(\mathrm{c}_i + \lambda_l\mathrm{a}_i)u_i^l}{u_{i_*^l}^l} \\ &= \frac{(\mathbf{c}_{\mathcal{B}^l} + \lambda_l\mathbf{a}_{\mathcal{B}^l})'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l} - (\mathrm{c}_{j^l} + \lambda_l\mathrm{a}_{j^l})}{u_{i_*^l}^l} \\ &= -\frac{\check{\mathrm{c}}_{j^l}^l + \lambda_l\check{\mathrm{a}}_{j^l}^l}{u_{i_*^l}^l}. \end{aligned}$$

Since $\lambda_l := -\check{\mathrm{c}}_{j^l}^l/\check{\mathrm{a}}_{j^l}^l$, $\kappa_{\lambda_l} = 0$, which proves (2.11).

A.3 Proof of Theorem 7

Let $\mathfrak{B}^l := \mathcal{B}^l \cup \{j^l\}$ for $l = 0, \cdots, J-1$, and $\mathfrak{B}^J := \mathcal{B}^J \cup \{N+1\}$, where $\mathcal{B}^l$, $\mathcal{B}^J$, andj^l are as defined in the simplex algorithm. We will show that, for any fixed $s \in [s_l, s_{l+1})$ (or $s \geq s_J$), $\mathfrak{B}^l$ (or $\mathfrak{B}^J$) is an optimal basic index set for the LP problem in (2.6).

For simplicity, let $j^J := N+1$, $\mathrm{c}_{N+1} := 0$, $\mathbf{A}_{N+1} := \mathbf{0}$, and $\mathrm{a}_{N+1} := 1$. The inverse of

$$\mathbb{A}_{\mathfrak{B}^l} \;=\; \begin{bmatrix} \mathbf{A}_{\mathcal{B}^l} & \mathbf{A}_{j^l} \\ \mathbf{a}_{\mathcal{B}^l}{}' & \mathrm{a}_{j^l} \end{bmatrix}$$

is given by

$$\mathbb{A}_{\mathfrak{B}^l}^{-1} \;=\; \begin{bmatrix} \mathbf{A}_{\mathcal{B}^l}^{-1} & \mathbf{0} \\ \mathbf{0}' & 0 \end{bmatrix} + \frac{1}{\mathrm{a}_{j^l} - \mathbf{a}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l}} \begin{bmatrix} -\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l} \\ 1 \end{bmatrix} \begin{bmatrix} -\mathbf{a}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1} \\ 1 \end{bmatrix}'$$

for $l = 0, \cdots, J$.

First, we show that $\mathbb{A}_{\mathfrak{B}^l}$ is a feasible basic index set of (2.6) for $s \in [s_l, s_{l+1}]$, i.e.

$$\mathbb{A}_{\mathfrak{B}^l}^{-1}(\mathbb{b} + s\mathbb{b}^{*\prime}) \geq \mathbf{0}. \tag{A.3}$$

Recalling that $\boldsymbol{z}_{\mathcal{B}^l}^l = \mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{b}$, $z_{j^l}^l = 0$, $s_l = \mathbf{a}'\boldsymbol{z}^l = (\mathbf{a}_{\mathcal{B}}^l{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{b})$, $\boldsymbol{d}_{\mathcal{B}^l}^l = -\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l}$, and $d_{j^l}^l = 1$, we have

$$\begin{aligned} \mathbb{A}_{\mathfrak{B}^l}^{-1}(\mathbb{b} + s\mathbb{b}^{*\prime}) \;&=\; \mathbb{A}_{\mathfrak{B}^l}^{-1}\left\{ \begin{bmatrix} \mathbf{b} \\ 0 \end{bmatrix} + s \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \right\} \\ &=\; \begin{bmatrix} \mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{b} \\ 0 \end{bmatrix} + \frac{(s - \mathbf{a}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{b})}{\mathrm{a}_{j^l} - \mathbf{a}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l}} \begin{bmatrix} -\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l} \\ 1 \end{bmatrix} \\ &=\; \begin{bmatrix} \boldsymbol{z}_{\mathcal{B}^l}^l \\ z_{j^l}^l \end{bmatrix} + \frac{s - s_l}{\mathrm{a}_{j^l} + \mathbf{a}_{\mathcal{B}^l}{}'\boldsymbol{d}_{\mathcal{B}^l}^l} \begin{bmatrix} \boldsymbol{d}_{\mathcal{B}^l}^l \\ d_{j^l}^l \end{bmatrix}. \end{aligned} \tag{A.4}$$

From $\boldsymbol{z}^{l+1} - \boldsymbol{z}^l = -(z_{i^l}^l / d_{i^l}^l)\boldsymbol{d}^l$ and $s_{l+1} - s_l = \mathbf{a}'(\boldsymbol{z}^{l+1} - \boldsymbol{z}^l) = -(z_{i^l}^l / d_{i^l}^l)(\mathrm{a}_{j^l} + \mathbf{a}_{\mathcal{B}^l}{}'\boldsymbol{d}_{\mathcal{B}^l}^l)$, it can be shown that

$$(A.4) = \begin{bmatrix} \boldsymbol{z}_{\mathcal{B}^l}^l \\ z_{j^l}^l \end{bmatrix} + \frac{s - s_l}{s_{l+1} - s_l} \left\{ \begin{bmatrix} \boldsymbol{z}_{\mathcal{B}^l}^{l+1} \\ z_{j^l}^{l+1} \end{bmatrix} - \begin{bmatrix} \boldsymbol{z}_{\mathcal{B}^l}^l \\ z_{j^l}^l \end{bmatrix} \right\}.$$

Thus, (A.4) is a convex combination of $\boldsymbol{z}^l$ and $\boldsymbol{z}^{l+1}$ for $s \in [s_l, s_{l+1}]$, and hence it is non-negative. This proves the feasibility of $\mathbb{A}_{\mathfrak{B}^l}$ for $s \in [s_l, s_{l+1}]$ and $l = 0, \ldots, J-1$. For $s \geq s^J$, we have

$$\begin{aligned} & \mathbb{A}_{\mathfrak{B}^J}^{-1} \left\{ \begin{bmatrix} \mathbf{b} \\ 0 \end{bmatrix} + s \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \right\} \\ = & \begin{bmatrix} \mathbf{A}_{\mathcal{B}^J}^{-1}\mathbf{b} \\ 0 \end{bmatrix} + (s - \mathbf{a}_{\mathcal{B}^J}{}'\mathbf{A}_{\mathcal{B}^J}^{-1}\mathbf{b}) \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \\ = & \begin{bmatrix} \mathbf{A}_{\mathcal{B}^J}^{-1}\mathbf{b} \\ 0 \end{bmatrix} + (s - s_J) \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \geq \mathbf{0}. \end{aligned}$$

