The Discrete Dantzig Selector: Estimating Sparse Linear Models via Mixed Integer Linear Optimization

RAHUL MAZUMDER [*1] AND PETER RADCHENKO [†2]

[1]Massachusetts Institute of Technology
[2]University of Southern California

January, 2017

Abstract

We propose a novel high-dimensional linear regression estimator: the *Discrete Dantzig Selector*, which minimizes the number of nonzero regression coefficients subject to a budget on the maximal absolute correlation between the features and residuals. Motivated by the significant advances in integer optimization over the past 10-15 years, we present a Mixed Integer Linear Optimization (MILO) approach to obtain *certifiably optimal* global solutions to this nonconvex optimization problem. The current state of algorithmics in integer optimization makes our proposal substantially more computationally attractive than the least squares subset selection framework based on integer *quadratic* optimization, recently proposed in [8] and the continuous nonconvex quadratic optimization framework of [33]. We propose new discrete first-order methods, which when paired with state-of-the-art MILO solvers, lead to good solutions for the *Discrete Dantzig Selector* problem for a given computational budget. We illustrate that our integrated approach provides globally optimal solutions in significantly shorter computation times, when compared to off-the-shelf MILO solvers. We demonstrate both theoretically and empirically that in a wide range of regimes the statistical properties of the *Discrete Dantzig Selector* are superior to those of popular ℓ_1-based approaches. We illustrate that our approach can handle problem instances with $p = 10{,}000$ features with certifiable optimality making it a highly scalable combinatorial variable selection approach in sparse linear modeling.

1 Introduction

We consider the familiar linear regression framework, with response vector $\mathbf{y} \in \mathbb{R}^{n\times 1}$, model matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$, regression coefficients $\boldsymbol{\beta} \in \mathbb{R}^{p\times 1}$ and errors $\boldsymbol{\epsilon} \in \mathbb{R}^{n\times 1}$: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$. We assume,

[*] Rahul Mazumder's research was partially supported by the Office of Naval Research: ONR-024511-00001 and a grant from the Moore Sloan Foundation. email: rahulmaz@mit.edu

[†] Peter Radchenko's research was partially supported by NSF Grant DMS-1209057. email: radchenk@usc.edu

unless otherwise mentioned, that the columns of $\mathbf{X}$, denoted by $\mathbf{x}_j$ for $j = 1, \ldots, p$, have been standardized to have zero means and unit ℓ_2-norm. In many modern statistical applications, the number of variables, p, is larger than the number of observations, n. In such cases, to carry out statistically meaningful estimation, it is often assumed that the number of nonzero elements in $\boldsymbol{\beta}$ is quite small [27]. The task is to obtain a good estimate, $\widehat{\boldsymbol{\beta}}$, which is sparse and serves as a good approximation to the underlying *true* regression coefficient. Of course, the basic problem of obtaining a sparse model with good data-fidelity is also of interest when the number of observations is comparable to or larger than p. In the sparse high-dimensional setting described above, two estimation approaches that have been very popular among statisticians and researchers in related fields are the Lasso [40] and the Dantzig Selector [16]. Both estimators can be expressed as solutions to convex optimization problems, which can be solved using computationally attractive procedures [11, 3, 30, 23], and come with strong theoretical guarantees [16, 9, 13]. For reasons that are explained later in this section, the primary motivation for our investigation in this paper is the Dantzig Selector, which is defined as the solution to the following linear optimization problem:

$$\min_{\boldsymbol{\beta}} \quad \|\boldsymbol{\beta}\|_1 \qquad \text{s.t.} \qquad \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \le \delta. \tag{1}$$

To distinguish this estimator from our proposed approach, we refer to it as the ℓ_1-Dantzig Selector. This estimator seeks to minimize the ℓ_1-complexity of the coefficient vector, subject to a constraint on the maximal absolute correlation between the corresponding residual vector and the predictors. The tuning parameter δ controls the amount of data-fidelity: a small value of δ corresponds to a good fit, and a larger value of δ leads to heavy shrinkage of the estimated regression coefficients. [16] point out several reasons as to why the feasibility set in (1) might serve as a good measure for data-fidelity. In particular, this set is invariant with respect to orthogonal transformations on the data $(\mathbf{y}, \mathbf{X})$. It can also be shown that δ controls[1] the residual sum of squares: the latter can be made arbitrarily close to the minimal least-squares value by decreasing δ. The ℓ_1-Dantzig Selector, like the Lasso, is used extensively as a model fitting routine to obtain a path of sparse linear models, as the data-fidelity parameter is allowed to vary [30], and allows a natural extension to more general response distributions [29]. Note that Problem (1) can be rewritten as a linear optimization problem and can be solved quite easily for problems with p in the order of thousands. Under some mild conditions, and even for p much larger than n, the corresponding estimator achieves a loss within a logarithmic factor of the ideal mean squared error achieved if the locations of the nonzero coordinates were known [16, 9].

The ℓ_1-Dantzig Selector, however, has limitations. In the presence of highly correlated covariates,

[1] More formally, we have:

$$\|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \ge \frac{\lambda_{\text{pmin}}(\mathbf{X})}{2np^{\frac{1}{2}}} \left(\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 - \|\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\text{LS}}\|_2^2 \right)^{\frac{1}{2}},$$

where, $\lambda_{\text{pmin}}(\mathbf{X})$ is the minimum *nonzero* singular value of $\mathbf{X}$, and $\widehat{\boldsymbol{\beta}}_{\text{LS}}$ is any least-squares solution — see Proposition A.1 in [21] for a proof of this result.

the estimator tends to choose a dense model, typically bringing in an important variable together with its correlated cousins, which does not significantly hurt the ℓ_1-norm of the corresponding coefficient vector. If one increases the data-fidelity threshold δ, the selected model becomes sparser, however, in the process, important variables might get left out. This is largely due to the nature of the bias imparted by the ℓ_1-norm, which penalizes both large and small coefficients in a similar fashion. Similar issues also arise in the case of Lasso [35, 26, 43, 13]. If the ℓ_0-pseudo-norm is used instead of the ℓ_1-norm, the aforementioned problems can be ameliorated: given multiple representations of the model with similar data-fidelity, the ℓ_0-pseudo-norm will always prefer the most parsimonious representation. In addition, the ℓ_0-pseudo-norm does not shrink the regression coefficients: once an important variable enters the model, it comes in unshrunk with its full effect, which, in turn, drains the effect of its correlated cousins and naturally leads to a sparser model.

Our Proposal. The preceding discussion suggests a natural question: what if we replace $\|\boldsymbol{\beta}\|_1$ in Problem (1) with $\|\boldsymbol{\beta}\|_0 := \sum_{i=1}^p \mathbf{1}(\beta_i \neq 0)$ – the number of nonzero entries in $\boldsymbol{\beta}$? This leads to the following discrete optimization problem, which also happens to define the estimator that we propose:

$$\min_{\boldsymbol{\beta}} \quad \|\boldsymbol{\beta}\|_0 \qquad \text{s.t.} \qquad \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta. \tag{2}$$

We refer to the above estimator as the *Discrete Dantzig Selector*. A couple of questions that may be asked at this point are:

- Is the estimator defined via Problem (2) computationally *tractable*?
- Does the *Discrete Dantzig Selector* lead to solutions with superior statistical properties, when compared to its ℓ_1 counterpart?

Addressing these questions and answering them affirmatively is the main focus of this paper.

The objective function in Problem (1), represented by $\|\boldsymbol{\beta}\|_1$, may be thought of as a *convexification* of the discrete quantity $\|\boldsymbol{\beta}\|_0$, which counts the number of nonzeros in the regression coefficient vector $\boldsymbol{\beta}$. The corresponding estimator seeks solutions with small ℓ_1-complexity. While this often leads to sparse solutions, i.e. those with few nonzero coefficients, the sparsity is an indirect consequence of minimizing $\|\boldsymbol{\beta}\|_1$. The *Discrete Dantzig Selector* on the other hand, targets sparsity *directly*, in its very formulation. Problem (2) can be reformulated as a Mixed Integer Linear Optimization (`MILO`) problem — due to the major advances in algorithmic research in `MILO` over the past 10-15 years, these methods are widely considered as a mature technology in a subfield of mathematical programming [41, 28]. Algorithmic advances coupled with hardware and software improvements have made `MILO` problems solvable to *certifiable* optimality for various problem sizes of practical interest. In this sense, it is perhaps appropriate to perceive `MILO` as a computationally *tractable* tool. The view of computational tractability we adopt here is *not* polynomial time tractability, but the ability of a method to provide high quality solutions with *provable* optimality certificates for problem types that are encountered in practice, in times that are appropriate for the applications being addressed. Our approach is aligned with an intriguing recent line of work in computational statistics: the use of Mixed Integer Optimization and, more broadly, modern optimization techniques to solve certain classes of discrete problems arising in statistical estimation tasks — see, for example, the recent works of [8, 6]. Further background on `MILO` appears in Section 2.1.

In this paper, we bring together recent advances from *diverse* areas of modern mathematical optimization methods: first-order techniques in convex optimization and `MILO` techniques. We provide a novel unified algorithmic approach that

(a) performs favorably over standalone of-the-shelf `MILO` solvers applicable for Problem (2), in terms of obtaining good quality solutions with provable certificates of optimality, and

(b) scales gracefully to problem sizes up to $p =$ 10,000 or even larger.

In an extensive series of experiments with synthetic and real data we demonstrate that our unified approach solves, to global optimality, instances of Problem (2) with $n \approx 500, p \approx 100$ in seconds, and underdetermined problems with $n \approx 900, p \approx$ 3,000 in minutes. While it takes marginally longer to provide *certificates* or guarantees of global optimality, the corresponding times are quite reasonable: in all the aforementioned instances the certificates of optimality are available within an hour. Our approach scales to several instances of problems with $n \leq p$ and p in the range 5,000 to 10,000, delivering optimal solutions in approximately an hour and proving optimality within at most two days, in all instances. We also find that the statistical properties of our estimates are substantially better than those of computationally friendlier alternatives, like the ℓ_1-Dantzig Selector, in terms of both the estimation error and the variable selection properties. Detailed results appear in Section 7.

Examples. To provide the reader with some intuition, we present a set of three examples, which illustrate the differences between the solutions to Problems (1) and (2). The following simple example[2] demonstrates how the ℓ_1-based method Dantzig Selector might experience difficulty in producing a sparse solution in cases where the signal predictors are highly correlated.

Example 1. Let $p = n + 1$. Take the first feature as $\mathbf{x}_1 = (1, \tau, ..., \tau)^\top$, take the (i, j)th entry of the feature matrix, $\mathbf{X}$, as $x_{ij} = 1$ if $i = j - 1$ and zero otherwise, for $i = 1, ..., n$ and $j \geq 2$, and set $\mathbf{y} = \mathbf{x}_1 - \mathbf{x}_2$.

The ℓ_1-norm of the *sparse* representation of the response, $\mathbf{y} = \mathbf{x}_1 - \mathbf{x}_2$, equals 2. Note that, given the available predictors, the response admits only one other exact representation, $\mathbf{y} = \tau\mathbf{x}_3 + ... + \tau\mathbf{x}_p$. The ℓ_1 cost for this *dense* representation is $\tau(n - 1)$, which is lower than the corresponding value for the sparse representation when τ is small. Consequently, as long as $\tau(n - 1) < 2$, both the Lasso and the ℓ_1-Dantzig selector select the dense representation of the

[2] this example was suggested to us by Emmanuel Candes

response. Alternatively, ℓ_0-based methods recover the sparse representation. More specifically, consider the solution to Problem (2): if the tuning parameter δ is set below $\tau/(1+\tau)$, then the estimator exactly recovers the sparse representation of the response.

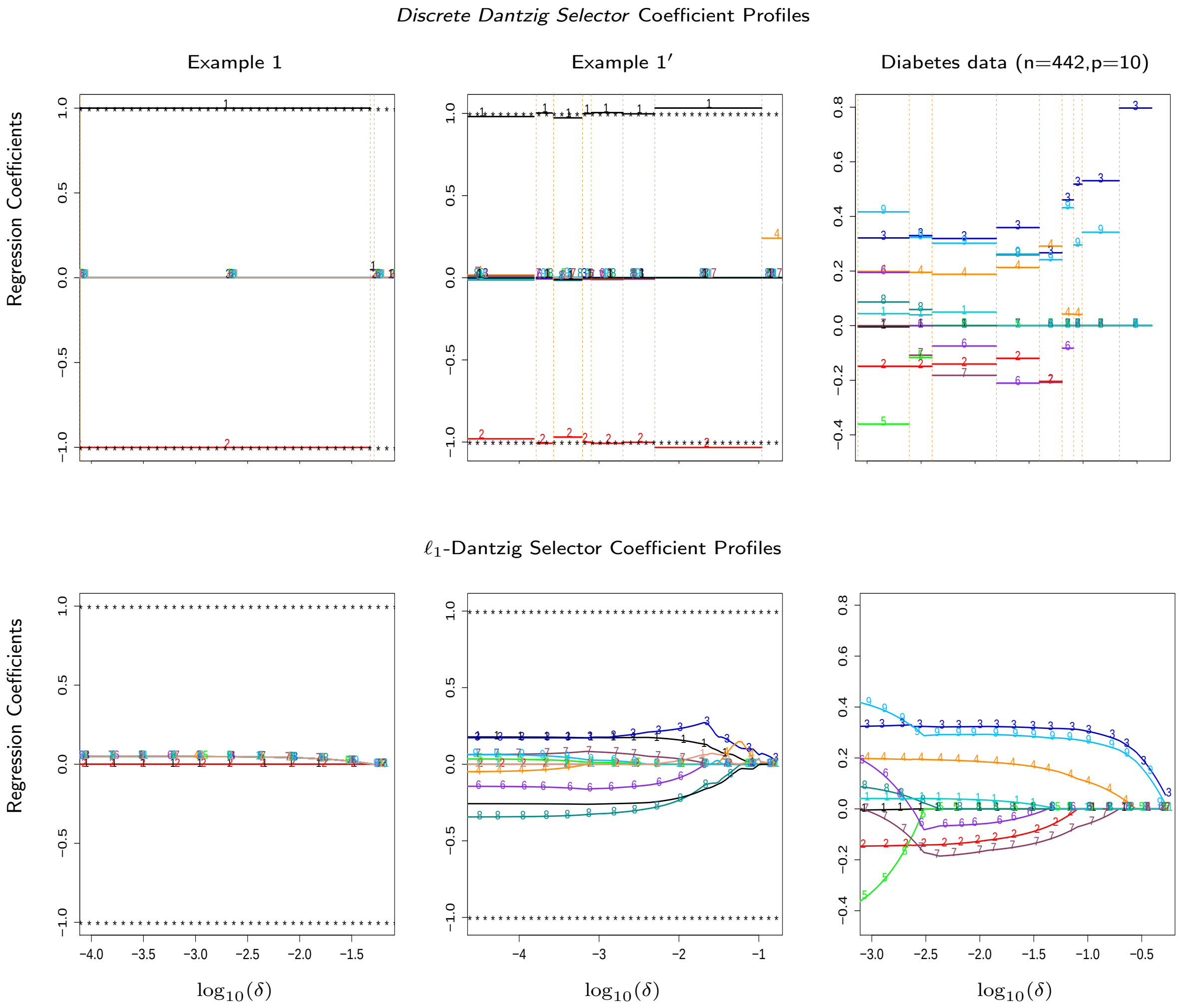

Figure 1: Coefficient profiles for the *Discrete Dantzig Selector* and the ℓ_1-Dantzig Selector, as a function of the data-fidelity parameter, δ. The dashed vertical lines in the top row of plots indicate the locations where the number of active variables changes. [Left Panel] corresponds to Example 1; [Middle Panel] corresponds to Example 1′; the "true" nonzero coefficients of $+1$ and -1 are shown as horizontal starred lines. [Right panel] corresponds to the path for the Diabetes dataset. The numbers overlain on the profiles indicate the different features.

Figure 1 demonstrates the difference between the *Discrete Dantzig Selector* and the ℓ_1-Dantzig selector, by displaying the coefficient profiles for both methods. Note that the profiles for the *Discrete Dantzig Selector* are constructed in a piece-wise constant fashion, where for each given model size, the displayed coefficients are taken from the solution corresponding to the lowest attainable value of $\|\mathbf{X}^\top(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}})\|_\infty$. The left panel of Figure 1 corresponds to Example 1, with $n = 10, p = 11$ and $\tau = 1/2p$, which we slightly modified by adding noise to the response: $\mathbf{y} = \mathbf{x}_1 - \mathbf{x}_2 + \boldsymbol{\epsilon}$. ϵ_i's are independently generated from a centered Gaussian distribution, corresponding to the Signal to Noise Ratio[3] (SNR) of 1.3×10^5.

Now consider Example $1'$, which is similar in spirit to Example 1. Here, the first two features, $\mathbf{x}_1$ and $\mathbf{x}_2$, are drawn from a centered bivariate Gaussian distribution with correlation 0.7. The remaining $p-2$ features are drawn from an independent standard Gaussian ensemble. All the features are standardized to have unit ℓ_2-norm, and the response is generated with SNR=1.4×10^3. The middle panel in Figure 1 displays the corresponding coefficient profiles with $n = 10, p = 12$. The *Discrete Dantzig Selector* exactly recovers the true model for a wide range of the tuning parameter, δ. As δ is decreased, and noise variables come into the model, their coefficients remain highly shrunk, while the coefficients for the signal variables remain near their true values. On the other hand, the ℓ_1-Dantzig selector is unable to recover the true model, and produces a large estimation error for all values of the tuning parameter.

The right panel in Figure 1 corresponds to the well known Diabetes dataset [19], where $n = 442$ and $p = 10$; here all the variables including the response were standardized to have unit ℓ_2-norm and zero mean. Note that, despite some similarities, the sequences of predictors entering the model are different for the two approaches.

We note that instead of formulation (2), one may prefer to minimize the data-fidelity term subject to a constraint on the number of nonzeros in $\boldsymbol{\beta}$:

$$\min_{\boldsymbol{\beta}} \quad \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 \leq k. \tag{3}$$

The framework developed in this paper may be adapted to Problem (3). In this paper, however, we focus on Problem (2).

Context and Related Work. A primary motivation of our work is derived from the recent work on the least squares variable selection problem [8], where the authors study

$$\min_{\boldsymbol{\beta}} \quad \frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 \leq k, \tag{4}$$

using mixed integer convex *quadratic* optimization (`MIQO`) methods. While certain statistical properties of solutions from Problems (2) and (4) are comparable; we observed in our computational experiments (see Section 7.1) that Problem (2) is orders of magnitude faster (often by a factor of hundreds) than Problem (4) in obtaining solutions with *certificates* of global optimality. In addition, a `MILO` formulation for Problem (2) consumes much less memory than a comparable `MIQO` formulation for Problem (4). The aforementioned computational superiority of `MILO` over `MIQO` should not come as a surprise. Indeed, it is quite well known in the integer programming community (see, for example, the nice review papers [28, 14]) that current algorithms for `MILO` problems are a much more *mature* technology than `MIQO`. Recently, [33] proposed MIPGO for nonconvex penalized least squares regression based on purely continuous nonconvex

[3] For a model generated as $y_i = \mu_i + \epsilon_i, i = 1, \ldots, n$; we define SNR as follows: SNR= $\text{Var}(\mu)/\text{Var}(\epsilon)$.

quadratic optimization. We demonstrate the substantial superiority of our proposal over MIPGO (in Section 7.1) in obtaining high quality statistical solutions within a given computational budget.

Thus, the superior computational scalability of the corresponding optimization methods forms a principal motivation to study Problem (2), as an effective estimation procedure for sparse linear regression. In addition, from a statistical viewpoint, in terms of estimating sparse regression models subject to good data-fidelity, the *Discrete Dantzig Selector* may be perceived as a natural, interpretable and useful alternative to least squares with variable selection, Problem (4) — in the same way as the ℓ_1-Dantzig Selector may be viewed as an appealing alternative to Lasso.

Contributions. Our main contributions can be summarized as follows:

1. We propose a new high-dimensional linear regression estimator: the *Discrete Dantzig Selector*, which minimizes the number of nonzero regression coefficients, subject to a budget on the maximal absolute correlation between the features and the residuals. We show that the estimator can be expressed as the solution to a `MILO` problem, a computationally *tractable* framework that delivers certifiably optimal global solutions; and is computationally more scalable than the recently proposed methods in [8, 5, 33].

