Accelerate the Warm-up Stage in the Lasso Computation via a Homotopic Approach

Yujie Zhao¹ and Xiaoming Huo²

¹Biostatistics and Research Decision Sciences Department, Merck & Co., Inc, PA, USA

²School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA

May 24, 2022

Abstract

In optimization, it is known that when the objective functions are strictly convex and well-conditioned, gradient-based approaches can be extremely effective, e.g., achieving the exponential rate of convergence. On the other hand, the existing Lassotype estimator cannot achieve the optimal rate due to the undesirable behavior of the absolute function at the origin. To expedite the warm-up stage, we proposed a homotopic method, which uses a sequence of surrogate functions to approximate the ℓ_1 penalty that is used in Lasso. The proposed algorithm is called the homotopy shrinkage yielding (HOSKY) algorithm. The surrogate functions will converge to the ℓ_1 penalty; while each surrogate function is strictly convex, which enables a provable faster numerical rate of convergence. We demonstrate that by meticulously designing the surrogate functions, one can prove a faster numerical convergence rate than any existing methods in the warm-up stage. Namely, the state-of-the-art algorithms can only guarantee $O(1/\epsilon)$ or $O(1/\sqrt{\epsilon})$ convergence rates, while we can prove an $O([\log(1/\epsilon)]^2)$ for the newly proposed algorithm. It's for warming-up: the aforementioned ϵ is within a specific range and can't go to zero. Our numerical simulations show that the new algorithm also performs better empirically.

Keywords— Lasso, homotopic method, convergence rate, ℓ_1 regularization

1 Introduction

In the framework of regression methods, the least absolute shrinkage and selection operator (Lasso) plays a crucial role as it performs both variable selection and model estimation. It was originally introduced in geophysics [35] and later by Robert Tibshirani [38] who coined the term. Its major objective is to select a reduced set of the known covariates for use in a predictive model.

In this paper, we focus on the calculation of the Lasso problem regarding the warm-up stage. The proposed algorithm is called homotopy shrinkage yielding (HOSKY) algorithm, which outputs an iterative estimator of the Lasso problem in the warm-up stage. Please note that we do not suggest a new algorithm for the Lasso problem. Instead, the proposed HOSKY algorithm serves as a good warm-up method. With the outputs from HOSKY, one can use them as initial points to solve the Lasso problem.

The main advantage of the proposed HOSKY algorithm is its lower computation complexity in the warm-up stage. Compared with the well-known ridge regression (a closed-form method), HOSKY has a much lower computation complexity (Section 4). Compared with existing algorithms to solve the Lasso problem, HOSKY's computational complexity is of order $O([\log(1/\epsilon)]^2)$, while the lowest order in the literature is $O(1/\sqrt{\epsilon})$. However, the overall convergence rate of HOSKY may not beat iterative algorithms – where $O(1/\sqrt{\epsilon})$ is the best – because HOSKY restricts in the warm-up stage. That is, the aforementioned ϵ does not converge to zero.

In the remaining of this section, we first introduce the problem formulation in Section 1.1, including a brief introduction of the Lasso model and the criterion to compare computational complexity. We summarize the existing literature in Section 1.2. Finally, we discuss our contributions in Section 1.3.

1.1 Problem Formulation

Let $\mathcal{D} = \{y \in \mathbb{R}^n, X \in \mathbb{R}^{n \times p}\}$ denote the available data, where y is the response vector and X is the model matrix (of predictors). Here n, p > 0 refers to the number of observations and covariates, respectively. In the discussed Lasso model, one usually has n < p. The regression model can be written as

$$y = X\beta^* + w$$
,

where $\beta^* \in \mathbb{R}^p$ is the ground truth of the regression coefficients desired to be estimated. And $w \in \mathbb{R}^n$ is the white-noise residual, i.e., $w_i \stackrel{i.i.d.}{\sim} N(0, \sigma^2)$ for any $i = 1, \ldots, n$. Accordingly, the Lasso estimator $\widehat{\beta}$ is commonly written as

$$\widehat{\beta} = \arg\min_{\beta} \left\{ F(\beta) := \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right\},\tag{1}$$

where parameter $\lambda > 0$ controls the trade-off between the sparsity and model's goodness of fit. In this paper, we don't consider the selection of parameter λ , which by itself has a large literature. Consequently, we don't include λ in the notation $F(\beta)$.

Under the above Lasso model, our objective is to develop an iterative algorithm to minimize the objective function $F(\beta)$ in equation (1) in the warm-up stage. And we denote the iterative estimator after k loops as $\beta^{(k)}$. Since we focus on the warm-up stage, so the number of loops k is bounded above in this paper.

For the aforementioned $\beta^{(k)}$, one key concern is its computational complexity. To fairly compare the computational complexity of $\beta^{(k)}$ from different iterative algorithms, we use *computational* complexity under ϵ -precision as the criterion throughout our paper, which is defined as follows.

Definition 1.1. For any pre-fixed $\epsilon > 0$, if we have

$$F(\beta^{(k)}) - F(\widehat{\beta}) \le \epsilon, \tag{2}$$

then we declare $\beta^{(k)}$ achieves the ϵ -precision. And we call the total number of numerical operations (such as plus, minus, multiplications and divisions) to achieve this ϵ -precision as a measure of the computational complexity.

The above definition indicates that the computational complexity is in terms of ϵ . And this correlation is usually adopted in the big O notation. For example, if the computational complexity of an algorithm is $O\left(np\frac{1}{\epsilon}\right)$, then it means that to achieve the ϵ -precision, the number of numeric operations can be upper bounded by a constant multiplies $np\frac{1}{\epsilon}$. In theory, an $O\left(np\log(\frac{1}{\epsilon})\right)$ algorithm is more computationally efficient than an $O\left(np\frac{1}{\epsilon}\right)$ algorithm. Moreover, an $O\left(np\log(\frac{1}{\epsilon})\right)$ algorithm has an even lower order of complexity. Although the order of computational complexity gives an upper bound of the number of numerical operations to achieve the ϵ -precision, it does not say

anything about the average performance of the algorithm. It is possible that an algorithm with larger upper bounds performs better in some cases than an algorithm with lower upper bounds.

In this paper, we focus on the computational complexity in the warm-up stage. Specifically, we don't require the value of ϵ in the ϵ -precision in (2) to go to 0. Instead, there is a lower bound of ϵ , which avoids it going to 0 and makes it a warm-up stage algorithm.

The reason for us to adopt the order of computational complexity instead of the running time is that the former records the number of numerical operations (like plus and minus), which is independent of different computer platforms. While running time, though is widely used, depends on different platforms. Consequently, the order of computational complexity provides a more reliable way for us to compare different algorithms.

1.2 Literature Review

In this section, we present several benchmarks to compare with our proposed HOSKY algorithm in the warm-up stage. The research can be mainly divided into six categories: (M1) ridge regression, (M2) first-order method, (M3) second-order method, (M4) coordinate descent method (a special first-order method), (M5) surrogate method, and (M6) path-following method. We notice that M1 gives a closed-form solution, which can be used in the warm-up stage of the Lasso computation. While M2-M6 are iterative algorithms and capable to solve the Lasso problem for the entire stage, i.e., $k \to \infty$ or $\epsilon \to 0$. In this paper, we will apply M2-M6 to the warm-up stage ($k \to \infty$ and $\epsilon \to 0$) and compare their computational complexity with our proposed HOSKY algorithm.

M1. The ridge regression [20] is frequently used as an initial point of the Lasso problem. It gives a closed-form solution

$$\left(X'X + 2\lambda nI\right)^{-1}X'y$$

in the warm-up stage to solve the Lasso problem. To get the above closed-form solution, the major computation lies in the inverse of the matrix $X'X + 2\lambda nI$. Accordingly, its computational complexity is of order $O(p^3)$, which does not depend on ϵ in Definition 1.1.

M2 The first-order method is often used to solve the Lasso problem. The first term in $F(\beta)$ is a nice quadratic function, which is numerically amenable. The challenge is rooted in the second term of $F(\beta)$, the ℓ_1 regularization term $\|\beta\|_1$, which is not differentiable at the origin. We review some representatives of the state-of-the-art algorithms to solve Lasso, which will serve as the benchmarks of our proposed algorithm.

Paper [12] approximates the first term of $F(\beta)$, i.e., $\frac{1}{2n}\|y-X\beta\|_2^2$, by its second-order Taylor expansion. Then, they use its first gradient and the Hessian matrix. In the Lasso problem, one usually has a large p with $p \gg n$. So it is computationally expensive to directly calculate its Hessian matrix, i.e., X'X/n, especially when p is large. To avoid the time-consuming calculation of the Hessian matrix, a key idea in [12] is to approximate the Hessian matrix by a diagonal matrix, whose diagonal entries are the maximal eigenvalue of X'X/n. After the quadratic approximation of the objective function, a proximal mapping is formed [7, 18, 33], where the soft-thresholding operator can be easily applied. The additional mathematical review of proximal mapping is available in A.

In the existing literature, we find many algorithms to solve Lasso follow the similar logic as [12] by using proximal gradient descent. And they have been referred to by a diverse set of names, including proximal algorithm, proximal point, and so on. And among them, [12] introduced above is a well-known representative, so we briefly introduce other proximal-related algorithms below. In the survey of [32], it can be seen that, many other widely-known statistical methods – including, the Majorization-Minimization (MM) [23, 21], and the Alternating Direction Method of Multipliers (ADMM) [5] – all fall into the proximal framework.

M3 The second-order algorithms are motivated by the first-order algorithms to learn more "history." Specifically speaking, the classical first-order algorithms use the gradient at the immediate previous solution. While the second-order algorithms, where [3] is a representative, take

advantage of the gradients at the previous two solutions, to learn from the "history." Since it uses historic information, it is also referred to as the *momentum* algorithm. Essentially, [3] falls into the framework of the Accelerate Gradient Descent (AGD), which is proposed by [29], and later widely applied into many optimization problems to speed up the convergence rate, seeing examples in [27, 3, 28, 17, 24, 11, 46], and many more. The mathematical details of the aforementioned (M1, M2) (as well as the coming ones) are provided in A.

- M4 The above two algorithms (M2, M3) update their estimates globally. On the contrary, the third category utilizes coordinate descent to update the estimate. This method is widely used and an corresponding R package named glmnet [15] has fueled its adoption. The coordinate descent method has been proposed for the Lasso problem many times, but only after [15], was its power fully appreciated. Early research work on the coordinate descent includes the discovery by [19, 42], and the convergence analysis by [40]. There are research work done on the applications of coordinate descent on Lasso problems, such as [16, 37, 14, 43, 10, 9], and so on. We choose [15] as a method to compare, since its implementation in the R package, glmnet, is very well-known by statisticians.
- M5 The aforementioned three categories (M1, M2, M3) targets directly at $F(\beta)$. Different from them, the fourth category aims to find a surrogate of $F(\beta)$. And the surrogate commonly happens to the ℓ_1 penalty term. Recall that the non-differentiability of the ℓ_1 penalty at the origin makes it hard to enable a fast convergence rate when applying the gradient descent method. Paper [36] proposes a surrogate function of the ℓ_1 penalty by taking advantage of the non-negative projection operator (seeing equations (2) and (3) in [36] for more details), where the surrogate function is twice differentiable. Consequently, the EM algorithms [13] are used for the optimization. Among this category, we select Smooth Lasso (SL) [26] as a benchmark, because it is developed recently and is an improved version of [36].
- M6 Another famous category utilizes the *path-following* idea [39, 34, 31]. The main idea of path-following (PF) is described as follows. It begins with a large penalty parameter λ , which leads all the estimated coefficients to 0. Then it tries to identify a sequence of decreasing penalty parameter λ , such that when λ is between two kink points, the support set (the set of non-zero entries of estimated β) remains unchanged. Moreover, the estimated β elementwisely is a linear function of λ . However, when one is over the kink point, the support is changed.

This type of algorithm has two major drawbacks. First, it is not guaranteed to work in general cases. As of now, the work in the current literature only establishes the path following algorithm in special situations (seeing A.5.1 for a concrete counter-example where the path following Lasso-algorithm fails). Second, the contemporary literature indicates that determining the number of loops in a path following algorithm is an open question [39, 34]. This indicates that there is no theoretical guarantee that the order of complexity of a path following approach is low, considering that the maximum number of loops can be as large as 2^p , where p is the number of predictors (seeing A.5.2 for a detailed discussion).

Given the above two drawbacks, we exclude the path-following algorithm as our benchmarks, since it doesn't work for general cases and there is no theoretical guarantee of its low computational complexity.

1.3 Our Contribution

The main contribution of the proposed HOSKY algorithm is its **provable lower order of computational complexity** in the warm-up stage, compared with the existing algorithms introduced in Section 1.2. The comparison criterion is the ϵ -precision in Definition 1.1, and the compared results are listed in Table 1. From Table 1, we find our proposed HOSKY algorithm achieves a log-polynomial order of $1/\epsilon$, while M2-M5 have a polynomial order of $1/\epsilon$. So, HOSKY has lower computational complexity than M2-M5. For M1, we will see its computational complexity is always higher than HOSKY's (Section 4). For M6, we excluded it from the comparison, since it only works

for a subset of the Lasso problem and there is no guarantee that its computational complexity is bounded.

We also admit that the overall computational complexity of HOSKY may not beat M2-M5, but it has a provable smaller computational complexity in the warm-up stage.

Table 1: The the provable upper bounds in convergence rate of M2-M4 and HOSKY for achieving the ϵ -precision in Definition 1.1.

- method ¹	$ISTA^2$	FISTA ³	CD^4	SL^5	HOSKY
Order of complexity	$O(p^2/\epsilon)$	$O(p^2/\sqrt{\epsilon})$	$O(p^2/\epsilon)$	$O(p^2/\epsilon)$	$O\left(\left[p^2\log(1/\epsilon)\right]^2\right)$

¹ Ridge regression is excluded since its complexity is not in terms of ϵ .

The organization of the rest of the paper is as follows. We develop our proposed HOSKY algorithm in Section 2. The related main theory is established in Section 3. Numerical examples are shown in Section 4. Some discussion are presented in Section 5. In A, we summarize some necessary technical details of these benchmark algorithms. A useful theorem on accelerated gradient descents is restated in B. All the technical proofs are relegated to C.

2 The Proposed Algorithm

In this section, we present our proposed iterative algorithm to solve the Lasso problem in the warm-up stage.

The main idea of the HOSKY algorithm is described as follows. Instead of optimizing $F(\beta)$ in (1) directly, we optimize a sequence of surrogate functions $F_{t_0}(\beta), F_{t_1}(\beta), \dots, F_{t_K}(\beta)$ in a sequential manner. That is, we optimize $F_{t_0}(\beta)$ first and then $F_{t_1}(\beta)$, until we arrive at $F_{t_K}(\beta)$. This sequence of surrogate functions $\{F_{t_k}(\beta)\}_{k=0,\dots,K}$ forms a homotopy path: it get closer and closer to $F(\beta)$ when k increases. Thus, by optimizing this sequence of surrogate functions, one can get an iterative estimator $\beta^{(k)}$ close to $\widehat{\beta}$. And this serves as a good initial point to solve (1). Note that this homotopy path is controlled by the elaborately designed hyper-parameter t_k , and we will discuss the formula of $\{t_k, F_{t_k}(\beta)\}_{k=0,\dots,K}$ in Section 2.2.

Details of the proposed HOSKY algorithm are discussed in the remainder of this section. In Section 2.1, we present the overview of our proposed algorithm. In Section 2.2, we present the design of $F_{t_k}(\beta)$. In Section 2.3, we describe the selection of the hyper-parameter t_k . In Section 2.4, we present our design of early stopping in the inner loop, which is critical to achieve the lower order of complexity.

Remark 2.1. We notice that the surrogate idea is also adopted in M5 in Section 1.2. But there are differences between M5 and HOSKY. One difference is that M5 only has one fixed surrogate function to approximate $F(\beta)$, while HOSKY has a sequence of surrogate functions $\{F_{t_k}(\beta)\}_{k=0,...,K}$, whose approximation is better.

2.1 Overview of the Proposed Algorithm

In this section, we give an overview of our proposed HOSKY algorithm. Our iterative algorithm has two layers of loops: an outer-loop and an inner-loop.