Next, we prove that $\mathbb{A}_{\mathfrak{B}^l}$ is an optimal basic index set of (2.6) for $s \in [s_l, s_{l+1}]$ by showing $\mathbb{C} - \mathbb{A}'(\mathbb{A}_{\mathfrak{B}^l}^{-1})'\mathbb{C}_{\mathfrak{B}^l} \geq \mathbf{0}$. For $i = 1, \cdots, N$, the ith element of $\mathbb{C} - \mathbb{A}'(\mathbb{A}_{\mathfrak{B}^l}^{-1})'\mathbb{C}_{\mathfrak{B}^l}$ is

$$\begin{aligned} & \mathrm{c}_i - \begin{bmatrix} \mathbf{c}_{\mathcal{B}^l} \\ \mathrm{c}_{j^l} \end{bmatrix}' \mathbb{A}_{\mathfrak{B}^l}^{-1} \begin{bmatrix} \mathbf{A}_i \\ \mathrm{a}_i \end{bmatrix} \\ = & \mathrm{c}_i - \mathbf{c}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_i - \frac{\mathrm{c}_{j^l} - \mathbf{c}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l}}{\mathrm{a}_{j^l} - \mathbf{a}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_{j^l}} (\mathrm{a}_i - \mathbf{a}_{\mathcal{B}^l}{}'\mathbf{A}_{\mathcal{B}^l}^{-1}\mathbf{A}_i) \\ = & \begin{cases} \check{\mathrm{c}}_i^l + \lambda_l \check{\mathrm{a}}_i^l & \text{for } i = 1, \cdots, N \\ \lambda_l & \text{for } i = N+1. \end{cases} \end{aligned}$$

Similarly, for $s \geq s^J$,

$$\begin{aligned} \mathrm{c}_i - \begin{bmatrix} \mathbf{c}_{\mathcal{B}^J} \\ 0 \end{bmatrix}' \mathbb{A}_{\mathfrak{B}^J}^{-1} \begin{bmatrix} \mathbf{A}_i \\ \mathrm{a}_i \end{bmatrix} &= \mathrm{c}_i - \mathbf{c}_{\mathcal{B}^J}{}'\mathbf{A}_{\mathcal{B}^J}^{-1}\mathbf{A}_i \\ &= \begin{cases} \check{\mathrm{c}}_i^J & \text{for } i = 1, \cdots, N \\ 0 & \text{for } i = N+1. \end{cases} \end{aligned}$$

Clearly, the optimality condition holds by the non-negativity of all the elements as defined in the simplex algorithm. This completes the proof.

A.4 Proof of Theorem 10

i) By (A.2), we can update the pivot rows of the tableau as follows:

$$\begin{aligned} &(\text{the } i\text{th pivot row of } \mathcal{B}^{l+1}) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (A.5)\\ = &\begin{cases} (\text{the } i\text{th pivot row of } \mathcal{B}^l) - \frac{u_i^l}{u_{i_*^l}^l} (\text{the } i_*^l\text{th pivot row of } \mathcal{B}^l) & \text{for } i \neq i_*^l; \\ \frac{1}{u_{i_*^l}^l} (\text{the } i_*^l\text{th pivot row of } \mathcal{B}^l) & \text{for } i = i_*^l. \end{cases} \end{aligned}$$

If $u_i^l = 0$, the ith pivot row of $\mathcal{B}^{l+1}$ is the same as the ith pivot row of $\mathcal{B}^l (\overset{L}{>} \mathbf{0})$. For $i = i_*^l$, the ith pivot row of $\mathcal{B}^{l+1}$ is $(1/u_{i_*^l}^l)$ (the ith pivot row of $\mathcal{B}^l$) $\overset{L}{>} \mathbf{0}$. If $i \neq i_*^l$ and $u_i^l < 0$, which imply $-u_i^l/u_{i_*^l}^l > 0$, the ith pivot row of $\mathcal{B}^{l+1} \overset{L}{>} \mathbf{0}$ since the sum of any two lexicographically positive vectors is still lexicographically positive. According to the tableau update algorithm, we have $u_{i_*^l}^l > 0$, where i_*^l is the index number of the lexicographically smallest pivot row among all the pivot rows for $\mathcal{B}^l$ with $u_i^l > 0$. For $i \neq i_*^l$ and $u_i^l > 0$, by the definition of i_*^l,

$$\frac{\text{the } i_*^l\text{th pivot row of } \mathcal{B}^l}{u_{i_*^l}^l} \overset{L}{<} \frac{\text{the } i\text{th pivot row of } \mathcal{B}^l}{u_i^l}.$$

This implies that

$$\begin{aligned} &(\text{the } i\text{th pivot row for } \mathcal{B}^{l+1}) \\ = \ &\big(\text{the } i\text{th pivot row of } \mathcal{B}^l\big) - \frac{u_i^l}{u_{i_*^l}^l} \big(\text{the } i_*^l\text{th pivot row of } \mathcal{B}^l\big) \overset{L}{>} \mathbf{0}. \end{aligned}$$

Therefore, all the updated pivot rows are lexicographically positive.

Remark If $z_{i^l}^l = 0$, (A.5) implies that $z_{B_i^l}^l = z_{B_i^l}^{l+1}$ for $i \neq i_*^l$, $i \in \mathcal{M}$. and $z_{j^l}^{l+1} = 0$. Hence $\boldsymbol{z}^{l+1} = \boldsymbol{z}^l$. On the other hand, if $z_{i^l}^l > 0$, $z_{j^l}^{l+1} = (z_{i^l}^l / u_{j^l}^l) > 0$ while $z_{j^l}^l = 0$ since $j^l \notin \mathcal{B}^l$. This implies $\boldsymbol{z}^{l+1} \neq \boldsymbol{z}^l$. Therefore, $\boldsymbol{z}^{l+1} = \boldsymbol{z}^l$ if and only if $z_{i^l}^l = 0$.

ii) When the basic index set $\mathcal{B}^l$ is updated to $\mathcal{B}^{l+1}$, $\check{\mathrm{c}}^l_{j^l} < 0$. Since $j^l \in \mathcal{B}^{l+1}$, $\check{\mathrm{c}}^{l+1}_{j^l} = 0$ Then, $(\mathbf{c}_{j^l} - \mathbf{c}'_{\mathcal{B}^{l+1}}\mathbf{A}^{-1}_{\mathcal{B}^{l+1}}\mathbf{A}_{j^l}) - (\mathbf{c}_{j^l} - \mathbf{c}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}_{j^l}) = (\check{\mathrm{c}}^{l+1}_{j^l} - \check{\mathrm{c}}^l_{j^l}) > 0$.