2. We develop new discrete first-order methods, motivated by recent algorithmic developments in first-order continuous *convex* optimization, to obtain high quality feasible solutions for the *Discrete Dantzig Selector* problem. These solutions are passed onto `MILO` solvers as warm-starts. Our proposal leads to advantages over the off-the-shelf state-of-the-art integer programming algorithms in terms of (a) obtaining superior upper bounds for a given computational budget and (b) aiding `MILO` solvers in obtaining tighter lower bounds and hence improved certificates of optimality. Exploiting problem specific information, we also propose enhanced `MILO` formulations, which further improve the algorithmic performance of `MILO` solvers.

3. We characterize the statistical properties of the *Discrete Dantzig Selector* and demonstrate both theoretically and empirically its advantages over ℓ_1-based approaches. Our results also apply to *approximate* solutions for the *Discrete Dantzig Selector* optimization problem.

4. Our approach obtains optimal solutions for $p \approx 500$ in a few minutes, $p \approx 3{,}000$ within fifteen minutes and for problems with $p = 10{,}000$ in an hour. Certificates of optimality are obtained at the expense of higher computation times—for instances with $p \approx 500$ they are obtained within half-hour, for $p \approx 3{,}000$ they are achieved around an hour and for $p = 10{,}000$ the certificates arrive in the range from three to forty hours. To the best of our knowledge, we present herein, the largest problem instances in subset selection for which certifiably optimal solutions can be obtained.

Roadmap. The remainder of the paper is organized as follows. Section 2 describes the optimization methodology behind the proposed approach, and discusses its connections with the ℓ_1-Dantzig Selector optimization problem. In Section 3, the statistical properties of the *Discrete Dantzig Selector* are analyzed from a theoretical point of view; the results are compared to the ℓ_1-Dantzig Selector and the Lasso. The framework of discrete first-order methods is described in Section 4. Additional discussion of MILO formulations together with problem specific enhancements, is presented in Section 5. Section 6 gathers numerical results on the computational performance of our algorithms in a variety of settings. An empirical analysis of the statistical properties of the *Discrete Dantzig Selector* is conducted in Section 7. Some technical details are provided in the Appendix.

2 Overview of the Proposed Methodology

Herein, we introduce and summarize the general aspects of the proposed methodology. Further details and enhancements are provided in Sections 4 and 5.

2.1 Mixed Integer Linear Optimization (MILO) Preliminaries

The general form of a MILO problem is as follows:

$$\begin{array}{lll} \min\limits_{\boldsymbol{\alpha}} & \mathbf{a}^\top \boldsymbol{\alpha} & \\ \text{s.t.} & \mathbf{A}\boldsymbol{\alpha} \leq \mathbf{b} & \\ & \alpha_i \in \{0,1\}, & i = 1,\ldots,m_1 \\ & \alpha_j \geq 0, & j = m_1+1,\ldots,m, \end{array}$$

where $\mathbf{a} \in \mathbb{R}^{m\times 1}, \mathbf{A} \in \mathbb{R}^{d\times m}$ and $\mathbf{b} \in \mathbb{R}^{d\times 1}$ are the problem data, the symbol "$\leq$" denotes element-wise inequalities, and we optimize over $(\alpha_1,\ldots,\alpha_m) := \boldsymbol{\alpha} \in \mathbb{R}^m$ containing both discrete $(\alpha_i, i = 1,\ldots,m_1)$ and continuous $(\alpha_i, i = m_1+1,\ldots,m)$ variables. For background on MILO, we refer the reader to [7, 31]. Some modern integer optimization solvers include Cplex, Glpk, Gurobi, Knitro, Mosek, Scip — see also [32].

As already alluded to in Section 1, there has been significant progress in the theory and practice of MILO over the past fifteen to twenty years. Specifically, the computational power of MILO solvers has undergone impressive advances over the past twenty-five years — the cumulative machine-independent speedup factor in MILO solvers between 1991 and 2015 is estimated to be 780,000 [10]. This progress can be attributed to the inclusion of both theoretical and practical advances into MILO solvers. Some of the main factors responsible for this speedup are advances in cutting plane theory, improved heuristic methods, disjunctive programming for branching rules, techniques for preprocessing MILOs, using linear optimization as a black box to be called

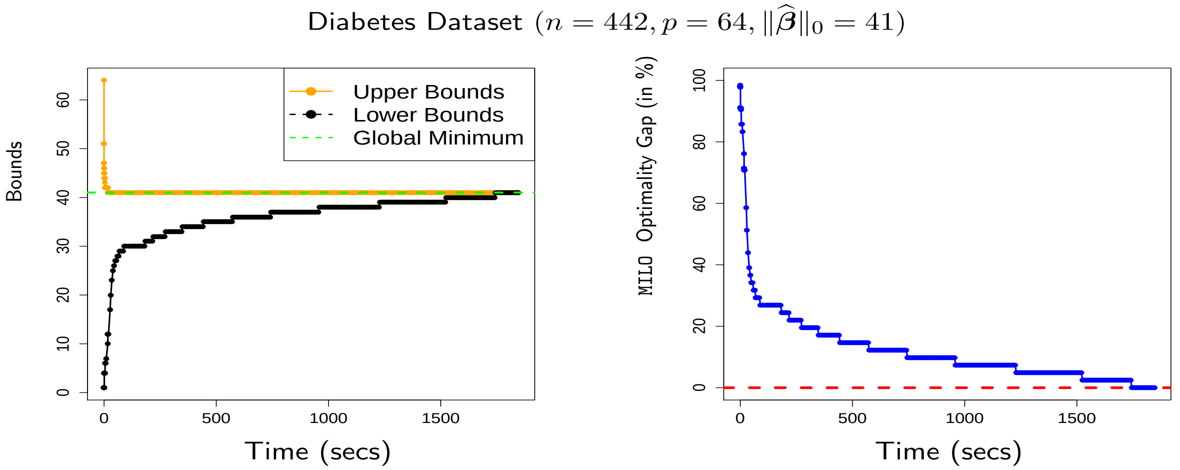

Figure 2: Typical evolution of a MILO algorithm for Problem (2), as a function of time. [Left Panel] displays the progress of Upper Bounds (UB) and Lower Bounds (LB) for the optimal value of the objective function. The upper bounds, which correspond to feasible solutions for Problem (2) are seen to stabilize at the global minimum within a few seconds. The lower bounds provide a certificate of *how far* the current solution might be from the global solution — these bounds progressively improve as the MILO algorithm explores more nodes in the branch and bound tree. Observe that the certificate of global optimality arrives at a later stage, even though the algorithm finds the global solution very quickly. [Right Panel] displays the evolution of the corresponding MILO Optimality Gap (in %), defined as (UB - LB)/UB, with time.

by MILO solvers, and improved linear optimization methods [10]. In addition, there have been substantial improvements in hardware speed: the overall hardware speedup from 1993 to 2015 is approximately estimated to be $10^{5.75} \sim 570{,}000$ [1]. When both hardware and software advances are combined, the overall speedup for MILO problems is estimated to be around 450 billion! One attractive feature of MILO solvers, which is a stark contrast to heuristic approaches, is that the former provide (a) feasible solutions, which are also upper bounds to the minimum objective value and (b) lower bounds for the optimal value of the objective function. As a MILO solver makes its way to the global optimum, the lower bounds become tighter, thereby providing improved certificates of sub-optimality (see Figure 2 for an illustration). This aspect of MILO solvers is quite useful, especially if one decides to stop the solver before reaching the global optimum. In the modern day world, MILO plays a key role in various impactful application areas of operations research: revenue management, air-traffic control, scheduling and matching tasks, production planning and others [42, 7]. In this paper, we show how the power of MILO can be used in the context of a problem of fundamental importance in statistics, namely, sparse linear model estimation — we build upon recent line of work in computational statistics, at the interface of modern discrete optimization and fundamental techniques in statistical modeling [6, 8].

2.2 `MILO` formulations for the *Discrete Dantzig Selector*

Assuming without loss of generality that Problem (2) has a minimizer which is bounded, it can be obtained by solving

$$
\begin{aligned}
\Gamma_1 := \quad \min_{\boldsymbol{\beta}} \quad & \qquad \|\boldsymbol{\beta}\|_0 \\
\text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta \\
& \qquad \|\boldsymbol{\beta}\|_\infty \leq \mathcal{M}_U,
\end{aligned}
\tag{5}
$$

where, $\mathcal{M}_U$ is a large but finite number [7]. We present a `MILO` formulation for Problem (5) (and also, Problem (2))

$$
\begin{aligned}
\min_{\boldsymbol{\beta}, \mathbf{z}} \quad & \qquad \sum_{i=1}^{p} z_i \\
\text{s.t.} \quad & -\delta \leq d_j - \langle \mathbf{q}_j, \boldsymbol{\beta} \rangle \leq \delta, \quad j = 1, \ldots, p \\
& -\mathcal{M}_U z_j \leq \beta_j \leq \mathcal{M}_U z_j, \quad j = 1, \ldots, p \\
& \qquad z_j \in \{0, 1\}, \qquad\quad j = 1, \ldots, p,
\end{aligned}
\tag{6}
$$

where the optimization variables are $\mathbf{z}$ (binary) and $\boldsymbol{\beta}$ (continuous); the problem data consists of $\mathbf{d} := \mathbf{X}^\top \mathbf{y} = (d_1, \ldots, d_p)^\top$ and $\mathbf{Q}_{p \times p} := \mathbf{X}^\top \mathbf{X} = [\mathbf{q}_1, \ldots, \mathbf{q}_p]$. Formulation (6) is often referred to as a "Big-M" formulation due to the presence of the parameter $\mathcal{M}_U$. The binary variable z_i controls whether β_i is zero or not: if $z_i = 0$ then $\beta_i = 0$ and if $z_i = 1$ then β_i is free to vary in the interval $[-\mathcal{M}_U, \mathcal{M}_U]$. The objective function $\sum_{i=1}^p z_i$ controls the number of nonzeros in the model. Figure 2 shows the performance of the above `MILO` formulation on the Diabetes dataset [19] with $n = 442, p = 64$ (here, we mean-centered and scaled $\mathbf{y}, \mathbf{x}_i$'s to have unit ℓ_2-norm).

Formulation (6) has intriguing connections to the ℓ_1-Dantzig Selector: the binary variables $z_i \in \{0, 1\}$ in Problem (6) can be relaxed into continuous variables $z_i \in [0, 1]$, leading to:

$$
\begin{aligned}
\Gamma_2 := \quad \min_{\boldsymbol{\beta}} \quad & \qquad \tfrac{1}{\mathcal{M}_U} \sum_{i=1}^{p} |\beta_j| \\
\text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta \\
& -\mathcal{M}_U \leq \beta_j \leq \mathcal{M}_U, \quad j = 1, \ldots, p.
\end{aligned}
\tag{7}
$$

Problem (7) modifies the ℓ_1-Dantzig Selector problem:

$$
\Gamma_3 := \min \ \frac{1}{\mathcal{M}_U} \|\boldsymbol{\beta}\|_1 \quad \text{s.t.} \quad \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta,
$$

by including an ℓ_∞-constraint on $\boldsymbol{\beta}$. It follows from the above that: $\Gamma_3 \leq \Gamma_2 \leq \Gamma_1$. The last inequality is typically strict, and, depending upon the data, the gap between the values, as well as between the corresponding optimal solutions, can be substantial, as illustrated in Figure 1. The above discussion provides another viewpoint for explaining the differences between the ℓ_1-Dantzig

Selector and the *Discrete Dantzig Selector* estimators. If $\mathcal{M}_U$ is taken to be large enough then $\Gamma_3 = \Gamma_2$.

We note that the constraint $\|\boldsymbol{\beta}\|_0 \leq k$ can also be expressed using Specially Ordered Sets [7] as an alternative to the "Big-M" formulation (6). This can be used when the user does not wish to specify a-priori any bound $\mathcal{M}_U$ on the regression coefficients and/or the coefficients have widely varying amplitudes. We discuss this further in Section 5.

We emphasize that $\mathcal{M}_U$ appearing in formulation (6) should *not* be interpreted as a statistical tuning parameter — it appears from a purely algorithmic viewpoint and, as we saw, has interesting connections to the ℓ_1-Dantzig Selector problem. $\mathcal{M}_U$ might be taken to be arbitrarily large (but finite) to obtain a solution to Problem (2). In Section 5 we describe several data driven methods to estimate $\mathcal{M}_U$, and we also discuss other structured formulations of Problem (6), which lead to improved algorithmic performance: they deliver tighter computational lower bounds in smaller amounts of time. We now proceed towards an analysis of the statistical properties of the *Discrete Dantzig Selector* and investigate its comparative advantages over its ℓ_1-counterpart.

3 Statistical Properties: Theory

In this section we study the statistical properties of the Discrete Dantzig estimator. In particular, we characterize its connections with the Best Subset selection estimator in the case of the orthonormal design, we investigate its oracle properties in the classical asymptotic regime and, finally, we analyze its estimation, prediction and variable selection performance in the high dimensional setting. To improve readability, all the technical proofs are presented in Section A in the Appendix.

3.1 Orthonormal Design

Here we assume $\mathbf{x}_j^\top \mathbf{x}_k$ equals 1 if $j = k$ and 0 otherwise. Note that such an assumption requires $n \geq p$. Our goal is to compare and connect the Discrete Dantzig estimator, $\widehat{\boldsymbol{\beta}}$, which solves the optimization problem (2), with the Best Subset selection estimator, $\widehat{\boldsymbol{\beta}}^{BS}$, which solves (4). In particular, we want to understand the relationship between the tuning parameter δ in the Discrete Dantzig optimization problem and the tuning parameter k, which controls the ℓ_0 norm of the Best Subset solution. Note that Problem (2) does not generally have a unique optimizer, so we use $\widehat{\boldsymbol{\beta}}$ to refer to just one of the Discrete Dantzig solutions.

Define $c_j = \mathbf{x}_j^\top \mathbf{y}$ for $j = 1, ..., p$. To simplify the presentation, and without loss of generality, suppose that the predictors are indexed in such a way that $|c_1| \geq |c_2| \geq ... \geq |c_p|$. Suppose also that $c_k \neq c_{k+1}$, to ensure uniqueness of the Best Subset solution.

Theorem 1. *1. The Best Subset selection estimator is uniquely defined as*

$$\widehat{\boldsymbol{\beta}}^{BS} = (c_1, \dots, c_k, 0, \dots, 0).$$

2. Suppose that $\delta \in [c_{k+1}, c_k)$. Then, the set of all the Discrete Dantzig solutions is

$$\{(c_1 + u_1, ..., c_k + u_k, 0, ..., 0)\,,\ |u_j| \leq \delta, j = 1, ..., k\}\,.$$

It follows that each Best Subset estimator is a Discrete Dantzig solution for an appropriately chosen δ. The coefficients of both estimators are obtained by the hard thresholding of the covariances c_j, however, the nonzero coefficients of the Discrete Dantzig estimator are allowed to deviate from the value of the covariance by an amount bounded above by δ.

3.2 Fixed p asymptotics

To avoid confusion, we refer to the true coefficient vector as $\boldsymbol{\beta}^*$. In this subsection we treat the number of predictors, p, as fixed and let the number of observations, n, tend to infinity. The standard assumption for deriving asymptotic results in this setting is that $\boldsymbol{\beta}^*$ does not depend on n, and $n^{-1}\mathbf{X}^\top\mathbf{X}$ converges to a non-singular covariance matrix C. We rewrite the above assumption to be consistent with the scaling $\|\mathbf{x}_j\|_2 = 1$ for $j = 1, ..., p$, which is used throughout this paper. Thus, we require that $\mathbf{X}^\top\mathbf{X}$ converges to C as n tends to infinity, and $\boldsymbol{\beta}^* = n^{1/2}\tilde{\boldsymbol{\beta}}^*$, for some fixed vector $\tilde{\boldsymbol{\beta}}^*$. We also impose a standard assumption that ϵ_i are i.i.d. with zero mean and finite variance. Given an index set $J \subseteq \{1, \dots, p\}$ we write $\mathbf{X}_J$ for the sub-matrix of $\mathbf{X}$ that consists of the columns identified by J. Let J^* denote the support of the vector $\boldsymbol{\beta}^*$. We define the Oracle estimator, $\widehat{\boldsymbol{\beta}}^O$, as the least-squares estimator computed using only the true predictors. In other words, the support of $\widehat{\boldsymbol{\beta}}^O$ equals J^*, and $\widehat{\boldsymbol{\beta}}^O_{J^*} = (\mathbf{X}_{J^*}^\top\mathbf{X}_{J^*})^{-1}\mathbf{X}_{J^*}^\top\mathbf{y}$.

Theorem 2. *Let $\delta \to \infty$ and $\delta = o(n^{1/2})$. Suppose that matrix C is invertible. Then, with probability tending to one,*

1. *The support of each solution to the Discrete Dantzig optimization problem equals J^*;*
2. *Both the true coefficient vector, $\boldsymbol{\beta}^*$, and the Oracle estimator, $\widehat{\boldsymbol{\beta}}^O$, belong to the set of solutions to the Discrete Dantzig optimization problem.*

Consider the polished version of the Discrete Danzig estimator, which is defined as follows: given a Discrete Dantzig solution with support $\widehat{J}$, the support of the polished estimator, $\widehat{\boldsymbol{\beta}}^P$, is set equal to $\widehat{J}$; on its support $\widehat{\boldsymbol{\beta}}^P$ is defined as the least-squares estimator using the corresponding predictors, i.e. $\widehat{\boldsymbol{\beta}}^P_{\widehat{J}} = (\mathbf{X}_{\widehat{J}}^\top\mathbf{X}_{\widehat{J}})^{-1}\mathbf{X}_{\widehat{J}}^\top\mathbf{y}$. Note that the value of $\widehat{\boldsymbol{\beta}}^P$ generally depends on the the choice of the Discrete Dantzig solution. However, this choice becomes irrelevant under the setting of Theorem 2. More specifically, with probability tending to one, every polished estimator coincides with the Oracle estimator.

Corollary 1. *Under the assumptions of Theorem 2, equality* $\widehat{\boldsymbol{\beta}}^P = \widehat{\boldsymbol{\beta}}^O$ *holds with probability tending to one.*

Consequently, estimator $\widehat{\boldsymbol{\beta}}^P$ satisfies the oracle property in the sense of Fan and Li [20].

3.3 High Dimensional Setting

Here we focus on the case where p is large, possibly much larger than n. We discuss the properties of the global as well as approximate solutions to Problem (2). We also comment on the estimator obtained from the closely related Problem (3), in which $\|\boldsymbol{\beta}\|_0$ is constrained, rather than minimized. We assume that the error terms in the underlying linear model are mean zero Gaussian with variance σ^2 and, as before, use $\boldsymbol{\beta}^*$ to refer to the true regression coefficient vector. We start with some notation. For every vector $\boldsymbol{\theta} \in \mathbb{R}^p$ and index set $J \subseteq \{1, ..., p\}$ we write $\boldsymbol{\theta}_J$ for the sub-vector of $\boldsymbol{\theta}$ determined by J.

Definition 1. *Given positive integers k and m, such that $m \in [k, p-k]$, and a positive c_0, let*

$$\begin{aligned}
\gamma(k) &= \min_{\boldsymbol{\theta} \neq 0,\, \|\boldsymbol{\theta}\|_0 \leq k} \frac{\|\mathbf{X}\boldsymbol{\theta}\|_2}{\|\boldsymbol{\theta}\|_2} \\
\kappa(k, c_0) &= \min_{|J_0| \leq k} \left\{ \min_{\boldsymbol{\theta} \neq 0,\, \|\boldsymbol{\theta}_{J_0^c}\|_1 \leq c_0 \|\boldsymbol{\theta}_{J_0}\|_1} \frac{\|\mathbf{X}\boldsymbol{\theta}\|_2}{\|\boldsymbol{\theta}_{J_0}\|_2} \right\} \\
\kappa(k, c_0, m) &= \min_{|J_0| \leq k} \left\{ \min_{\boldsymbol{\theta} \neq 0,\, \|\boldsymbol{\theta}_{J_0^c}\|_1 \leq c_0 \|\boldsymbol{\theta}_{J_0}\|_1} \frac{\|\mathbf{X}\boldsymbol{\theta}\|_2}{\|\boldsymbol{\theta}_{J_{01}}\|_2} \right\},
\end{aligned}$$

where $J_0 \subseteq \{1, ..., p\}$ and $J_{01} := J_0 \cup J_1$, with J_1 identifying the m largest (in magnitude) coordinates of $\boldsymbol{\theta}$ outside of J_0.