In outer-loops, we iterate the sequence of the surrogate functions $F_{t_0}(\beta), F_{t_1}(\beta), \dots, F_{t_K}(\beta)$, where

$$F_{t_k}(\beta) = \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_{t_k}(\beta). \tag{3}$$

The difference between $F_{t_k}(\beta)$ and $F(\beta)$ lies in the last item: it is $\lambda f_{t_k}(\beta)$ in $F_{t_k}(\beta)$, while it is $\lambda \|\beta\|_1$ in $F(\beta)$. Here $f_{t_k}(\beta)$ is a surrogate function of the target function $\|\beta\|_1$ and $t_k > 0$ is a

^{2,3,4,5} These methods fall in the framework of M2, M3, M4, M5 in Section 1.2, respectively.

hyper-parameter controlling the closeness between $\|\beta\|_1$ and $f_{t_k}(\beta)$. Since the design of $f_{t_k}(\beta)$ and t_k are not obvious, so we postpone it to Section 2.2. In the above outer-loops, by iterating k, it forms a homotopic path with $F_{t_k}(\beta)$ get closer and closer to $F(\beta)$ as k increases. And at the beginning of each outer-loop, it takes the stopping position from the previous outer-loop.

In an inner-loop of the k-th outer loop, we iteratively minimize $F_{t_k}(\beta)$ in (3) by the accelerated gradient descent (AGD) method. To save computations, we do not require AGD to minimize $F_{t_k}(\beta)$ until convergence. Instead, we only require AGD to minimize $F_{t_k}(\beta)$ when a pre-specified precision $\tilde{\epsilon}_k$ arrives. Mathematically, we stop the AGD inner-loops when

$$F_{t_k}(\beta^{(k)[s]}) - F_{\min,k} < \widetilde{\epsilon}_k. \tag{4}$$

Here $\beta^{(k)[s]}$ denotes the AGD iterative estimator in the sth inner-loop of the kth outer-loop, and we have $F_{\min,k} = \min_{\beta} F_{t_k}(\beta)$.

For both the outer-loops and inner-loops, we find the surrogate function $f_{t_k}(\beta)$ plays a vital role. There are two advantages to substitute $\|\beta\|_1$ with $f_{t_k}(\beta)$ in a homotopic path. First, the surrogate function $F_{t_k}(\beta)$ will be strongly convex and well conditioned (see Section 2.2). Consequently, a lower order of complexity becomes achievable in the inner-loop. Second, if when we increase the outer-loop index k, the surrogate function $f_{t_k}(\beta)$ gets closer and closer to $\|\beta\|_1$. Given the above advantages, one can prove a log-polynomial computational complexity for this algorithm at the warm-up stage. The reason we can only use it at the warm-up stage is that, even if $f_{t_k}(\beta)$ is close to $\|\beta\|_1$, the iterative solution $\beta^{(k)}$ of our algorithm is still not the original solution β under the Lasso model. To summarize the above key idea of our proposed HOSKY algorithm, we present the pseudo code in Algorithm 1.

Algorithm 1: Pseudo code of the proposed Homotopy-Shrinkage algorithm

Input:

6

- 1. Response vector $y \in \mathbb{R}^n$;
- 2. Model matrix $X \in \mathbb{R}^{n \times p}$;
- 3. A parameter λ that relates to the Lasso model;
- 4. A pre-specified precision ϵ in (2).

Output: an estimate of β , which satisfies the ϵ -precision.

```
1 Hyper-parameter initialization (Section 2.3)
```

```
2 \blacktriangleright Outer-Loop: \blacktriangleleft while the precision \epsilon is not achieved do k = k + 1
```

 $\kappa = \kappa + 1$ Set the current objective function as $F_{t_k}(\beta)$ to minimize in the inner-loop.

▶ Inner-Loop: \triangleleft while the precision $\widetilde{\epsilon}_k$ is not achieved do

continue iterate AGD on $F_{t_k}(\beta)$.

2.2 Design of Surrogate Functions

This section discusses the design of the surrogate function $f_t(\beta)$ for a general t > 0. In our proposed algorithm, $t \in \{t_0, t_1, \ldots, t_K\}$, where K is decided by the pre-specified ϵ in equation (2). This surrogate function replaces the $\|\beta\|_1$ in (1). And then, as the outer-loop goes on, the new objective function $F_t(\beta) = \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_t(\beta)$ gets closer and closer to $F(\beta)$. Consequently, one will get an estimate of β which is close to $\widehat{\beta}$.

The motivations to design the surrogate function $f_t(\beta)$ include two. First, we hope the surrogates $F_t(\beta)$ are strongly convex and well-conditioned. If so, the gradient descent method can achieve a very fast convergence rate. However, for the original objective function $F(\beta)$ in Lasso, it is not strongly convex given the ℓ_1 norm ($\|\beta\|_1$ in our paper). Second, it is widely acknowledged that the quadratic function (such as $\|\beta\|_2^2$) can be easily proved to be strongly convex. Motivated by the aforementioned two facts, we try to replace $\|\beta\|_1$ by $f_t(\beta)$, which is quadratic near 0 and almost linear outside. By making this replacement, the surrogates $F_t(\beta)$ can be strongly convex. Yet, it is nontrivial to find a good surrogate function $f_t(\beta)$. We list the requirements of $f_t(\beta)$ in Condition 2.2.

Condition 2.2. Desirable conditions for function $f_t(x)$ are in the following.

- 1. When t gets closer to 0, we have $f_t(x)$ close to the absolute value function |x|.
- 2. Function $f_t(x)$ has the second derivative with respective to x.
- 3. For fixed t > 0, function $x \mapsto f_t(x)$ is quadratic on [-t, t], here \mapsto indicates that the left hand side (i.e., x) is the variable in the function in the right hand side (i.e., $f_t(x)$). We following this convention in the rest of this paper.
- 4. Function $x \mapsto f_t(x)$ is C^1 . Here C^1 is the set of all continuously differentiable functions.

Following the requirements in Condition 2.2, we design $f_t(x)$ in the following equation with $x \in \mathbb{R}$.

$$f_t(x) = \begin{cases} \frac{1}{3t^3} \left[\log(1+t) \right]^2 x^2, & \text{if } |x| \le t, \\ \left[\frac{\log(1+t)}{t} \right]^2 |x| + \frac{1}{3|x|} \left[\log(1+t) \right]^2 - \frac{1}{t} \left[\log(1+t) \right]^2, & \text{otherwise.} \end{cases}$$
 (5)

It is also worth noting that when $x = (x_1, \dots, x_d)' \in \mathbb{R}^d$, then we have $f_t(x) = \sum_{i=1}^d f_t(x_i)$.

A nice property of the above $f_t(x)$ is that it is quadratic near 0 and almost linear outside. Fig. 1 displays this surrogate function $f_t(x)$. It can be seen that, when t gets smaller, $f_t(x)$ become closer to the counterparts of the function |x|. Besides, the difference between $f_t(\beta)$ and $||\beta||_1$ has both a upper bound and a lower bound as shown in Lemma 2.3.

Lemma 2.3. Suppose that from the beginning of our algorithm to the end of our algorithm, we have that $\beta_i^{(k)} \leq B$ for any $i \in \{1, 2 \dots p\}$ and $k \in \{1, 2, \dots, K\}$. Then for any k, the surrogate function $f_{t_k}(x)$ in equation (5) has the following property:

$$f_{t_k}(B) - B \le f_{t_k}(x) - |x| \le 0.$$
 (6)

Proof. See C.2.
$$\Box$$

The right hand side of inequality (6) shows that f(t) is always below |x|. The left hand side of inequality (6) shows that f(t) is not too below from |x|. This inequality guarantees the estimator of HOSKY is close to $\widehat{\beta}$ because their objective function is close.

Remark 2.4. The design of $f_t(x)$ in equation (5) is not unique but needs to satisfy some special requirements. Generally speaking, we can assume that $f_t(x)$ has the following format:

$$f_t(x) = \begin{cases} d(t)x^2, & \text{if } |x| \le t, \\ a(t)|x| + b(t)g(x) + c(t), & \text{otherwise.} \end{cases}$$
 (7)

The requirement in Condition 2.2 is equivalently transformed into:

- 1. both $x \mapsto f_t(x)$ and $t \mapsto f_t(x)$ are C^1 .
- 2. a(0) = 1, b(0) = 0, c(0) = 0, so that $f_0(x) = |x|$.

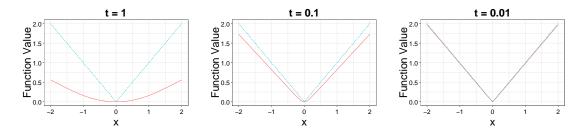


Figure 1: The red and blue solid line represent $f_t(x)$ and |x|, respectively.

Besides, we wish the second derivative of $f_t(x)$ has the format of $f_t''(x) = h(t) \max\{t, |x|\}^v$, where h(t) is a function of t and v is a constant. Accordingly, it is reasonable to suppose that $g(x) = \frac{1}{(1-v)(2-v)}x^{2-v}$. Combining all the requests above, one has

$$a(t) = \frac{\upsilon}{1+\upsilon} t^{1+\upsilon} b(t).$$

Since a(0) = 1, we choose $b(t) = \frac{1+v}{v} \left[\log(1+t) \right]^{1+v}$. Other choice of b(t) can be $\sin(\cdot)$ function or other functions, which makes $t^{1-v}b(t)$ as an constant when t = 0.

Remark 2.5. Our idea is similar to [8] in appearance, however, the differences are as follows.

- 1. [8] aimed at ℓ_p penalty, where $p \notin \{1, 2, +\infty\}$, while we focus on the p = 1, which is not discussed in [8] and the theory in [8] is not easily-extendable to the situation when p = 1.
- 2. [8] minimizes a linear function instead of the quadratic residual $\frac{1}{2n} ||y X\beta||_2^2$, where the Hessian matrix of the objective function needs different treatment.

2.3 Selection of Hyper-parameters

For our proposed HOSKY algorithm, it has a sequence of objective function $\{F_{t_k}(\beta)\}_{k=0,...,K}$. Accordingly, there is a sequence of hyper-parameter $\{t_k\}_{k=0,...,K}$. In this section, we will first discuss the initial value of the hyper-parameter, i.e., t_0 . Then we will discuss the correlation between two neighboring hyper-parameters, i.e., the correlation between t_{k-1} and t_k for any k=1,...,K.

For the initial value of the hyper-parameter t_0 , a natural motivation is to keep it relatively small. This is because, with an unnecessarily large t_0 , one would end up with more outer-loops if one starts. Consequently, one gets more numerical operations, which leads to higher computational complexity. So a minimal value of t_0 is desired. In our proposed HOSKY algorithm, we design such minimal t_0 to ensure that when $t = t_0$, the initial estimator $\beta^{(0)}$ is going to be bounded by t_0 entrywisely. Under this motivation, we design the minimal value of t_0 in equation (8) of Lemma 2.6.

Lemma 2.6. Suppose in a Lasso problem, we have the response vector $y \in \mathbb{R}^n$ and a model matrix $X \in \mathbb{R}^{n \times p}$. For our proposed algorithm, there exist a value t_0 that satisfies the following:

$$t_0 \in \left\{ t : \left| \sum_{j=1}^p M(t)_{ij} (X'y/n)_j \right| \le t \right\}, \quad \forall i = 1, \dots, p,$$
 (8)

where $M(t) = \left(\frac{X'X}{n} + \frac{\lambda}{3t^3} \left[\log(1+t)\right]^2 I\right)^{-1}$. When one chooses the aforementioned t_0 as the initial point in the proposed algorithm, one has $\left|\beta_i^{(0)}\right| \leq t_0$ for any $1 \leq i \leq p$, where $\beta_i^{(0)}$ denotes the ith entry in the vector $\beta^{(0)} = M(t_0)X'y/n$.

Proof. See C.1. \Box

For the correlation between t_{k-1} and t_k for k = 1, ..., K, there is a closed-from correlation: $t_{k+1} = t_k(1-h)$. Here $h \in (0,1)$ is set to be a predetermined values. In this way, one gets t_k smaller and smaller as k increases. Consequently, the surrogate function $f_{t_k}(\beta)$ become closer and closer to $\|\beta\|_1$ (recall Fig. 1).

2.4 Stopping Rule of the Inner-Loop

In this section, we discuss the stopping rule of the inner-loop. Recall that, in the inner loop, our objective is to minimize equation (3) with respect to β by using the AGD algorithm. To save computation, we do not iterative the AGD algorithm until convergence, instead, we stop the AGD algorithm once a pre-specified precision $\tilde{\epsilon}_k$ is achieved as shown in equation (4). This pre-specified precision $\tilde{\epsilon}_k$ is set to control the convergence of the AGD algorithm is not very poor, and we set

$$\widetilde{\epsilon}_k = \frac{\lambda p}{3B} \left[\log(1 + t_k) \right]^2. \tag{9}$$

The justification of our choice of $\tilde{\epsilon}_k$ is elaborated in our proof, whose detailed derivation can be found in C.4.

It is worth noting that, theoretically, our algorithm can achieve the order of complexity of $O\left((\log(1/\epsilon))^2\right)$ in the warm-up stage. Yet, in practice, it may not be implementable. The matter of fact is that the stoping rule of inner-loop requires knowing the value of $F_{\min,k}$, which is not possible. However, we may have to use some alternatives, such as stopping the inner-loop after a fixed number of inner-loops. In simulations, we will show that this alternative works.

As a summary of our proposed algorithm, We re-describe its simple pseudo-code in Algorithm 1 into Algorithm 2, including some of the additional details that are discussed above. In particular, we elaborate on the detailed steps of the inner-loop as shown in line 4 of Algorithm 1.

3 Order of Complexity of the HOSKY Algorithm

This section discusses the order of computational complexity of the proposed HOSKY algorithm. Recall that the order of computational complexity is defined as the number of numerical operations needed to achieve the ϵ -precision (see Definition 1.1), and always comes in a big $O(\cdot)$ notation. Because our proposed HOSKY algorithm involves two layers of loops, the order of computational complexity is in the order of the product of (i) the number of inner-loops, (ii) the number of numerical operations in each inner-loop, (iii) the number of outer-loops. In the remainder of this section, we will discuss (i), (ii), and (iii), respectively.

First, the number of inner-loops can be found in Lemma 3.1.

Lemma 3.1 (Number of Inner-Loops). Recall that a Lasso problem has a response vector $y \in \mathbb{R}^n$ and a model matrix $X \in \mathbb{R}^{n \times p}$. To minimize the Lasso objective function $F(\beta) = \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$, our proposed HOSKY algorithm minimizes $F_{t_k}(\beta) = \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_{t_k}(\beta)$, in the kth outer-loop by the AGD algorithms. Instead of converging to the minimizer of $F_{t_k}(\beta)$, we apply an early stopping rule (4) with $\tilde{\epsilon}_k$ set as in (9). It is guaranteed that, under the condition of Lemma 2.3, one can achieve the inter-loop precision ϵ_k defined in (4) after $C_1 \log(1/\tilde{\epsilon}_k)$ inner-loops, where C_1 is a constant that does not depend on the value of k.

Proof. See C.3.
$$\Box$$

Second, we notice the number of numerical operations in one inner-loop is of order $O(p^2)$. This is because the computation is dominated by the matrix multiplication of $\beta' \nabla F_{t_k}(\underline{\beta}^{(k)[s]})$ as shown in Line 9 in Algorithm 2.

Finally, the minimal number of outer-loops is summarized in Lemma 3.2.

Algorithm 2: A detailed version of our proposed algorithm

Input:

- 1. A response vector $y \in \mathbb{R}^n$;
- 2. A model matrix $X \in \mathbb{R}^{n \times p}$;
- 3. A parameter λ that relates to the Lasso model;
- 4. A pre-specified precision ϵ in (2).

Output: an estimator of β , noted as $\beta^{(k)}$, which achieves the ϵ -precision.