Similarly as the proof of (2.11),

$$\left(\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^{l+1}}\mathbf{A}^{-1}_{\mathcal{B}^{l+1}}\mathbf{A}\right) - \left(\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}\right) = \kappa^l \mathbf{e}'_{i^l_*}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A},$$

where $\kappa^l := (\mathbf{c}'_{\mathcal{B}^l}\boldsymbol{u}^l - \mathrm{c}_{j^l})/u^l_{i^l_*}$. $\mathbf{e}'_{i^l_*}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}$ is the i^l_*th pivot row for $\mathcal{B}^l$, which is lexicographically positive. Since the j^lth entry of $\mathbf{e}'_{i^l_*}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}$ is strictly positive, that of $(\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^{l+1}}\mathbf{A}^{-1}_{\mathcal{B}^{l+1}}\mathbf{A}) - (\mathbf{c}' - \mathbf{c}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A})$ must share the same sign with κ^l. Thus, we have $\kappa^l > 0$. Then the updated cost row is given as

$$\begin{aligned} & \left[-\mathbf{c}'_{\mathcal{B}^{l+1}}\mathbf{A}^{-1}_{\mathcal{B}^{l+1}}\mathbf{b}, \mathbf{c}' - \mathbf{c}'_{\mathcal{B}^{l+1}}\mathbf{A}^{-1}_{\mathcal{B}^{l+1}}\mathbf{A}\right] \\ = & \left[-\mathbf{c}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{b}, \mathbf{c}' - \mathbf{c}'_{\mathcal{B}^l}\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}\right] + \kappa^l \mathbf{e}'_{i^l_*}\left[\mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{b}, \mathbf{A}^{-1}_{\mathcal{B}^l}\mathbf{A}\right]. \end{aligned}$$

Clearly, the cost row for $\mathcal{B}^{l+1}$ is lexicographically greater than that for $\mathcal{B}^l$.

A.5 Proof of (5.2)

$$\begin{aligned} & \Pr\{Y_i \neq \text{sign}(\mathbf{x}_i\boldsymbol{\beta})\} \\ = \; & \Pr\{\mathbf{x}_i\boldsymbol{\beta} > 0 \text{ and } \mathbf{x}_i\boldsymbol{\beta} + \epsilon_i < 0\} + \Pr\{\mathbf{x}_i\boldsymbol{\beta} < 0 \text{ and } \mathbf{x}_i\boldsymbol{\beta} + \epsilon_i > 0\} \\ = \; & 2\Pr\{\mathbf{x}_i\boldsymbol{\beta} > 0 \text{ and } \mathbf{x}_i\boldsymbol{\beta} + \epsilon_i < 0\} \end{aligned}$$

Let $\mathbf{W} := (W_1, W_2)' := (\frac{\mathbf{x}_i\boldsymbol{\beta}}{\sqrt{V}}, \frac{\epsilon_i}{\sigma})'$, where $V = var(\mathbf{x}_i\boldsymbol{\beta})$. By the model settings, we have

$$\mathbf{W} := \begin{bmatrix} W_1 \\ W_2 \end{bmatrix} \sim N(\mathbf{0}, \mathbf{I}).$$

Then,

$$
\begin{aligned}
& \Pr\{Y_i \neq \text{sign}(\mathbf{x}_i\boldsymbol{\beta})\} \\
= \; & 2\Pr\{W_1 > 0 \text{ and } \sqrt{V}W_1 + \sigma W_2 < 0\} \\
= \; & 2\int_0^{\infty}\int_{-\infty}^{-\frac{\sqrt{V}w_1}{\sigma}} \frac{1}{2\pi}\exp\left\{-\frac{1}{2}(w_1^2+w_2^2)\right\} dw_2 dw_1 \quad \text{(by using polar coordinates)} \\
= \; & \frac{1}{\pi}\int_{\arctan\frac{\sqrt{V}}{\sigma}}^{\frac{\pi}{2}}\int_0^{\infty} r\exp\{-r^2/2\} dr d\theta \\
= \; & \frac{1}{2} - \frac{1}{\pi}\arctan\sqrt{\frac{V}{\sigma^2}},
\end{aligned}
$$

where $w_1 := r\cos\theta$ and $w_2 := -r\sin\theta$.

APPENDIX B

PARAMETRIC QUADRATIC PROGRAMMING

Consider the standard for of a Parametric Quadratic Programming (PQP) problem given by

$$\begin{aligned} \min_{\boldsymbol{z}} \quad & \frac{1}{2}\boldsymbol{z}'\mathbf{C}\boldsymbol{z} + \lambda \mathbf{c}'\boldsymbol{z} \\ \text{s.t.} \quad & \mathbf{A}\boldsymbol{z} = \mathbf{b} \text{ and } \boldsymbol{z} \geq \mathbf{0}, \end{aligned} \tag{B.1}$$

where $\mathbf{C}$ is a nonnegative definite matrix. Wolfe (1959) proposed a simplex method to solve the problem. This section is a summary of the Wolfe simplex algorithm.

Theorem 13 *The PQP problem in* (B.1) *is solvable if and only if there exist* $\boldsymbol{z}^* \geq \mathbf{0}$*,* $\boldsymbol{v}^* \geq \mathbf{0}$*, and* $\boldsymbol{u}^*$ *with* $\mathbf{A}\boldsymbol{z}^* = \mathbf{b}$ *such that*

$$\begin{aligned} & \boldsymbol{z}^{*\prime}\boldsymbol{v}^* = 0 \\ & \mathbf{C}\boldsymbol{z}^* - \boldsymbol{v}^* + \mathbf{A}'\boldsymbol{u}^* = -\lambda \mathbf{c}. \end{aligned} \tag{B.2}$$

If the condition is satisfied, $\boldsymbol{z}^*$ *is a solution of the problem in* (B.1).