We use s^* to denote $\|\boldsymbol{\beta}^*\|_0$. As we discuss in the next subsection, quantities $[\kappa(s^*, c_0)]^{-1}$ and $[\kappa(s^*, c_0, m)]^{-1}$, for $m \geq s^*$ and $c_0 \geq 1$, appear in the error bounds for the Lasso and the original Dantzig selector, while $[\gamma(2s^*)]^{-1}$ appears in the bounds for *Discrete Dantzig Selector.* The following result establishes some useful relationships for these quantities.

Proposition 1. *For all positive integers k and m, with $m \in [k, p-k]$, and all $c_0 \geq 1$ the following holds: $\gamma(2k) \geq \kappa(k, c_0)/\sqrt{2}$ and $\gamma(2k) \geq \kappa(k, c_0, m)$.*

Recall the setting of Example 1. When $\tau(n-1) < 2$, the ℓ_1 methods, such as the original Dantzig selector, fail to recover the sparse representation of the response. Note also that $\kappa(2, c_0) = \kappa(2, c_0, m) = 0$, for $m \geq 2$ and $c_0 \geq 1$. On the other hand, $\gamma(4) > 0$ for $p > 4$, and the *Discrete Dantzig Selector* succeeds in recovering the correct sparse representation, for every sufficiently small value of the tuning parameter, δ.

The following theorem establishes several useful bounds for the *Discrete Dantzig Selector.*

Theorem 3. *Suppose that $\widehat{\boldsymbol{\beta}}$ solves optimization problem (2) for $\delta = \sigma\sqrt{2(1+a)\log p}$, where $a \geq$*

0. *The following bounds hold with probability bounded below by* $1 - (p^a\sqrt{\pi \log p})^{-1}$*:*

$$\begin{aligned}
\|\widehat{\boldsymbol{\beta}}\|_0 &\leq s^* \\
\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_1 &\leq 4\,[\gamma(2s^*)]^{-2}\, s^* \delta \\
\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2 &\leq 8\,[\gamma(2s^*)]^{-4}\, s^* \delta^2 \\
n^{-1}\|\mathbf{X}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)\|_2^2 &\leq 8[\gamma(2s^*)]^{-2} s^* \delta^2.
\end{aligned}$$

Remark. It follows from the proof of Theorem 3 that the above result

(i) holds uniformly over the set $\{\boldsymbol{\beta}^* : \|\boldsymbol{\beta}^*\|_0 \leq s^*\}$;

(ii) also holds for the solution to Problem (3) with $k = s^*$.

We now compare the above bounds to those established for the popular ℓ_1-based approaches. Under the assumed scaling of the predictors, and for every positive integer m, such that $m \in [s^*, p - s^*]$, Theorem 7.1 in [9] gives the following error bounds for the ℓ_1-Dantzig Selector estimator, $\widehat{\boldsymbol{\beta}}_{\text{DS}}$:

$$\begin{aligned}
\|\widehat{\boldsymbol{\beta}}_{\text{DS}} - \boldsymbol{\beta}^*\|_1 &\leq 8\,[\kappa(s^*, 1)]^{-2}\, s^* \delta \\
\|\widehat{\boldsymbol{\beta}}_{\text{DS}} - \boldsymbol{\beta}^*\|_2^2 &\leq 16\left(1 + \sqrt{s/m}\right)^2 [\kappa(s^*, 1, m)]^{-4}\, s^* \delta^2 \\
n^{-1}\|\mathbf{X}(\widehat{\boldsymbol{\beta}}_{\text{DS}} - \boldsymbol{\beta}^*)\|_2^2 &\leq 16[\kappa(s^*, 1)]^{-2} s^* \delta^2.
\end{aligned} \tag{8}$$

By Proposition 1, the right hand sides of the above inequalities are at least as large as the corresponding bounds in Theorem 3. Moreover, the differences in the two sets of bounds can potentially be quite significant. Consider the setting of Example 1 for illustration. The upper bounds in Theorem 3 are finite for $p > 4$, while the three bounds in display (8) are infinite.

Examining Theorem 7.2 in [9], we conclude that the corresponding error bounds for the Lasso are at least as large as those given in display (8). The same result also provides an upper bound on the ℓ_0-pseudo-norm of the Lasso estimator, $\widehat{\boldsymbol{\beta}}_{\text{Lasso}}$:

$$\|\widehat{\boldsymbol{\beta}}_{\text{Lasso}}\|_0 \leq \frac{64\phi_{\max}}{[\kappa(s^*, 3)]^2} s^*,$$

where $\phi_{\max}$ is the maximum eigenvalue of the matrix $\mathbf{X}^\top \mathbf{X}$. Note that the right-hand side of the above bound is infinite in the setting of Example 1. In general, this upper bound is at least 64 times as large as the one for the *Discrete Dantzig Selector* estimator. We informally summarize the above findings as follows: when compared to the ℓ_1-based approaches, the *Discrete Dantzig Selector* satisfies as good or better estimation and prediction error bounds, while achieving significantly higher level of sparsity.

We can sharpen the bounds in Theorem 3 by making them dependent on the support of $\boldsymbol{\beta}^*$. More specifically, given an index set J^* we define

$$\tilde{\gamma}(J^*) = \min_{J \subset \{1,\ldots,p\}, |J|=2s^*, J \supseteq J^*} \; \min_{\boldsymbol{\theta} \neq 0, \boldsymbol{\theta}_{J^c}=0} \frac{\|\mathbf{X}\boldsymbol{\theta}\|_2}{\|\boldsymbol{\theta}\|_2}.$$

Then, Theorem 3 holds with $\gamma(2s^*)$ replaced by $\tilde{\gamma}(\{k : \beta_k^* \neq 0\})$, and the corresponding result is uniform over $\boldsymbol{\beta}^*$.

The following corollary to Theorem 3 shows that the *Discrete Dantzig Selector* successfully recovers the support of the true coefficient vector, provided the nonzero coefficients are appropriately bounded away from zero. Define $|\beta^*|_{\min} = \min\{|\beta_k^*|, \beta_k^* \neq 0\}$.

Corollary 2. *If $|\beta^*|_{\min} > 4\sigma \left[\gamma(2s^*)\right]^{-2} \sqrt{(1+a)s^* \log p}$, then the estimator from Theorem 3 exactly recovers the support of $\boldsymbol{\beta}^*$, with probability bounded below by $1 - (p^a\sqrt{\pi \log p})^{-1}$.*

We now consider an estimator $\widehat{\boldsymbol{\beta}}$ that is a feasible solution to the optimization problem (2), but not necessarily the optimal solution. Recall that our algorithms produce $\widehat{\boldsymbol{\beta}}$ together with a lower bound on the minimum value of the objective function, $\|\boldsymbol{\beta}\|_0$. We denote this lower bound by $\widehat{s}_{LB}$. The next result shows that if the algorithm is stopped when $\|\widehat{\boldsymbol{\beta}}\|_0$ is within a prespecified multiplicative factor of $\widehat{s}_{LB}$, the bounds from Theorem 3 continue to hold after an appropriate adjustment. The corresponding proof follows the argument in the proof of Theorem 3, making only minor modifications.

Theorem 4. *Suppose that $\widehat{\boldsymbol{\beta}}$ is a feasible solution to the optimization problem* (2)*, corresponding to $\delta = \sigma\sqrt{2(1+a)\log p}$, where $a \geq 0$, such that $\|\widehat{\boldsymbol{\beta}}\|_0 \leq (1+\psi)\widehat{s}_{LB}$. Then, the following bounds hold with probability bounded below by $1 - (p^a\sqrt{\pi \log p})^{-1}$:*

$$\begin{aligned}
\|\widehat{\boldsymbol{\beta}}\|_0 &\leq (1+\psi)s^* \\
\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_1 &\leq 2(2+\psi)\left[\gamma\left([2+\psi]s^*\right)\right]^{-2} s^*\delta \\
\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2 &\leq 4(2+\psi)\left[\gamma([2+\psi]s^*)\right]^{-4} s^*\delta^2 \\
n^{-1}\|\mathbf{X}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)\|_2^2 &\leq 4(2+\psi)[\gamma([2+\psi]s^*)]^{-2}s^*\delta^2.
\end{aligned}$$

Note that the constant ψ is typically quite small in practice, for example, 0.1 (see the right panel in Figure 2 for an illustration of the evolution of ψ over time for the Diabetes dataset.) Thus, the corresponding effect on the error bounds is generally minor.

4 Obtaining Good Solutions via Discrete First-Order Methods

In this section we propose new algorithms, referred to as *discrete first-order methods*, which deliver good upper bounds for Problem (2). It is important to note that unlike the `MILO` framework, these algorithms do not provide lower bounds. Instead, the solutions obtained by our methods are passed to `MILO` solvers as warm-starts. The proposed algorithms are inspired by recent advances of first-order methods in convex optimization [37, 36, 38], and can be viewed as their nonconvex adaptations. We summarize their key advantages:

- They provide excellent upper bounds to Problem (2) with low computational cost, time and memory requirements.
- `MILO` solvers accept these solutions as warm-starts and consequently improve upon them. This hybrid approach outperforms the stand-alone capabilities of an off-the-shelf `MILO` solver, producing high quality upper bounds in amounts of time that are orders of magnitude smaller.
- The solutions obtained can be used to improve the overall run-time of `MILO` solvers, including certificates of global optimality.

We validate our proposed methods on several synthetic and real-data datasets.

4.1 Discrete First-Order Methods

Problem (2) involves the minimization of a discontinuous objective function over a polyhedral set. Thus, it is not directly amenable to simple proximal gradient type algorithms [36, 37, 8]. We propose two algorithms: Algorithm 1 (see Section 4.1.1) and Algorithm 2 (see Section 4.1.2), both of which can be used as stand-alone solvers for obtaining good quality upper bounds to Problem (2). We also present a hybrid method, Algorithm 3, which combines the strengths of both Algorithms 1 and 2, by using the solution obtained from Algorithm 1 as an initialization to Algorithm 2. In our experiments Algorithm 3 showed the best empirical performance. Section 6 presents numerical results illustrating the performance of our framework. We emphasize that Algorithms 1—3 only provide good upper bounds, they do not certify the quality of solutions via lower bounds. A main purpose of these algorithms is to provide good quality upper bounds to initialize the `MILO` solvers — the latter, in turn, are often found to improve upon the upper bounds obtained from these first-order algorithms, at the cost of more (but still reasonable) computation times.

4.1.1 The Variable Splitting Method

We present our first discrete first-order method based on a classical method in nonlinear optimization: the Alternating Direction Method of Multipliers (aka ADMM) [4] popularly used in the context of convex optimization–we refer the reader to [12] for a nice exposition on this topic. We choose this method because of its simplicity and good performance in practice, as seen in our experiments. To apply this algorithm, we *decouple* the feasible set $\{\boldsymbol{\beta} : \|\mathbf{X}^\top(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta\}$ and the discontinuous function $\boldsymbol{\beta} \mapsto \|\boldsymbol{\beta}\|_0$. Observe that Problem (2) can be equivalently rewritten as:

$$
\begin{aligned}
\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & \|\boldsymbol{\beta}\|_0 \\
\text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y}-\mathbf{X}\boldsymbol{\alpha})\|_\infty \leq \delta \\
& \boldsymbol{\alpha} = \boldsymbol{\beta}.
\end{aligned} \tag{9}
$$

We consider the Augmented Lagrangian given by:

$$
\mathcal{L}_\lambda(\boldsymbol{\beta},\boldsymbol{\alpha};\boldsymbol{\nu}) := \|\boldsymbol{\beta}\|_0 + \frac{\lambda}{2}\|\boldsymbol{\beta}-\boldsymbol{\alpha}\|_2^2 + \langle \boldsymbol{\nu}, \boldsymbol{\alpha}-\boldsymbol{\beta}\rangle \tag{10}
$$

for some value of $\lambda > 0$, where, $\boldsymbol{\nu}$ may be thought of as a "dual" variable[4] that along with λ controls the proximity between $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. The ADMM procedure leads to the following update sequence:

$$
\boldsymbol{\beta}_{k+1} \in \operatorname*{arg\,min}_{\boldsymbol{\beta}} \; \mathcal{L}_\lambda(\boldsymbol{\beta},\boldsymbol{\alpha}_k;\boldsymbol{\nu}_k) \tag{11}
$$

$$
\boldsymbol{\alpha}_{k+1} \in \operatorname*{arg\,min}_{\boldsymbol{\alpha}:\|\mathbf{X}^\top(\mathbf{y}-\mathbf{X}\boldsymbol{\alpha})\|_\infty \leq \delta} \; \mathcal{L}_\lambda(\boldsymbol{\beta}_{k+1},\boldsymbol{\alpha};\boldsymbol{\nu}_k) \tag{12}
$$

$$
\boldsymbol{\nu}_{k+1} = \boldsymbol{\nu}_k + \lambda\left(\boldsymbol{\alpha}_{k+1} - \boldsymbol{\beta}_{k+1}\right). \tag{13}
$$

Step (11) can be performed via a hard thresholding operation [18] and (12) involves a simple projection onto the polyhedron $\{\boldsymbol{\alpha} : \|\mathbf{X}^\top(\mathbf{y}-\mathbf{X}\boldsymbol{\alpha})\|_\infty \leq \delta\}$, which can be done efficiently, as detailed in Section B.1 (Appendix).

Algorithm 1

(1.) Input $(\boldsymbol{\beta}_1, \boldsymbol{\alpha}_1, \boldsymbol{\nu}_1)$, choose $\lambda > 0$, and repeat the following steps until convergence.

(2.) Update $(\boldsymbol{\beta}_k, \boldsymbol{\alpha}_k, \boldsymbol{\nu}_k)$ to $(\boldsymbol{\beta}_{k+1}, \boldsymbol{\alpha}_{k+1}, \boldsymbol{\nu}_{k+1})$ via (11)–(13).

(3.) Stop if $\|\boldsymbol{\beta}_{k+1} - \boldsymbol{\beta}_k\|_2 \leq \tau_1\|\boldsymbol{\beta}_k\|_2$ and[5] $\|\boldsymbol{\beta}_k - \boldsymbol{\alpha}_k\|_2 \leq \tau_2 \max\{\|\boldsymbol{\beta}_k\|_2, \|\boldsymbol{\alpha}_k\|_2\}$, otherwise go to Step 2.

We found Algorithm 1 to work quite well in our experiments. The algorithm may be sensitive to the choice of λ — affecting the solution and the time until convergence. We recommend using multiple values of λ, and choosing the best solution among them. In Section 4.1.3 we address these shortcomings and describe modifications that lead to improvements in practice.

[4] Following the terminology in [12], if instead of $\|\boldsymbol{\beta}\|_0$ we had a convex function, then $\boldsymbol{\nu}$ would be a dual variable, and its corresponding update step (13) would be the dual update. We will with a slight abuse of terminology use the term "dual" here.

[5] Here, τ_1, τ_2 are tolerances for convergence, typically, taken to be equal and set to 10^{-4}.

4.1.2 Sequential Linear Optimization

We now describe another nonlinear optimization algorithm for obtaining upper bounds for Problem (2), motivated by ideas popularly used in nonconvex penalized regression (see, for example, [35] and references therein). Let us consider a family of nonconvex functions, $\rho_\gamma(|\beta|)$, parametrized by $\gamma \in (0, \bar{\gamma}]$, such that $\gamma = \bar{\gamma}$ corresponds to $\rho_\gamma(|\beta|) = |\beta|$, and, as γ decreases to 0, $\rho_\gamma(|\beta|)$ becomes a progressively better approximation to $\mathbf{1}(\beta \neq 0)$. In other words,

$$\|\boldsymbol{\beta}\|_0 = \lim_{\gamma \to 0+} \sum_{i=1}^{p} \rho_\gamma(|\beta_i|). \tag{14}$$

We make the following assumption about $\rho_\gamma(\cdot)$:

Assumption (A): $\rho_\gamma(\beta)$ is symmetric in β around zero and continuous. Let $\rho_\gamma(\beta) \geq 0$ and $\rho_\gamma(0) = 0$. For every γ, the map $|\beta| \mapsto \rho_\gamma(|\beta|)$ is concave and differentiable on $(0, \infty)$.

Some popular choices of $\rho_\gamma(\cdot)$ are $\rho_\gamma(t) = \log(\frac{t}{\gamma} + 1)/\log(\frac{1}{\gamma} + 1)$ and $\rho_\gamma(t) = t^\gamma$ for $t \geq 0$. We refer the reader to [35] (and references therein) for more context and examples of nonconvex penalty functions used in sparse linear regression.

We propose to compute good upper bounds for the following continuous nonconvex optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & h(\boldsymbol{\beta}) := \sum_{i=1}^{p} \rho_\gamma(|\beta_i|) \\ \text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta, \end{aligned} \tag{15}$$

especially for $\gamma \approx 0+$. In light of (14), this leads to good upper bounds for Problem (2). Note that the concavity of $|\beta| \mapsto \rho_\gamma(|\beta|)$ leads to the following upper bound (for all $\boldsymbol{\beta}$ and $\widetilde{\boldsymbol{\beta}}$):

$$\begin{aligned} h(\boldsymbol{\beta}) &= \sum_{i=1}^{p} \rho_\gamma(|\beta_i|) \\ &\leq \underbrace{\sum_{i=1}^{p} \rho_\gamma(|\widetilde{\beta}_i|) + \sum_{i=1}^{p} \rho'_\gamma(|\widetilde{\beta}_i|) \Big(|\beta_i| - |\widetilde{\beta}_i|\Big)}_{:= \bar{h}(\boldsymbol{\beta}; \widetilde{\boldsymbol{\beta}})}, \end{aligned} \tag{16}$$

where $\rho'_\gamma(\cdot)$ denotes the derivative of $|\beta| \mapsto \rho_\gamma(|\beta|)$, with the convention that $\rho'_\gamma(0) = \infty$ if the derivative is unbounded as $|\beta| \to 0+$. Inequality (16) suggests that we sequentially minimize an upper bound to the objective function in (15). This leads to the following iterative scheme:

$$\begin{aligned} \boldsymbol{\beta}^{k+1} \in \quad & \underset{\boldsymbol{\beta}}{\arg\min} && \sum_{i=1}^{p} \rho'_{\gamma}(|\beta_i^k|)|\beta_i| \\ & \text{s.t.} && \|\mathbf{X}^{\top}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_{\infty} \leq \delta, \end{aligned} \tag{17}$$

where we assume, without loss of generality, that $\boldsymbol{\beta}^1$ is feasible for Problem (17). The above sequential approximation of the function $h(\boldsymbol{\beta})$ is similar to the popular reweighted ℓ_1-minimization method, used in signal processing [15] for sparse linear model estimation.

We now present a simple finite time convergence rate of the iterative process (17) in terms of reaching an approximate first order stationary point of Problem (15). Towards this end, we introduce the following quantity:

$$\begin{aligned} \Delta(\boldsymbol{\theta}) := \quad & \min_{\boldsymbol{\beta}} && \sum_{i=1}^{p} \rho'_{\gamma}(|\theta_i|)\left(|\beta_i| - |\theta_i|\right) \\ & \text{s.t.} && \|\mathbf{X}^{\top}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_{\infty} \leq \delta, \end{aligned} \tag{18}$$

which we use to define a first-order stationary point for Problem (15).

Definition 2. *Suppose $\widehat{\boldsymbol{\theta}}$ is feasible for Problem* (15)*. We say that $\widehat{\boldsymbol{\theta}}$ is a first-order stationary point for Problem* (15)*, if $\Delta(\widehat{\boldsymbol{\theta}}) = 0$. For $\phi > 0$, $\widehat{\boldsymbol{\theta}}$ is said to be an ϕ accurate first-order stationary point if $\Delta(\widehat{\boldsymbol{\theta}}) \geq -\phi$.*

We note that $\Delta(\boldsymbol{\theta})$ (assuming that $\boldsymbol{\theta}$ is feasible for Problem (15)) is a measure of how *far* $\widehat{\boldsymbol{\theta}}$ is from a first order stationary point of Problem (15)–if $\Delta(\boldsymbol{\theta}) < 0$, then the current estimate $\boldsymbol{\theta}$ can be improved, if $\Delta(\boldsymbol{\theta}) = 0$ then the solution cannot be improved via update (17). We refer the reader to Section B.4 for a more detailed explanation. The following theorem (for a proof see Section B.4) presents a finite time convergence rate analysis of the sequence (17) to a first-order stationary point for Problem (15).