1 initialization
$$t_0$$
, h , $k = 1$, $\beta^{(0)} = \left[X'X + \frac{2n\lambda[\log(1+t_0)]^2}{3t_0^2} I \right]^{-1} X'y$

2 \blacktriangleright Outer-loop: \blacktriangleleft while $F(\beta^{(k-1)}) - F_{\min} > \epsilon$ do

3 $|s = 1|$
4 $|\beta^{(k)[0]} = \beta^{(k-1)}|$
5 $|\bar{\beta}^{(k)[0]} = \beta^{(k-1)}|$
6 $|\epsilon_k = \frac{\lambda p}{3B} [\log(1+t_k)]^2|$
7 \blacktriangleright Inner-loop: \blacktriangleleft while $F_{t_k}(\bar{\beta}^{(k)[s-1]}) - F_{\min,k} > \bar{\epsilon}_k$ do

8 $|\underline{\beta}^{(k)[s]} = (1-q_s)\bar{\beta}^{(k)[s-1]} + q_s\beta^{(k)[s-1]}|$
9 $|\underline{\beta}^{(k)[s]} = \arg\min_{\beta} \left\{ \gamma_s \left[\beta' \nabla F_{t_k}(\underline{\beta}^{(k)[s]}) + \mu_k V(\underline{\beta}^{(k)[s]}, \beta) \right] + V(\beta^{(k)[s-1]}, \beta) \right\}$
10 $|\underline{\beta}^{(k)[s]} = (1-\alpha_s)\bar{\beta}^{(k)[s-1]} + \alpha_s\beta^{(k)[s]}$
11 $|\underline{\beta}^{(k)} = \bar{\beta}^{(k)[s]}|$
12 $|\beta^{(k)} = \bar{\beta}^{(k)[s]}|$
13 $|t_k = t_{k-1}(1-h)|$
14 $|t_k = t_{k-1}(1-h)|$

In line 2, $F_{\min} = \min_{\beta} F(\beta)$.

In line 4 and the rest of this paper, we use parenthesis (k) to denote the kth outer-loop, and we use bracket [s] to denote the sth inner-loop.

In line 7, $F_{\min,k} = \min_{\beta} F_{t_k}(\beta)$.

In line 8, in this paper, we choose q_s as $q_s = q = \frac{\alpha_k - \mu_k/L_k}{1 - \mu_k/L_k}$ for s = 1, 2, ..., where $\alpha_k = \sqrt{\frac{\mu_k}{L_k}}$. And L_k, μ_k is defined as $\|\nabla F_{t_k}(x) - \nabla F_{t_k}(y)\|_2^2 \le L_k \|x - y\|_2$, $F_{t_k}(y) \ge F_{t_k}(x) + \nabla F_{t_k}(x)(y - x) + \frac{\mu_k}{2} \|y - x\|_2^2$. In line 9, we choose γ_s as $\gamma_s = \gamma = \frac{\alpha}{\mu_k(1 - \alpha_k)}$ for s = 1, 2, ... Here V(x, z) is defined as $V(x, z) = v(z) - [v(x) + \nabla v(x)'(z - x)]$, with $v(x) = \|x\|_2^2/2$.

Lemma 3.2 (Number of outer-loop). With the conditions in Lemma 2.3 being satisfied, then when $k \ge \frac{-1}{\log(1-h)} \log\left(\frac{\lambda pt_0(2B+1)}{\epsilon}\right)$, our proposed algorithm finds a point $\beta^{(k)}$ such that (2) is satisfied.

Proof. See C.4.
$$\Box$$

With all the above blocks, we develop the main theory, i.e., the order of computational complexity to achieve the ϵ -precision of our proposed HOSKY algorithm.

Theorem 3.3 (Main Theory). Under the conditions in Lemma 2.3, we can find $\beta^{(k)}$ such that the ϵ -precision defined in (2) is satisfied after

$$p^2 O\left(\left[\frac{-1}{\log(1-h)}\log\left(\frac{\lambda p t_0(2B+1)}{\epsilon}\right)\right]^2\right)$$

numerical operations.

Proof. See C.5.
$$\Box$$

As we can see from Theorem 3.3, the computational complexity of HOSKY is of **log-polynomial** of $1/\epsilon$. However, the compared benchmarks in Section 1.2 is **polynomial** of $1/\epsilon$. So we can declare the proposed HOSKY algorithm has lower computational complexity in the warm-up stage.

4 Numerical Examples

In this section, we compare the performance of HOSKY with three benchmarks through various numerical experiments. The selected three benchmarks are ridge regression in M1, ISTA in M2, and FISTA in M3. Here we exclude CD in M4 and SL in M5 since they share the same order of computational complexity as ISTA. And we exclude PF in M6 given its possible unbounded computational complexity.

In this section, two numerical examples are shown. The first example is the image de-noising, and the second example is the sparse linear regressions.

4.1 Simulation 1: Image De-nosing

In this simulation, we compare our proposed HOSKY algorithm with M1-M3 in the application of image de-noising. The example image we investigated is a 13×26 batman image. The image goes through a Gaussian blur of size 9×9 and standard deviation 4 followed by an additive zero-mean white Gaussian noise with standard deviation 10^{-3} . The original and observed images are given in Fig. 2 (a) and (b), respectively.

For these experiments, we assume reflexive (Neumann) boundary conditions. We then test M1-M3 and HOSKY for solving problem (1), where y represents the vectorized observed image, and X = RW with R as the matrix representing the blur operator and W as the inverse of a three-stage Haar wavelet transform. The regularization parameter is select to be $\lambda = 10^{-4}$.

The de-noising results are summarized in Figure 2(c)-(f), where one can easily find that the image produced by HOSKY is of better quality than those created by M1-M3. Besides, the computational complexity of HOSKY is lower than M1-M3, as shown in Table 2. Though it is not exactly lower than FISTA, they are on the same scale. And the small difference might be caused by the hidden constant in the big $O(\cdot)$ notation.

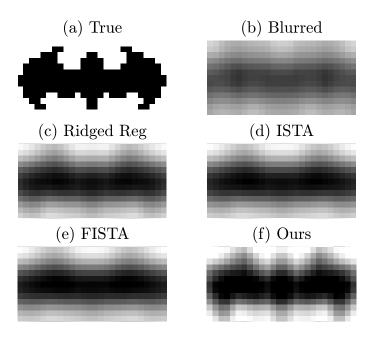


Figure 2: True, blurred and denoised images by ridged regression, ISTA, FISTA and HOSKY.

method	ridge regression ¹	$ISTA^2$	$FISTA^3$	HOSKY
Number of Numerical Operations	26,314,540	2,098,802	663,601	867,357

^{1,2,3} Ridge regression, ISTA, FISTA fall in M1, M2, M3 introduced in Section 1.2, respectively.

Table 2: The computational complexity to de-noise Figure 2 (b)

4.2 Simulation 2: Sparse Linear Regressions

In this section, we compare HOSKY with M1-M3 under the scenario to estimate true parameters in the sparse linear regression models. The data generation mechanism is described as follows, which follows [15]. First, we generate Gaussian data with n observations and p covariates, with each predictor is associated with a random vector $X_j \in \mathbb{R}^n$, and the model matrix is $X = (X_1, \dots, X_j, \dots, X_p)$. Here we assume that the random vector X_j follows the multivariate normal distribution with zero mean, variances being equal to 1, and identical population correlation ρ , that is, the covariance matrix of X_j has 1 on its diagonal and ρ for its reminder entries. In this simulation, we set $\rho = 0.1$. The response values were generated by

$$y = \sum_{j=1}^{p} X_j \beta_j + qz. \tag{10}$$

For the value of β_i ($\forall 1 \leq i \leq p$), we discuss two scenarios:

- Scenario 1: $\beta_i = (-1)^i \exp(-2(i-1)/20);$
- Scenario 2: $\beta_i = (-1)^i \exp(-2(i-1)/20) \mathbf{1} \{i \le 10\}$, where $\mathbf{1} \{\cdot\}$ is the identity function, i.e., $\mathbf{1} \{x \in A\} = 1$ if $x \in A$, and $\mathbf{1} \{x \in A\} = 0$ otherwise.

Both scenarios are constructed to have alternating signs and to be exponentially decreasing. And the difference between these two scenarios lies in the sparsity in the second scenario, i.e., its $\beta_i = 0$ when i > 10. This parameter setting assumes that most of the entries in β is zero, which renders a case with sparse truth. Besides, $z = (z_1 \cdots z_p)'$ is the white noise with z_i satisfying the standard normal distribution N(0,1). The quantity q is chosen so that the signal-to-noise ratio is 3.0. The turning parameter λ is set to be 10^{-3} . And in our simulation, we investigate two sets of (n,p) values, i.e., (n = 50, p = 20) and (n = 50, p = 80).

The simulation results are summarized in Table 3 and visualized in Fig. 3. In Table 3, the values in cells are the number of operations to achieve the different ϵ -precision in Definition 1.1. In Fig. 3, the blue line, red line, and yellow line represent the number of numerical operations of ISTA, FISTA, and HOSKY, respectively. The x-axis is the $\log(1/\epsilon)$ and the y-axis is the logarithms of the number of numerical operations to achieve the corresponding ϵ -precision.

From Table 3 and Fig. 3, we find there are some common properties shared among different methods. For example, iterative algorithms – like ISTA, FISTA, and HOSKY – take more numerical operations to achieve the targeted precision ϵ , when ϵ gets smaller. While for closed-form algorithms like ridge regression, its numbers of numerical operations are independent of ϵ .

From Table 3 and Fig. 3, we also find differences among different methods. Generally speaking, HOSKY has fewer numerical operations than M1-M3. For example, under the first scenario with $\epsilon=0.005, n=50, p=80$, HOSKY only requires 36,457 operations, however, M1-M3 need 373,414, 190,131 and 55,133 operations, respectively. And we also notice that when n=50, p=20 with large ϵ , the number of numerical operations of M2-M3 and HOSKY are very similar. This is because when ϵ is small, the hidden constant before the complexity $(O(1/\epsilon)$ for ISTA, $O(1/\sqrt{\epsilon})$ for FISTA, and $O(1/(\log(1/\epsilon)^2))$ for HOSKY) are dominated.

5 Discussion

In our theoretical result, we required the presence of a constant $\tau := t_K > 0$, such that $t_k \ge \tau$ for all k. Such a condition prevents the hyper-parameter t from converging to zero. It will be interesting to study whether there is a way to relax this condition. In Section 5.1, we show that when the τ is chosen to be small enough, an early-stopped homotopic approach will find the support of the global solution, therefore, one can simply run the ordinary regression on this support set, without losing anything.

In Section 5.2, we discuss other seemingly similar homotopic ideas, and articulate the differences between theirs and the work that is presented in this paper.

Table 3: Numerical complexity of ISTA, FISTA, HOSKY in the first simulation

14010 0. 114	morroar	compic	tity Of I			ODILL	11 0110 111	.bu billiu.	1001011
	Precision ϵ								
method	0.05	0.03	0.02	0.01	0.009	0.008	0.007	0.006	0.005
	Scenario 1 $(n = 50, p = 20)$								
Ridged Regression	7,354	7,354	7,354	7,354	7,354	7,354	7,354	7,354	7,354
ISTA	5,070	6,016	7,005	9,585	10, 101	10,703	11,434	12,294	13,369
FISTA	4,781	5,117	5,453	6,461	6,685	6,797	7,021	7,133	7,357
HOSKY	5,478	5,478	5,479	5,479	5,479	5,479	5,479	5,479	6,005
	Scenario 1 $(n = 50, p = 80)$								
Ridged Regression	373,414	373,414	373,414	373,414	373,414	373,414	373,414	373,414	373,414
ISTA	37,400	50,277	65,273	109,772	119,226	130,799	145,306	164,377	190, 131
FISTA	31,237	34,533	37,417	45,657	47,305	48,541	50,189	52,249	55, 133
HOSKY	30,919	30,919	32,765	34,611	34,611	34,611	36,457	36,457	36,457
			Scen	nario 2: (n	= 50, p =	: 20)			
Ridged Regression	7,354	7,354	7,354	7,354	7,354	7,354	7,354	7,354	7,354
ISTA	5,242	6,274	7,263	9,370	9,757	10,187	10,703	11,305	12,122
FISTA	4,781	$\boldsymbol{5,229}$	5,565	6,125	6,349	6,461	6,573	6,797	7,021
HOSKY	5,479	5,479	5,479	6,005	6,005	6,005	6,005	6,005	6,005
	Scenario 2: $(n = 50, p = 80)$								
Ridged Regression	373,414	373,414	373,414	373,414	373,414	373,414	373,414	373,414	373,414
ISTA	39,519	55,330	72,119	112,869	120,693	130,473	142,698	158,346	179,373
FISTA	31,649	35,769	39,065	45,657	46,893	48,129	49,365	51,013	53,485
HOSKY	30,918	32,763	32,763	32,763	32,763	32,763	32,763	32,763	32,763

¹ There is the parameters settings of our HOSKY algorithm: $t_0 = 3, h = 0.1, \lambda = 1e - 3, \beta^{(0)} = \mathbf{1}_{p \times 1}$.

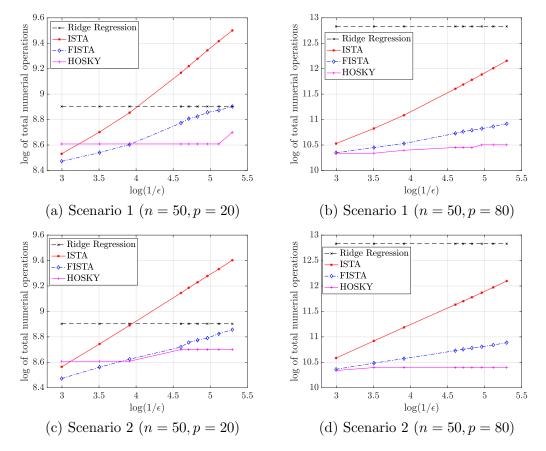


Figure 3: Number of Operations of ISTA, FISTA and HOSKY under different ϵ

5.1 Support Recovery and the Need for Hyper-parameter t to Converge to Zero

In Theorem 3.1, we assume that there is a constant $\tau > 0$, such that $t_k \geq \tau$ for all k. Such a condition prevents the hyper-parameter t from converging to zero. Therefore, our result just applies to the warm-up stage of a homotopic approach to solving the Lasso problem. In this subsection, we show that under some standard conditions that have appeared in the literature, as long as we set τ to be small enough, the associated algorithm will find a solution with both small "prediction error" and small "estimation error." The mathematical meaning of "prediction error" is

$$\frac{1}{n} \left\| X \left(\widetilde{\beta} - \widehat{\beta} \right) \right\|_{2}^{2}$$

where $\widetilde{\beta} = \arg\min_{\beta} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_t(\beta)$ for a general t and $f_t(\beta)$ defined in (5), and $\widehat{\beta} = \arg\min_{\beta} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$. And the mathematical meaning of "estimation error" in our paper is

$$\left\|\widetilde{\beta}-\widehat{\beta}\right\|_{2}^{2}$$
.

In the remaining of this section, we will give two propositions, where we develop the conditions where we would have a small prediction error and estimation error, respectively.

We begin with the prediction error, i.e., $\frac{1}{n} \| X \left(\widetilde{\beta} - \widehat{\beta} \right) \|_2^2$. In the Proposition 5.1, we declare that there are no additional conditions needed to guarantee the small prediction error. That is, as long as we converge $t \to 0$, our proposed algorithm can guarantee the prediction error goes to zero as well.

Proposition 5.1. For our proposed algorithm, when $t \to 0$, we have the perdition error $\frac{1}{n} \| X \left(\widetilde{\beta} - \widehat{\beta} \right) \|_2^2 \to 0$, where $\widetilde{\beta} = \arg \min_{\beta} \frac{1}{2n} \| y - X\beta \|_2^2 + \lambda f_t(\beta)$ for a general t and $f_t(\beta)$ defined in (5). And $\widehat{\beta} = \arg \min_{\beta} \frac{1}{2n} \| y - X\beta \|_2^2 + \lambda \|\beta\|_1$.