The key to this theorem is the Karush-Kuhn-Tucker (KKT) optimality conditions, which can be found in many optimization textbooks (e.g., Mangasarian (1994); Bertsimas (1999)).

The conditions in (B.2) are sufficient for the optimality of $\boldsymbol{z}^*$, which can be proved easily by showing that, for any $\boldsymbol{w} \geq \mathbf{0}$ with $\mathbf{A}\boldsymbol{w} = \mathbf{b}$, we have

$$\begin{aligned}
& \left(\frac{1}{2}\boldsymbol{w}'\mathbf{C}\boldsymbol{w} + \lambda\mathbf{c}'\boldsymbol{w}\right) - \left(\frac{1}{2}\boldsymbol{z}^{*\prime}\mathbf{C}\boldsymbol{z}^* + \lambda\mathbf{c}'\boldsymbol{z}^*\right) \\
= & \frac{1}{2}(\boldsymbol{w}-\boldsymbol{z}^*)'\mathbf{C}(\boldsymbol{w}-\boldsymbol{z}^*) + \boldsymbol{z}^{*\prime}\mathbf{C}(\boldsymbol{w}-\boldsymbol{z}^*) + \lambda\mathbf{c}'(\boldsymbol{w}-\boldsymbol{z}^*) \\
= & \frac{1}{2}(\boldsymbol{w}-\boldsymbol{z}^*)'\mathbf{C}(\boldsymbol{w}-\boldsymbol{z}^*) + (\boldsymbol{w}-\boldsymbol{z}^*)'(\mathbf{C}\boldsymbol{z}^* + \lambda\mathbf{c}) \\
= & \frac{1}{2}(\boldsymbol{w}-\boldsymbol{z}^*)'\mathbf{C}(\boldsymbol{w}-\boldsymbol{z}^*) + (\boldsymbol{w}-\boldsymbol{z}^*)'(\boldsymbol{v}^* - \mathbf{A}'\boldsymbol{u}^*) \\
= & \frac{1}{2}(\boldsymbol{w}-\boldsymbol{z}^*)'\mathbf{C}(\boldsymbol{w}-\boldsymbol{z}^*) + \boldsymbol{w}'\boldsymbol{v}^* \\
\geq & \ 0.
\end{aligned}$$

Proving that they are also necessary is more technical and not provided here, but it can be done by checking certain regularity conditions associated with KKT Theorem.

The solution of a PQP problem with given $\lambda = \lambda^* \geq 0$ can be found by solving the following associated LP problem

$$\begin{aligned}
\min \quad & \mathbf{1}'\boldsymbol{w} \\
\text{s.t.} \quad & [-\mathbf{I}, \mathbf{C}, \mathbf{A}', -\mathbf{A}', \mathbf{I}](\boldsymbol{v}', \boldsymbol{z}', (\boldsymbol{u}^+)', (\boldsymbol{u}^-)', \boldsymbol{w}')' = -\lambda^*\mathbf{c} \\
& (\boldsymbol{v}', \boldsymbol{z}', (\boldsymbol{u}^+)', (\boldsymbol{u}^-)', \boldsymbol{w}')' \geq \mathbf{0}
\end{aligned}$$

under a side condition $\boldsymbol{v}'\boldsymbol{z} = \mathbf{0}$. Note that the LP problem is not a parametric LP, so we can apply a regular simplex algorithm to solve it (see Bertsimas and Tsitsiklis (1997) Chapter 3). With the simplex algorithm, the side condition can be satisfied by controlling the entry element of the basic index set at each iteration. The control ensures that the index of v_i (or z_i), the ith element of $\boldsymbol{v}$ (or $\boldsymbol{z}$), can not enter if z_i (or v_i) is in the current basic index set for $i = 1, \cdots, n$. By Theorem 13, if the PQP problem is solvable, the solution $\boldsymbol{w}$ for the

LP problem has to be $\mathbf{0}$, and the corresponding solutions for $\boldsymbol{v}$, $\boldsymbol{z}$, and $\boldsymbol{u}$ have to satisfy the conditions in (B.2). Thus, that $\boldsymbol{z}$ solves the PQP problem at $\lambda = \lambda^*$.

Furthermore, in order to efficiently generating the entire solution path for the PQP problem, we can first obtain the optimal basic index set for the previous LP problem at $\lambda = 0$, and use this set as the initial basic index set for a simplex algorithm that solves the following LP problem

$$\begin{aligned} \max \quad & \lambda \\ \text{s.t.} \quad & [-\mathbf{I}, \mathbf{C}, \mathbf{A}', -\mathbf{A}', \mathbf{c}'](\boldsymbol{v}', \boldsymbol{z}', (\boldsymbol{u}^+)', (\boldsymbol{u}^-)', \lambda)' = \mathbf{0} \\ & (\boldsymbol{v}', \boldsymbol{z}', (\boldsymbol{u}^+)', (\boldsymbol{u}^-)', \lambda)' \geq \mathbf{0}. \end{aligned}$$

Because the basic index set has the same basic matrices associated with it for both of the two LP problems, its feasibility for the former LP problem guarantees that for the latter. Then, by Theorem 13 and with the control of optimal basic index set (for the side condition $\boldsymbol{v}'\boldsymbol{z} = \mathbf{0}$), each iteration of the simplex algorithm for solving the latter LP problem produces a joint solution for the original PQP problem, and therefore sequentially generates its entire solution path.

APPENDIX C

PARAMETRIC LINEAR PROGRAMMING PACKAGE FOR REGULARIZATION METHODS

Software Description `lpRegPath` is an R package (see `http://www.r-project.org/` for more information) designed for solving a family of regularization problems that satisfy certain conditions on their loss and penalty.

By incorporating an additive penalty, regularization methods modify many commonly used statistical procedures to deal with both ill-posedness and over-fitting. For instance, LASSO (Tibshirani; 1996) is a modified version of linear regression via l_1-norm penalty on regression coefficients. Such modification can ensure that the solution is unique and continuous in the data. In general, a regularization method involves a tuning parameter which controls the trade-off between goodness of fit and model complexity. Instead of solving the regularization problem for a fixed value of the tuning parameter, `lpRegPath` implements an algorithm that generates the entire solution path as a function of the tuning parameter. Such a solution path offers rich information about how constrained models evolve with features and what features are persistent in the data, which are of general interest in data analysis. In addition to facilitating computation and tuning, the path-finding algorithm for feature selection can equip the data analyst with a useful tool for visualizing

a path of the fitted models instead of a single one. With the aid of risk measures, such a path can portray a full spectrum of potentially good models for selection and averaging.