Theorem 5. *Consider Problem* (15) *for a fixed $\gamma > 0$, with Assumption (A) on $\rho_{\gamma}(\cdot)$ in place. The update sequence $\boldsymbol{\beta}^k$, defined via* (17)*, leads to a decreasing sequence of objective values for Problem* (15)*: $h(\boldsymbol{\beta}^{k+1}) \leq h(\boldsymbol{\beta}^k)$ for all $k \geq 1$. In addition, for every $\mathcal{K} > 0$ we have the following finite-time convergence rate:*

$$\min_{1 \leq k \leq \mathcal{K}} \left\{ -\Delta(\boldsymbol{\beta}^k) \right\} \leq \frac{1}{\mathcal{K}} \left(h(\boldsymbol{\beta}^1) - \widehat{h} \right),$$

where the sequence of objective function values satisfies $h(\boldsymbol{\beta}^k) \downarrow \widehat{h}$ as $k \to \infty$.

We emphasize that the result in Theorem 5 pertains to the performance of the sequence (17) as a numerical optimization scheme, and has no direct implication on the statistical properties of the sequence. Theorem 5 implies that for any $\phi > 0$, it takes at most $\mathcal{K} = O(\frac{1}{\phi})$ many iterations to reach a ϕ-accurate first-order stationary point, i.e., there exists a $1 \leq k^* \leq \mathcal{K}$ such that

$\Delta(\boldsymbol{\beta}^{k^*}) > -\phi$. The sequence $\boldsymbol{\beta}^k$ leads to an estimate $\widehat{\boldsymbol{\beta}}_\gamma$, an upper bound for Problem (15) for a fixed γ. Since our intent is to obtain a good solution to Problem (2), we make use of property (14). This suggests that we obtain a good upper bound to Problem (15) for a small value of $\gamma \approx 0+$. Instead of applying iteration (17) for a pre-specified (small) value of γ, we recommend using a continuation strategy in practice. We take a sequence of decreasing values of $\gamma \in \{\gamma_1, \ldots, \gamma_N\}$, where $\gamma_i > \gamma_{i+1}$. We use $\widehat{\boldsymbol{\beta}}_{\gamma_i}$ as a warm-start for obtaining a good solution (upper bound) to Problem (15) for a smaller value of $\gamma = \gamma_{i+1}$. In our numerical experiments, this continuation strategy seems to work well. The method is summarized below.

Algorithm 2

(1.) Take a decreasing sequence of γ values $\{\gamma_1, \ldots, \gamma_N\}$; initialize with $\boldsymbol{\beta}^0 = \mathbf{0}$; and fix a value of Tol $= 10^{-5}$ (say). Set $\kappa = 1$ and $\gamma = \gamma_\kappa$.

(2.) Use the update sequence rule (17) until some convergence criterion is met: $\left(-\Delta(\boldsymbol{\beta}^k)\right) <$ Tol. Let $\widehat{\boldsymbol{\beta}}_\gamma$ denote the estimate of $\boldsymbol{\beta}$, upon convergence.

(3.) Set $\kappa \leftarrow \kappa + 1$, $\gamma = \gamma_\kappa$ and $\boldsymbol{\beta}^0 \leftarrow \widehat{\boldsymbol{\beta}}_\gamma$. If $\kappa \leq N$, then goto Step 2. If $\kappa > N$, exit with $\widehat{\boldsymbol{\beta}}_\gamma$ as an upper bound to Problem (2).

The linear optimization Problem (17) can be solved quite efficiently using simplex methods. For larger problems, i.e. p larger than a few thousand, we recommend using modern first-order method as described in Section B.2. Since Algorithm 2 requires solving several instances of related problems of the form (17), the warm-start capabilities of simplex methods and first-order methods lead to computational benefits.

4.1.3 Algorithm 3: Combining the Strengths of Algorithm 1 and Algorithm 2

In our empirical studies we observed that Algorithm 1 is more effective in obtaining good upper bounds than Algorithm 2 for a given time limit. Algorithm 2, on the other hand, has stronger convergence guarantees than Algorithm 1. Algorithm 1 leads to an estimate of $\boldsymbol{\beta}$ that is sparse but approximately satisfies[6] the feasibility constraint of Problem (2). Algorithm 2, in contrast, leads to solutions that are both sparse and feasible — these advantages make Algorithm 2 an important tool in our framework. We propose to combine the best features of Algorithms 1 and 2 to develop a hybrid variant: Algorithm 3, which we recommend to use in practice. Algorithm 3 is simple but very effective: it *uses* the solution obtained from Algorithm 1, say, $\widehat{\boldsymbol{\beta}}^{(1)}$, to create a set $\mathcal{I} \subset \{1, \ldots, p\}$, which includes the nonzeros in $\widehat{\boldsymbol{\beta}}^{(1)}$, and then applies Algorithm 2 on this set $\mathcal{I}$ — the details of this method are presented in Section B.5 (in the Appendix).

[6] This is because Algorithm 1 delivers a pair, $\boldsymbol{\alpha}, \boldsymbol{\beta}$, which are approximately equal: $\boldsymbol{\alpha} \approx \boldsymbol{\beta}$; $\boldsymbol{\alpha}$ is feasible for Problem (2) but need not be exactly sparse; $\boldsymbol{\beta}$, on the other hand, is sparse but approximately feasible.

5 Structured MILO Formulations and Certificates of Optimality

This section is dedicated to enhancements of the basic *Discrete Dantzig Selector* formulation (6), presented in Section 2.2. These are particularly useful in delivering tighter lower bounds, thereby providing certificates of global optimality in shorter times.

Note that formulation (6) requires the specification of $\mathcal{M}_U$ large enough to include the solution of *Discrete Dantzig Selector*. We mention another MILO formulation for Problem (2), based on Specially Ordered Sets [7]. We introduce binary variables $z_i \in \{0, 1\}$, which satisfy the condition $(1 - z_i)\beta_i = 0$ for all $i = 1, \ldots, p$ — in other words, if $z_i = 0$, then $\beta_i = 0$, and if $z_i = 1$, then β_i is unconstrained. This condition can be modeled via integer optimization using Specially Ordered Sets of Type 1 (SOS-1). More specifically,

$$(1 - z_i)\beta_i = 0 \quad \Longleftrightarrow \quad (\beta_i, 1 - z_i) : \text{SOS-1},$$

for every $i = 1, \ldots, p$. This leads to the following MILO formulation for Problem (2):

$$\begin{array}{lll} \min\limits_{\boldsymbol{\beta}, \mathbf{z}} & \sum\limits_{i=1}^{p} z_i & \\ \text{s.t.} & -\delta \leq d_j - \langle \mathbf{q}_j, \boldsymbol{\beta} \rangle \leq \delta & j = 1, \ldots, p \\ & (\beta_j, 1 - z_j) : \text{SOS-1} & j = 1, \ldots, p \\ & z_j \in \{0, 1\} & j = 1, \ldots, p, \end{array} \tag{19}$$

where, we use the notation as used in Problem (6). Observe that unlike (6), Problem (19) does not contain any parameter $\mathcal{M}_U$ in its formulation. Problem (19) may be preferred over Problem (6) when the different nonzero values of $|\widehat{\beta}_i|$'s have widely different amplitudes. In general, however, we found empirically that the algorithmic performances of formulations (19) and (6) are comparable. The MILO formulations (6) and (19) are found to work quite well in obtaining good *upper bounds* for up to $p = 10{,}000$, once they are warm-started via the discrete first-order methods described in Section 4.1. If additional problem-specific information which we refer to as "intelligence" is supplied to the MILO formulations (6) and (19), the results are found to improve substantially — as shown in Section 6.2. More specifically, we use the term "intelligence" to broadly refer to two components:

(a) providing an advanced warm-start to the MILO solver, obtained via our discrete first-order methods

(b) arming the MILO solver with information in the form of interval bounds on the regression coefficients β_j, predictions $\mathbf{x}_i^\top \boldsymbol{\beta}$, and also bounds on $\|\boldsymbol{\beta}\|_1$ and $\|\mathbf{X}\boldsymbol{\beta}\|_1$.

We note that the resulting formulation with the additional bounds as suggested in (b), above, should lead to a solution for Problem (2). We, thus, present the following *structured* version of formulation (2):

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{z}} \quad & \sum_{i=1}^{p} z_i & & \\
\text{s.t.} \quad & -\delta \leq d_j - \langle \mathbf{x}_j, \boldsymbol{\xi} \rangle \leq \delta & j = 1, \ldots, p \qquad & \text{(20a)} \\
& -\mathcal{M}_U^j z_j \leq \beta_j \leq \mathcal{M}_U^j z_j & j = 1, \ldots, p \qquad & \text{(20b)} \\
& z_j \in \{0, 1\} & j = 1, \ldots, p \qquad & \\
& \boldsymbol{\xi} = \mathbf{X}\boldsymbol{\beta} & & \text{(20c)} \\
& -\mathcal{M}_U^j \leq \beta_j \leq \mathcal{M}_U^j & j = 1, \ldots, p \qquad & \text{(20d)} \\
& -\mathcal{M}_U^{\xi,i} \leq \xi_i \leq \mathcal{M}_U^{\xi,i} & i = 1, \ldots, n \qquad & \text{(20e)} \\
& \|\boldsymbol{\beta}\|_1 \leq \mathcal{M}_\ell & & \\
& \sum_{i=1}^{n} |\xi_i| \leq \mathcal{M}_\ell^{\xi}, & & \text{(20f)}
\end{aligned}$$

where the optimization variables are $\boldsymbol{\beta} \in \mathbb{R}^p, \mathbf{z} \in \{0,1\}^p, \boldsymbol{\xi} \in \mathbb{R}^n$, and the parameters $\mathcal{M}_U^i, \mathcal{M}_U^{\xi,i}, \mathcal{M}_\ell, \mathcal{M}_\ell^{\xi}$ control, respectively, upper bounds on $|\beta_i|$, $|\langle \mathbf{x}_i, \boldsymbol{\beta} \rangle|$, $\|\boldsymbol{\beta}\|_1$ and $\|\mathbf{X}\boldsymbol{\beta}\|_1$. We note that the parameter $\mathcal{M}_U$ in Problem (6) is such that $\mathcal{M}_U \geq \max_j \mathcal{M}_U^j$ for $j = 1, \ldots, p$. Problem (20) is equivalent to the following constrained version of Problem (2):

$$\begin{aligned}
\min_{\boldsymbol{\beta}} \quad & \|\boldsymbol{\beta}\|_0 & \\
\text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta & \\
& -\mathcal{M}_U^j \leq \beta_j \leq \mathcal{M}_U^j & j = 1, \ldots, p \\
& |\langle \mathbf{x}_i, \boldsymbol{\beta} \rangle| \leq \mathcal{M}_U^{\xi,i} & i = 1, \ldots, n \\
& \|\boldsymbol{\beta}\|_1 \leq \mathcal{M}_\ell & \\
& \|\mathbf{X}\boldsymbol{\beta}\|_1 \leq \mathcal{M}_\ell^{\xi}. &
\end{aligned} \qquad (21)$$

Section 5.1 presents several strategies to compute these parameters such that a solution to Problem (21) is also a solution to Problem (2).

We present a few variations of formulation (20) that might be preferred from a computational viewpoint, depending upon the problem instance under consideration. For large values of p and n (approximately a few thousand), the constraints appearing in (20a) and (20c) may be replaced by:

$$-\delta \leq d_j - \langle \tilde{\mathbf{x}}_j, \boldsymbol{\xi} \rangle \leq \delta, \;\; \boldsymbol{\xi} = \tilde{\mathbf{X}}\boldsymbol{\beta},$$

where, $\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} = \mathbf{X}^\top \mathbf{X}$ and $\tilde{\mathbf{X}}$ is triangular — this leads to a sparse representation of the constraints appearing in (20). When n is large and p is smaller, it may be useful to perform a variable reduction by removing the variable $\boldsymbol{\xi}$ from (20). This will replace constraint (20a) by $-\delta \leq$

$d_j - \langle \mathbf{q}_j, \boldsymbol{\beta} \rangle \leq \delta$, and constraints (20c), (20e) and (20f) will be dropped. Constraints (20b) imply the bounds indicated in the constraints (20d), hence the constraints (20d) may be dropped in favor of a formulation with fewer constraints.

We note that formulation (20) is an optimization problem with many more continuous variables than formulation (6). This implies that the `MILO` solver needs to do more work at every node, by solving larger convex linear programs. However, the advantage is that the resulting formulation is more structured, and, thus, tighter lower bounds may be obtained by exploring fewer nodes. Section 6.2 presents some computational results illustrating the performance of the above framework.

5.1 Specification of Parameters

We present herein, several data-driven ways to compute the parameters in formulation (20). The methods presented here are quite different from those proposed in [8], where, the authors rely crucially on being able to compute analytic expressions for least squares solutions for a given subset size—such expressions are not available for Problem (2).

5.1.1 Specification of Parameters via Linear Optimization

We present several methods based on linear optimization that can be used to estimate the parameters appearing in Problem (20), in such a way that these estimates lead to $\widehat{\boldsymbol{\beta}}$, a solution to Problem (2).

Bounds on $\widehat{\beta}_i$'s. Consider the following pair of linear optimization problems:

$$\begin{aligned} \mu_i^+ := & \quad \max_{\boldsymbol{\beta}} \;\; \beta_i \\ \text{s.t.} & \quad \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta, \\ & \\ \mu_i^- := & \quad \min_{\boldsymbol{\beta}} \;\; \beta_i \\ \text{s.t.} & \quad \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta, \end{aligned} \tag{22}$$

for $i = 1, \ldots, p$. Note that μ_i^+ and μ_i^- provide upper and lower bounds on $\widehat{\beta}_i$ for every i. μ_i^+ is typically a strict upper bound to $\widehat{\beta}_i$, because (22) does not account for the fact that solutions to Problem (2) are sparse. Similarly, μ_i^- is a lower bound to $\widehat{\beta}_i$, and it is easy to see that $\mathcal{M}_U^i = \max\{|\mu_i^+|, |\mu_i^-|\}$ is an upper bound to $|\widehat{\beta}_i|$. Note that solutions to Problem (22) are finite only if the feasible set is bounded. If $n > p$ and if the entries of $\mathbf{X}$ are drawn from a continuous probability measure, then the bounds are finite with probability one. The above bounds can be made tighter by using information about upper bounds on Problem (2) as obtained via the discrete first-order methods. We describe such methods in Section B.6 (Appendix). Once upper bounds on $|\widehat{\beta}_i|$, i.e. $\mathcal{M}_U^i$, are obtained, they can be used to compute bounds on $\|\widehat{\boldsymbol{\beta}}\|_\infty$ and $\|\widehat{\boldsymbol{\beta}}\|_1$ as follows:

$$\|\widehat{\boldsymbol{\beta}}\|_\infty \leq \mathcal{M}_U = \max_{i=1,\ldots,p} \ \mathcal{M}_U^i \ \text{ and } \ \|\widehat{\boldsymbol{\beta}}\|_1 \leq \sum_{i=1}^{\alpha_0} \mathcal{M}_U^{(i)},$$

where, α_0 denotes an upper bound to Problem (2) and $\mathcal{M}_U^{(1)} \geq \mathcal{M}_U^{(2)} \geq \ldots \geq \mathcal{M}_U^{(p)}$.

Bounds on $\langle \mathbf{x}_i, \widehat{\boldsymbol{\beta}} \rangle$'s. Bounds on $\langle \mathbf{x}_i, \widehat{\boldsymbol{\beta}} \rangle$ can be obtained by solving the following pair of linear optimization problems:

$$\begin{aligned} v_i^+(\alpha_0) := \quad & \max_{\boldsymbol{\beta}} \ \langle \mathbf{x}_i, \boldsymbol{\beta} \rangle \\ \text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta \\ & \|\boldsymbol{\beta}\|_\infty \leq \mathcal{M}_U \\ & \|\boldsymbol{\beta}\|_1 \leq \mathcal{M}_U \alpha_0 \end{aligned} \tag{23}$$

$$\begin{aligned} v_i^-(\alpha_0) := \quad & \min_{\boldsymbol{\beta}} \ \langle \mathbf{x}_i, \boldsymbol{\beta} \rangle \\ \text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta \\ & \|\boldsymbol{\beta}\|_\infty \leq \mathcal{M}_U \\ & \|\boldsymbol{\beta}\|_1 \leq \mathcal{M}_U \alpha_0, \end{aligned}$$

for every $i = 1, \ldots, n$.

Analogous to the bounds derived via Problem (22), it is also possible to compute more conservative bounds on $\langle \mathbf{x}_i, \widehat{\boldsymbol{\beta}} \rangle$, by dropping the constraints $\|\boldsymbol{\beta}\|_\infty \leq \mathcal{M}_U$ and $\|\boldsymbol{\beta}\|_1 \leq \mathcal{M}_U \alpha_0$ in Problem (23). This gives nontrivial bounds even for the under-determined $n < p$ case, as long as $\mathbf{X}$ has rank n (this is in contrast with the bounds from Problem (22) being vacuous when $n < p$). It is also possible to estimate bounds on $\langle \mathbf{x}_i, \widehat{\boldsymbol{\beta}} \rangle$ by including an additional constraint: $\|\mathbf{X}\boldsymbol{\beta}\|_\infty \leq \mathcal{M}_U^\xi$, and using an iterative method as described in the Appendix, Section B.6 (see Step-1–Step-4) while computing bounds on the regression coefficients.

The quantity $\mathrm{v}_i = \max\{v_i^+(\alpha_0), -v_i^-(\alpha_0)\}$ provides an upper bound to $|\langle \mathbf{x}_i, \widehat{\boldsymbol{\beta}} \rangle|$. In particular, this leads to the following upper bounds:

$$\|\mathbf{X}\widehat{\boldsymbol{\beta}}\|_\infty \leq \max_{i=1,\ldots,n} \mathrm{v}_i \quad \text{ and } \quad \|\mathbf{X}\widehat{\boldsymbol{\beta}}\|_1 \leq \sum_{i=1}^{n} \mathrm{v}_i,$$

leading to a data-driven method to estimate bounds appearing in (20).

Computational Cost. Computing the quantities appearing in (22), (37) and (23) requires solving at least $2(p+n)$ linear optimization problems. However, these individual problems are quite simple to parallelize and they need to be solved once, before proceeding to solve Problem (20). These linear optimization problems can be solved by simplex based solvers quite easily for p in the lower thousands (typically less than a minute with Gurobi's simplex solver).

5.1.2 Specification of Parameters from warm-starts

We present herein, simple practical methods to compute the parameter values by using good upper bounds to Problem (2). Let $\widehat{\boldsymbol{\beta}}^0$ denote a solution that corresponds to a good upper bound to Problem (2). $\widehat{\boldsymbol{\beta}}^0$ can be obtained from Algorithm 3, for example. One can also use the solution obtained from Algorithm 3 as a warm-start to Problem (19) and allow it to run for a few minutes — the resulting estimate may be used as $\widehat{\boldsymbol{\beta}}^0$. The parameters appearing in the bounds can be based on $\widehat{\boldsymbol{\beta}}^0$, as follows. To be on the conservative side, we recommend setting the same bound for all the $\mathcal{M}_U^i$'s: for example, they can all be assigned the value $\tau\|\widehat{\boldsymbol{\beta}}^0\|_\infty$. Similarly, a conservative bound for all the $\mathcal{M}_U^{\xi,i}$'s is given by $\tau\|\mathbf{X}\widehat{\boldsymbol{\beta}}^0\|_\infty$. In addition, we can set $\mathcal{M}_\ell = \min\left\{\tau\|\widehat{\boldsymbol{\beta}}^0\|_0\|\widehat{\boldsymbol{\beta}}^0\|_\infty, \tau\|\widehat{\boldsymbol{\beta}}^0\|_1\right\}$ and $\mathcal{M}_\ell^\xi = \tau\|\mathbf{X}\widehat{\boldsymbol{\beta}}^0\|_\infty$, for some value of $\tau \in \{1.5, 2\}$.