Proof. See C.6.
$$\Box$$

After developing the prediction error, we now discuss the estimation error. A nice property of Lasso is that, it can potentially achieve the sparse estimation when n < p, i.e., most of the entries in the Lasso estimator $\widehat{\beta}$ are zero, and only a few of them are non-zero. The index set of these non-zero entries are called *support set*, i.e., $S = \left\{i : \text{if } \widehat{\beta}_i \neq 0 \ \forall \ i=1,2,\ldots,p\right\}$. To show how Lasso can realize the sparse estimation, we take X = I as an illustration example, where I is the identity matrix. (More complicated model matrix X can also be used, but here we use X = I to create an example.) Then we have the linear regression model as

$$y = \beta + w$$

where y is the response vector, and w is the white noise. The Lasso estimator of the above linear regression model is

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{2n} \left\| y - X\beta \right\|_2^2 + \lambda \left\| \beta \right\|_1.$$

It can be verified that

$$\widehat{\beta}_i = \begin{cases} \operatorname{sign}(y_i)(|y_i| - n\lambda), & \text{if } |y_i| > n\lambda; \\ 0, & \text{otherwise,} \end{cases}$$

is the solution of the Lasso problem. Note that $\widehat{\beta}$ is sparse if y has many components with small magnitudes. However, if we consider $f_t(\beta)$, instead of the ℓ_1 penalty $\|\beta\|_1$, we have

$$\widetilde{\beta} = \arg\min_{\beta} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_t(\beta).$$

We can show that $\widetilde{\beta}_i = 0$ if and only if $y_i = 0$. This shows that $\widetilde{\beta}$ is not guaranteed to be sparse. Although $\widetilde{\beta}$ is not sparse, we can still verify that $\widetilde{\beta}$ has very small estimation error under some specific assumptions of the model matrix X.

Proposition 5.2. Suppose the model matrix X in the Lasso problem has the following three properties:

- 1. $\|(X_S'X_S)^{-1}X_S'\|_F$ can be bounded by a constant, where $S = \{i : \widehat{\beta}_i \neq 0, \forall i = 1, 2, ..., p\}$ with $\widehat{\beta} = \frac{1}{2n} \|y X\beta\|_2^2 + \lambda \|\beta\|_1$. And $\|\cdot\|_F$ is the Frobenius norm defined as $\|A_{m \times n}\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}$, where a_{ij} is the (i,j)th entry in matrix A.
- 2. $\|X_{S^c}^{\dagger}\|_F$ can be bounded by a constant, where S^c is the complement set of S. And $X_{S^c}^{\dagger}$ is the pseudo-inverse of matrix X_{S^c} . The mathematical meaning of pseudo-inverse is that, suppose $X_{S^c} = U\Sigma V$, which is the singular value decomposition (SVD) of X_{S^c} . Then $X_{S^c}^{\dagger} = V'\Sigma^{\dagger}U'$. For the rectangular diagonal matrix Σ , we get Σ^{\dagger} by taking the reciprocal of each non-zero element on the diagonal, leaving the zeros in place, and then transposing the matrix.
- 3. $\sigma_{\max}(\Sigma_1) < \min\{2, 2\sigma_{\min}(\Sigma_2)\}$, where $\sigma_{\max}(\Sigma_1)$ returns the maximal absolute diagonal values of matrix Σ_1 , and $\sigma_{\min}(\Sigma_2)$ returns the minimal absolute diagonal values of matrix Σ_2 .

 Matrix Σ_1 is the diagonal matrix in the SVD of matrix $(X_S'X_S)^{-1}X_S'X_{S^c} + (X_{S^c}^{\dagger}X_S)'$, i.e., $(X_S'X_S)^{-1}X_S'X_{S^c} + (X_{S^c}^{\dagger}X_S)' = U_1\Sigma_1V_1$. Matrix Σ_2 is the diagonal matrix of the SVD of the matrix $\frac{1}{2}X_{S^c}^{\dagger}X_{S^c} + \frac{1}{2}(X_{S^c}^{\dagger}X_{S^c})'$, i.e., $\frac{1}{2}X_{S^c}^{\dagger}X_{S^c} + \frac{1}{2}(X_{S^c}^{\dagger}X_{S^c})' = U_2\Sigma_2V_2$.

Then we have $\left\|\widetilde{\beta} - \widehat{\beta}\right\|_2^2 \to 0$ when $t \to 0$.

Proof. See C.7.
$$\Box$$

We notice that the above proposition requires a strong condition on the model matrix X to achieve the support recovery. Releasing the conditions in the above proposition is an interesting future research topic.

5.2 Other Related Homotopic Ideas

It is worth noting that, in the recent research, some researchers also realize the log-polynomial order of complexity (seeing [44, 25, 41, 45, 30]) in a framework similar to Lasso-algorithms. However, we would like to clarify that, there are some essential differences between our paper and these papers. First, the problem formulation in these papers is different from ours. The problem formulation these papers solve is that, they start at some initial objective problem, which is a Lasso-type objective function:

$$\frac{1}{2n} \|y - X\beta\|_2^2 + \lambda^{(0)} \|\beta\|_1, \tag{11}$$

and then they gradually decrease the large $\lambda^{(0)}$ until the target regularization $\lambda^{(\text{target})}$ is reached. When the $\lambda^{(\text{target})}$ is reached, the algorithm is stopped. However, this algorithmic solution is not optimal in (11). In other words, the solution of these papers is not exactly the Lasso solution. While in our paper, our objective function stays the same as (1) from the beginning to the end of our algorithm. Therefore, the solution we iteratively calculated is the minimizer of the Lasso problem in (1). In addition to the difference in the objective function, the assumptions between our algorithm and these papers are also different. Specifically, these papers require more additional assumptions than ours, such as the restricted isometry property (RIP), which is used to ensure that all solution path is sparse. Finally, through both our paper and these papers are called the "homotopic" method, the definition of the "homotopic" is different. Specifically, these papers use

the homotopic path in the penalty parameter λ : they start from a very large λ and then shrinkage to the target λ . This type of method is also called "path following" in other papers, such as [34], [1], [2] and etc, instead of "homotopic path". However, our paper uses the homotopic path in the ℓ_1 penalty $\lambda \|\beta\|_1$: we replace the ℓ_1 regularization term with a surrogate function, and then by adjusting the parameters in the surrogates, to get the surrogate approximates more close to the original ℓ_1 regularization term.

Acknowledgement

This project is partially supported by the Transdisciplinary Research Institute for Advancing Data Science (TRIAD), http://triad.gatech.edu, which is a part of the TRIPODS program at NSF and locates at Georgia Tech, enabled by the NSF grant CCF-1740776. The authors are also partially sponsored by NSF grants 1613152 and 2015363.

References

- [1] G. I. Allen. Automatic feature selection via weighted kernels and regularization. *Journal of Computational and Graphical Statistics*, 22(2):284–299, 2013.
- [2] G. I. Allen, C. Peterson, M. Vannucci, and M. Maletić-Savatić. Regularized partial least squares with an application to NMR spectroscopy. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 6(4):302–314, 2013.
- [3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM journal on imaging sciences, 2(1):183–202, 2009.
- [4] A. Beck and L. Tetruashvili. On the convergence of block coordinate descent type methods. SIAM journal on Optimization, 23(4):2037–2060, 2013.
- [5] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine learning, 3(1):1–122, 2011.
- [6] S. Boyd and L. Vandenberghe. Numerical linear algebra background. 2010. http://www.seas.ucla.edu/-vandenbe/ee236b/lectures/num-lin-alg.pdf, 2016.
- [7] L. M. Brègman. Relaxation method for finding a common point of convex sets and its application to optimization problems. In *Doklady Akademii Nauk*, volume 171, pages 1019–1022. Russian Academy of Sciences, 1966.
- [8] S. Bubeck, M. B. Cohen, Y. T. Lee, and Y. Li. An homotopy method for ℓ_p regression provably beyond self-concordance and in input-sparsity time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1130–1137. ACM, 2018.
- [9] J. Chen and F. Fang. Semiparametric likelihood for estimating equations with non-ignorable non-response by non-response instrument. *Journal of Nonparametric Statistics*, 31(2):420–434, 2019.
- [10] J. Chen, J. Shao, and F. Fang. Instrument search in pseudo-likelihood approach for nonignorable nonresponse. *Annals of the Institute of Statistical Mathematics*, 73(3):519–533, 2021.
- [11] J. Chen, B. Xie, and J. Shao. Pseudo likelihood and dimension reduction for data with nonignorable nonresponse. *Statistical Theory and Related Fields*, 2(2):196–205, 2018.

- [12] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences, 57(11):1413–1457, 2004.
- [13] M. A. Figueiredo. Adaptive sparseness for supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 25(9):1150–1159, 2003.
- [14] J. Friedman, T. Hastie, H. Höfling, R. Tibshirani, et al. Pathwise coordinate optimization. The annals of applied statistics, 1(2):302–332, 2007.
- [15] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.
- [16] W. J. Fu. Penalized regressions: the bridge versus the lasso. Journal of computational and graphical statistics, 7(3):397–416, 1998.
- [17] S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2):59–99, 2016.
- [18] M. R. Hestenes. Multiplier and gradient methods. Journal of optimization theory and applications, 4(5):303–320, 1969.
- [19] C. Hildreth. A quadratic programming procedure. Naval research logistics quarterly, 4(1):79–85, 1957.
- [20] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.
- [21] D. R. Hunter and R. Li. Variable selection using mm algorithms. *Annals of statistics*, 33(4):1617, 2005.
- [22] G. Lan. Lectures on optimization. methods for machine learning. H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 2019.
- [23] K. Lange, D. R. Hunter, and I. Yang. Optimization transfer using surrogate objective functions. Journal of computational and graphical statistics, 9(1):1–20, 2000.
- [24] H. Li and Z. Lin. Accelerated proximal gradient methods for nonconvex programming. In Advances in neural information processing systems, pages 379–387, 2015.
- [25] Q. Lin and L. Xiao. An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization. In *International Conference on Machine Learning*, pages 73–81, 2014.
- [26] S. Mukherjee and C. S. Seelamantula. Convergence rate analysis of smoothed Lasso. In Communication (NCC), 2016 Twenty Second National Conference on, pages 1–6. IEEE, 2016.
- [27] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.
- [28] Y. Nesterov. Gradient methods for minimizing composite functions. Mathematical Programming, 140(1):125–161, 2013.
- [29] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate o (1/k²). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.
- [30] H. Pang, H. Liu, R. J. Vanderbei, and T. Zhao. Parametric simplex method for sparse learning. In Advances in Neural Information Processing Systems, pages 188–197, 2017.

- [31] M. Y. Park and T. Hastie. L1-regularization path algorithm for generalized linear models. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 69(4):659–677, 2007.
- [32] N. G. Polson, J. G. Scott, B. T. Willard, et al. Proximal algorithms in statistics and machine learning. *Statistical Science*, 30(4):559–581, 2015.
- [33] R. T. Rockafellar. Monotone operators and the proximal point algorithm. SIAM journal on control and optimization, 14(5):877–898, 1976.
- [34] S. Rosset and J. Zhu. Piecewise linear regularized solution paths. *The Annals of Statistics*, pages 1012–1030, 2007.
- [35] F. Santosa and W. W. Symes. Linear inversion of band-limited reflection seismograms. SIAM Journal on Scientific and Statistical Computing, 7(4):1307–1330, 1986.
- [36] M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for 11 regularization: A comparative study and two new approaches. In *European Conference on Machine Learning*, pages 286–297. Springer, 2007.
- [37] S. K. Shevade and S. S. Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.
- [38] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- [39] R. J. Tibshirani, J. Taylor, et al. The solution path of the generalized Lasso. *The Annals of Statistics*, 39(3):1335–1371, 2011.
- [40] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. Journal of optimization theory and applications, 109(3):475–494, 2001.
- [41] Z. Wang, H. Liu, and T. Zhang. Optimal computational and statistical rates of convergence for sparse nonconvex learning problems. Annals of statistics, 42(6):2164, 2014.
- [42] J. Warga. Minimizing certain convex functions. Journal of the Society for Industrial and Applied Mathematics, 11(3):588–593, 1963.
- [43] T. T. Wu, K. Lange, et al. Coordinate descent algorithms for lasso penalized regression. The Annals of Applied Statistics, 2(1):224–244, 2008.
- [44] L. Xiao and T. Zhang. A proximal-gradient homotopy method for the sparse least-squares problem. SIAM Journal on Optimization, 23(2):1062–1091, 2013.
- [45] T. Zhao, H. Liu, T. Zhang, et al. Pathwise coordinate optimization for sparse learning: Algorithm and theory. *The Annals of Statistics*, 46(1):180–218, 2018.
- [46] Y. Zhao, X. Huo, and Y. Mei. Identification of underlying dynamic system from noisy data with splines. arXiv preprint arXiv:2103.10231, 2021.

SUPPLEMENTARY MATERIAL

Review of some State-of-the-art Algorithms Α

In this section, we will show the algorithm mechanism of these four representative we select, namely ISTA [12] in Section A.1, FISTA [3] in Section A.2, CD [15] in Section A.3, and SL [26] in Section A.4. For each algorithm, we show (i) their number of operations in a loop, (ii) the number of loops to meet the ϵ -precision in equation (2), (iii) and their according order of complexity.

A.1Iterative Shrinkage-Thresholding Algorithms (ISTA)

ISTA aims at the minimization of a summation of two functions, g + f, where the first function $g:\mathbb{R}^p\to\mathbb{R}$ is continuous convex and the other function $f:\mathbb{R}^p\to\mathbb{R}$ is smooth convex with a Lipschitz continuous gradient. Recall the definition of Lipschitz continuous gradient as follows:

$$\|\nabla f(x) - \nabla f(y)\|_2 \le L\|x - y\|_2.$$

If we let $g(\beta) = \lambda \|\beta\|_1$ and $f(\beta) = \frac{1}{2n} \|Y - X\beta\|_2^2$ with the Lipschitz continuous gradient L taking the largest eigenvalue of matrix X'X/n, noted as $\sigma_{\max}(X'X/n)$, then Lasso is a special case of ISTA. The key point of ISTA lies in the updating rule from $\beta^{(k)}$ to $\beta^{(k+1)}$, i.e., $\beta^{(k)} \to \beta^{(k+1)}$. It is

realized by updating $\beta^{(k+1)}$ through the quadratic approximation function of $f(\beta)$ at value $\beta^{(k)}$:

$$\beta^{(k+1)} = \arg\min_{\beta} f(\beta^{(k)}) + \langle (\beta - \beta^{(k)}), \nabla f(\beta^{(k)}) \rangle + \frac{\sigma_{\max}(X'X/n)}{2} \|\beta - \beta^{(k)}\|_{2}^{2} + \lambda \|\beta\|_{1}.$$
 (12)

Simple algebra shows that (ignoring constant terms in β), minimization of equation (12) is equivalent to the minimization problem in the following equation:

$$\beta^{(k+1)} = \arg\min_{\beta} \frac{\sigma_{\max}(X'X/n)}{2} \left\| \beta - (\beta^{(k)} - \frac{\frac{1}{n}(X'X\beta^{(k)} - X'y)}{\sigma_{\max}(X'X/n)}) \right\|_{2}^{2} + \lambda \|\beta\|_{1}, \tag{13}$$

where the soft-thresholding function in equation (14) can be used to solve the problem in equation (13):

$$S(x,\alpha) = \begin{cases} x - \alpha, & \text{if } x \ge \alpha, \\ x + \alpha, & \text{if } x \le -\alpha, \\ 0, & \text{otherwise.} \end{cases}$$
 (14)

The summary of ISTA algorithm is presented in Algorithm 3.

Algorithm 3: Iterative Shrinkage-Thresholding Algorithms (ISTA

Input: $y_{n\times 1}, X_{n\times p}, L = \sigma_{\max}(X'X/n)$

Output: an estimator of β satisfies the ϵ -precision, noted as $\beta^{(k)}$

- 1 initialization:
- $\beta^{(0)}, k = 0$
- 3 while $F(\beta^{(k)}) F(\widehat{\beta}) > \epsilon$ do 4 $\beta^{(k+1)} = S(\beta^{(k)} \frac{1}{nL}(X'X\beta^{(k)} X'y), \lambda/L)$ 5 k = k+1

It can be seen from line 4 in Algorithm 3 that the number of operations in one loop of ISTA is $O(p^2)$. This is because that the main computation of each loop in ISTA is the matrix multiplication in $X'X\beta^{(k)}$. Note that the matrix X'X can be pre-calculated and saved, therefore, the order of computational complexity is p(2p-1) [6].

In addition to the operations in each loop, we also develop the convergence analysis of ISTA in the following equation [3, Theorem 3.1]. To make it more clear, we list [3, Theorem 3.1] below with several changes of notation. The notations are changed to be consistent with the terminology that are used in this paper.