The regularization methods for feature selection that the package can currently handle include l_1-norm Support Vector Machine (SVM) (Zhu et al.; 2004), l_1-norm Quantile Regression (Li and Zhu; 2008), and functional component selection for kernel methods (e.g., $\boldsymbol{\theta}$-step of multi-category SVM (Lee et al.; 2006)). For each of the regularization methods, `lpRegPath` generates the entire solution path, empirically evaluates the performance of all the models in the path, selects the most plausible model or features, outputs numerical as well as graphical summaries, and make predictions for new data sets.

The family of the regularization methods in consideration can be rephrased as parametric linear programming (LP) problems as noted in Yao and Lee (2007). To take advantages of the commonality in the problems for computational efficiency, the package has the core module that implements a tailored tableau-simplex method for generating the solution path of the parametric LP. In addition, for each regularization method, it has a shell function that identifies the corresponding components in the standard form of the LP and calls the same core module for computation. Due to this structural division of the core module and shell functions, other regularization procedures with convex and piecewise linearity can be easily incorporated in the package. For example, the path-finding algorithm can be extended to Dantzig selector (Candes and Tao; 2007) and Support Vector regression with ϵ-insensitive loss (Vapnik; 1998) by writing relevant shell functions.

Methodology To further illustrate the core-shell relationship, we mathematically describe the connections between the parametric LP and regularization methods. Let $\mathcal{L}(y, f(\boldsymbol{x}))$ denote a convex loss function for the prediction error and $J(f)$ be a convex penalty functional that measures the model complexity. The empirical risk with respect to $\mathcal{L}$ is defined

by $L(\boldsymbol{Y}, f(\mathbf{X})) := \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, f(\mathbf{x}_i))$, where $\boldsymbol{Y} := (y_1, \cdots, y_n)'$ and $\mathbf{X} := (\mathbf{x}_1', \cdots, \mathbf{x}_n')'$. Formally, the regularization problems are defined to be

$$\begin{aligned} &\min L(\boldsymbol{Y}, f) + \lambda \cdot J(f) \text{ or} \\ &\min L(\boldsymbol{Y}, f) \\ &\text{subject to } J(f) \leq s \text{ with respect to } f \in \mathcal{F}, \end{aligned} \tag{C.1}$$

where λ and s are the pre-specified nonnegative regularization parameters, and $\mathcal{F}$ denotes a model space. Its solution can be viewed as a function of λ or s, called the solution path.

Consider the model space $\mathcal{F} := \{f(\mathbf{x}; \boldsymbol{\beta}) : \boldsymbol{\beta} \in \mathcal{D}\}$ with $\boldsymbol{\beta}$ and $\mathcal{D}$ respectively being the model parameter and the parameter space. The regularization problem in (C.1) can be transformed into a parametric LP problem, if both of L and J are convex piecewise linear functions with respect to $\boldsymbol{\beta}$, and $\mathcal{D}$ is a polyhedron. Take the l_1-norm SVM as an example which can be written as

$$\min_{\beta_0, \boldsymbol{\beta}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \xi_i^+ + \lambda \|\boldsymbol{\beta}\|_1 \right\} \text{ s.t. } \xi_i = 1 - y_i(\beta_0 + \boldsymbol{\beta}'\mathbf{x}_i) \text{ for } i = 1, \cdots, n.$$

The LP formulations for (C.1) under the conditions L and J are respectively

$$\begin{cases} \min\limits_{\boldsymbol{z} \in \mathcal{R}^N} & (\mathbf{c} + \lambda \mathbf{a})'\boldsymbol{z} \\ \text{subject to} & \mathbf{A}\boldsymbol{z} = \mathbf{b} \\ & \boldsymbol{z} \geq \mathbf{0}, \end{cases} \quad \text{or} \quad \begin{cases} \min\limits_{\boldsymbol{z} \in \mathcal{R}^N} & \mathbf{c}'\boldsymbol{z} \\ \text{subject to} & \mathbf{A}\boldsymbol{z} = \mathbf{b} \\ & \mathbf{a}'\boldsymbol{z} \leq s \\ & \boldsymbol{z} \geq \mathbf{0}, \end{cases} \tag{C.2}$$

where $\boldsymbol{z}$ is an N-vector of variables, $\mathbf{c}$ and $\mathbf{a}$ are fixed N-vectors, $\mathbf{b}$ is a fixed M-vector, and $\mathbf{A}$ is an $M \times N$ fixed matrix. The transformation is implemented by shell functions in `lpRegPath`. The former in (C.2) is the standard form of the parametric-cost LP, whose solution is a piecewise constant function of λ. And the latter can be viewed as a special case of the parametric right-hand-side LP, whose solution is a piecewise linear function of s. To characterize the solutions completely, it is sufficient to identify the joint solutions

at a finite sequence of λ_l. By the correspondence between the two formulations in (C.2), solving one of them will produce the solution to the other.

Tableau-simplex algorithm is a reliable and efficient way to generate the solution path for a parametric-cost LP problem (Murty; 1983; Bertsimas and Tsitsiklis; 1997). By using the commonality in the structure of $\mathbf{A}$ (that is $\mathbf{A} = [\mathbf{A}^*, \mathbf{I}, -\mathbf{I}]$) for the regularization methods in consideration, the tableau-simplex algorithm can be tailored into a faster algorithm. Finding each joint solution, the computational complexity of the tailored algorithm is less than $O(M(N-2M))$ compared with $O(MN)$ for the original one. Such a simplification is the theoretical basis of the core program in `lpRegPath`.

By connecting regularization with parametric LP, the package proposes a broad and unified paradigm that can be adopted to solve a wide family of regularization problems through the same core program. To the best of my knowledge, none of the published R packages deals with generic parametric LP problems, although standard LP can be solved by `lpSolve`. Therefore, `lpRegPath` can be useful for both regularization methods and parametric LP algorithms.