The method described above leads to parameter specific bounds as a simple by-product of our general algorithmic framework. Unlike the methods in Section 5.1.1, it requires no additional computation. On the other hand, the bounds in Section 5.1.1 are conservative, because they are implied by the bounds from Problem (2).

6 Numerical Experiments: Algorithmic Performance

In this section, we report extensive numerical experiments that demonstrate: (a) the usefulness of the discrete first-order methods (Section 4.1) in obtaining good quality upper bounds, especially when they are used to provide warm-starts to `MILO` solvers — this is shown in Section 6.1; and (b) how advanced warm-starts, coupled with the enhanced formulations presented in Section 5.1, can be used to improve the overall run-time for off-the-shelf `MILO` solvers, when proving global optimality for the *Discrete Dantzig Selector* problem — this is shown in Section 6.2.

All computations were carried out on Columbia University's high performance computing (HPC) facility, `http://hpc.cc.columbia.edu/`, on the `Yeti` cluster computing environment. The discrete first-order methods were implemented in Matlab 2014a, and we used Gurobi [25] version 6.0.3. For all experiments in Sections 6.1 and 6.2 (except the large scale examples) we used 16GB of memory.

Type-1 ($n = 100, p = 1000$)

Data Fidelity	Time	Quality of Upper Bounds	
Parameter	(in secs)	With Warm	Vanilla
$1.5\bar{\delta}$	2 (*)	0	–
	120	0	0
	500	0	0
$\bar{\delta}$	90 (*)	0	42.85
	120	0	42.85
	500	0	28.57
$0.5\bar{\delta}$	57 (*)	0	200
	120	0	86.66
	500	0	26.66
$0.2\bar{\delta}$	120	3.12	25
	210 (*)	0	15.62
	500	0	15.62

Type-2 ($n = 300, p = 1000$)

Data Fidelity	Time	Quality of Upper Bounds	
Parameter	(in secs)	With Warm	Vanilla
$1.5\bar{\delta}$	50 (*)	0	–
	120	0	214.28
	500	0	14.28
$\bar{\delta}$	120	6.66	146.66
	132(*)	0	73.33
	500	0	0
$0.5\bar{\delta}$	35 (*)	0	–
	120	0	29.62
	500	0	25.92
$0.2\bar{\delta}$	40 (*)	0	–
	120	0	73.44
	500	0	23.43

Type-3 ($n = 600, p = 2000$)

Data Fidelity	Time	Quality of Upper Bounds	
Parameter	(in secs)	With Warm	Vanilla
$1.5\bar{\delta}$	500	7.14	–
	530 (*)	0	–
	950	0	28.57
$\bar{\delta}$	500	11.11	–
	875 (*)	0	137.03
	950	0	33.33
$0.5\bar{\delta}$	55 (*)	0	–
	500	0	–
	950	0	120.37
$0.2\bar{\delta}$	500	0.8	–
	560 (*)	0	–
	950	0	77.6

Type-4 ($n = 58, p = 2000$)

Data Fidelity	Time	Quality of Upper Bounds	
Parameter	(in secs)	With Warm	Vanilla
$1.2\bar{\delta}$	300 (*)	0	220
	370	0	0
	600	0	0
$\bar{\delta}$	300	16.66	–
	367 (*)	0	216.66
	600	0	0
$0.5\bar{\delta}$	300	5	–
	560 (*)	0	95
	600	0	95
$0.2\bar{\delta}$	145 (*)	0	–
	300	0	165
	600	0	60

Table 1: Tables showing "Quality of Upper Bounds", defined as $100 \times (h_{\text{alg}} - \widehat{h})/\widehat{h}$, where h_{alg} refers to the objective value obtained by algorithm "alg" (at the given time), and $\widehat{h}$ is the best objective value found in the entire run-time duration of all the algorithms. Two cases of "alg" $\in$ {"With Warm", "Vanilla"} have been considered: "With Warm" denotes `MILO` warm-started with a solution from Algorithm 3 (the structured formulations of Section 5 are not used here); "Vanilla" denotes a `MILO` solver without any warm-start specification. "With Warm" is found to obtain the best upper bound for a given computation time-limit in all the instances. In several instances, `MILO` is found to improve the solution obtained via Algorithm 3, after accepting it as a warm-start. For Type-1,2 the total time limit was 500 secs; for Type-3 it was 950 secs, and for Type-4 the algorithms were considered for a total time limit of 600 secs. For method "With Warm", the times reported show the overall time taken by Algorithm 3 and the `MILO` algorithm. An asterisk "(*)" indicates that the best solution is obtained at that time. A "–" means that no feasible solution was obtained by the algorithm in that time.

6.1 Obtaining Good Quality Upper Bounds

From a practical viewpoint, being able to obtain good quality upper bounds to Problem (2) is, perhaps, of foremost importance. To demonstrate the effectiveness of our computational framework in this regard, we perform a series of experiments on the data-types described below.

`Type-Synth`: We generate a Gaussian ensemble $\mathbf{X}_{n\times p} \sim \mathrm{MVN}(\mathbf{0}, \boldsymbol{\Sigma})$, where $\sigma_{ij} = \rho^{|i-j|}$ for some value of $\rho \in [0, 1)$, with the convention that $0^0 = 1$. The underlying true regression coefficient vector, $\boldsymbol{\beta}^* \in \mathbb{R}^p$, has $\beta^*_j = 1$ for k^* equi-spaced values of $j \in \{1, \ldots, p\}$ and $\beta^*_j = 0$ for the remaining values of j.

Type-1: This is of `Type-Synth` with $n = 100$, $p = 1000$, $\rho = 0$, $k^* = 10$. We studied Problem (2) for four different values of the parameter δ set at $\bar{\delta}(1.5, 1, 0.5, 0.2)$ with $\bar{\delta}$ being defined below.

Type-2: This is of `Type-Synth` with $n = 300$, $p = 1000$, $\rho = 0.8$, $k^* = 25$. Here δ values were set as $\bar{\delta}(1.5, 1, 0.5, 0.2)$.

Type-3: This is of `Type-Synth` with $n = 600$, $p = 2000$, $\rho = 0.8$, $k^* = 40$. Here δ values were set as $\bar{\delta}(1.5, 1, 0.5, 0.2)$.

Type-4: This is a semi-synthetic dataset: we considered the Radiation sensitivity gene expression dataset[7] from Ch. 16 of the book [26]. The features were randomly downsampled to $p = 2000$ and there were $n = 58$ observations. We generated response $\mathbf{y}$ based on a linear model with $\|\boldsymbol{\beta}^*\|_0 = 10$, $\beta^*_j = 1$ for $j \leq 10$, and $\beta^0_j = 0$ for $j > 10$. Here δ values were set as $\bar{\delta}(1.2, 1, 0.5, 0.2)$.

Type-5: This is of `Type-Synth` with $n = 1000$, $p = 3000$, $\rho = 0$, $k^* = 10$; we considered one value of δ, which was set to $\bar{\delta}$.

In each of the above examples, after $\mathbf{X}$ was generated, we standardized its columns to have zero mean and unit ℓ_2-norm. Then, the response was generated as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\epsilon}$, where $\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$, and σ^2 was adjusted to match the selected value of SNR$= \mathrm{Var}(\mathbf{x}^\top\boldsymbol{\beta}^*)/\sigma^2$ (taken as 3 in all the above cases); the reference value of the tuning parameter was set to $\bar{\delta} = \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^*)\|_\infty$.

We studied different first-order algorithms described in Section 4. Algorithm 3 was empirically seen to have the best performance over its constituents, Algorithms 1 and 2, when used separately. Hence, we used Algorithm 3 in all the experiments to obtain good upper bounds to Problem (2). The solution obtained from Algorithm 3 was passed as a warm-start to the `MILO` formulation (6) (for a large value of $\mathcal{M}_U = 10^3$) — this hybrid `MILO` approach is denoted by "With Warm" in Table 1. We compared this method with the vanilla `MILO` formulation (6) (for a large value of $\mathcal{M}_U = 10^3$), which was implemented without any warm-start information. Table 1 shows the

[7] We downloaded the dataset from the website `http://statweb.stanford.edu/~tibs/ElemStatLearn/datasets/`

objective values obtained by these two methods — the MILO algorithm aided with advanced warm-starts was found to perform the best across all the examples. For the hybrid approach ("With Warm"), in many of the instances, the solution obtained by Algorithm 3 was further improved by MILO. In some cases, the vanilla MILO approach took a while before it was able to find a feasible solution. For example, in the Type-5 setting (which does not appear in Table 1) the best solution was delivered by Algorithm 3 within one minute; in contrast, the vanilla MILO algorithm failed to find a feasible solution within 1000 seconds.

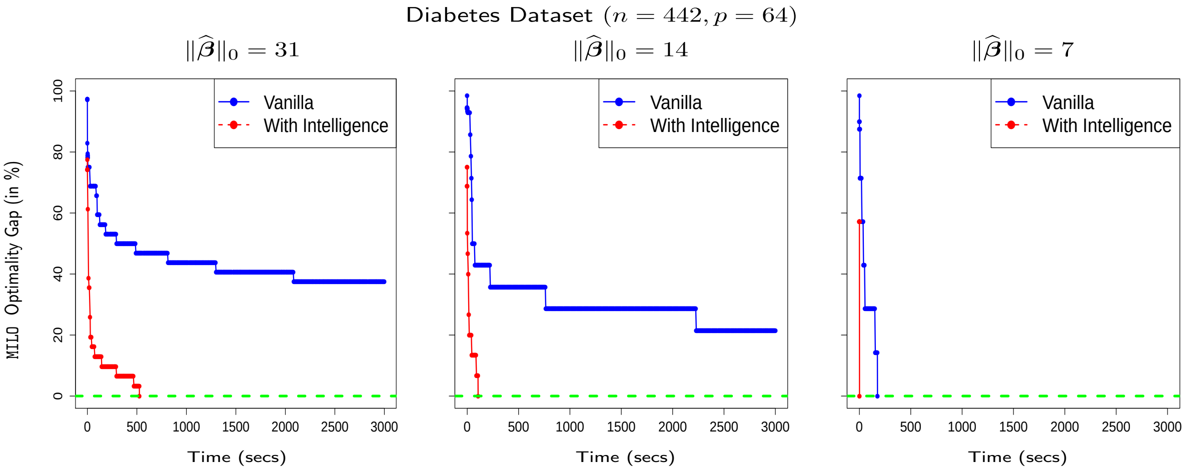

Figure 3: The evolution of MILO Optimality gaps (defined in Figure 2) as functions of time (in secs) for the MILO methods with and without problem-specific information. We consider three different values of the data fidelity parameter δ, leading to solutions with different number of nonzeros as mentioned in the figure panels. Here, "With Intelligence" refers to a MILO algorithm aided with an advanced warm-start, in addition to the bounds specified in Section 5, and "Vanilla" refers to a MILO algorithm without any such additional information. MILO is found to benefit significantly from additional problem-specific information.

6.2 Lower Bounds and Certificates of Optimality

Here we demonstrate how our framework delivers certifiably optimal solutions to Problem (2). In our first series of experiments we considered the popular diabetes dataset [19], which we examined with interaction terms included, giving us $p = 64$ and $n = 442$. All the features and the response were mean-centered and standardized to have unit ℓ_2-norm. Figure 3 shows the performance of two versions of MILO – "With Intelligence" and "Vanilla". "With Intelligence" refers to MILO formulation (20), where a MILO solver is provided with an advanced warm-start, say, $\widehat{\boldsymbol{\beta}}^0$. The parameter specifications are obtained based on the method in Section 5.1.2; here, we used the box constraints and ℓ_1-constraint on $\boldsymbol{\beta}$. The "Vanilla" version of MILO was not provided with any such problem-specific information — we used formulation (6), as in Section 6.1. Our experimental results (Figure 3) show that "Intelligence" significantly enhances the performance of the MILO solver, in terms of proving global optimality. Usually, we observe that for a fixed n, p with $n > p$ the time to certify optimality is smaller when k is small or close to p – intuitively, this

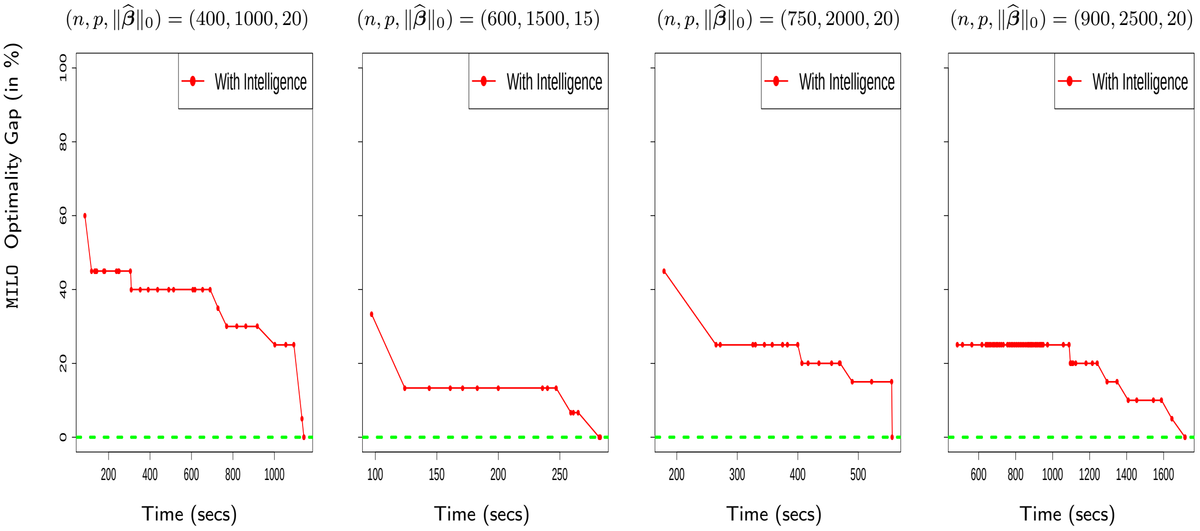

Figure 4: The evolution of `MILO` Optimality gaps (defined in Figure 2) as a function of time (after a warm-start was supplied to it) for different synthetic examples with varying problem-sizes. "With Intelligence" refers to `MILO` aided with an advanced warm-start information, in addition to the bounds specified in Section 5, as described in the text. The "Vanilla" version of the `MILO` was found to perform worse, and is, thus, not shown in the figures. In all these instances, the ℓ_1-Dantzig Selector resulted in denser solutions.

is due to the "search-space" being small. The computation time increases as k becomes closer to $p/2$. This is reflected in Figures 2 and 3.

6.2.1 Moderate Scale Examples

We considered some examples of `Type-Synth` for $\rho = 0$ and different values of n, p, k^*; in all the examples, we set $\delta = \|\mathbf{X}^\top(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^*)\|_\infty$. The results for `MILO` with intelligence are displayed in Figure 4. We obtained an advanced warm-start ($\widehat{\boldsymbol{\beta}}^0$) from a combination of Algorithm 3 and `MILO` formulation (19), where the latter was allowed to run for an overall time limit of 500 seconds. The warm start $\widehat{\boldsymbol{\beta}}^0$ was used to initialize formulation (20) — the parameter specifications in the formulation were obtained based on the method in Section 5.1.2. We also experimented with the version of formulation (20) that considers only box constraints on $\boldsymbol{\beta}$; the results were often found to be roughly similar — both methods certified optimality, though there were some differences in the total run-time (roughly around a few minutes). For all the synthetic examples presented in Figure 4, the vanilla version of `MILO` took much longer to prove optimality and hence they are not shown in Figure 4.

In all these instances, the ℓ_1-Dantzig Selector, not surprisingly, resulted in a solution that was more dense than the corresponding *Discrete Dantzig Selector*.

			(Synthetic Examples)			
n	p	k^*	Upper Bound	Lower Bound	MILO Gap	Time (hrs)
2,000	5,000	30	30	30	0	8.3
4,000	5,000	60	60	60	0	20.0
7,000	7,000	20	20	20	0	21.7
6,000	7,000	60	60	60	0	20.0
4,000	8,000	20	20	20	0	41.9
3,000	8,000	20	20	20	0	18.3
3,000	9,000	20	20	20	0	47.2
4,000	9,000	20	20	20	0	44.4
1,000	10,000	10	10	10	0	14.2
5,000	10,000	10	10	10	0	2.5
5,000	10,000	20	20	20	0	47.5
10,000	10,000	30	30	27	10%	42.5
			(Real Data Examples)			
n	p	k^*	Upper Bound	Lower Bound	MILO Gap	Time (hrs)
6,000	4,500	20	20	20	0	5.0
6,000	4,500	40	40	37	10%	12.5

Table 2: Table showing times taken to reach global optimal solutions for several problem instances (both synthetic and real data), with p up to 10,000. For each instance, we list the upper bound, the lower bound, the corresponding `MILO` (Optimality) Gap and the time taken to reach the listed lower bound by the `MILO` solver equipped with "Intelligence" (as described in Figure 4). The times (in hours) refer to those taken by the `MILO` solver after being provided with a warm-start. In all the instances, the best upper bound was obtained around approximately one hour, however, it took much longer to obtain a certificate of global optimality via the (almost) matching lower bounds. The `MILO` Gap was found to be zero in all the instances apart from the two where the algorithm was terminated upon obtaining a lower bound within 10% of the upper bound. The results demonstrate that certifiably optimal *Discrete Dantzig Selector* solutions can be obtained for large scale instances.

6.2.2 Large Scale Examples

We consider several large scale examples with p ranging in 4,500 to 10,000: these problem-sizes are orders of magnitude greater than those considered in [8]. These computations were performed with 100 GB memory.

We studied a host of synthetic examples, all generated as in Section 6.2.1. We also considered a semi-synthetic dataset derived from the well-known Gisette data `http://archive.ics.uci.edu/ml/datasets/Gisette`. Here, we generated a response $\mathbf{y}$, based on the Gisette data covariates (each feature was standardized to have zero mean and unit ℓ_2 norm) — here, $n = 6000$ and $p = 4500$, we set $\beta^*_i = 1, i = 1, \ldots, k^*$ and the remaining $\beta^*_i = 0$; SNR=3 and considered two instances with $k^* = 20, 30$.

The algorithmic set-up was similar to that used in Section 6.2.1. The results are presented in Table 2. In all these instances, the ℓ_1-Dantzig Selector resulted in a solution that was more dense than those obtained via the *Discrete Dantzig Selector*. For all the synthetic examples, the `MILO` solver delivered solutions that matched the optimal solution of the data generating mechanism. Typically, the time taken to prove optimality marginally increases with larger values of k^* (for a fixed n, p); for a fixed k^*, p the times taken to certify optimality increases with decreasing values of n. The examples demonstrated in this paper, show the largest instances of discrete optimization problems for exact variable selection, that can be solved to provable optimality.

7 Numerical Experiments: Statistical Properties

We conducted a series of synthetic experiments to understand the statistical properties of the *Discrete Dantzig Selector* and compare them to those of the ℓ_1-Dantzig Selector and variants.

We used the following datasets in our analysis.

Example-A: This is of `Type-Synth` with $n = 200$, $p = 500$, $\rho = 0$ and $k^* = 20$.

Example-B: This dataset was similar to the one taken in Example-A, but the amplitudes and signs of the true regression coefficients were allowed to vary: the twenty nonzero β_j^*'s were equally spaced in the interval $[-10, 10]$.

Example-C: This is of `Type-Synth` with $n = 100, p = 500, \rho = 0.85, k^* = 10$.

Example-D: We set $n = 100, p = 300$ and let $\mathbf{X} \sim \text{MVN}(\mathbf{0}, \boldsymbol{\Sigma})$, where $\sigma_{12} = \sigma_{21} = 0.7$, and all the remaining σ_{jk} are equal to zero. We also took $\beta_1^* = 1$ and $\beta_2^* = -1$, with the remaining coefficients β_j^* set to zero, resulting in $k^* = 2$. (This example is a larger version of Example $1'$ described in Section 1 and illustrated in Figure 1.)

In each of the above cases, after $\mathbf{X}$ was generated, we standardized its columns to have unit ℓ_2-norm. Then, the response was generated as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\epsilon}$, where $\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$, and σ^2 was adjusted to match the selected value of SNR, which was varied across $\{3, 10\}$ in the examples.