Theorem A.1. Let $\{\beta^{(k)}\}$ be the sequence generated by Line 4 in Algorithm 3. Then for any $k \geq 1$, we have

$$F(\beta^{(k)}) - F(\widehat{\beta}) \le \frac{\sigma_{\max}(X'X/n)\|\beta^{(0)} - \widehat{\beta}\|_2^2}{2k}.$$
 (15)

Therefore, to achieve the ϵ -precision, i.e., $F(\beta^{(k)}) - F(\widehat{\beta}) \leq \epsilon$, at least $\frac{\sigma_{\max}(X'X/n)\|\beta^{(0)} - \widehat{\beta}\|_2^2}{2\epsilon}$ loops are required, which leads to the order of complexity $O(\frac{\sigma_{\max}(X'X/n)\|\beta^{(0)} - \widehat{\beta}\|_2^2}{2\epsilon}p^2) = O(p^2/\epsilon)$.

A.2 Fast Iterative Shrinkage-Thresholding Algorithms (FISTA)

Motivated by ISTA, [3] developed another algorithm called Fast Iterative Shrinkage-Thresholding Algorithms (FISTA). The main difference of ISTA and FISTA is that FISTA employs an auxiliary variable $\alpha^{(k)}$ to update from $\beta^{(k)}$ to $\beta^{(k+1)}$ in the second-order Taylor expansion step (i.e., the one in equation (12)); More specifically, they have

$$\beta^{(k+1)} = \arg\min_{\alpha} f(\alpha^{(k)}) + \langle (\alpha - \alpha^{(k)}), \nabla f(\alpha^{(k)}) \rangle + \frac{\sigma_{\max}(X'X/n)}{2} \|\alpha - \alpha^{(k)}\|_{2}^{2} + \lambda \|\alpha\|_{1}, \quad (16)$$

where $\alpha^{(k)}$ is a specific linear combination of the previous two estimator $\beta^{(k-1)}$, $\beta^{(k-2)}$, in particular, we have $\alpha^{(k)} = \beta^{(k-1)} + \frac{t_{k-1}-1}{t_k}(\beta^{(k-1)} - \beta^{(k-2)})$. FISTA falls in the framework of Accelerate Gradient Descent(AGD), as it takes additional past information to utilize an extra gradient step via the auxiliary sequence $\alpha^{(k)}$, which is constructed by adding a "momentum" term $\beta^{(k-1)} - \beta^{(k-2)}$ that incorporates the effect of second-order changes. For completeness, the FISTA is shown in Algorithm 4.

Algorithm 4: Fast Iterative Shrinkage-Thresholding Algorithms (FISTA)

```
Input: y_{n \times 1}, X_{n \times p}, L = \sigma_{\max}(X'X/n)

Output: an estimator of \beta, noted as \beta^{(k)}, which satisfies the \epsilon-precision.

initialization;

\beta^{(0)}, t_1 = 1, k = 0

while F(\beta^{(k)}) - F(\widehat{\beta}) > \epsilon do

\beta^{(k)} = S(\alpha^{(k)} - \frac{1}{nL}(X'X\alpha^{(k)} - X'y), \lambda/L)

t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}

\alpha^{(k+1)} = \beta^{(k)} + \frac{t_k-1}{t_{k+1}}(\beta^{(k)} - \beta^{(k-1)})

k = k+1
```

Obviously, the main computational effort in both ISTA and FISTA remains the same, namely, in the soft-thresholding operation of line 4 in Algorithm 3 and 4. The number of operations in each loop of FISTA is still $O(p^2)$. Although for both ISTA and FISTA, they have the same number of operation in one loop, FISTA has improved convergence rate than ISTA, which is shown in the following theorem [3, Theorem 4.4].

Theorem A.2. Let $\{\alpha^{(k)}\}$, $\{\beta^{(k)}\}$ be a sequence generated by Line 6 and Line 4 in Algorithm 4, respectively. Then for any $k \geq 1$, we have that

$$F(\beta^{(k)}) - F(\widehat{\beta}) \le \frac{2\sigma_{\max}(X'X/n)\|\beta^{(0)} - \widehat{\beta}\|_2^2}{(k+1)^2}.$$
 (17)

Consequently, FISTA has a faster convergence rate than ISTA, which improves from O(1/k) to $O(1/k^2)$. This is because that, to update from $\beta^{(k-1)}$ to $\beta^{(k)}$, ISTA only considers $\beta^{(k-1)}$, however, FISTA takes both $\beta^{(k-1)}$ and $\beta^{(k-2)}$ into account. To achieve the precision $F(\beta^{(k)}) - F(\widehat{\beta}) \le \epsilon$, at least $\frac{2\sigma_{\max}(X'X/n)\|\beta^{(0)} - \widehat{\beta}\|_2^2}{\sqrt{\epsilon}}$ loops are required, which leads to an order of complexity of $O(\frac{2\sigma_{\max}(X'X/n)\|\beta^{(0)} - \widehat{\beta}\|_2^2}{\sqrt{\epsilon}}p^2) = O(p^2/\sqrt{\epsilon})$.

A.3 Coordinate Descent (CD)

The updating rule in both ISTA and FISTA involve all coordinates simultaneously. In contrast, [15] proposed a Lasso-algorithm that cyclically chooses one coordinate at a time and performs a simple analytical update. Such an approach is called coordinate gradient descent.

The updating rule (from $\beta^{(k)}$ to $\beta^{(k+1)}$) in CD is that, it optimizes with respect to only the *j*th entry of $\beta^{(k+1)}$ ($j=1,\dots,p$) where the gradient at $\beta_j^{(k)}$ in the following equation is used for the updating process:

$$\frac{\partial}{\partial \beta_j} F(\beta^{(k)}) = \frac{1}{n} \left(e_j' X' X \beta^{(k)} - y' X e_j \right) + \lambda \operatorname{sign}(\beta_j)$$
(18)

where e_j is a vector of length p, whose entries are all zero expect that the jth entry is equal to 1. Imposing the gradient in equation (18) to be 0, we can solve for $\beta_i^{(k+1)}$ as follows:

$$\beta_j^{(k+1)} = S\left(y'Xe_j - \sum_{l \neq j} (X'X)_{jl} \beta_k^{(k)}, n\lambda\right) / (X'X)_{jj},$$

where $S(\cdot)$ is the soft-thresholding function defined in equation (14). This algorithm has been implemented into the a R package, *glmnet*, and we summarize it in Algorithm 5.

Algorithm 5: Coordinate Descent(CD)

Input: $y_{n\times 1}, X_{n\times p}, \lambda$

Output: an estimator of β , noted as $\beta^{(k)}$, which satisfies the ϵ -precision.

- 1 initialization;
- $\beta^{(0)}, k = 0$
- з while $F(\beta^{(k)}) F(\widehat{\beta}) > \epsilon$ do
- for $j = 1 \cdots p$ do $\begin{bmatrix} \beta_j^{(k+1)} = S\left(y'Xe_j \sum_{l \neq j} (X'X)_{jl} \beta_k^{(k)}, n\lambda\right) \middle/ (X'X)_{jj} \end{bmatrix}$

After reviewing the algorithm of CD, we develop the order of complexity of CD. Firstly, the number of operations in each loop of CD is $O(p^2)$. It can be explained by the following two reasons. (i) While updating $\beta_j^{(k+1)}$ (line 5 in Algorithm 5), it costs O(p) operations because of $\sum_{l\neq j} (X'X)_{jl} \beta_k^{(k)}$. (ii) From line 4 in Algorithm 5, we can see that all p entries of $\beta^{(k+1)}$ are updated one by one. Combining (i) and (ii), we can see that the number of operations need in one loop of CD is of the order $O(p^2)$.

The convergence rate of CD is derived as a corollary in [4, Corollary 3.8] and here we list the corollary as a theorem below. We changed several notations to adopt the terminology in this paper:

Theorem A.3. Let $\{\beta^{(k)}\}$ be the sequence generated by the Line 5 in Algorithm 5. Then we have that

$$F(\beta^{(k)}) - F(\widehat{\beta}) \le \frac{4\sigma_{\max}(X'X/n)(1+p)\|\beta^{(0)} - \widehat{\beta}\|_2^2}{k + (8/p)}.$$
 (19)

The above equation shows that, to achieve the precision ϵ -precision, at least

$$\frac{4\sigma_{\max}(X'X/n)(1+p)\|\beta^{(0)}-\widehat{\beta}\|_2^2}{\epsilon}-\frac{8}{p}$$

loops are required, which leads to an order of complexity of

$$O(\left[\frac{4\sigma_{\max}(X'X/n)(1+p)\|\beta^{(0)} - \widehat{\beta}\|_{2}^{2}}{\epsilon} - \frac{8}{p}\right]p^{2}) = O(p^{2}/\epsilon - 8p) = O(p^{2}/\epsilon).$$

A.4 Smooth Lasso (SL)

The aforementioned Lasso-algorithms all aim exactly at minimizing the function $F(\beta)$. On the contrary, [26] used an approximate objective function to solve the Lasso. Their method is called a Smooth-Lasso (SL) algorithm. The main idea of SL is that it use a smooth function— $\phi_{\alpha}(u) = \frac{2}{u}\log(1+e^{\alpha u}) - u$ — to approximate the ℓ_1 penalty, and Accelerated Gradient Descent (AGD) algorithm is applied after the replacement. Therefore, the objective function of SL becomes $F_{\alpha}(\beta) = \frac{1}{2n}||y-X\beta||_2^2 + \lambda \sum_{i=1}^p \phi_{\alpha}(\beta_i)$. The pseudo code of SL is displayed in Algorithm 6.

Algorithm 6: Smooth Lasso (SL)

```
Input: y_{n\times 1}, X_{n\times p}, \ \mu = \left[\sigma_{\max}^2(X/\sqrt{n}) + \lambda\alpha/2\right]^{-1}
Output: an estimator of \beta, noted as \beta^{(k)}, which satisfies the \epsilon-precision.

initialization;

\beta^{(0)}, k = 0

while F(\beta^{(k)}) - F(\widehat{\beta}) > \epsilon do

w^{(k+1)} = \beta^{(k)} + \frac{k-2}{k+1} (\beta^{(k)} - \beta^{(k-1)})

\beta^{(k+1)} = w^{(k+1)} - \mu \nabla F_{\alpha}(w^{(k)})

k = k+1
```

For the computational effort, it mainly lies in the calculation of $\nabla F_{\alpha}(w) = \frac{X'X}{n}w - \frac{X'y}{n} + v$, where the v is a vector of length p, whose ith entry is $\frac{-2}{w_i^2}\log(1+e^{\alpha w_i}) + \frac{2\alpha e^{\alpha w_i}}{w_i(1+e^{\alpha w_i})} - 1$. Accordingly, the main computational effort of each loop of SL is the matrix multiplication in $X'Xw^{(k)}$, which cost $O(p^2)$ operations. On the other side, proved by [26], the approximation error of $\beta^{(k)}$ in SL is shown in equation (20).

Theorem A.4. Let $\{\beta^{(k)}\}$ be a sequence generated as in Line 5 of Algorithm 6. Then we have

$$F(\beta^{(k)}) - F(\widehat{\beta}) \le \frac{4\|\beta^{(0)} - \widehat{\beta}\|_2^2 \sigma_{\max}^2(\frac{X}{\sqrt{n}})}{k^2} + \frac{4\sqrt{2\lambda n \log 2}\|\beta^{(0)} - \widehat{\beta}\|_2}{k}.$$
 (20)

So to achieve the ϵ -precision, SL needs $O(1/\epsilon)$, which results in the order of complexity $O(p^2/\epsilon)$.

A.5 Path Following Lasso-Algorithm

As mentioned in Section 1, the path following Lasso-algorithm has two drawbacks. First, it is not guaranteed to work in general cases. Second, there is no theoretical guarantee that the order of complexity of a path following Lasso-algorithm is low, considering that the maximum number of loops can be as large as 2^p , where p is the number of predictors. In this section, we provide mathematical details to support the above two drawbacks. The structure of this section is described as follows. In Section A.5.1, we provide a counter example that the path following Lasso-algorithm is not workable, which represents a general category of design matrix X and coefficient β . In Section A.5.2, we provide mathematical details to support the second drawback of the path following Lasso-algorithm.

A.5.1 Details to Support the first Drawback of Path Following Lasso Algorithm

In this section, we provide a counter example that the path following Lasso-algorithm is not workable. This counter example represents a general category of design matrix X and coefficient β . We use

the following counterexample to argue that a path following approach does not work in the most general setting.

Before representing the concrete counter example, let us discuss the key step in designing a path following Lasso-algorithm. For a general solution derived by path following Lasso-algorithm, i.e., $\hat{\beta}(\lambda)$, it is the minimizer of (1), so it must satisfy the first order condition of (1):

$$q - \lambda \operatorname{sign}(\widehat{\beta}(\lambda)) = X' X \widehat{\beta}(\lambda), \tag{21}$$

where q = X'y and is $sign(\widehat{\beta}(\lambda))$ a vector, whose *i*th component is the sign function of $\widehat{\beta}(\lambda)$:

$$\operatorname{sign}(\beta_i(\lambda)) = \begin{cases} 1 & \text{if } \beta_i(\lambda) > 0 \\ -1 & \text{if } \beta_i(\lambda) < 0 \\ [-1, 1] & \text{if } \beta_i(\lambda) = 0 \end{cases}.$$

If we divide the indices of q, β, X into $S = \{i : \widehat{\beta}_i(\lambda) \neq 0, \forall i = 1, ..., p\}$ and its complements S^c , then we can rewrite equation (21) as

$$\begin{pmatrix} q_S \\ q_{S^c} \end{pmatrix} - \begin{pmatrix} \lambda \operatorname{sign}(\widehat{\beta}_S(\lambda)) \\ \lambda \operatorname{sign}(\widehat{\beta}_{S^c}(\lambda)) \end{pmatrix} = \begin{pmatrix} X_S^\top X_S & X_S^\top X_{S^c} \\ X_{S^c}^\top X_S & X_{S^c}^\top X_{S^c} \end{pmatrix} \begin{pmatrix} \widehat{\beta}_S(\lambda) \\ 0 \end{pmatrix},$$

where $\widehat{\beta}_S(\lambda)$ is the subvector of β only contains elements whose indices are in S and $\widehat{\beta}_{S^c}(\lambda)$ is the complement of β_S . Besides, $\operatorname{sign}(\widehat{\beta}_S(\lambda))$ is the subset of $\operatorname{sign}(\widehat{\beta}(\lambda))$, only contains the elements whose indices are in S, and $\operatorname{sign}(\widehat{\beta}_{S^c}(\lambda))$ is the complement to $\operatorname{sign}(\widehat{\beta}_S(\lambda))$. Matrix X_S is the columns of X whose indices are in S, and X_{S^c} is the complement of X_S .

Suppose we are interested in parameter estimated under λ and $\lambda - \Delta(\Delta \in (0, \lambda))$, i.e., $\widehat{\beta}(\lambda)$, $\widehat{\beta}(\lambda - \Delta)$. Then $\widehat{\beta}(\lambda)$, $\widehat{\beta}(\lambda - \Delta)$ must satisfy the following two system of equations:

$$\begin{cases}
q_S - \lambda \operatorname{sign}(\widehat{\beta}_S(\lambda)) &= X_S' X_S \widehat{\beta}_S(\lambda) \\
q_{S^c} - \lambda \operatorname{sign}(\widehat{\beta}_{S^c}(\lambda)) &= X_{S^c}' X_S \widehat{\beta}_S(\lambda)
\end{cases} ,$$
(22)

$$\begin{cases}
q_S - (\lambda - \Delta)\operatorname{sign}(\widehat{\beta}_S(\lambda - \Delta)) &= X_S' X_S \widehat{\beta}_S(\lambda - \Delta) \\
q_{S^c} - (\lambda - \Delta)\operatorname{sign}(\widehat{\beta}_{S^c}(\lambda - \Delta)) &= X_{S^c}' X_S \widehat{\beta}_S(\lambda - \Delta)
\end{cases}$$
(23)

From the above two system of equations, we have the following:

$$-(\lambda - \Delta)\operatorname{sign}(\widehat{\beta}_{S^c}(\lambda - \Delta)) = -\lambda\operatorname{sign}(\widehat{\beta}_{S^c}(\lambda)) + \Delta X'_{S^c}X_S(X'_SX_S)^{-1}\operatorname{sign}(\widehat{\beta}_S(\lambda)). \tag{24}$$

That is, if one decrease λ to $\lambda - \Delta$, one must strictly follow (24).