Getting Started with lpRegPath

Before using `lpRegPath`, we should have R ready, which can be obtained at `http://cran.r-project.org`. The current version of package `lpRegPath` is downloadable at `http://www.stat.osu.edu/~yao/software.html` for both WINDOWS and UNIX systems. Under WINDOWS environment, we can install `lpRegPath` by clicking the menu item "Packages→Install package(s) from local zip files..."

```
> utils:::menuInstallLocal()
package 'lpRegPath' successfully unpacked and MD5 sums
checked updating HTML package descriptions
```

The installation does not load `lpRegPath` for a running R session. To load the package, use the command:

```
> require(lpRegPath)
Loading required package: lpRegPath
```

The reference manual is available online. One can look up the usage of particular commands and the relevant examples by typing

```
> ? lpRegPath
```

For illustration of the package, take l_1-norm SVM as an example. First, generate simulated data with the following code:

```
set.seed(980);
n=400; p=10;  sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(sign(x%*%beta+rnorm(n,0,sd)))
```

Then, compute and display the solution path of the associated l_1-norm SVM with the commands:

```
sp=path.svm.L1(y, x)
plot.path(sp); plot.path(sp, xtype="lam")
```

Figure C.1 shows the solution path in terms of s and λ respectively.

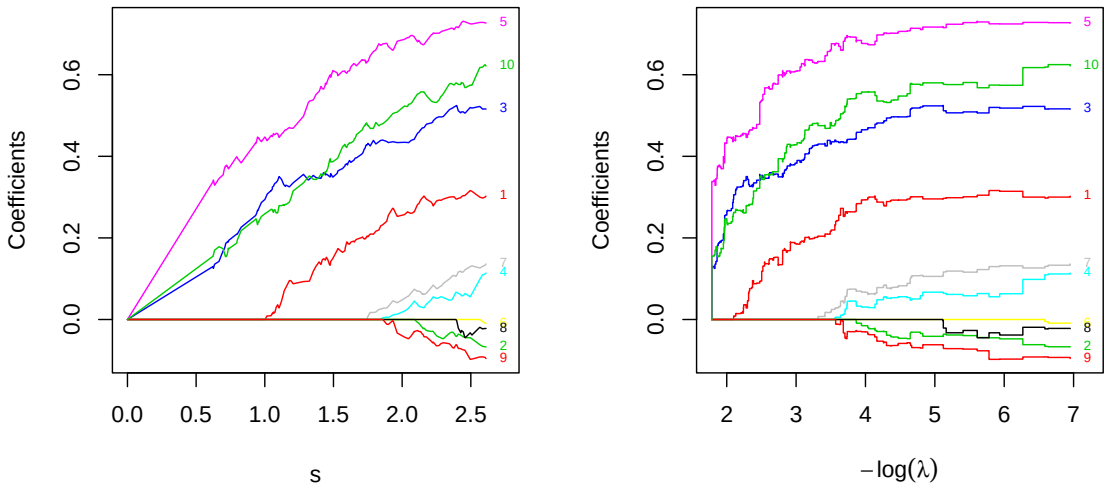

Figure C.1: An example of a solution-path plot of coefficients for l_1-norm SVM.

As described before, a solution path represents a set of optimal solution functions in terms of tuning parameter. "sp" in the example is a list that stores the information of all the joint solutions, from which the entire solution path can be derived.

```
names(sp)
[1] "coeff" "n_iter" "s_penalty" "lambda" "intercept"
```

Using the object-oriented-programming facility in R (see R Language Definition (2008) Chapter 2), we introduced two new class attributes: "path" and "risk" for `lpRegPath`, with which the solution paths and risk curves for a variety of solution path methods can be plotted by calling the generic "plot" function. For example,

```
plot(sp); plot(sp, xtype="lam")
```

`lpRegPath` estimates risk curves by calling "risk" methods, which carry out cross validation to evaluate the predictive performance of the models/classifiers in a solution path. For example, to get a five-fold cross validated error rate curve with l_1-norm SVM, use

```
Est.risk=risk.svm.L1(y, x, fold=5, repetition=10)
plot.risk(Est.risk); plot.risk(Est.risk, xtype="lam")
```

or

```
plot(Est.risk); plot(Est.risk, xtype="lam")
```

Figure C.2 displays an example of the risk curves. To make estimation of the risk more reliable, one may repeat cross validation for different splits of the data and take the average of the resulting risk estimates. The number of replicates is controlled by the input argument "repetition".

Given risk curves, we can select the "best" tuning parameters in the solution paths in terms of either s or λ.

```
> Est.risk$optER.s
                  s minimum risk
0-1 loss   1.427117    0.3325104
```

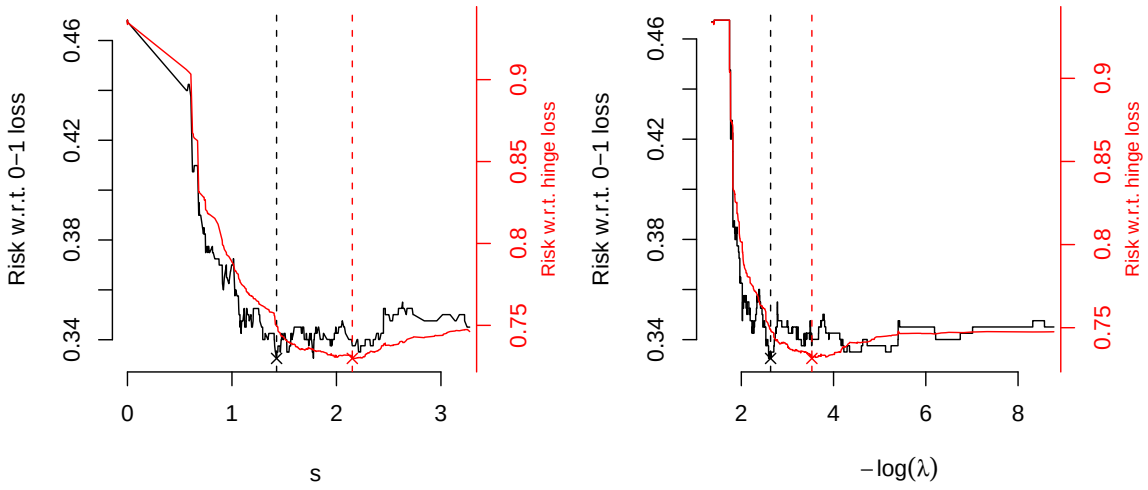

Figure C.2: An example of risk curves for l_1-norm SVM.

```
hinge loss 2.151852    0.7298276
```

```
> Est.risk$optER.lam;
                lambda minimum risk
0-1 loss    0.07169844    0.3325104
hinge loss 0.02930853    0.7315266
```

By combining the solution path with the "best" tuning parameter, we can select the "best" model/classifier. The "best" model/classifier usually involves only a subset of available covariates or features, which amounts to variable/feature selection. This can be done by

```
> model=select(sp, Est.risk)
> names(model)
```

```
[1] "opt.model" "intercept"
```

The previous sequence of commands for analysis is streamlined into one function, "svm.L1". Figure C.3 is the plot generated by "svm.L1", which computes the solution path, estimates the risk and identifies the "best" classifiers in terms of 0-1 loss and hinge loss respectively.

```
> fit=svm.L1(y,x, fold=5, rep=10)
> names(fit)
[1] "opt.model" "sp" "est.risk.path" "intercept"
```

So far we have provided an example for l_1-norm SVM. The usage of the functions for other methods is similar. Their examples can be found in the manual of `lpRegPath`.