We considered the following estimators in our analysis:

- "Warm" — this method applies a heuristic strategy to obtain upper bounds to Problem (2). We used[8] Algorithm 2, described in Section 4.1.2.
- "L0-DS" — the solution obtained from "Warm" is taken as a warm-start to a `MILO` solver and subsequently allowed to run with a time limit of 4000 seconds.
- "L0-DS-Pol" — this is a "polished" version of the *Discrete Dantzig Selector* estimator "L0-DS" and is obtained by performing a simple least squares fit on the support of the "L0-DS" estimate.
- "L1-DS" — this is the original ℓ_1-Dantzig Selector.
- "L1-DS-Pol" — this is a polished version of the "L1-DS".

[8] This is similar to a re-weighted ℓ_1-minimization [15] method applied to Problem (2). We took the penalty $\rho_\gamma(|\beta|) \propto \log(|\beta|/\gamma + 1)$ on a geometrically decreasing grid of ten γ values: $\gamma_i = 10^{-2} \times 0.8^{i-1}$ for $i = 1, \ldots, 10$.

Each of the above estimators were computed on a range of approximately thirty different δ values around $\bar{\delta} = \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}^\top\boldsymbol{\beta}^*)\|_\infty$. We considered ten different replications (based on different $\boldsymbol{\epsilon}$ realizations) and took the median of the results. The optimal tuning parameter (δ^{opt}) for every model was selected based on the value of δ that minimized the estimation error with respect to the true regression coefficients. For this chosen value of δ^{opt} we considered different metrics to assess the performance of the different estimators. We computed the squared ℓ_2-error in estimating the regression coefficients: $\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2$. We also considered the "Variable Selection error", which is defined as $\sum_{j=1}^{p} 1(\widehat{S}_j \neq S_j^*)$, where $\widehat{S}_j$ is the jth coordinate of $\widehat{\mathbf{S}} := \text{Supp}(\widehat{\boldsymbol{\beta}})$, and S_j^* is the jth element of $\mathbf{S}^* = \text{Supp}(\boldsymbol{\beta}^*)$. Finally, we computed the "number of nonzeros", which refers to the number of nonzero coefficients in $\widehat{\boldsymbol{\beta}}$.

A collection of representative results with SNR=10 is displayed in Figure 5. The error bars correspond to standard errors, the width being set to $2\hat{s}/\sqrt{N}$ where, $\hat{s}$ is the mean absolute deviation around the median and N denotes the number of replicates (here, ten). A larger display of additional examples with varying SNR values is presented in Table 4 in Appendix C, where we also report the "Prediction Error", defined as $\|\mathbf{X}\widehat{\boldsymbol{\beta}} - \mathbf{X}\boldsymbol{\beta}^*\|_2^2/\|\mathbf{X}\boldsymbol{\beta}^*\|_2^2$. In Table 5, given in the same section, we provide comparisons with the polished version of the ℓ_1-Dantzig Selector. Our experiments show that polishing the ℓ_1-Dantzig Selector may lead to marginally better solutions relative to the original Dantzig Selector, but the corresponding statistical performance is inferior to that of the estimates based on the *Discrete Dantzig Selector* estimator.

We note that the performance of the Lasso was found to be quite similar to that of the ℓ_1-Dantzig Selector. The statistical performance of the subset selection procedure (4), as described in [8], was found to be similar to that of the *Discrete Dantzig Selector* for $p \leq 1000$. Because the main focus of the paper is to show that *Discrete Dantzig Selector* is a computationally tractable procedure, which delivers estimates with better statistical properties than its ℓ_1 counterpart, we restrict our numerical studies to the methods listed above.

Summary of Findings. Based on the experimental results, we observe that the *Discrete Dantzig Selector* and its polished variant perform quite well, when compared to the competing methods in terms of estimating $\boldsymbol{\beta}^*$; they also demonstrate superior variable selection properties (not surprisingly, the ℓ_0 methods obtain the sparsest models across all the examples). "Warm" does not perform very well when compared to "L0-DS", even though both methods attempt to solve Problem (2) — this suggests that estimators based on rigorous optimization procedures have better statistical properties. We observe that "L0-DS" and "L0-DS-Pol" possess similar variable selection properties, however, the latter may lead to better estimators of $\boldsymbol{\beta}^*$ and $\mathbf{X}\boldsymbol{\beta}^*$, due to the least squares post-processing. Polishing of the ℓ_1-Dantzig Selector may not lead to better solutions, due to the weak variable selection properties of "L1-DS". In some cases, when the value of ρ is quite large and, consequently, the covariates are highly correlated (see Example-C, Example-D), the basic problem of variable selection becomes difficult: instead of choosing a "signal" variable, the "L0-DS" chooses its correlated surrogate. In these cases, as expected, we observe that the

Example-A

$(n = 200, p = 500)$

Example-C

$(n = 100, p = 500)$

Example-D

$(n = 100, p = 300)$

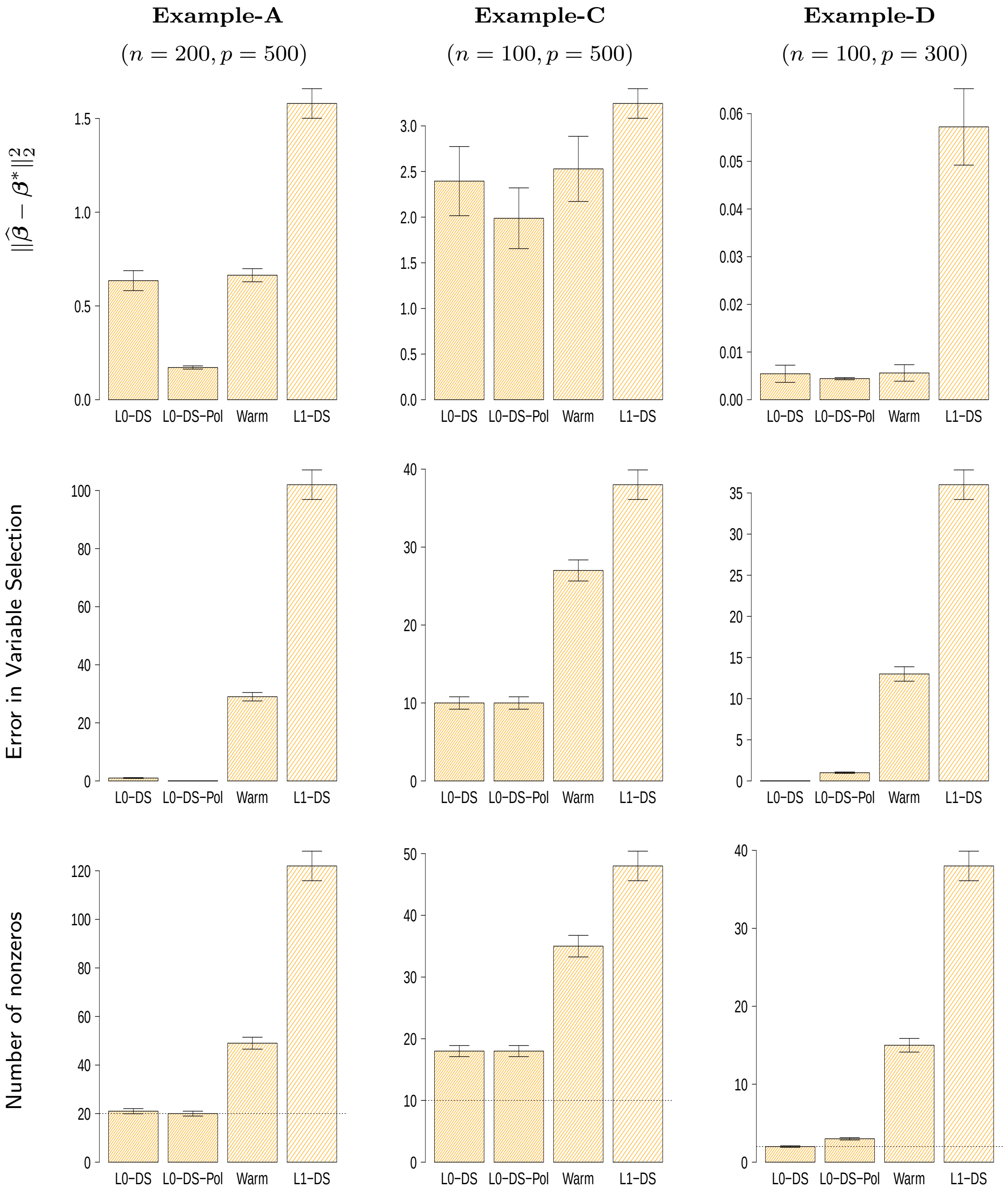

Figure 5: The statistical performance of the *Discrete Dantzig Selector* (L0-DS); its polished version, which does a least squares refitting on the support (L0-DS-Pol); the heuristic estimates delivered by Algorithm 2 (Warm); and the original ℓ_1-Dantzig Selector (L1-DS). We display three different metrics: [top panel] squared ℓ_2-error in estimating $\boldsymbol{\beta}$, [middle panel] the 0-1 variable selection error and [bottom panel] the number of nonzeros in the optimally selected model; the horizontal dotted line shows the number of nonzeros in the "true" model. We observe that the *Discrete Dantzig Selector* based approaches perform very well in terms of obtaining a model with high quality estimation and variable selection properties. Their models are substantially sparser than those for the ℓ_1-based methods. The heuristic approach, "Warm" which approximately optimizes Problem (2), falls short in obtaining high quality statistical estimates. A number of additional experiments (with results similar to this figure), are presented in Section C of the Appendix.

Example	Metric	Time (secs)	L0-DS-Pol	MIPGO
$\rho = 0$	Variable Selection Error	60	0	20
$n = 100$		200	0	14
$p = 200$	$\Vert\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\Vert_2^2$	60	0.309	20.00
$k^* = 20$		200	0.309	8.79
$\rho = 0.7$	Variable Selection Error	60	0	10
$n = 100$		200	0	4
$p = 200$	$\Vert\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\Vert_2^2$	60	0.063	0.271
$k^* = 10$		200	0.063	0.076
$\rho = 0.7$	Variable Selection Error	100	0	9
$n = 300$		500	0	5
$p = 600$	$\Vert\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\Vert_2^2$	100	0.094	9.142
$k^* = 20$		500	0.094	1.548
$\rho = 0.7$	Variable Selection Error	150	0	25
$n = 300$		1000	0	16
$p = 1000$	$\Vert\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\Vert_2^2$	150	0.175	25.00
$k^* = 25$		1000	0.175	14.893

Table 3: In all the above instances, the generated data is of `Type-Synth`, with SNR=10. Both the *Discrete Dantzig Selector* and MIPGO (with the MCP penalty) methods were run on twenty different values of the tuning parameter, and the best solutions are reported. L0-DS-Pol refers to the least squares solution obtained on the variables selected by *Discrete Dantzig Selector*. For every value of the tuning parameter, each method was run for a time budget of t_1, t_2 seconds with $t_1 < t_2$ specified in the "Time" column. The *Discrete Dantzig Selector* method reaches the best solution within t_1 seconds, much earlier than its competitor. The MIPGO method is seen to take orders of magnitude longer to get a solution of the same quality as the *Discrete Dantzig Selector*— the differences become more pronounced with increasing problem size.

"L0-DS" incurs relatively large variable selection error–the prediction accuracy of these models however, demonstrate a more optimistic picture than the variable selection properties (See Table 4).

7.1 Comparisons with Least Squares Subset Selection

We discuss some comparisons of our proposal with the recently proposed methods: Problem (4) by [8] and MIPGO [33].

For underdetermined problems (with $n < p$) the authors in [8] (see Section 5.3.2 in [8]) point out that `MIQO` solvers take a long time to *certify* optimality, by producing matching upper and lower bounds. For problems with $n \leq 100$ and $p = 1000$, [8] demonstrated how the `MIQO` methods for Problem (4) could certify *local* optimality[9] in a (small) bounding box around a candidate solution. We observed in our computational experiments that Problem (2) is orders of magnitude faster than (4) for underdetermined problems, in obtaining solutions with *certificates* of global

[9] We note that certifying local optimality i.e., optimality in a neighborhood of a candidate solution is also an NP-hard problem.

optimality. On several randomly generated problem instances generated as per `Type-Synth` with $n = 100$, $p = 200$, $\rho = 0$, $k^* = 10$ and SNR=10, Problem (2) was solved to global optimality, i.e., zero optimality gap with a median time of about 4 minutes. On the same instances, the `MIQO` formulation for Problem (4) took more than 7 hours of computation time to obtain similar optimality certificates. In addition, a `MILO` formulation for Problem (2) consumes much less memory than a comparable `MIQO` formulation for Problem (4). For example, on problem instances with $n = p \in \{1.5, 2, 2.5, 3\} \times 10^3$, we observed that Problem (4) requires at least twice as much memory as that for Problem (2), within the first 800 seconds of computation time. The memory requirement for a `MIQO` for Problem (4) with $n = p = 3000$ was more than 12GB.

MIPGO [33] is a discrete optimization framework for minimizing a regularized version of the least squares loss, with a nonconvex quadratic penalty (for example, SCAD or MCP). This corresponds to a nonconvex *quadratic* optimization problem, which the authors express as a discrete linear optimization problem via linear complementary constraints [24]. This representation results in many more binary variables (several multiples of p) than that required for the *Discrete Dantzig Selector*. For example, in the case of the MCP penalty, the paper [33] presents a `MILO` with $4p$ binary variables and many more continuous variables. In particular, with $n = 300$ and $p = 1000$, the MIPGO solver[10] creates a problem with 38,000 variables and 27,000 equality constraints, which after presolve reduces to a problem with approximately 7,000 continuous and 4,000 binary variables. These optimization problems are substantially larger than (6), which is a `MILO` with p continuous and as many binary variables. As a result, the MIPGO formulation seems to become computationally expensive as the dimensionality of the problem increases. This is illustrated in Table 3 – we show on several synthetic instances that, with a particular time budget, the *Discrete Dantzig Selector* formulation (6) equipped with a warm-start from Algorithm 2, leads to better solutions than MIPGO. The quality of solutions produced by MIPGO is found to improve with more computation time, but the time taken can be substantially larger than that of the *Discrete Dantzig Selector*. The synthetic datasets for Table 3 were generated the same way as in Section 6. We set the concavity parameter for the MCP penalty in the MIPGO code to its default value of $a = 2$.

Acknowledgements

The authors thank the anonymous referees for their helpful comments that led to improvements in the manuscript. The authors also thank Emmanuel Candes and Robert Freund for helpful suggestions and encouragement. R.M. thanks Jonathan Goetz, Juan-Pablo Vielma and Dimitris Bertsimas for helpful discussions. R.M.'s research was partially supported by ONR N000141512342 and a grant from the Moore Sloan Foundation. Peter Radchenko's research was partially supported

[10] We used the code of [33], obtained from the first author's website. The numbers are read off from the Gurobi log report.

by NSF Grant DMS-1209057.

A Proofs for Section 3

Proof of Theorem 1. Let $\tilde{\epsilon}$ denote the projection of $\mathbf{y}$ onto the orthogonal complement to the space spanned by the predictor vectors $\mathbf{x}_j, j = 1, ...p$. Note that

$$\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 = \sum_{j=1}^{p}(c_j - \beta_j)^2 + \|\tilde{\epsilon}\|^2.$$

Thus, under the constraint $\|\boldsymbol{\beta}\|_0 \leq k$, the smallest sum of squares is achieved by setting $\beta_j = c_j$ for $j = 1, ..., k$ and $\beta_j = 0$ for $j = k+1, ..., p$. This completes the proof of part 1.

Note that

$$|\mathbf{x}_j^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})| = |c_j - \beta_j|.$$

Thus, the constraint $\|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta$ is satisfied if and only if $\beta_j \in [c_j - \delta, c_j + \delta]$ for $j = 1, ..., p$. In order to minimize $\|\boldsymbol{\beta}\|_0$, coefficients β_j for which $0 \in [c_j - \delta, c_j + \delta]$ are set to zero. Thus, $\beta_j = 0$ if and only if $|c_j| \leq \delta$, which implies $\beta_j = 0$ for $j > k$. This completes the proof of part 2.

Proof of Theorem 2. Throughout this proof we omit the words "with probability tending to one" to improve the presentation. The constraint $\|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta$ implies $\|\mathbf{X}^\top\mathbf{X}(\boldsymbol{\beta}^* - \boldsymbol{\beta}) + \mathbf{X}^\top\boldsymbol{\epsilon}\|_\infty \leq \delta$. Recall that $\delta = o(n^{1/2})$ and note that $\|\mathbf{X}^\top\boldsymbol{\epsilon}\|_\infty = O_p(1)$, due to the scaling of the predictors and the assumptions on the $\boldsymbol{\epsilon}$. Consequently, $\|\mathbf{X}^\top\mathbf{X}(\boldsymbol{\beta}^* - \boldsymbol{\beta})\|_2 = o_p(n^{1/2})$. Because $\mathbf{X}^\top\mathbf{X}$ converges to an invertible matrix C, we conclude that there exists a $o_p(1)$ sequence of random variables b_n, such that the bound

$$\|n^{-1/2}\widehat{\boldsymbol{\beta}} - n^{-1/2}\boldsymbol{\beta}^*\|_2 \leq b_n$$

simultaneously holds for all the Discrete Dantzig solutions $\widehat{\boldsymbol{\beta}}$. Recall that $\boldsymbol{\beta}^* = n^{1/2}\tilde{\boldsymbol{\beta}}^*$, for some fixed vector $\widetilde{\boldsymbol{\beta}}^*$. Consequently, $\beta_j^* \neq 0$ implies $\widehat{\beta}_j \neq 0$ for $j = 1, ..., p$. In other words the support of each Discrete Dantzig solution contains J^*. It is only left to show that the cardinality of each such support cannot be greater than $|J^*|$. Note that with probability tending to one, $\boldsymbol{\beta}^*$ is feasible for the Discrete Dantzig optimization problem. Indeed,

$$\|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^*)\|_\infty = \|\mathbf{X}^\top\boldsymbol{\epsilon}\|_\infty = O_p(1),$$

which is bounded above by δ, due to the assumption $\delta \to \infty$. Thus, inequality $\|\widehat{\boldsymbol{\beta}}\|_0 \leq |J^*|$ holds for each Dantzig Selector solution, which completes the proof of part 1.

In the paragraph above we deduced that the minimum value of the Discrete Dantzig objective function equals $|J^*|$. We also showed that $\boldsymbol{\beta}^*$ is feasible for the Discrete Dantzig optimization problem. A similar argument establishes the feasibility of $\boldsymbol{\beta}^O$. Consequently, $\boldsymbol{\beta}^*$ and $\boldsymbol{\beta}^O$ are indeed Discrete Dantzig solutions, which completes the proof of part 2.

Proof of Proposition 1. Consider an arbitrary nonzero $\boldsymbol{\theta} \in \mathbb{R}^p$, such that $\|\boldsymbol{\theta}\|_0 \leq 2k$. Let J_0 be the index set of the k largest, in magnitude, coordinates of $\boldsymbol{\theta}$. Observe that $\|\boldsymbol{\theta}_{J_0^c}\|_1 \leq \|\boldsymbol{\theta}_{J_0}\|_1$ and $\|\boldsymbol{\theta}\|_2 = \|\boldsymbol{\theta}_{J_{01}}\|_2$, because $m \geq k$. Thus

$$\frac{\|\mathbf{X}\boldsymbol{\theta}\|_2}{\|\boldsymbol{\theta}\|_2} = \frac{\|\mathbf{X}\boldsymbol{\theta}\|_2}{\|\boldsymbol{\theta}_{J_{01}}\|_2} \geq \kappa(k, c_0, m),$$

for $c_0 \geq 1$, which implies $\gamma(2k) \geq \kappa(k, c_0, m)$. Also note that

$$2\frac{\|\mathbf{X}\boldsymbol{\theta}\|_2^2}{\|\boldsymbol{\theta}\|_2^2} \geq \frac{\|\mathbf{X}\boldsymbol{\theta}\|_2^2}{\|\boldsymbol{\theta}_{J_0}\|_2^2} \geq [\kappa(k, c_0)]^2,$$

for $c_0 \geq 1$, which gives $\gamma(2k) \geq \kappa(k, c_0)/\sqrt{2}$.

Proof of Theorem 3 and Corollary 2. Note that $\mathbf{X}^\top \boldsymbol{\epsilon}$ is a mean zero Gaussian vector, such that the variance of each component is σ^2. Consequently, it follows from well-known maximal inequalities for Gaussian variables that the bound $\|\mathbf{X}^\top \boldsymbol{\epsilon}\|_\infty \leq \delta$ holds with probability at least $1 - (p^a \sqrt{\pi \log p})^{-1}$. The rest of the proof is conducted on the set where the above bound is valid. Note that on this set $\boldsymbol{\beta}^*$ is a feasible solution for the optimization problem (2), which implies $\|\widehat{\boldsymbol{\beta}}\|_0 \leq \|\boldsymbol{\beta}^*\|_0$. Recall that we denote $\|\boldsymbol{\beta}^*\|_0$ by s^* and derive the following inequalities:

$$\begin{aligned}
&\gamma(2s^*)^2 \left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_2^2 \leq \left\|\mathbf{X}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)\right\|_2^2 \\
&= \ (\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)^\top \mathbf{X}^\top \mathbf{X}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*) \\
&\leq \ \left\|\mathbf{X}^\top \mathbf{X}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)\right\|_\infty \left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_1 \\
&\leq \ \left(\left\|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})\right\|_\infty + n^{-1}\left\|\mathbf{X}^\top \boldsymbol{\epsilon}\right\|_\infty\right) \left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_1.
\end{aligned}$$

Because both $\|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})\|_\infty$ and $\|\mathbf{X}^\top \boldsymbol{\epsilon}\|_\infty$ are bounded above by δ, we derive

$$\left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_2^2 \leq \gamma(2s^*)^{-2} 2\delta \left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_1.$$

Applying inequality $\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_1 \leq (2s^*)^{1/2} \|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2$ to either the left or the right hand side of the above display yields the ℓ_1 and the ℓ_2 estimation bounds, respectively, in the statement of Theorem 3.

Finally, to establish the prediction error bound, observe the following inequality:

$$\left\|\mathbf{X}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)\right\|_2^2 \leq 2\delta \left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_1,$$

which is is a direct consequence of the two displays given above. We then complete the proof of Theorem 3 by combining the above display with the inequalities

$$\begin{aligned} \left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_1 &\leq (2s^*)^{1/2} \left\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\right\|_2 \\ &\leq (2s^*)^{1/2} \gamma(2s^*)^{-1} \left\|\mathbf{X}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)\right\|_2 . \end{aligned}$$

Corollary 2 follows directly from the ℓ_1 estimation bound in Theorem 3.

Proof of Theorem 4. We again focus on the set of high probability, where inequality $\|\mathbf{X}^\top \boldsymbol{\epsilon}\|_\infty \leq \delta$ holds. Because $\boldsymbol{\beta}^*$ is a feasible solution to the optimization problem (2), we have $\|\boldsymbol{\beta}^*\|_0 \geq \widehat{s}_{LB}$. Thus, $\|\widehat{\boldsymbol{\beta}}\|_0 \leq (1+\psi)\|\boldsymbol{\beta}^*\|_0$. The rest of the proof is identical to the one for Theorem 3, with one exception: the bound $\|\widehat{\boldsymbol{\beta}}\|_0 \leq \|\boldsymbol{\beta}^*\|_0$ contains an additional factor $(1+\psi)$.

B Additional Algorithm Details and Proofs

B.1 Details on Algorithm 1

Update (11) in the ADMM algorithm can be performed via a hard thresholding operation [18], as it is of the form:

$$\widehat{\boldsymbol{\beta}}(\lambda') := \arg\min_{\boldsymbol{\beta}} \|\boldsymbol{\beta} - \mathbf{c}\|_2^2 + \lambda' \|\boldsymbol{\beta}\|_0, \tag{24}$$

for an appropriately chosen $\lambda' = \frac{2}{\lambda}$ and $\mathbf{c} = \boldsymbol{\alpha} + \frac{1}{\lambda}\boldsymbol{\nu}$; a solution is given by $\widehat{\beta}_j(\lambda') = c_j \mathbf{1}(|c_j| > \sqrt{\lambda'}), j = 1, \ldots, p$. The update step (12) involves the following projection:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \;\; & f(\boldsymbol{\alpha}) := \|\boldsymbol{\alpha} - \bar{\mathbf{c}}\|_2^2 \\ \text{s.t.} \;\; & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\alpha})\|_\infty \leq \delta, \end{aligned} \tag{25}$$

where $\bar{\mathbf{c}} = \boldsymbol{\beta} - \boldsymbol{\nu}/\lambda$. While the projection (25) can be computed by using standard quadratic programming methods, in our experience, we found them[11] to be quite time consuming for larger problems ($p \geq 1000$), especially because this projection needs to be computed for every iteration (indexed by k) of (11)–(13). Thus, we recommend using specialized first-order methods — these methods also naturally make use of warm-start information, which is particularly useful to us due to the iterative nature of the updates (11)–(13). Unless $\mathbf{X}$ has uncorrelated columns, it is not straightforward to solve (25) in its primal form — we thus consider a dual of Problem (25),

[11] Our reference is GUROBI's quadratic programming solver.

for which we apply first-order methods for convex composite minimization [36]. To improve the flow of presentation, we relegate the description of a more general first-order method, which also applies to Problem (25), to Section B.3 in the Appendix. We repeat steps (11)–(13) until an (approximate) convergence criterion is met — see for example [34, 12] for convergence results for the general method. We terminate the algorithm as soon as the successive changes in the $\boldsymbol{\beta}$ updates become small and one has approximate primal feasibility (see Step (3) in Algorithm 1).

B.2 Additional Details on Algorithm 2: Solving Problem (17)

Observe that Problem (17) is of the composite form [36]:

$$\min_{\boldsymbol{\theta}} \quad f_1(\boldsymbol{\theta}) + f_2(\boldsymbol{\theta}) \quad \text{s.t.} \quad \boldsymbol{\theta} \in \mathcal{C}, \tag{26}$$

where the function $f_1(\boldsymbol{\theta})$ is smooth, with its gradient Lipschitz continuous: $\|\nabla f_1(\boldsymbol{\theta}) - \nabla f_1(\boldsymbol{\theta}')\| \leq L\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|$; $f_2(\boldsymbol{\theta})$ is nonsmooth and $\mathcal{C}$ is a convex set. In our specific case, the smooth component is the zero function, $f_2(\boldsymbol{\theta}) = \sum_{i=1}^{p} w_i|\theta_i|$ and $\mathcal{C} = \{\boldsymbol{\theta} : \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})\|_\infty \leq \delta\}$. Thus, one may appeal to first-order optimization methods [2, 37, 36] for composite function minimization. This requires solving, at every iteration, a problem of the form:

$$\begin{array}{rll} \boldsymbol{\theta}^{m+1} \in & \underset{\boldsymbol{\theta}}{\arg\min} & \frac{L}{2}\|\boldsymbol{\theta} - \overline{\boldsymbol{\theta}}^m\|_2^2 + \sum\limits_{i=1}^{p} w_i|\theta_i| \\ & \text{s.t.} & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})\|_\infty \leq \delta, \end{array} \tag{27}$$

for some choice of $L > 0$ and $\overline{\boldsymbol{\theta}}^m$, and $w_i = |\rho_\gamma'(|\beta_i^k|)|$. If $\overline{\boldsymbol{\theta}}^m = \boldsymbol{\theta}^m$, then the above update sequence becomes identical to proximal gradient descent [2]. One may also use accelerated gradient descent methods, with a momentum term. We describe in Section B.3 first-order gradient methods that can be used to compute solutions to Problem (27). The sequence $\boldsymbol{\theta}^m$, defined via (27), leads to the solution of Problem (17) as $m \to \infty$, providing a $O(\frac{1}{m})$-suboptimal solution in m many iterations if one uses standard proximal gradient descent methods; the convergence rate can be improved to $O(\frac{1}{m^2})$ if one uses the accelerated gradient descent version of the algorithm.

Instead of choosing $f_1(\boldsymbol{\theta}) \equiv 0$ one may also choose $f_1(\boldsymbol{\theta}) = \frac{\tau}{2}\|\boldsymbol{\theta}\|_2^2$, for a small value of $\tau > 0$. Interestingly, for small values of τ the minimizer to Problem (26) is also a minimizer of the problem with the choice $f_1(\boldsymbol{\theta}) = 0$. This equivalence of solutions which holds true in much more generality is often known as *exact* regularization of convex programs in the mathematical programming literature — see for example [22]. Even if the two problems are not equivalent, the choice of $f_1(\boldsymbol{\theta}) = \frac{\tau}{2}\|\boldsymbol{\theta}\|_2^2$ always serves as an approximate solution to Problem (17). With this choice of $f_1(\boldsymbol{\theta})$, one needs to solve a problem of the form:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \frac{\tau}{2}\|\boldsymbol{\theta}\|_2^2 + \sum_{i=1}^{p} w_i|\theta_i| \\ \text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})\|_\infty \leq \delta. \end{aligned}$$

A solution to the above problem can be computed by considering its dual and applying (accelerated) proximal gradient methods on the dual formulation, as described in Section B.3. In this approach, a two-stage iterative algorithm of the form (27) described above, is not required.

Algorithm 2 suggests that we solve Problem (17) repeatedly for different values of γ — it turns out that the overall cost for solving all these problems is quite small. This is because (a) the problems do not change much across different values of γ; and (b) for a fixed γ, while moving across different values of k, the linear optimization problems are quite similar since the weights $|\rho'_\gamma(|\beta_i^k|)|$ do not change much across k. Thus the solutions obtained from one linear optimization problem can be used as a warm-start to solve the next linear optimization problem. This is found to reduce the overall computation time. Both the first-order methods (described above) and simplex methods can gracefully take advantage of warm-starts.

B.3 Dual Gradient Method

Here we describe how to solve a problem of the form:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \ & \tfrac{1}{2}\|\boldsymbol{\alpha} - \bar{\mathbf{c}}\|_2^2 + \sum_{i=1}^{p} w_i|\alpha_i| \\ \text{s.t.} \ & \|\mathbf{A}\boldsymbol{\alpha} - \mathbf{b}\|_\infty \leq \delta, \end{aligned} \tag{28}$$

where we assume that $w_i \geq 0$ and the set $\{\boldsymbol{\alpha} : \|\mathbf{A}\boldsymbol{\alpha} - \mathbf{b}\|_\infty \leq \delta\}$ is nonempty. Note that the constraint set in (28) makes solving the primal form (28) challenging. However, due to the strong convexity of the objective a dual is smooth (has Lipschitz continuous gradient) and the non differentiability nicely separates across the dual variables – we thus use dual proximal gradient algorithms to optimize (28). This trick is often used in optimization and signal processing, see for example [17].

To derive a dual for Problem (28), we note that it can be written equivalently as:

$$\begin{aligned} \min_{\boldsymbol{\alpha},\boldsymbol{\zeta}} \ & \tfrac{1}{2}\|\boldsymbol{\alpha} - \bar{\mathbf{c}}\|_2^2 + \sum_{i=1}^{p} w_i|\alpha_i| \\ \text{s.t.} \ & \|\boldsymbol{\zeta}\|_\infty \leq \delta, \\ & \boldsymbol{\zeta} = \mathbf{A}\boldsymbol{\alpha} - \mathbf{b}. \end{aligned}$$

The minimum of the above problem can be obtained by maximizing a *dual* problem, obtained by dualizing the equality constraints $\boldsymbol{\zeta} = \mathbf{A}\boldsymbol{\alpha} - \mathbf{b}$; this consequently leads to the following problem:

$$g(\boldsymbol{\mu}) := \min_{\boldsymbol{\zeta},\boldsymbol{\alpha}:\|\boldsymbol{\zeta}\|_\infty \leq \delta} \Big(\tfrac{1}{2}\|\boldsymbol{\alpha} - \overline{\mathbf{c}}\|_2^2 + \langle \boldsymbol{\mu}, \boldsymbol{\zeta} - (\mathbf{A}\boldsymbol{\alpha} - \mathbf{b})\rangle \\ + \sum_{i=1}^p w_i|\alpha_i|\Big).$$

The above can be simplified to:

$$\begin{aligned} & g(\boldsymbol{\mu}) \\ =& \min_{\boldsymbol{\alpha}} \left(\tfrac{1}{2}\|\boldsymbol{\alpha} - \overline{\mathbf{c}}\|_2^2 - \|\boldsymbol{\mu}\|_1\delta - \langle \boldsymbol{\mu}, (\mathbf{A}\boldsymbol{\alpha} - \mathbf{b})\rangle + \sum_{i=1}^p w_i|\alpha_i| \right) \\ =& g_1(\boldsymbol{\mu}) - \delta\|\boldsymbol{\mu}\|_1, \end{aligned}$$

where,

$$g_1(\boldsymbol{\mu}) = \min_{\boldsymbol{\alpha}} \left(\tfrac{1}{2}\|\boldsymbol{\alpha} - \overline{\mathbf{c}}\|_2^2 - \langle \boldsymbol{\mu}, (\mathbf{A}\boldsymbol{\alpha} - \mathbf{b})\rangle + \sum_{i=1}^p w_i|\alpha_i| \right).$$

Note that $\widehat{\boldsymbol{\alpha}}$, the unique minimizer of the above problem, is given by:

$$\begin{aligned} \widehat{\boldsymbol{\alpha}} =& \arg\min_{\boldsymbol{\alpha}} \ \tfrac{1}{2}\|\boldsymbol{\alpha} - (\overline{\mathbf{c}} + \mathbf{A}^\top\boldsymbol{\mu})\|_2^2 + \sum_{i=1}^p w_i|\alpha_i|, \\ \text{i.e.,} \quad \widehat{\alpha}_i =& \operatorname{sgn}(\overline{c}_i + \mathbf{a}_i^\top\boldsymbol{\mu}) \cdot \max\left\{|\overline{c}_i + \mathbf{a}_i^\top\boldsymbol{\mu}| - w_i, 0\right\}, \end{aligned}$$

for $i = 1, \ldots, p$, where $\mathbf{a}_i$ is the ith column of $\mathbf{A}$. It follows from standard convex analysis [39] that the function $\boldsymbol{\mu} \mapsto g_1(\boldsymbol{\mu})$ is differentiable with its gradient given by:

$$\nabla g_1(\boldsymbol{\mu}) = -(\mathbf{A}\widehat{\boldsymbol{\alpha}} - \mathbf{b}),$$

and its gradient is Lipschitz continuous:

$$\|\nabla g_1(\boldsymbol{\mu}) - \nabla g_1(\boldsymbol{\mu}')\| \leq \|\mathbf{A}\|_2^2\|\boldsymbol{\mu} - \boldsymbol{\mu}'\|,$$

where $\|\mathbf{A}\|_2$ denotes the largest singular value of $\mathbf{A}$ and for a vector $\mathbf{u}$, the term $\|\mathbf{u}\|$ denotes the usual ℓ_2-norm of $\mathbf{u}$.

By using standard quadratic programming duality theory [11], the minimum of Problem (28) can be obtained by maximizing the unconstrained dual problem $g(\boldsymbol{\mu})$ in the dual variable $\boldsymbol{\mu}$, which is equivalent to the following minimization problem:

$$\min_{\boldsymbol{\mu}} \ -g(\boldsymbol{\mu}) = \min_{\boldsymbol{\mu}} \left(-g_1(\boldsymbol{\mu}) + \delta\|\boldsymbol{\mu}\|_1\right). \tag{29}$$

This problem is of the composite form [36], and proximal gradient descent methods [37, 36, 38] apply to it directly.

For the special case of Problem (25), the method described above applies with $w_i = 0, i = 1, \ldots, p$. Clearly, (29) is an ℓ_1-regularized quadratic program, with the primal dual relationship being: $\boldsymbol{\alpha} = \overline{\mathbf{c}} + \mathbf{A}^\top \boldsymbol{\mu}$.

Note that Problem (28) needs to be solved several times during the course of Algorithm 1 and Algorithm 2, across the different iterations. Fortunately, these problems are not completely unrelated, in fact, they are quite "similar". In Algorithm 2 the weights w_i change; and in Algorithm 1, the parameter $\overline{\mathbf{c}}$ changes. Since the problems are similar, it is not unreasonable to expect that the optimal dual variables corresponding to these two problems do not change much. Thus, it is useful to initialize the dual variable $\boldsymbol{\mu}$ for one instantiation of Problem (28) with the (dual) solution obtained from another instantiation of Problem (28). This simple strategy leads to substantial performance gains over solving the problems independent of one another.

B.4 Proof of Theorem 5

Proof. Note that the sequence $\boldsymbol{\beta}^k$, defined via (17) satisfies the following relationship:

$$\begin{aligned} h(\boldsymbol{\beta}^k) =& \overline{h}(\boldsymbol{\beta}^k; \boldsymbol{\beta}^k) \\ \geq & \min_{\boldsymbol{\beta}} \ \overline{h}(\boldsymbol{\beta}; \boldsymbol{\beta}^k) \ \text{ s.t. } \ \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta \\ =& \overline{h}(\boldsymbol{\beta}^{k+1}; \boldsymbol{\beta}^k). \end{aligned}$$

Observing that

$$\overline{h}(\boldsymbol{\beta}^{k+1}; \boldsymbol{\beta}^k) \geq h(\boldsymbol{\beta}^{k+1}),$$

we have:

$$h(\boldsymbol{\beta}^k) = \overline{h}(\boldsymbol{\beta}^k; \boldsymbol{\beta}^k) \geq \overline{h}(\boldsymbol{\beta}^{k+1}; \boldsymbol{\beta}^k) \geq h(\boldsymbol{\beta}^{k+1}), \tag{30}$$

and, thus, the sequence $h(\boldsymbol{\beta}^k)$ is decreasing. Subtracting $h(\boldsymbol{\beta}^k)$ from all sides of the above inequality, we derive:

$$0 \geq \overline{h}(\boldsymbol{\beta}^{k+1}; \boldsymbol{\beta}^k) - h(\boldsymbol{\beta}^k) \geq h(\boldsymbol{\beta}^{k+1}) - h(\boldsymbol{\beta}^k).$$

The first part of the above display gives us, using (16):

$$\begin{aligned} 0 \geq & \overline{h}(\boldsymbol{\beta}^{k+1}; \boldsymbol{\beta}^k) - h(\boldsymbol{\beta}^k) \\ =& \sum_{i=1}^p \left\langle \rho'_\gamma(|\beta_i^k|), |\beta_i^{k+1}| - |\beta_i^k| \right\rangle = \Delta(\boldsymbol{\beta}^k), \end{aligned}$$

which means $\Delta(\boldsymbol{\beta}^k) \leq 0$ for all k. If $\Delta(\boldsymbol{\beta}^k) < 0$, then $\boldsymbol{\beta}^{k+1}$ leads to a strictly improved value of the objective function. If $\Delta(\boldsymbol{\beta}^k) = 0$, then $\boldsymbol{\beta}^k$ is a fixed point of the above update equation. Hence, $\Delta(\boldsymbol{\beta}^k)$ is a measure of how far $\boldsymbol{\beta}^k$ is from a first-order stationary point of Problem (15).

The display in (30) shows that the objective values are decreasing, and, because the objective values are all bounded below (by zero), the decreasing sequence converges.

In addition, we have that

$$h(\boldsymbol{\beta}^k) - h(\boldsymbol{\beta}^{k+1}) \geq -\Delta(\boldsymbol{\beta}^k).$$

Adding the above for $k = 1, \ldots, \mathcal{K}$, we have:

$$\begin{aligned} h(\boldsymbol{\beta}^1) - h(\boldsymbol{\beta}^{\mathcal{K}+1}) \quad &\geq \quad \sum_{\mathcal{K} \geq k \geq 1} \left\{-\Delta(\boldsymbol{\beta}^k)\right\} \\ &\geq \quad \mathcal{K} \min_{1 \leq k \leq \mathcal{K}} \left\{-\Delta(\boldsymbol{\beta}^k)\right\}, \end{aligned} \tag{31}$$

which leads to the following convergence rate:

$$\min_{1 \leq k \leq \mathcal{K}} \left\{-\Delta(\boldsymbol{\beta}^k)\right\} \leq \frac{1}{\mathcal{K}} \left(h(\boldsymbol{\beta}^1) - h(\boldsymbol{\beta}^{\mathcal{K}+1})\right) \tag{32}$$

$$\leq \frac{1}{\mathcal{K}} \left(h(\boldsymbol{\beta}^1) - \widehat{h}\right), \tag{33}$$

where (33) follows from (32) by using the observation that $h(\boldsymbol{\beta}^k) \downarrow \widehat{h}$.

□

B.5 Additional details on Algorithm 3

We seek an upper bound to a simple variant of Problem (15):

$$\begin{aligned} \min_{\boldsymbol{\beta}} \;\; & h(\boldsymbol{\beta}) := \sum_{i=1}^{p} \rho_\gamma(|\beta_i|) \\ \text{s.t.} \;\; & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta \\ & \beta_i = 0, i \in \mathcal{I}^c, \end{aligned} \tag{34}$$

where $\text{Supp}(\widehat{\boldsymbol{\beta}}^{(1)}) := \{i : \widehat{\beta}_i^{(1)} \neq 0, i = 1, \ldots, p\} \subset \mathcal{I}$, and $\mathcal{I}^c$ is the complement of $\mathcal{I}$. We assume, of course, that the feasible set in Problem (34) is nonempty. A simple method for constructing $\mathcal{I}$, which we found to be quite useful in practice, is presented below. Let $\mathcal{B} \subset \{1, \ldots, p\}$ and $\mathcal{B}^c$ denote its complement. We define the following set:

$$\mathcal{F}(\mathcal{B}) := \left\{\boldsymbol{\beta} : \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \leq \delta, \beta_i = 0, i \in \mathcal{B}^c\right\}.$$

Let $\widehat{\boldsymbol{\alpha}}^{(1)}, \widehat{\boldsymbol{\beta}}^{(1)}$ be the solutions produced by Algorithm 1. Suppose we let B denote the support of $\widehat{\boldsymbol{\beta}}^{(1)}$; the size of B is typically much smaller than p. If $\mathcal{F}(B)$ is nonempty, we take $\mathcal{I} = B$. Note, however, that $\mathcal{F}(B)$ may be empty, because $\widehat{\boldsymbol{\alpha}}^{(1)}, \widehat{\boldsymbol{\beta}}^{(1)}$ are only approximately equal: $\widehat{\boldsymbol{\alpha}}^{(1)} \approx \widehat{\boldsymbol{\beta}}^{(1)}$. In this case, we need expand the set B, so that the set $\mathcal{F}(B)$ becomes nonempty. There may be several ways to do this, but we found the following simple method to be quite useful in our numerical experiments.

1. If $\mathcal{F}(B)$ is empty, we consider the set $\{|\widehat{\alpha}_i^{(1)}|, i \in B^c\}$ and find the index of the largest element in this set, which we denote by: $\widehat{i} \in \arg\max_{i \in B^c} \ |\widehat{\alpha}_i^{(1)}|$.
2. Make B larger by including this new feature $\widehat{i}$: we thus have $B \leftarrow B \cup \{\widehat{i}\}$.
3. Check if the resulting set $\mathcal{F}(B)$ is nonempty, if not, we repeat the above steps until $\mathcal{F}(B)$ becomes nonempty.
4. We let $\mathcal{I}$ be the resulting set B obtained upon termination: $\mathcal{I} = B$.

Problem (34), which is an optimization problem with fewer variables than Problem (15) is found to deliver solutions that are better upper bounds to Problem (2). This also leads to better and numerically more robust solutions than those available directly from Algorithm 1. The general algorithmic framework via sequential linear optimization, presented in Section 4.1.2, readily applies to obtain good upper bounds to Problem (34).

B.6 Tighter bounds on $\hat{\beta}_i$'s

The bounds described via (22) can be sharpened by making use of good upper bounds to the solution of Problem (2). Towards this end, we need to reformulate (6). Note that in Problem (6), if we take $\mathcal{M}_U$ to be to be sufficiently large then this will lead to a solution for Problem (2). We rewrite Problem (6) as follows:

$$\begin{aligned} \min_{\boldsymbol{\beta}, \mathbf{z}, \alpha} \quad & \alpha \\ \text{s.t.} \quad & \sum_{i=1}^p z_i \leq \alpha \\ & -\delta \leq d_j - \langle \mathbf{q}_j, \boldsymbol{\beta} \rangle \leq \delta, \quad j = 1, \ldots, p \\ & -\mathcal{M}_U z_j \leq \beta_j \leq \mathcal{M}_U z_j, \quad j = 1, \ldots, p \\ & z_j \in \{0, 1\}, \qquad j = 1, \ldots, p, \end{aligned} \tag{35}$$

where the optimization variables are $\boldsymbol{\beta}, \mathbf{z} \in \mathbb{R}^p$ and $\alpha \in \mathbb{R}$.

For a fixed α, consider the feasible set of Problem (35):

$$\mathcal{S}_\alpha = \left\{ (\boldsymbol{\beta}, \mathbf{z}) : \begin{array}{c} \sum_{j=1}^{p} z_j \le \alpha, \;\; \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \le \delta \\ |\beta_j| \le \mathcal{M}_U z_j, \;\; z_j \in \{0,1\}, j = 1, \ldots, p \end{array} \right\}.$$

Observe that

$$\mathcal{S}_\alpha \subset \overline{\mathcal{S}}_\alpha, \tag{36}$$

where

$$\overline{\mathcal{S}}_\alpha = \left\{ (\boldsymbol{\beta}, \mathbf{z}) : \begin{array}{c} \sum_{j=1}^{p} z_j \le \alpha, \;\; \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \le \delta \\ |\beta_j| \le \mathcal{M}_U z_j, \;\; z_j \in [0,1], j = 1, \ldots, p \end{array} \right\}$$

is obtained by relaxing the binary variables $z_j \in \{0,1\}$ into the continuous variables $z_j \in [0,1]$, for all $j = 1, \ldots, p$. Noting that $\mathcal{S}_\alpha \subset \mathcal{S}_{\alpha'}$ for $\alpha \le \alpha'$; and using this along with (36) we have

$$\mathcal{S}_{\alpha^*} \subset \mathcal{S}_{\alpha_0} \subset \overline{\mathcal{S}}_{\alpha_0},$$

where α^* is the optimum value, and α_0 is an upper bound to Problem (35), and hence $\alpha_0 \ge \alpha^*$. The above inequality leads to the following chain of inequalities:

$$\begin{aligned} \min_{(\boldsymbol{\beta},\mathbf{z}) \in \mathcal{S}_{\alpha^*}} \beta_i \ge \min_{(\boldsymbol{\beta},\mathbf{z}) \in \mathcal{S}_{\alpha_0}} \beta_i \ge \min_{(\boldsymbol{\beta},\mathbf{z}) \in \overline{\mathcal{S}}_{\alpha_0}} \beta_i := \mu_i^-(\alpha_0) \\ \max_{(\boldsymbol{\beta},\mathbf{z}) \in \mathcal{S}_{\alpha^*}} \beta_i \le \max_{(\boldsymbol{\beta},\mathbf{z}) \in \mathcal{S}_{\alpha_0}} \beta_i \le \max_{(\boldsymbol{\beta},\mathbf{z}) \in \overline{\mathcal{S}}_{\alpha_0}} \beta_i := \mu_i^+(\alpha_0). \end{aligned}$$

The quantities at the right end above, i.e. $\mu_i^-(\alpha_0)$ and $\mu_i^+(\alpha_0)$, can be computed by solving a pair of linear optimization problems:

$$\begin{aligned} \mu_i^+(\alpha_0) := \quad & \max_{\boldsymbol{\beta}} \;\; \beta_i \\ \text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \le \delta, \\ & \|\boldsymbol{\beta}\|_\infty \le \mathcal{M}_U, \\ & \|\boldsymbol{\beta}\|_1 \le \mathcal{M}_U \alpha_0, \\ \\ \mu_i^-(\alpha_0) := \quad & \min_{\boldsymbol{\beta}} \;\; \beta_i \\ \text{s.t.} \quad & \|\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|_\infty \le \delta, \\ & \|\boldsymbol{\beta}\|_\infty \le \mathcal{M}_U, \\ & \|\boldsymbol{\beta}\|_1 \le \mathcal{M}_U \alpha_0. \end{aligned} \tag{37}$$

The quantities $\mu_i^-(\alpha_0)$ and $\mu_i^+(\alpha_0)$ are lower and upper bounds, respectively, for $\widehat{\beta}_i$ — the bounds depend upon α_0 and $\mathcal{M}_U$. Note that $\mu_i(\alpha_0) := \max\left\{\mu_i^+(\alpha_0), -\mu_i^-(\alpha_0)\right\}$ provides an upper bound to $|\widehat{\beta}_i|$, which consequently leads to an improved estimate for $\|\widehat{\boldsymbol{\beta}}\|_\infty$ — this suggests a way to adaptively refine $\mathcal{M}_U$, and, thus, $\mu_i^-(\alpha_0)$ and $\mu_i^+(\alpha_0)$.

C Additional Experiments

This section complements the experimental results shown in the main body of the paper. Table 4 is an elaborate version of the representative results displayed in Figure 5. Here, we consider different values of SNR and also display the prediction errors. The results show that *Discrete Dantzig Selector* outperforms the ℓ_1-Dantzig Selector based methods in terms of estimating the true underlying regression coefficients, and does so with better variable selection properties.

Example-A $(n = 200, p = 500)$

Metric	SNR	L0-DS	L0-DS-Pol	L1-DS	Warm
$\|\widehat{\beta} - \beta^*\|_2^2$	3	3.858 (0.466)	1.123 (0.254)	5.266 (0.162)	3.429 (0.19)
Variable Selection Error	3	6 (0.543)	16 (0.844)	102 (1.327)	43 (1.327)
Prediction Error	3	0.158 (0.013)	0.581 (0.038)	0.18 (0.004)	0.133 (0.004)
Number of Nonzeros	3	24 (0.274)	35 (0.663)	122 (1.327)	63 (1.266)
$\|\widehat{\beta} - \beta^*\|_2^2$	10	0.635 (0.053)	0.172 (0)	1.58 (0.049)	0.664 (0.035)
Variable Selection Error	10	1 (0.137)	0 (0)	102 (1.327)	29 (1.371)
Prediction Error	10	0.031 (0.002)	0.249 (0)	0.054 (0.001)	0.028 (0.001)
Number of Nonzeros	10	21 (0.137)	20 (0)	122 (1.327)	49 (1.371)

Example-B $(n = 200, p = 500)$

Metric	SNR	L0-DS	L0-DS-Pol	L1-DS	Warm
$\|\widehat{\beta} - \beta^*\|_2^2$	3	150.386 (8.993)	64.17 (4.888)	137.073 (4.511)	102.583 (5.711)
Variable Selection Error	3	11 (0.247)	16 (0.573)	38 (1.266)	14 (0.658)
Prediction Error	3	0.155 (0.007)	0.109 (0.003)	0.132 (0.005)	0.123 (0.005)
Number of Nonzeros	3	16 (0.362)	23 (0.362)	46 (1.447)	22 (0.693)
$\|\widehat{\beta} - \beta^*\|_2^2$	10	45.599 (2.508)	13.773 (1.39)	47.113 (1.016)	38.256 (2.068)
Variable Selection Error	10	9 (0.411)	11 (0.482)	36 (1.387)	14 (0.724)
Prediction Error	10	0.045 (0.001)	0.052 (0.001)	0.043 (0.001)	0.035 (0.001)
Number of Nonzeros	10	17 (0.151)	23 (0.392)	50 (1.266)	26 (0.663)

Example-C $(n = 100, p = 500)$

Metric	SNR	L0-DS	L0-DS-Pol	L1-DS	Warm
$\|\widehat{\beta} - \beta^*\|_2^2$	3	7.322 (0.65)	6.823 (0.693)	6.452 (0.263)	7.674 (0.347)
Variable Selection Error	3	10 (0.573)	9 (0.814)	43 (1.538)	21 (1.116)
Prediction Error	3	0.168 (0.015)	0.966 (0.051)	0.172 (0.004)	0.193 (0.008)
Number of Nonzeros	3	13 (0.271)	13 (0.392)	47 (1.658)	22 (1.096)
$\|\widehat{\beta} - \beta^*\|_2^2$	10	2.395 (0.379)	1.988 (0.333)	3.245 (0.097)	2.529 (0.357)
Variable Selection Error	10	10 (0.795)	10 (0.795)	38 (1.357)	27 (1.343)
Prediction Error	10	0.054 (0.004)	0.515 (0.036)	0.056 (0.002)	0.051 (0.004)
Number of Nonzeros	10	18 (0.685)	18 (0.685)	48 (1.357)	35 (1.266)

Example-D $(n = 100, p = 300)$

Metric	SNR	L0-DS	L0-DS-Pol	L1-DS	Warm
$\|\widehat{\beta} - \beta^*\|_2^2$	3	0.019 (0.006)	0.015 (0)	0.191 (0.027)	0.018 (0.007)
Variable Selection Error	3	0 (0)	0 (0)	36 (0.693)	12 (0.85)
Prediction Error	3	0.033 (0.005)	0.955 (0)	0.101 (0.007)	0.028 (0.003)
Number of Nonzeros	3	2 (0)	2 (0)	38 (0.693)	14 (0.85)
$\|\widehat{\beta} - \beta^*\|_2^2$	10	0.005 (0.002)	0.004 (0)	0.057 (0.008)	0.006 (0.002)
Variable Selection Error	10	0 (0)	1 (0.082)	36 (0.693)	13 (0.877)
Prediction Error	10	0.006 (0.002)	0.527 (0.006)	0.03 (0.002)	0.007 (0.001)
Number of Nonzeros	10	2 (0)	3 (0.082)	38 (0.693)	15 (0.877)

Table 4: Tables showing the statistical performance of four different methods: "L0-DS", "L0-DS-Pol", "L1-DS" and "Warm", described in Section 7. The standard errors are given in parentheses (they are computed in the same fashion as in Figure 5). The *Discrete Dantzig Selector* based methods deliver models with good accuracy in estimating the regression coefficients, and the estimated models are sparser than those for the ℓ_1-based method and the method based on Algorithm 2, which is a heuristic strategy to approximate good upper bounds for the *Discrete Dantzig Selector* problem.

An important advantage of the *Discrete Dantzig Selector* based methods is that they deliver models that are very sparse. The polished version of the *Discrete Dantzig Selector* is found to exhibit better statistical performance than the original *Discrete Dantzig Selector* estimator. Table 5 compares the polished versions of the *Discrete Dantzig Selector* and the ℓ_1-Dantzig Selector, and finds that the performance of the former approach is significantly better.

Example-A $(n = 200, p = 500)$

Metric	SNR	L0-DS-Pol	L1-DS-Pol
$\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2$	3	1.123 (0.254)	2.163 (0.139)
Variable Selection Error	3	16 (0.844)	30 (1.116)
Prediction Error	3	0.581 (0.038)	0.777 (0.02)
Number of Nonzeros	3	35 (0.663)	50 (1.146)
$\|\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2$	10	0.172 (0)	0.282 (0.011)
Variable Selection Error	10	0 (0)	11 (0.392)
Prediction Error	10	0.249 (0)	0.292 (0.003)
Number of Nonzeros	10	20 (0)	31 (0.392)

Table 5: Tables comparing the polished version of *Discrete Dantzig Selector* with the polished version of ℓ_1-Dantzig Selector. The standard errors are given in parentheses. The statistical performance of ℓ_1-Dantzig Selector is inferior, most likely due to its weaker variable selection properties.

References

[1] Top500 Supercomputer Sites, Directory page for Top500 lists. Result for each list since June 1993. `http://www.top500.org/statistics/sublist/`. Accessed: 2013-12-04.

[2] 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.

[3] S. R. Becker, E. J. Candès, and M. C. Grant. Templates for convex cone problems with applications to sparse signal recovery. *Mathematical Programming Computation*, 3(3):165–218, 2011.

[4] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, Massachusetts, 2nd edition, 1999.

[5] D. Bertsimas and A. King. Or forum – an algorithmic approach to linear regression. *Operations Research*, 2015.

[6] D. Bertsimas and R. Mazumder. Least quantile regression via modern optimization. *Annals of Statistics*, 42(6):2494–2525, 2014.

[7] D. Bertsimas and R. Weismantel. *Optimization over integers.* Dynamic Ideas Belmont, 2005.

[8] D. Bertsimas, A. King, and R. Mazumder. Best subset selection via a modern optimization lens. *Annals of Statistics*, 44(2):813–852, 2016.

[9] P. Bickel, Y. Ritov, and A. Tsybakov. Simultaneous analysis of lasso and dantzig selector. *Annals of Statistics*, 37:1705–1732, 2009.

[10] R. E. Bixby. A brief history of linear and mixed-integer programming computation. *Documenta Mathematica, Extra Volume: Optimization Stories*, pages 107–121, 2012.

[11] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, Cambridge, 2004.

[12] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. *Foundations and Trends in Machine Learning.* Number 3(1). Now Publishers, 2011.

[13] P. Bühlmann and S. van-de-Geer. *Statistics for high-dimensional data.* Springer, 2011.

[14] S. Burer and A. Saxena. The MILP road to MIQCP. In *Mixed Integer Nonlinear Programming*, pages 373–405. Springer, 2012.

[15] E. Candes, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

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

[17] P. L. Combettes, DJ. Dũng, and B. C. Vũ. Dualization of signal recovery problems. *Set-Valued and Variational Analysis*, 18(3-4):373–404, 2010.

[18] D. Donoho and I. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81:425–455, 1994.

[19] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression (with discussion). *Annals of Statistics*, 32(2):407–499, 2004. ISSN 0090-5364.

[20] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360(13), 2001.

[21] R. M. Freund, P. Grigas, and R. Mazumder. A new perspective on boosting in linear regression via subgradient optimization and relatives. *Annals of Statistics (to appear)*, 2017.

[22] M. P. Friedlander and P. Tseng. Exact regularization of convex programs. *SIAM Journal on Optimization*, 18(4):1326–1350, 2007.

[23] J. Friedman, T. Hastie, H. Hoefling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 2(1):302–332, 2007.

[24] F. Giannessi and E. Tomasin. Nonconvex quadratic programs, linear complementarity problems, and integer linear programs. In *5th Conference on Optimization Techniques Part I*, pages 437–449. Springer, 1973.

[25] I. Gurobi Optimization. Gurobi optimizer reference manual, 2015. URL `http://www.gurobi.com`.

[26] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning, Second Edition: Data Mining, Inference, and Prediction (Springer Series in Statistics)*. Springer New York, 2 edition, 2009. ISBN 0387848576.

[27] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. CRC Press, FL, 2015.

[28] R. Hemmecke, M. Köppe, J. Lee, and R. Weismantel. Nonlinear integer programming. In *50 Years of Integer Programming 1958-2008*, pages 561–618. Springer, 2010.

[29] G. M. James and P. Radchenko. A generalized Dantzig selector with shrinkage tuning. *Biometrika*, 96:323–337, 2009.

[30] G. M. James, P. Radchenko, and J. Lv. Dasso: connections between the Dantzig selector and lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(1):127–142, 2009.

[31] M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, and L. A. Wolsey. *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-art*. Springer Science & Business Media, 2009.

[32] J. T. Linderoth and A. Lodi. MILP software. *Wiley encyclopedia of operations research and management science*, 2010.

[33] H. Liu, T. Yao, and R. Li. Global solutions to folded concave penalized nonconvex learning. *Annals of Statistics*, 44 (2):629–659, 2016.

[34] Z. Lu and Y. Zhang. Sparse approximation via penalty decomposition methods. *SIAM Journal on Optimization*, 23(4):2448–2478, 2013.

[35] R. Mazumder, J. Friedman, and T. Hastie. Sparsenet: Coordinate descent with non-convex penalties. *Journal of the American Statistical Association*, 117(495):1125–1138, 2011.

[36] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140 (1):125–161, 2013.

[37] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer, Norwell, 2004.

[38] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):123–231, 2013.

[39] R. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, 1996.

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

[41] J. P. Vielma. Mixed integer linear programming formulation techniques. *SIAM Review*, 57(1):3–57, 2015.

[42] H. P. Williams. *Model building in mathematical programming*. John Wiley & Sons, 2013.

[43] C.-H. Zhang and J. Huang. The sparsity and bias of the lasso selection in high-dimensional linear regression. *Annals of Statistics*, 36(4):1567–1594, 2008.