Following the above key step in the path following Lasso-algorithm, we represent a counter example as follows. Suppose $\beta_1 > \beta_2 > \beta_3 > \beta_4 = \beta_5 = \ldots = \beta_p = 0$. The model matrix $X = (X_1, X_2, X_3, \ldots, X_p)$, where $X_1, X_2 \in \mathbb{R}^n$ is the first two columns from a orthogonal matrix $(X_1, X_2, \widetilde{X}_3, \ldots, \widetilde{X}_p)$, and for $j \geq 3$, we have $X_j = \alpha_j X_1 + (1 - \alpha_j) X_2 + \sqrt{1 - \alpha_j^2 - (1 - \alpha_j)^2} \widetilde{X}_j$ with $\alpha_j \in (0, 1)$. The response vector y is generated by

$$y = \sum_{j=1}^{p} \beta_j X_j.$$

If β_1, β_2 are very large number, say, 200, 100, and β_3 is not that large, say, 1. Then the following algorithm works as follows:

- Loop 0: We start with $\lambda_0 = +\infty$, then we know that $\widehat{\beta}(\lambda_0) = 0$ and $S_0 = \emptyset$.
- Loop 1: When λ changes from $\lambda_0 = +\infty$ to $\lambda_1 = ||q||_{\infty}$, from (21), we know that $S_1 = \{1\}$.

- Loop 2: Similar to the first loop, when λ decrease to λ_2 , we have $S_2 = \{1, 2\}$.
- Loop 3: This is where problem happens. From (24), we know that $\forall \lambda_2 \Delta \in (\lambda_3, \lambda_2]$, we have

$$\operatorname{sign}(\widehat{\beta}_{S_2^c}(\lambda-\Delta)) = X_{S_2^c}' X_{S_2} (X_{S_2}' X_{S_2})^{-1} \operatorname{sign}(\widehat{\beta}_{S_2}(\lambda)).$$

Since $\operatorname{sign}(\widehat{\beta}_{S_2}(\lambda)) = (1,1)'$ and $X_{S_2} = (X_1,X_2), X_{S_2^c} = (X_3,\ldots,X_p)$, we have the right hand side of the above equation as a all-one vector, i.e, $(1,1,\ldots,1)^{\top}$. To make the left hand $\operatorname{side} \operatorname{sign}(\widehat{\beta}_{S_2^c}(\lambda_2 - \Delta))$ equals to $(1,1,\ldots,1)'$, we can only take $\Delta = \lambda_2$, which gives us $S_3 = \{1,2,3,\ldots,p\}$.

However, from the data generalization, we know that the true support set is $\{1,2,3\}$. Therefore, one will not be able to develop a path following algorithm to realize correct support set recovery. At least not in the sense of inserting one at a time to the support set. In the above example, since a path following approach can only visit three possible subsets, it won't solve the Lasso problem in general.

A.5.2 Details to Support the Second Drawback of Path Following Lasso-Algorithm

In this section, we provide more technical details to support the second drawback of path following Lasso-Algorithm. Recall the main idea of path following Lasso-Algorithm is that, it begins with a large λ_0 , which makes the estimated $\widehat{\beta}(\lambda_0) = 0$, and accordingly its support set $S_0 = \emptyset$ (empty set). Then it tries to identify a sequence of the penalty parameter λ as follows:

$$\lambda_0 > \lambda_1 > \lambda_2 > \ldots > \lambda_{T-1} > \lambda_T = 0.$$

such that for any $k \geq 1$, when we have $\lambda \in [\lambda_k, \lambda_{k-1}]$, the support of $\widehat{\beta}(\lambda)$ (which is a function of λ) i.e., S_k , remains unchanged. Moreover, within the interval $[\lambda_k, \lambda_{k-1}]$, vector $\widehat{\beta}(\lambda)$ elementwisely is a linear function of λ . However, when one is over the kink point, the support is changed/enlarged, i.e., we have $S_k \neq S_{k-1}$ or even $S_k \subseteq S_{k-1}$.

A point deserves attention is that, if T, the total number of kink points is small, then the path following algorithm is efficient, i.e., it only requires $O(nTp^2)$ numerical operations. In particular, if the size of supports are strictly increasing, i.e., we have

$$|S_{k-1}| < |S_k| \ \forall k > 1,$$

then we have $T \leq p$, and accordingly the computational complexity can be bounded by $O(np^3)$. However, it turns out bounding the value of T is an open question. In recent papers such as [39, 34], we can see that bounding T is an open problem.

B An Important Theorem

Our proof will rely on a result on the number of steps in achieving certain accuracy in using the accelerate gradient descent (AGD) when the objective function is strongly convex. The result is the Theorem 3.7 in [22]. We represent the theorem here for readers' convenience. We introduce some notations first. Suppose ones wants to minimize a convex function $f: X \to \mathbb{R}$ in a feasible closed convex set $X \in \mathbb{R}^p$. We further assume that f is a differentiable convex function with Lipschitz continuous gradients L, i.e., $\forall x, y \in X$, we have

$$\|\nabla f(x) - \nabla f(y)\|_{2} \le L \|x - y\|_{2}$$

where $\nabla f(x)$ represents the gradient of function f(x). Furthermore, we assume that f is a strongly convex function, i.e., $\forall x, y \in X$, there exist $\mu > 0$, such that we have

$$f(y) \ge f(x) + \nabla f(x)(y - x) + \frac{\mu}{2} \|y - x\|_2^2$$
.

This type of function f is called the L-smooth and μ -strongly convex function. Recall that our objective is to solve the following problem:

$$\min_{x \in X} f(x).$$

In the following, we present one version of the accelerated gradient descent (AGD) algorithm. Given $(x^{(t-1)}, \bar{x}^{(t-1)}) \in X \times X$ for t = 1, 2, ..., we set

$$\underline{x}^{(t)} = (1 - q_t)\bar{x}^{(t-1)} + q_t x^{(t-1)} \tag{25}$$

$$x^{(t)} = \arg\min_{x \in X} \left\{ \gamma_t \left[x' \nabla f\left(\underline{x}^{(t)}\right) + \mu V\left(\underline{x}^{(t)}, x\right) \right] + V\left(x^{(t-1)}, x\right) \right\}$$
 (26)

$$\bar{x}^{(t)} = (1 - \alpha_t) \, \bar{x}^{(t-1)} + \alpha_t x^{(t)}, \tag{27}$$

for some $q_t \in [0,1], \gamma_t \geq 0$, and $\alpha_t \in [0,1]$. And here V(x,z) is the prox-function (or Bregman's distance), i.e.,

$$V(x, z) = v(z) - [v(x) + (z - x)'\nabla v(x)],$$

with $v(x) = ||x||_2^2/2$. By applying AGD as shown in (25)-(27), the following theorem presents an inequality that can be utilized to determine the number of loops when certain precision of a solution is given.

Theorem B.1. Let $(\underline{x}^{(t)}, x^{(t)}, \bar{x}^{(t)}) \in X \times X \times X$ be generated by accelerated gradient descent method in (25)-(27). If $\alpha_t = \alpha, \gamma_t = \gamma$ and $q_t = q$, for $t = 1, \ldots, k$, satisfy $\alpha \geq q$, $\frac{L(\alpha - q)}{1 - q} \leq \mu$, $\frac{Lq(1 - \alpha)}{1 - q} \leq \frac{1}{\gamma}$, and $\frac{1}{2(1 - \alpha)} \leq \mu + \frac{1}{2}$, then for any $x \in X$, we have

$$f\left(\bar{x}^{(k)}\right) - f(x) + \alpha\left(\mu + \frac{1}{\gamma}\right)V\left(x^{(k-1)}, x\right) \leq (1 - \alpha)^k \left[f\left(\bar{x}^{(0)}\right) - f(x) + \alpha\left(\mu + \frac{1}{\gamma}\right)V\left(x^{(1)}, x\right),\right].$$

In particular, if

$$\alpha = \sqrt{\frac{\mu}{L}}, q = \frac{\alpha - \mu/L}{1 - \mu/L}, \gamma = \frac{\alpha}{\mu(1 - \alpha)}$$

then for any $x \in X$, we have

$$f\left(\bar{x}^{(k)}\right) - f(x) + \alpha\left(\mu + \frac{1}{\gamma}\right)V\left(x^{(k-1)}, x\right) \le \left(1 - \sqrt{\frac{\mu}{L}}\right)^k \left[f\left(\bar{x}^{(0)}\right) - f(x) + \alpha\left(\mu + \frac{1}{\gamma}\right)V\left(x^{(1)}, x\right)\right]. \tag{28}$$

The above theorem gives a convergence rate of AGD under the scenario when the objective function is strongly convex. This result will be utilized in the proof of Theorem 3.1, which can be found in C.3.

C Proofs

C.1 Proof of a Lemma

The proof of Lemma 2.6 is as follows.

Proof. In this proof, we will do two parts.

First, we will prove the existence of the initial point t_0 stated in (8). We know that

$$\lim_{t \to +\infty} \frac{\sum_{j=1}^{p} M(t)_{ij} (X'y/n)_{j}}{t} = \lim_{t \to +\infty} \sum_{j=1}^{p} \left(\left[\frac{X'X}{n} + \frac{\lambda \left[\log(1+t) \right]^{2}}{3t^{3}} I \right]^{-1} \right)_{ij} \left(\frac{X'y}{n} \right)_{j} \frac{1}{t}$$

$$= \lim_{t \to +\infty} \sum_{j=1}^{p} \left(\left[\frac{X'Xt}{n} + \frac{\lambda \left[\log(1+t) \right]^{2}}{3t^{2}} I \right]^{-1} \right)_{ij} \left(\frac{X'y}{n} \right)_{j}.$$

$$= 0.$$

The above indicates that when t is very large, the t_0 defined in (8) will exist. Next, we will verify that, if t_0 is chosen as shown in (8), i.e,

$$t_0 \in \left\{ t : \left| \sum_{j=1}^p M(t)_{ij} (X'y/n)_j \right| \le t, \forall i = 1, \dots, p \right\},$$

we have $\left|\beta_i^{(0)}\right| < t_0$. It can be verified that,

$$M(t)\frac{X'y}{n} = \arg\min_{\beta} \left\{ \underbrace{\frac{1}{2n} \|y - X\beta\|_{2}^{2} + \frac{1}{3t^{3}} \left[\log(1+t)\right]^{2} \beta' \beta}_{\mathcal{G}(\beta)} \right\}, \tag{29}$$

where $\mathcal{G}(\beta)$ is a special case of $F_t(\beta)$ when t is large enough to include all the coefficient β_i into the interval [-t,t]. Utilizing the above fact that the minimizer in (29) when $t=t_0$ satisfies the condition that its coordinates are within $[-t_0, t_0]$, we have

$$|\beta_i(t_0)| = \left| \left(M(t_0) \frac{X'y}{n} \right)_i \right| = \left| \sum_{j=1}^p M(t_0)_{ij} (X'y/n)_j \right| \le t_0.$$

Thus, if we choose t_0 as shown in (8), i.e.,

$$t_0 \in \left\{ t : \left| \sum_{j=1}^p M(t)_{ij} (X'y/n)_j \right| \le t, \forall i = 1, \dots, p \right\},$$

we can verify that $\forall i = 1, 2, \dots p, |\beta_i(t_0)| \leq t_0$, i.e., $|\beta_i^{(0)}| \leq t_0$.

C.2Proof of a Lemma

The Proof of Lemma 2.3 is as follows.

Proof. Because $f_t(x)$ is a even function, we only consider the positive x in the remaining of the proof.

First, when $0 \le x \le t$, one has

$$f_t(x) - x = \frac{1}{3t^3} \left[\log(1+t) \right]^2 x^2 - x,$$

which is a quadratic function with the axis of symmetry, $\frac{3t^3}{2[\log(1+t)]^2}$ being larger than t. Therefore, one has

$$\frac{1}{3t} [\log(1+t)]^2 - t \le f_t(x) - x \le 0.$$

Then we discuss the scenario when x > t, where

$$f_t(x) - x = \left[\left[\frac{\log(1+t)}{t} \right]^2 - 1 \right] x + \frac{1}{3x} \left[\log(1+t) \right]^2 - \frac{1}{t} \left[\log(1+t) \right]^2,$$

which is a decreasing function of variable x. Therefore,

$$f_t(B) - B = [f_t(x) - x]|_{x=B} \le f_t(x) - x \le [f_t(x) - x]|_{x=t} = f_t(t) - t,$$

where we further have $f_t(t) - t = \frac{1}{3t} \left[\log(1+t) \right]^2 - t \le 0$. By the combination of two scenario $(x \le t \text{ and } x > t)$, we prove the statement in equation

C.3 Proof of a Theorem

The proof of Theorem 3.1 can be found below.

Proof. To begin with, we revisit some notations in linear algebra. For matrix A, we use A_{ij} to indicate the (i, j)th entry in matrix A. Besides, its maximal/minimal eigenvalue is $\lambda_{\min}(A)/\lambda_{\min}(A)$, respectively.

It is known that, the condition number of function $F_{t_k}(\beta) = \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_{t_k}(\beta)$ is defined by the ratio between the maximal and minimal eigenvalue of its Hessian. Recall that, the (i, j)th entry of the Hessian matrix of the surrogate function $f_{t_k}(\beta)$, noted as $H_{t_k,i,j}$, is

$$H_{t_k,i,j} = \begin{cases} \frac{2}{3} \left[\log(1+t_k) \right]^2 \max\{|\beta_i|,t_k\}^{-3}, & \text{if } i = j, \\ 0, & \text{otherwise.} \end{cases}$$

Note that the Hessian matrix H_{t_k} is diagonal and positive definite; therefore one can easily find its minimum and maximum eigenvalues. So the condition number of function $F_{t_k}(\beta) = \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_{t_k}(\beta)$, noted as κ_k , is

$$\kappa_{k} = \frac{\lambda_{\max} \left(\frac{X'X}{n} + \lambda H_{t_{k}} \right)}{\lambda_{\min} \left(\frac{X'X}{n} + \lambda H_{t_{k}} \right)}$$

$$\leq \frac{\lambda_{\max} \left(\frac{X'X}{n} \right) + \lambda \lambda_{\max} \left(H_{t_{k}} \right)}{\lambda_{\min} \left(\frac{X'X}{n} \right) + \lambda \lambda_{\min} \left(H_{t_{k}} \right)}$$

$$\leq \frac{\lambda_{\max} \left(\frac{X'X}{n} \right) + \lambda \lambda_{\max} \left(H_{t_{k}} \right)}{\lambda \lambda_{\min} \left(H_{t_{k}} \right)}$$

$$= \frac{\lambda_{\max} \left(\frac{X'X}{n} \right) + \frac{2\lambda}{3t_{k}^{3}} \left[\log(1 + t_{k}) \right]^{2}}{\frac{2\lambda}{3x^{3}} \left[\log(1 + t_{k}) \right]^{2}}$$

$$= \frac{3x^{3} \lambda_{\max} \left(\frac{X'X}{n} \right)}{2\lambda \left[\log(1 + t_{k}) \right]^{2}} + \frac{x^{3}}{t_{k}^{3}}$$

$$\leq \frac{3B^{3} \lambda_{\max} \left(\frac{X'X}{n} \right)}{2\lambda \left[\log(1 + \tau_{k}) \right]^{2}} + \left(\frac{B}{\tau} \right)^{3}.$$
(33)

Equation (30) is due to the definition of the condition number. Inequality (31) is because of the two fact. First, for the maximal eigenvalue of summation of two matrix A + B, i.e., $\lambda_{\max}(A + B)$, is no more than summation of maximal eigenvalue separately, $\lambda_{\max}(A) + \lambda_{\max}(B)$. Second, similar to the maximal eigenvalue, the minimal eigenvalue follows the similar rule that $\lambda_{\min}(A + B) \ge \lambda_{\min}(A) + \lambda_{\min}(B)$. The equality in (32) is due to the fact that matrix H_{t_k} is diagonal with positive diagonal entries. The x in (32) refers to

$$x = \max\{|\beta_i| : \beta_i \text{ is the } i\text{th entry in } \beta\}.$$

Inequality (33) is because that $t_k \ge \tau$ and we assume that throughout the algorithm, all elements in $\beta^{(k)}(k=1,2...)$ is bounded by B.

By calling Theorem 3.7 in [22] (i.e., the theorem in B in this paper), we can prove the statement in Theorem 3.1. The details of the proof are listed as follows. Recall that we want to minimize $F_{t_k}(\beta)$ for a fixed k. From the previous analysis, we can find that $F_{t_k}(\beta)$ is L_k -smooth and μ_k -strongly convex, where $L_k = \lambda_{\max}\left(\frac{X'X}{n} + \lambda H_{t_k}\right)$ and $\mu_k = \lambda_{\min}\left(\frac{X'X}{n} + \lambda H_{t_k}\right)$. Consequently, the

condition number in the kth outer-loop $\kappa_k = \frac{L_k}{\mu_k}$ can be upper bounded by $\frac{3B^3 \lambda_{\max}\left(\frac{X'X}{n}\right)}{2\lambda[\log(1+\tau)]^2} + \left(\frac{B}{\tau}\right)^3$ for any $\{k = 0, 1, 2, \ldots : t_k \ge \tau\}$.

For a fixed k, when applying AGD to minimize $F_{t_k}(\beta)$, our steps, which are line 8-10 in Algorithm 2, follows the AGD steps that are presented in (25)-(27), by setting $\alpha_k = \sqrt{\frac{\mu_k}{L_k}}, q_k = \frac{\alpha_k - \mu_k/L_k}{1 - \mu_k/L_k}, \gamma_k = \frac{\alpha_k}{\mu_k(1 - \alpha_k)}$. According to (28) in Theorem B.1, if

$$(1 - \alpha_{k})^{s} \underbrace{\left[F_{t_{k}}(\bar{\beta}^{(k)[0]}) - F_{k,\min} + \alpha_{k} \left(\mu_{k} + \frac{1}{\gamma_{k}} \right) V(\beta^{(k-1)[1]}, \widehat{\beta}_{k}) \right]}_{C_{k}} - \underbrace{\alpha_{k} \left(\mu_{k} + \frac{1}{\gamma_{k}} \right) V(\beta^{(k-1)[s-1]}, \widehat{\beta}_{k})}_{D_{k}} \leq \widetilde{\epsilon}_{k}, \tag{34}$$

then $F_{t_k}(\beta^{(k)[s]}) - F_{k,\min} \leq \widetilde{\epsilon}_k$ with a given $\widetilde{\epsilon}_k$. Here in (34), we have $\widehat{\beta}_k = \arg\min_{\beta} F_{t_k}(\beta)$ and function $V(\cdot, \cdot)$ has been defined in B.

We then solve the inequality in (34) to get an explicit formula for the quantity s. To achieve this goal, we simplify (34) first. Note that quantities C_k and D_k are defined via underlining in (34). It can be verified that $D_k \geq 0$. This is because $v(x) = ||x||_2^2/2$ (recall the definition of v(x) in B) is a convex function, i.e., we have

$$V(\beta^{(k-1)[s-1]}, \widehat{\beta}_k) = v(\widehat{\beta}_k) - \left[v(\beta^{(k-1)[s-1]}) + (\widehat{\beta}_k - \beta^{(k-1)[s-1]})' \nabla v(\beta^{(k-1)[s-1]})\right] \geq 0.$$

Since $\mathcal{D}_k > 0$, if we have

$$(1 - \alpha_k)^s C_k \leq \widetilde{\epsilon}_k$$

then the inequality in (34) will be satisfied. By introducing simple linear algebra, the above inequality can be rewritten as

$$(1 - \alpha_k)^s \le \frac{\widetilde{\epsilon}_k}{\mathcal{C}_k}.$$

By taking logarithm of both sides, we have

$$s \log (1 - \alpha_k) \le \log \left(\frac{\widetilde{\epsilon}_k}{C_k}\right),$$

which gives

$$s \ge \frac{-\log\left(\frac{\tilde{\epsilon}_k}{C_k}\right)}{-\log\left(1 - \alpha_k\right)} = \frac{\log\left(\frac{C_k}{\tilde{\epsilon}_k}\right)}{-\log\left(1 - \alpha_k\right)}.$$
 (35)

Furthermore, we know $\log \left(\frac{1}{1-x}\right) \ge x$ for 0 < x < 1, so if

$$s \ge \frac{\log\left(\frac{C_k}{\tilde{\epsilon}_k}\right)}{C_k},\tag{36}$$

then the inequality in (35) holds. In summary, if we have (36), then we have $F_{t_k}(\beta^{(k)[s]}) - F_{t_k}(\widehat{\beta}_k) < \widetilde{\epsilon}_k$.

Now we will show that, both $\frac{1}{\alpha_k} = \sqrt{\frac{L_k}{\mu_k}}$ and $\log(\mathcal{C}_k)$ in (36) can be bounded by a constant that does not depend on k (or equivalently, t_k). First, we prove that $\frac{1}{\alpha_k} = \sqrt{\frac{L_k}{\mu_k}}$ can be bound. This is essentially the argument that have been used in the step (33). Second, we prove that \mathcal{C}_k is also bounded. Because we have

$$C_k = \underbrace{F_{t_k}(\beta^{(k)[0]}) - F_{k,\min}}_{C_{k,1}} + \underbrace{\alpha_k \left(\mu_k + \frac{1}{\gamma_k}\right)}_{C_{k,2}} \underbrace{V(\beta^{(k-1)[1]}, \widehat{\beta}_k)}_{C_{k,3}}.$$

Note that quantities $C_{k,1}$, $C_{k,2}$, and $C_{k,3}$ are defined via underlining in the above equation. It is evident that $C_{k,1}$ and $C_{k,3}$ are bounded. For $C_{k,2}$, we have

$$C_{k,2} = \mu_k,$$

because we set $\gamma_k = \frac{\alpha_k}{\mu_k(1-\alpha_k)}$. Since μ_k is bounded above by a constant, quantity $\mathcal{C}_{k,2}$ is bounded as well. By combining the above several block, we know $\log(\mathcal{C}_k)$ is bounded.

In conclusion, after $C_1 \log(1/\tilde{\epsilon}_k)$ inner-loops, one is guaranteed to achieve the following precision

$$F_{t_k}(\beta^{(k)}) - F_{\min,k} \le \widetilde{\epsilon}_k,$$

where $\widetilde{\epsilon}_k = \frac{\lambda p}{3B} \left[\log(1 + t_k) \right]^2$ and C_1 is a constant that does not depend on the value of t_k (or k).

C.4 Proof of a Theorem

The proof of Theorem 3.2 is as follows.

Proof. We start by showing that, for any $t \geq 0$, one has

$$F(\beta^{(k)}) - F(\widehat{\beta}) \le \lambda p(2B+1)t_k.$$

This is because of the following sequence of inequalities for any $\beta \in \mathbb{R}^p$:

$$F(\beta^{(k)}) = \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \|\beta^{(k)}\|_{1}$$

$$= \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} |\beta_{i}^{(k)}|$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}^{(k)}) - \lambda p [f_{t_{k}}(x) - x] |_{x=B}$$

$$= \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}^{(k)}) +$$

$$\lambda pB \left[1 - \left[\frac{\log(1 + t_{k})}{t_{k}}\right]^{2}\right] - \frac{\lambda p}{3B} [\log(1 + t_{k})]^{2} + \frac{\lambda p}{t_{k}} [\log(1 + t_{k})]^{2}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}^{(k)}) +$$

$$\lambda pB \left[1 - \left(\frac{1}{1 + t_{k}}\right)^{2}\right] - \frac{\lambda p}{3B} [\log(1 + t_{k})]^{2} + \lambda pt_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}^{(k)}) + 2\lambda pBt_{k} - \frac{\lambda p}{3B} [\log(1 + t_{k})]^{2} + \lambda pt_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}^{(k)}) + \lambda p(2B + 1)t_{k} - \frac{\lambda p}{3B} [\log(1 + t_{k})]^{2} + \lambda pt_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}^{(k)}) + \lambda p(2B + 1)t_{k} - \frac{\lambda p}{3B} [\log(1 + t_{k})]^{2} + \tilde{\epsilon}_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}^{(k)}) + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}) + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}) + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \sum_{i=1}^{p} f_{t_{k}}(\beta_{i}) + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta^{(k)}\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

$$\leq \frac{1}{2n} \|y - X\beta\|_{2}^{2} + \lambda \|\beta\|_{1} + \lambda p(2B + 1)t_{k}$$

where inequality (37) is due to the left side hand of inequality (6), i.e., $[f_{t_k}(x) - |x|]|_{x=B} \le f_t(x) - |x|$. And equation (37) is by plugging in the value of $[f_{t_k}(x) - x]|_{x=t_0}$. Inequality (39) utilizes the inequality that $\frac{t_k}{1+t_k} \le \log(1+t_k)$ and inequality $\log(1+t_k) \le t_k$. Inequality (40) uses inequality $1-\frac{1}{(1+t_k)^2} \le 2t_k$. Inequality (41) is because that we assume the precision in kth inner-loop is $F_{t_k}(\beta^{(k)}) - F_{t_k}(\widehat{\beta}^{(k)}) \le \widetilde{\epsilon}_k$. Equation (42) is owing to the fact that we set $\widetilde{\epsilon}_k = \frac{\lambda p}{3B} [\log(1+t_k)]^2$. Inequality (44) is due to the right hand side of inequality (6), i.e., $f_t(x) - |x| \le 0$. Inequality (43) is because $\widehat{\beta}^{(k)}$ is the minimizer of $F_{t_k}(\beta)$, so $F_{t_k}(\widehat{\beta}^{(k)}) < F_{t_k}(\widehat{\beta})$. Inequality (44) is because $f_{t_k}(x) - |x| \le 0$ in Lemma 2.3.

Through the above series of equalities and inequalities, we know that

$$F(\beta^{(k)}) - F(\widehat{\beta}) \le \lambda p(2B+1)t_k. \tag{45}$$

Besides, in the statement of the theorem, we have

$$k \ge \frac{-1}{\log(1-h)}\log\left(\frac{\lambda p(2B+1)t_0}{\epsilon}\right)$$

which is equivalent to

$$\lambda p(2B+1)t_k \leq \epsilon$$
.

So the right side of inequality (45) isn't larger than ϵ . Thus, we prove that, when $k \geq \frac{-1}{\log(1-h)} \log \left(\frac{\lambda p(2B+1)t_0}{\epsilon}\right)$, we have $F(\beta^{(k)}) - F(\widehat{\beta}) \leq \epsilon$.

C.5 Proof of a Theorem

The proof of Theorem 3.3 is as follows.

Proof. The total number of numeric operations is determined by three factors, namely (1) the number of out-loops, (2) the number of inner-loops, and (3) the number of numeric operations in each inner-loops. We adopt the assumption that different basic operations can be treated equally. We have discussed (1) and (2) in Section 3, and we discuss (3) briefly here. The main computational cost of an inner-loop in our proposed algorithm lies in Line 9 of Algorithm 2, which is the matrix multiplication in $\frac{\partial}{\partial \beta^{(k)[s]}} F_{t_k}(\beta^{(k)[s]}) = \frac{X'X}{n} \beta^{(k)[s]} - \frac{X'y}{n} + \frac{\partial}{\partial \beta^{(k)[s]}} f_{t_k}(\beta^{(k)[s]})$. With matrix $\frac{X'X}{n}, \frac{X'y}{n}$ being pre-calculated and stored at the beginning of the execution, the calculation of $\frac{\partial}{\partial \beta^{(k)[i]}} F_{t_k}(\beta^{(k)[s]})$ requires $O(p^2)$ operations.

Now we count the total number of numerical operations that are need in our proposed method to achieve the ϵ precision. We know that to achieve $F(\beta^{(k)}) - F_{\min} < \epsilon$, we need at least (Theorem 3.2)

$$N \stackrel{\Delta}{=} \frac{-1}{\log(1-h)} \log \left(\frac{\lambda p(2B+1)t_0}{\epsilon} \right)$$

outer-loops. Furthermore, we know that the number inner-loop in an inner-loop k is $O(\log(\frac{1}{\tilde{\epsilon}_k}))$ with a hidden constant which can be universally bounded, and the number of operations in each inner-loop is p^2 . Therefore, the total number of numerical operations to get the estimator $\beta^{(k)}$ with

precision $F(\beta^{(k)}) - F(\widehat{\beta}) \leq \epsilon$ can be upper bounded by the following quantity:

$$p^{2} \sum_{k=1}^{N} \log \left(\frac{1}{\tilde{\epsilon}_{k}} \right) = p^{2} \sum_{k=1}^{N} \log \left(\frac{3B}{\lambda p} \left[\log(1 + t_{k}) \right]^{2} \right]^{-1} \right)$$

$$= p^{2} \sum_{k=1}^{N} \log \left(\frac{3B}{\lambda p} \right) - p^{2} \sum_{k=1}^{N} \log \left([\log(1 + t_{k})]^{2} \right)$$

$$= p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} \sum_{k=1}^{N} \log (\log(1 + t_{k}))$$

$$\leq p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} \sum_{k=1}^{N} \log \left(\frac{t_{k}}{1 + t_{k}} \right)$$

$$= p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} \sum_{k=1}^{N} \log \left(t_{k} \right) + 2p^{2} \sum_{k=1}^{N} \log \left(1 + t_{k} \right)$$

$$= p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} \sum_{k=1}^{N} \log \left(t_{0} (1 - h)^{k} \right) + 2p^{2} \sum_{k=1}^{N} \log \left(1 + t_{k} \right)$$

$$\leq p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} \sum_{k=1}^{N} \log \left(t_{0} (1 - h)^{k} \right) + 2p^{2} \sum_{k=1}^{N} t_{k}$$

$$= p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} \sum_{k=1}^{N} \log \left(t_{0} (1 - h)^{k} \right) + 2p^{2} \sum_{k=1}^{N} t_{0} (1 - h)^{k}$$

$$= p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} N \log \left(t_{0} \right) - 2p^{2} \log \left(1 - h \right) \sum_{k=1}^{N} k + 2p^{2} \sum_{k=1}^{N} t_{0} (1 - h)^{k}$$

$$= p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} N \log \left(t_{0} \right) - 2p^{2} \log \left(1 - h \right) \sum_{k=1}^{N} k + 2p^{2} \sum_{k=1}^{N} t_{0} (1 - h)^{k}$$

$$= p^{2} N \log \left(\frac{3B}{\lambda p} \right) - 2p^{2} N \log \left(t_{0} \right) - 2p^{2} \log \left(1 - h \right) \frac{(N+1)N}{2}$$

$$+ 2p^{2} \frac{t_{0} \left[1 - (1 - h)^{N} \right]}{h}$$

$$= O(N^{2})$$

where equality (46) is derived by plugging in that $\tilde{\epsilon}_k = \frac{\lambda p}{3B} [\log(1+t_k)]^2$. To be more exactly, there is a hidden constant related to the big O notation in $O\left(\log\left(\frac{1}{\tilde{\epsilon}_k}\right)\right)$ in equality (46), however, as mention in the proof of Theorem 3.1, this hidden constant can be bounded universally. So in equality (46), we omit this hidden constant. Inequality (47) is derived due to the inequality that $\log(1+x) \geq \frac{x}{1+x}$ for $x \geq 0$. Inequality (48) is derived due to the inequality that $\log(1+x) \leq x$ for x > 0.

C.6 Proof of a Proposition

The proof of Proposition 5.1 is as follows.

Proof. Because $\hat{\beta}$ is the minimizer of $\frac{1}{2} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$, we can get its first-order condition as:

$$\frac{1}{n}\left(X'X\widehat{\beta} + X'y\right) + \lambda \operatorname{sign}\left(\widehat{\beta}\right) = 0 \tag{49}$$

And because $\widetilde{\beta}$ is the minimizer of $\frac{1}{2} \|y - X\beta\|_2^2 + \lambda f_t(\beta)$, we can get its first-order condition as:

$$\frac{1}{n}\left(X'X\widetilde{\beta} + X'y\right) + \lambda \nabla f_t\left(\widetilde{\beta}\right) = 0,\tag{50}$$

where $\nabla f_t\left(\widetilde{\beta}\right)$ is the gradient of $f_t\left(\widetilde{\beta}\right)$. By subtracting (49) from (50), we have

$$\frac{1}{n}X'X\left(\widetilde{\beta}-\widehat{\beta}\right)+\lambda\left[\nabla f_{t}\left(\beta\right)-\operatorname{sign}\left(\widehat{\beta}\right)\right]=0.$$

By left multiplying $(\widetilde{\beta} - \widehat{\beta})'$ on both sides of the above equation, we have

$$\frac{1}{n}\left(\widetilde{\beta}-\widehat{\beta}\right)'X'X\left(\widetilde{\beta}-\widehat{\beta}\right)+\lambda\left(\widetilde{\beta}-\widehat{\beta}\right)'\left[\nabla f_t\left(\widetilde{\beta}\right)-\operatorname{sign}\left(\widehat{\beta}\right)\right]=0.$$

The above is equivalent to

$$\frac{1}{n} \left(\widetilde{\beta} - \widehat{\beta} \right)' X' X \left(\widetilde{\beta} - \widehat{\beta} \right) = -\lambda \left(\widetilde{\beta} - \widehat{\beta} \right)' \left[\nabla f_t \left(\widetilde{\beta} \right) - \operatorname{sign} \left(\widehat{\beta} \right) \right]
= -\lambda \left(\widetilde{\beta} - \widehat{\beta} \right)' \nabla f_t \left(\widetilde{\beta} \right) + \lambda \left(\widetilde{\beta} - \widehat{\beta} \right)' \operatorname{sign} \left(\widehat{\beta} \right)
= -\lambda \left(\widetilde{\beta} - \widehat{\beta} \right)' \nabla f_t \left(\widetilde{\beta} \right) + \lambda \widetilde{\beta}' \operatorname{sign} \left(\widehat{\beta} \right) - \lambda \widetilde{\beta}' \operatorname{sign} \left(\widehat{\beta} \right)
= -\lambda \left(\widetilde{\beta} - \widehat{\beta} \right)' \nabla f_t \left(\widetilde{\beta} \right) + \lambda \widetilde{\beta}' \operatorname{sign} \left(\widehat{\beta} \right) - \lambda \left\| \widehat{\beta} \right\|_{1}.$$

Because $f_t(\beta)$ is a convex function, we have

$$\frac{1}{n} \left(\widetilde{\beta} - \widehat{\beta} \right)' X' X \left(\widetilde{\beta} - \widehat{\beta} \right) \le -\lambda \left[f_t \left(\widehat{\beta} \right) - f_t \left(\widetilde{\beta} \right) \right] + \lambda \widetilde{\beta}' \operatorname{sign} \left(\widehat{\beta} \right) - \lambda \left\| \widehat{\beta} \right\|_{1}.$$

So we have

$$\frac{1}{n} \left\| X \left(\widetilde{\beta} - \widehat{\beta} \right) \right\|_{2}^{2} \leq -\lambda \left[f_{t} \left(\widehat{\beta} \right) - f_{t} \left(\widetilde{\beta} \right) \right] + \lambda \left\| \widetilde{\beta} \right\|_{1} - \lambda \left\| \widehat{\beta} \right\|_{1}.$$

When $t \to 0$, we have $f_t(\beta)$ very close to $\|\beta\|_1$, so we have $\frac{1}{n} \|X(\widetilde{\beta} - \widehat{\beta})\|_2^2 \to 0$.

C.7 Proof of a Proposition

The proof of Proposition 5.2 is as follows.

Proof. From Proposition 5.1, we know that

$$\left\| X \left(\widetilde{\beta} - \widehat{\beta} \right) \right\|_{2}^{2} \to 0$$

when $t \to 0$, where $\widehat{\beta} = \arg\min_{\beta} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$, and $\widetilde{\beta} = \arg\min_{\beta} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda f_t(\beta)$. The above can be written as $X_S \left(\widetilde{\beta}_S - \widehat{\beta}_S\right) + X_{S^c} \widetilde{\beta}_{S^c} = \delta, \tag{51}$

where S is the support set of $\widehat{\beta}$ and $\|\delta\|_2^2 \approx 0$. By left multiplying $(X_S'X_S)^{-1}X_S'$ on both sides of (51), we have

$$\left(\widetilde{\beta}_S - \widehat{\beta}_S\right) + \left(X_S'X_S\right)^{-1} X_S'X_{S^c}\widetilde{\beta}_{S^c} = \left(X_S'X_S\right)^{-1} X_S'\delta,\tag{52}$$

By left multiplying $X_{S^c}^{\dagger}$ on both sides of (51), we have

$$X_{S^c}^{\dagger} X_S \left(\widetilde{\beta}_S - \widehat{\beta}_S \right) + X_{S^c}^{\dagger} X_{S^c} \widetilde{\beta}_{S^c} = X_{S^c}^{\dagger} \delta, \tag{53}$$

where $X_{S^c}^{\dagger}$ is the *pseudo-inverse* of matrix X_{S^c} . The mathematical meaning of pseudo-inverse is that, suppose $X_{S^c} = U\Sigma V$, which is the singular value decomposition (SVD) of X_{S^c} . Then

 $X_{S^c}^{\dagger} = V' \Sigma^{\dagger} U'$. For the rectangular diagonal matrix Σ , we get Σ^{\dagger} by taking the reciprocal of each non-zero elements on the diagonal, leaving the zeros in place, and then transposing the matrix.

By reorganizing (52) and (53) into block matrix, we have

$$\underbrace{\begin{pmatrix} I & (X_S'X_S)^{-1} X_S'X_{S^c} \\ X_{S^c}^{\dagger} X_S & X_{S^c}^{\dagger} X_{S^c} \end{pmatrix}}_{M} \begin{pmatrix} \widetilde{\beta}_S - \widehat{\beta}_S \\ \widetilde{\beta}_{S^c} \end{pmatrix} = \begin{pmatrix} (X_S'X_S)^{-1} X_S'\delta \\ X_{S^c}^{\dagger} \delta \end{pmatrix}.$$

Through this system of equations, we can solve $\left\| \begin{pmatrix} \widetilde{\beta}_S - \widehat{\beta}_S \\ \widetilde{\beta}_{S^c} \end{pmatrix} \right\|_2^2$ as

$$\left\| \left(\begin{array}{c} \widetilde{\beta}_S - \widehat{\beta}_S \\ \widetilde{\beta}_{S^c} \end{array} \right) \right\|_2^2 = \left\| \widetilde{\beta}_S - \widehat{\beta}_S \right\|_2^2 + \left\| \widetilde{\beta}_{S^c} \right\|_2^2 = \left\| M^{-1} \left(\begin{array}{c} (X_S' X_S)^{-1} X_S' \delta \\ X_{S^c}^\dagger \delta \end{array} \right) \right\|_2^2.$$

Because for a matrix A and vector x, we have $||Ax||_2^2 \le ||A||_F^2 ||x||_2^2$, we can bound $||\widetilde{\beta}_S - \widehat{\beta}_S||_2^2 + ||\widetilde{\beta}_{S^c}||_2^2$ as

$$\begin{aligned} \left\| \widetilde{\beta}_{S} - \widehat{\beta}_{S} \right\|_{2}^{2} + \left\| \widetilde{\beta}_{S^{c}} \right\|_{2}^{2} & \leq \| M^{-1} \|_{F}^{2} \left\| \left(\frac{(X'_{S}X_{S})^{-1} X'_{S} \delta}{X_{S^{c}}^{\dagger} \delta} \right) \right\|_{2}^{2} \\ & = \| M^{-1} \|_{F}^{2} \left(\left\| (X'_{S}X_{S})^{-1} X'_{S} \delta \right\|_{2}^{2} + \left\| X_{S^{c}}^{\dagger} \delta \right\|_{2}^{2} \right). \end{aligned}$$

Because $\|M^{-1}\|_F \leq \sqrt{\operatorname{rank}(M^{-1})} \|M^{-1}\|_2$, we can further bound $\|\widetilde{\beta}_S - \widehat{\beta}_S\|_2^2 + \|\widetilde{\beta}_{S^c}\|_2^2$ as

$$\|\widetilde{\beta}_{S} - \widehat{\beta}_{S}\|_{2}^{2} + \|\widetilde{\beta}_{S^{c}}\|_{2}^{2}$$

$$\leq \operatorname{rank}(M^{-1}) \|M^{-1}\|_{2}^{2} \left(\|(X_{S}'X_{S})^{-1} X_{S}'\delta\|_{2}^{2} + \|X_{S^{c}}^{\dagger}\delta\|_{2}^{2} \right)$$

$$= \operatorname{rank}(M^{-1}) \left[\frac{1}{\lambda_{\min}(M)} \right]^{2} \left(\|(X_{S}'X_{S})^{-1} X_{S}'\delta\|_{2}^{2} + \|X_{S^{c}}^{\dagger}\delta\|_{2}^{2} \right). \tag{54}$$

For $\left\| \left(X_S' X_S \right)^{-1} X_S' \delta \right\|_2^2$ in (54), we have

$$\left\| \underbrace{(X_S'X_S)^{-1} X_S'}_{Q} \delta \right\|_{2}^{2} = \|Q\delta\|_{2}^{2}$$

$$= \sum_{i=1}^{|S|} (q_i'\delta)^{2}$$

$$\leq \sum_{i=1}^{|S|} \|q_i\|_{2}^{2} \|\delta\|_{2}^{2}$$

$$= \|Q\|_{F}^{2} \|\delta\|_{2}^{2},$$

where q_i' denotes the *i*th row in matrix Q, and Q denotes $(X_S'X_S)^{-1}X_S'$. Because $\|Q\|_F^2$ is bounded and $\|\delta\|_2^2 \to 0$, we have $\|(X_S'X_S)^{-1}X_S'\delta\|_2^2 \to 0$.

For $\left\|X_{S^c}^{\dagger}\delta\right\|_2^2$ in (54), following the similar logic, we have

$$\left\|X_{S^c}^{\dagger}\delta\right\|_2^2 \leq \left\|X_{S^c}^{\dagger}\right\|_F^2 \left\|\delta\right\|_2^2$$

Because $\left\|X_{S^c}^{\dagger}\right\|_F^2$ is bounded and $\left\|\delta\right\|_2^2 \to 0$, we have $\left\|X_{S^c}^{\dagger}\delta\right\|_2^2$.

For $\lambda_{\min}(M)$ in (54), let's start with a general eigenvalue of matrix M, and we denote the eigenvalue of M as $\lambda(M)$. If we prove that all the eigenvalue of matrix M is strictly larger than 0, than $\frac{1}{\lambda_{\min}}(M)$ can be bounded. This is equivalent to prove that $M - \lambda(M)I$ is positive semidefinite for any eigenvalue $\lambda(M)$.

If we denote $M^* = \frac{M+M'}{2}$, then we notice that $\lambda(M) = \lambda(M^*)$. We will verify that $M^* - \lambda(M)I$ is positive semidefinite under the conditions of Proposition 5.2. To verify it, we know that for any α, β , we have

$$\left(\begin{array}{cc} \alpha' & \beta' \end{array}\right) M^* \left(\begin{array}{c} \alpha \\ \beta \end{array}\right)$$

$$= \left(\begin{array}{cc} \alpha' & \beta' \end{array}\right) \left(\begin{array}{cc} (1-\lambda)I & \frac{A+B'}{2} \\ \frac{A'+B}{2} & \frac{1}{2}X_{S^c}^{\dagger}X_{S^c} + \frac{1}{2}\left(X_{S^c}^{\dagger}X_{S^c}\right)' - \lambda I \right) \left(\begin{array}{c} \alpha \\ \beta \end{array}\right)$$

$$= \left(1-\lambda\right) \|\alpha\|_2^2 + \beta' \left[\frac{1}{2}X_{S^c}^{\dagger}X_{S^c} + \frac{1}{2}\left(X_{S^c}^{\dagger}X_{S^c}\right)' - \lambda I \right] \beta + \alpha'(A+B')\beta,$$
 (55)

where $A = (X_S'X_S)^{-1} X_S'X_{S^c}$, $B = X_{S^c}^{\dagger}X_S$. For the last term in (55), we can apply SVD to A + B', i.e., $A + B' = U_1\Sigma_1V_1$, then we have

$$\begin{aligned} |\alpha'(A+B')\beta| &= \alpha' U_1 \Sigma_1 V_1 \beta \\ &\leq \sigma_{\max}(\Sigma_1) \langle \alpha' U_1, V_1 \beta \rangle \\ &\leq \sigma_{\max}(\Sigma_1) \left\| \alpha' U_1 \right\|_2 \left\| V_1 \beta \right\|_2 \\ &\leq \sigma_{\max}(\Sigma_1) \left\| \alpha' \right\|_2 \left\| \beta \right\|_2 \\ &\leq \frac{1}{2} \sigma_{\max}(\Sigma_1) \left(\left\| \alpha' \right\|_2^2 + \left\| \beta \right\|_2^2 \right), \end{aligned}$$

where $\sigma_{\max}(\Sigma_1)$ is the maximal absolute value in the diagonal entry of Σ_1 .

By plugging the above result into (55), we have

$$\left(\begin{array}{c} \alpha' \quad \beta' \end{array}\right) M^* \left(\begin{array}{c} \alpha \\ \beta \end{array}\right) \geq \left(1 - \lambda\right) \|\alpha\|_2^2 + \beta' \left[\frac{1}{2} X_{S^c}^{\dagger} X_{S^c} + \frac{1}{2} \left(X_{S^c}^{\dagger} X_{S^c}\right)' - \lambda I\right] \beta$$

$$- |\alpha'(A + B')\beta|$$

$$\geq \left(1 - \lambda\right) \|\alpha\|_2^2 + \beta' \left[\frac{1}{2} X_{S^c}^{\dagger} X_{S^c} + \frac{1}{2} \left(X_{S^c}^{\dagger} X_{S^c}\right)' - \lambda I\right] \beta$$

$$- \frac{1}{2} \sigma_{\max}(\Sigma_1) \left(\|\alpha'\|_2^2 + \|\beta\|_2^2\right)$$

$$= \left(1 - \lambda - \frac{1}{2} \sigma_{\max}(\Sigma_1)\right) \|\alpha\|_2^2 +$$

$$\beta' \left[\frac{1}{2} X_{S^c}^{\dagger} X_{S^c} + \frac{1}{2} \left(X_{S^c}^{\dagger} X_{S^c}\right)' - \left(\lambda + \frac{1}{2} \sigma_{\max}(\Sigma_1)\right) I\right] \beta,$$

$$(56)$$

where $\sigma_{\max}\left(\Sigma_{1}\right)$ is the maximal absolute diagonal value of matrix Σ_{1} . Because we have $\sigma\left(\Sigma_{1}\right)<2$, so the first term in (56) is greater than 0. Besides, because the minimal singular value of $\frac{1}{2}X_{S^{c}}^{\dagger}X_{S^{c}}+\frac{1}{2}\left(X_{S^{c}}^{\dagger}X_{S^{c}}\right)'$ is larger than $\frac{1}{2}\sigma_{\max}(\Sigma_{1})$, i.e., $\frac{1}{2}X_{S^{c}}^{\dagger}X_{S^{c}}+\frac{1}{2}\left(X_{S^{c}}^{\dagger}X_{S^{c}}\right)'=U_{2}\Sigma_{2}V_{2}$ and

 $2\sigma_{\min}(\Sigma_2) > \sigma_{\max}(\Sigma_1)$, the second term in (56) is also greater than 0. Thus, we prove that M^* is a positive semidefinite matrix, whose eigenvalue would be strictly larger than 0. According, M, which shares the same eigenvalue with M^* also has eigenvalues strictly larger than 0. So we have $\frac{1}{\lambda_{\min}(M)}$ bounded.

In conclusion, because $\lambda_{\min}(M)$ is bounded, $\|(X_S'X_S)^{-1}X_S'\delta\|_2^2 \to 0$, and $\|X_{S^c}^{\dagger}\delta\|_2^2 \to 0$, we have

 $\left\|\widetilde{\beta}-\widehat{\beta}\right\|_2^2=\left\|\widetilde{\beta}_S-\widehat{\beta}_S\right\|_2^2+\left\|\widetilde{\beta}_{S^c}\right\|_2^2\to 0.$