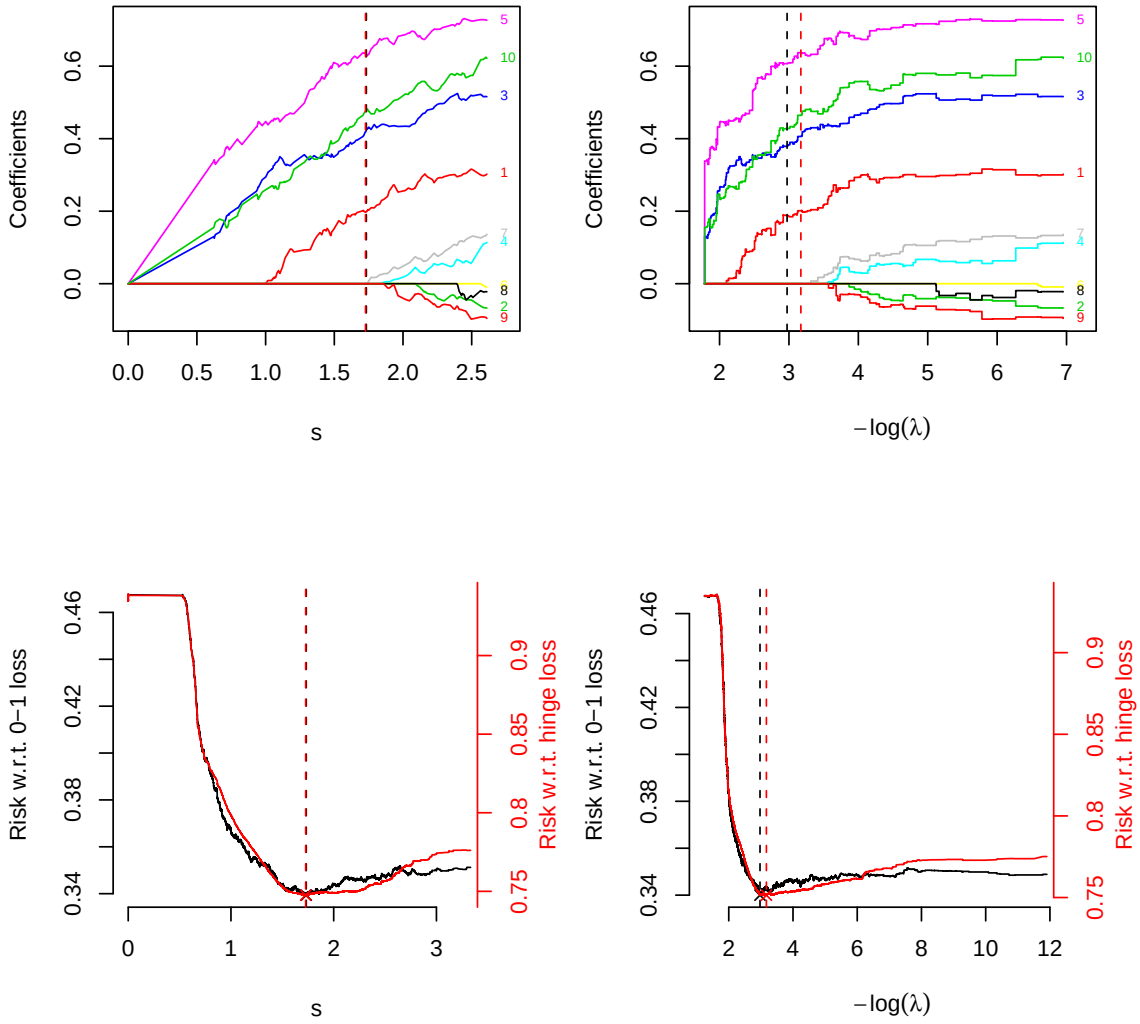

Figure C.3: An example of risk curves for l_1-norm SVM.

REFERENCES

[1] Ambroise, C. and McLachlan, G. (2002). Selection bias in gene extraction on the basis of microarray gene-expression data. *Proceedings of the National Academy of Sciences USA 99*, pages 6562–6566.

[2] Bertsimas, D. (1999). *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, second printing (in 2003) of the second edition.

[3] Bertsimas, D. and Tsitsiklis, J. (1997). *Introduction to Linear Programming*. Athena Scientific, Belmont, Massachusetts.

[4] Bloomfield, P. and Steiger, W. (1980). Least absolute deviations curve-fitting. *SIAM Journal on Scientific Computing*, 1:290–301.

[5] Bondell, H. and Reich, B. (2008). Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar. *Biometrics*, 64:115–123.

[6] Bradley, P. S. and Mangasarian, O. L. (1998). Feature selection via concave minimization and support vector machines. In Shavlik, J., editor, *Machine Learning Proceedings of the Fifteenth International Conference*, pages 82–90, San Francisco, California. Morgan Kaufmann.

[7] Candes, E. and Tao, T. (2007). The Dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, 35:2313–2351.

[8] Chen, S., Donoho, D., and Saunders, M. (1999). Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61.

[9] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20:273–297.

[10] Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression (with discussion). *The Annals of Statistics*, 32:407–451.

[11] Fisher, W. D. (1961). A note on curve fitting with minimum deviations by linear programming. *Journal of the American Statistical Association*, 56:359–362.

[12] Freed, N. and Glover, F. (1981a). Applications and implementation: A linear programming approach to the discriminant problem. *Decision Sciences*, 12:68–74.

[13] Freed, N. and Glover, F. (1981b). Simple but powerful goal programming motheds for discriminant problems. *European Journal of Operational Research*, 7:44–66.

[14] Gal, T. (1979). *Postoptimal Analyses, Parametric Programming, and Related Topics*. McGraw-Hill International, New York.

[15] Gass, S. and Saaty, T. (1955a). The computational algorithm for the parametric objective function. *Naval Research Logistics Quarterly*, 2:39–45.

[16] Gass, S. and Saaty, T. (1955b). The parametric objective function (part 2). *Journal of the Operations Research Society of America*, 3:395–401.

[17] Gu, C. (2002). *Smoothing Spline ANOVA Models*. Springer-Verlag, New York.

[18] Gunn, S. R. and Kandola, J. S. (2002). Structural modelling with sparse kernels. *Journal of Machine Learning Research*, 48:137–163.

[19] Hastie, T., Rosset, S., Tibshirani, R., and Zhu, J. (2004). The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5:1391–1415.

[20] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer Verlag, New York.

[21] Hoerl, A. and Kennard, R. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67.

[22] Jagannathan, R. (1966). A simplex-type algorithm for linear and quadratic programming–a parametric procedure. *Econometrica*, 34:460–471.

[23] James, G., Radchenko, P., and Lv, L. (2008). Dasso: Connections between the dantzig selector and lasso. Journal of the Royal Statistical Society, Series B (To appear).

[24] Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, 30:81–93.

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

[26] Koenker, R. (2005b). *Quantile Regression (Econometric Society Monographs)*. Cambridge University Press.

[27] Koenker, R. and Bassett, G. (1978). Regression quantiles. *Econometrica*, 1:33–50.

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

[29] Lee, Y. and Cui, Z. (2006). Characterizing the solution path of multicategory support vector machine. *Statistica Sinica*, 16:391–409.

[30] Lee, Y., Kim, Y., Lee, S., and Koo, J.-Y. (2006). Structured Multicategory Support Vector Machine with ANOVA decomposition. *Biometrika*, 93:555–571.

[31] Lee, Y., Lin, Y., and Wahba, G. (2004). Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99:67 – 81.

[32] Li, Y. and Zhu, J. (2008). L1-norm quantile regressions. *Journal of Computational and Graphical Statistics*, 17:163–185.

[33] Lin, Y. and Zhang, H. (2006). Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, 34:2272–2297.

[34] Mangasarian, O. (1994). *Nonlinear Programming*. SIAM, Philadelphia, PA, corrected (republication of the original book published by McGraw-Hill, Now York, in 1969) edition.

[35] Meinshausen, N., Rocha, G., and Yu, B. (2007). A tale of three cousins: Lasso, l2boosting, and Dantzig (discussion on Candes and Tao's Dantzig selector paper). *The Annals of Statistics*, 35:2372–2384.

[36] Micchelli, C. and Pontil, M. (2005). Learning the kernel function via regularization. *Journal of Machine Learning Research*, 6:1099–1125.

[37] Murty, K. (1983). *Linear Programming*. Wiley, New York, NY.

[38] Narula, S. and Wellington, J. (1977). Prediction, linear regression and minimum sum of relative errors. *Technometrics*, 19:185–190.

[39] Panne, C. v. d. and Whinston, A. (1969). The symmetric formulation of the simplex method for quadratic programming. *Econometrica*, 37:507–527.

[40] Park, C., Park, K., and Koo, J.-Y. (2006). *Structured nonparametric quantile estimation*. Korea University.

[41] Pistikopoulos, E., Georgiadis, M., and Dua, V., editors (2007). *Multi Parametric Programming: Theory, Algorithms and Applications (Process Systems Engineering)*. Wiley VCH, Weinheim.

[42] R Development Core Team (2008). *R Language Definition*, Version 2.7.2.

[43] Rosset, S. and Zhu, J. (2007). Piecewise linear regularized solution paths. *The Annals of Statistics*.

[44] Rusin, M. H. (1971). A revised simplex method for quadratic programming. *SIAM Journal on Applied Mathematics*, 20:143–160.

[45] Saaty, T. and Gass, S. (1954). The parametric objective function, part 1. *Operational Research*, 2:316–319.

[46] Sharma, P., Sahni, N., Tibshirani, R., Skaane, P., Urdal, P., Berghagen, H., Jensen, M., Kristiansen, L., Moen, C. S. P., Zaka, A., Arnes, J., Sauer, T., Akslen, L., Schlichting, E., Borresen-Dale, A., and Lonneborg, A. (2005). Early detection of breast cancer based on gene-expression patterns in peripheral blood cells. *Breast Cancer Research*, 7:634–644.

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

[48] Tibshirani, R., Hastie, T., and Narasimhan, B., C. G. (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression. *PNAS*, 99:6567–6572.

[49] Vapnik, V. (1998). *Statistical Learning Theory*. Wiley, New York: NY.

[50] Wagner, H. M. (1959). Linear programming techniques for regression analysis. *Journal of the American Statistical Association*, 54:206–212.

[51] Wahba, G. (1990). *Spline Models for Observational Data*, volume 59 of *Applied Mathematics*. SIAM, Philadelphia: PA.

[52] Wang, L. and Shen, X. (2006). Multi-category support vector machines, feature selestion and solution path. *Statistica Sinica*, 16:617–633.

[53] Wang, L. and Shen, X. (2007). On l_1-norm multi-category support vector machines: Methodology and theroy. *Journal of the American Statistical Association*, 102:583–594.

[54] Wolfe, P. (1959). The simplex method for quadratic programming. *Econometrica*, 27:382–398.

[55] Yao, Y. and Lee, Y. (2007). Another look at linear programming for feature selection via methods of regularization. Technical Report 800, Department of Statistics, The Ohio State University.

[56] Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67.

[57] Zhang, H. (2006). Variable selection for support vector machines via smoothing spline ANOVA. *Statistical Sinica*, 16:659–674.

[58] Zhang, H., Liu, Y., Wu, Y., and Zhu, J. (2006). *Variable selection for multicategory SVM via sup-norm regularization*. North Carolina State University.

[59] Zhang, H., Liu, Y., Wu, Y., and Zhu, J. (2008). Variable selection for multicategory svm via sup-norm regularization. *Electronic Journal of Statistics*, 2:149–167.

[60] Zhu, J., Rosset, S., Hastie, T., and Tibshirani, R. (2004). 1-norm support vector machines. In Thrun, S., Saul, L., and Schölkopf, B., editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA.