Principal Component Hierarchy for Sparse Quadratic Programs

Robbie Vreugdenhil [1] **Viet Anh Nguyen** [2 3] **Armin Eftekhari** [4] **Peyman Mohajerin Esfahani** [1]

Abstract

We propose a novel approximation hierarchy for cardinality-constrained, convex quadratic programs that exploits the rank-dominating eigenvectors of the quadratic matrix. Each level of approximation admits a min-max characterization whose objective function can be optimized over the binary variables analytically, while preserving convexity in the continuous variables. Exploiting this property, we propose two scalable optimization algorithms, coined as the "*best response*" and the "*dual program*", that can efficiently screen the potential indices of the nonzero elements of the original program. We show that the proposed methods are competitive with the existing screening methods in the current sparse regression literature, and it is particularly fast on instances with high number of measurements in experiments with both synthetic and real datasets.

1. Introduction

Sparsity is a powerful inductive bias that improves the interpretability and performance of many regression models (Ribeiro et al., 2016; Hastie et al., 2009; Bertsimas et al., 2017). Recent years have witnessed a growing interest in sparsity-based methods and algorithms for sparse recovery, mostly in the setting of sparse linear regression (Atamturk & Gomez, 2020; Bertsimas & van Parys, 2017; Hazimeh et al., 2020; Hastie et al., 2017).

In this paper we study the more general problem of sparse linearly-constrained quadratic programming with a regularization term. Sparsity is imposed in this context by controlling the ℓ_0 norm of the estimator (Miller, 2002). More specifically, we consider the problem

$$\begin{array}{lll} J^\star \triangleq & \min & \langle c, x\rangle + \langle x, Qx\rangle + \eta^{-1}\|x\|_2^2 \\ & \text{s.t.} & x \in \mathbb{R}^n \\ & & Ax \leq b, \quad \|x\|_0 \leq s, \end{array} \tag{$\mathcal{P}$}$$

where $Q \in \mathbb{S}_+^n$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and the integer $s \leq n$ specifies the target sparsity level. The ridge regularization term $\eta^{-1}\|x\|_2^2$ (Hoerl & Kennard, 1970) in the objective function reduces the mean squared error when the data is affected by noise and/or uncertainty (Ghaoui & Lebret, 1997; Mazumder et al., 2020).

Problem ($\mathcal{P}$) arises in a wide range of applications, including sparse linear regression, model predictive control (Aguilera et al., 2014), portfolio optimization (Bertsimas & Cory-Wright, 2020), binary quadratic programming and principal component analysis (Bertsimas et al., 2020b).

Motivation for our approach. Throughout, suppose that Q in problem ($\mathcal{P}$) admits the eigendecomposition $Q = \sum_{i=1}^n \lambda_i v_i v_i^\top$, where $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$ are the eigenvalues of Q. Our approach to solve ($\mathcal{P}$) is centered around the following key observation:

> **Observation.** For many real-world applications, the matrix Q is *nearly low-rank*.

The concept of *nearly low-rank* is contextual. In this paper, we say that Q is nearly low-rank if a low-rank matrix can approximate Q to a reasonable accuracy, where the accuracy is measured by a matrix norm.

It is instructive to demonstrate the nearly low-rank property of Q in the context of linear regression. Given N training samples $\{(\xi_i, \omega_i)\}_{i=1}^N \subset \mathbb{R}^n \times \mathbb{R}$, the sparse ridge regression problem

$$\min_{x:\|x\|_0\leq s} \quad \frac{1}{N}\sum_{i=1}^N \|\xi_i - x^\top \omega_i\|_2^2 + \eta^{-1}\|x\|_2^2 \tag{1}$$

coincides with ($\mathcal{P}$) for the choice of

$$Q = \frac{1}{N}\sum_{i=1}^N \omega_i\omega_i^\top, \quad c = \frac{1}{N}\sum_{i=1}^N \xi_i\omega_i. \tag{2}$$

[1]Delft Center for Systems and Control, Delft University of Technology [2]Department of Management Science and Engineering, Stanford University [3]VinAI Research, Vietnam [4]Department of Mathematics and Mathematical Statistics, Umea University. Correspondence to: Robbie Vreugdenhil <robbievreugdenhil@gmail.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

In high-dimensional regression ($N < n$), the matrix Q is automatically rank-deficient. Moreover, Q can also be nearly low-rank even when $N > n$. As a numerical example, for the UCI Superconductivity dataset[1], we randomly select 70% of the dataset as training samples, and then compute the matrix Q, as specified in (2). Figure 1 plots the empirical distribution of the largest eigenvalues of Q, taken over 100 independent replications. On average, the ratio λ_1/λ_{10} between the 1st largest and the 10th largest eigenvalues of Q is 97.2. We can also observe that the magnitude of the eigenvalues decays relatively fast for this dataset.

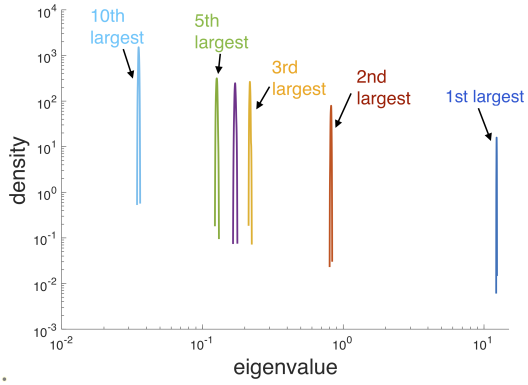


Figure 1. Empirical distribution of the eigenvalues of the matrix Q for the Superconductivity dataset (plotted in log-log scale).

Next, let us recall the low-rank truncation of the matrix Q. For $k \leq n$, the approximation of Q using its k leading eigenvectors, denoted by Q_k, is given by

$$Q_k \triangleq \sum_{i=1}^{k} \lambda_i v_i v_i^\top.$$

Following the same sampling procedure described above, Figure 2 depicts how the average truncation error, measured by the Frobenius norm $\|Q_k - Q\|_F$, rapidly decreases as k increases. In this figure, we also report the minimum dimension $\hat{k}$ so that the reconstruction error of $Q_{\hat{k}}$ falls below 10% that of the rank-1 approximation Q_1. We observe that $\hat{k} \ll n$ for many UCI regression datasets. This nearly low-rank behavior is typical in data sciences (Udell & Townsend, 2019).

Contributions. We summarize the contributions as follows.

(i) **Hierarchy of approximations and min-max characterization:** We exploit the nearly low-rank nature of the matrix Q and propose a hierarchy of approximations for cardinality-constrained convex quadratic programs. This hierarchy enables us to strike a balance between the scalability of the solver and the quality of the solution. We further show that each k-leading spectral truncation can be characterized as a min-max problem (Proposition 3.1).

(ii) **Scalable and certifiable algorithms:** We propose two scalable algorithms that enjoy optimality certificates if the min-max characterization admits a saddle point (Propositions 3.3 and 3.5). The proposed algorithms build on a desirable feature of the objective function of the min-max characterization, in which the minimizer over the binary variables admits a closed-form solution (Lemma 3.2).

(iii) **Safe screening:** If the min-max characterizations do not admit a saddle point, the proposed iterative algorithms can serve as a screening method to reduce the variables in the original problem. We investigate the effectiveness of our algorithms through an in-depth numerical comparison with the recent sparse regression literature including the safe screening procedures of Atamturk & Gomez (2020) and the warm start of Bertsimas & van Parys (2017). Moreover, we also benchmark against direct optimization methods Beck & Eldar (2012) and Yuan et al. (2020). Experiments on both synthetic and real datasets reveal that our algorithms deliver promising results (Section 4).

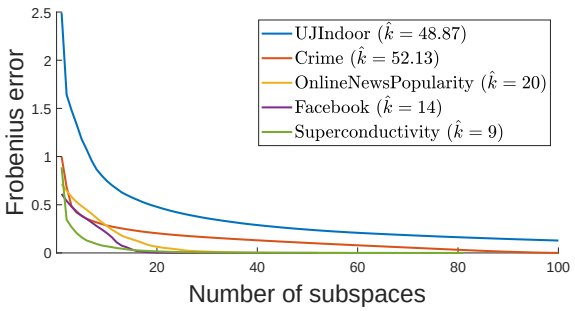


Figure 2. Empirical Frobenius reconstruction error $\|Q_k - Q\|_F$ averaged over 100 independent replications.

We note that using leading principal components in regression has a long-standing history (Næs & Martens, 1988; Hastie et al., 2009; Baye & Parker, 1984). However, to the best of our knowledge, it has never been applied in the context of cardinality-constrained convex quadratic problems. Compared to the existing solution procedures in the literature, the performance of our methods, measured by both the objective value and the screening capacity, is consistent across a wider range of input parameters. Moreover, in sparse regression, our methods can scale to problems of

[1]Available at https://archive.ics.uci.edu/ml/datasets/Superconductivty+Data

large sample size N because the complexity of our methods does not depend on N poorly.

This paper unfolds as follows. Section 2 provides a brief overview over the landscape of the cardinality-constrained quadratic problems. Section 3 devises two distinctive algorithms that leverage the principal component approximation of the matrix Q. Section 4 reports an in-depth numerical comparison between our algorithms and the current sparse regression literature.

Notations. For any matrix u, we use $\sqrt{u}$ and $|u|$ to denote the component-wise square root and absolute value of u, respectively. For a vector u, we use $\mathrm{diag}(u)$ to denote the diagonal matrix formed by u. For an integer n, we also denote with $[n]$ the set of integers $\{1, \ldots, n\}$. Given an index set $\mathbb{A} \subset [n]$, the binary vector $u = \mathbb{1}_n(\mathbb{A}) \in \{0,1\}^n$ is defined as $u_j = 1$ if and only if $j \in \mathbb{A}$. In words, $\mathbb{1}_n(\mathbb{A})$ is the indicator vector of the index set $\mathbb{A}$.

2. Literature Review

Sparse regression. In general, the sparse regression problem is NP-hard (Natarajan, 1995). Until recently the sparse optimization literature largely focused on convex heuristics for sparse regression, e.g., Lasso (ℓ_1) (Tibshirani, 1996) and Elastic Net ($\ell_1 - \ell_2$) (Zou & Hastie, 2005). Despite their scalability, convex heuristic approaches are inherently biased since the ℓ_1-norm penalizes large and small coefficients uniformly. In contrast, solvers that directly tackle sparse regression do not suffer from unwanted shrinkage and have enjoyed a resurgence of interest (Bertsimas & van Parys, 2017; Hazimeh et al., 2020; Dedieu et al., 2020; Gomez & Prokopyev, 2018), thanks to the recent breakthroughs in mixed-integer programming (Bertsimas et al., 2016). These direct approaches are also the focus of this paper.

In particular, Bertsimas & van Parys (2017) devise a convex reformulation of the sparse regression problem using duality theory, the solution of which provides a warm start for a branch-and-cut algorithm. This method can solve sparse regression problems at the scale of $n \approx 10^5$ while the earlier work (Bertsimas et al., 2016) only goes to sizes of $n \approx 10^3$. However, as pointed out by Xie & Deng (2020), the performance of these algorithms depends critically on the speed of the commercial solvers and varies significantly from one dataset to another. Therefore we focus on the warm start method in the approach of Bertsimas & van Parys (2017), which makes use of the kernel matrix $[\omega_i^\top \omega_j]_{i,j}$. Notice that the size of this kernel matrix scales with the number of samples. On the contrary, our proposed solution procedures use only the resulting matrix Q, whose size does not depend on the number of samples.

Convex approximations of the sparse regression can however be used as a safe screening procedure as demonstrated by Atamturk & Gomez (2020). Safe screening methods aim to identify the support of the solution set of ($\mathcal{P}$) and reduce the problem dimension n before invoking an MIQP solver. Indeed, if we correctly rule out any single suboptimal dimension, the solution space would be cut by half. In this sense, the expected speedup for the MIQP solver is exponential (Atamturk & Gomez, 2020).

Sparse quadratic programs. We are only aware of a few papers that solve sparse quadratic programs exactly. Beck & Eldar (2012) and Beck & Hallak (2015) devise coordinate descent type algorithms based on the concept of coordinate-wise optimality, which updates the support at each iteration and, in particular, can also be used to solve sparse quadratic programs. Yuan et al. (2020) considers a block decomposition algorithm that combines combinatorial search and coordinate descent. Specifically, this method uses a random or greedy strategy to find the working set and then performs a global combinatorial search over the working set based on the original objective function. Recently, Bertsimas & Cory-Wright (2020) and Bertsimas et al. (2020b) apply the advances in exact sparse regression to sparse quadratic programs to solve problems of higher dimension. Both essentially rewrite ($\mathcal{P}$) as a sparse regression problem and then solve it with a modified version of the branch-and-cut algorithm of Bertsimas & van Parys (2017). This procedure can also accommodate linear constraints.

Given the existing literature mentioned above, to our best of knowledge, our approach is the first to identify and exploit the low-rank structure of Q in the original problem ($\mathcal{P}$) by using the leading principal component approximation of Q, notably in the context of sparse quadratic programming.

3. Principal Component Approximation

The key idea of this study is to leverage the principal component approximation of the matrix Q in ($\mathcal{P}$) in order to deploy the duality technique from convex optimization in a more efficient manner. To this end, we introduce additional continuous variables y along with the equality constraints $\sqrt{\lambda_i} y_i = \sqrt{\lambda_i} \langle v_i, x \rangle$ where λ_i and v_i are, respectively, the eigenvalues and eigenvectors of the matrix Q. Throughout, we denote $V = [v_1, \ldots, v_k] \in \mathbb{R}^{d \times k}$. Using these definitions, program ($\mathcal{P}$) can be approximated via

$$
\begin{array}{rll}
J_k^\star \triangleq & \min & \langle c, x \rangle + \sum_{i=1}^k \lambda_i y_i^2 + \eta^{-1} \|x\|_2^2 \\
& \text{s.t.} & x \in \mathbb{R}^n,\ y \in \mathbb{R}^k \\
& & Ax \leq b, \quad \|x\|_0 \leq s \\
& & \sqrt{\lambda_i} y_i = \sqrt{\lambda_i} \langle v_i, x \rangle, \quad i \in [k].
\end{array}
\qquad (\mathcal{P}_k)
$$

The last equality constraints of ($\mathcal{P}_k$) are scaled using the coefficients $\sqrt{\lambda_i}$ to improve numerical stability. Since the matrix Q is positive semidefinite, we have the hierarchy of

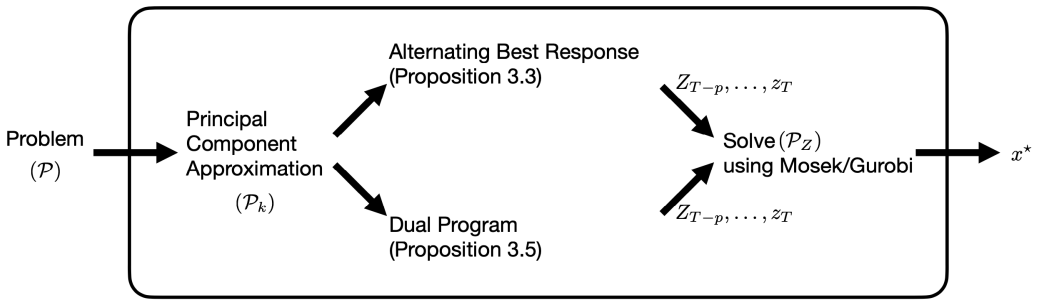


Figure 3. Schematic overview of the principal component approximation to sparsity-constrained quadratic programs.

approximations in which the sequence of optimal values $J_k^\star$ preserves the order

$$J_1^\star \leq J_2^\star \leq \cdots \leq J_n^\star = J^\star,$$

where $J^\star$ is the optimal value of program $(\mathcal{P})$. In fact, one can observe that the two programs $(\mathcal{P})$ and $(\mathcal{P}_k)$ are equivalent when $k = n$. Next, we use the standard convex duality to turn program $(\mathcal{P}_k)$ into a min-max optimization problem.

Proposition 3.1 (Min-max characterization)**.** *For each $k \leq n$, the optimal value $J_k^\star$ of $(\mathcal{P}_k)$ is equal to*

$$J_k^\star = \min_{\substack{z\in\{0,1\}^n \\ \sum z_j \leq s}} \max_{\substack{\alpha\in\mathbb{R}^k \\ \beta\in\mathbb{R}^m_+}} L(z,\alpha,\beta), \qquad (\mathcal{M}_k)$$

where the objective function L is defined as

$$L(z,\alpha,\beta) \triangleq -\beta^\top b - \frac{1}{4}\|\alpha\|_2^2 \qquad (3)$$
$$-\frac{\eta}{4}(c + V\sqrt{\Lambda}\alpha + A^\top\beta)^\top \mathrm{diag}(z)(c + V\sqrt{\Lambda}\alpha + A^\top\beta),$$

in which $\Lambda = \mathrm{diag}\{\lambda_1, \cdots, \lambda_k\}$ is a diagonal matrix whose elements on the main diagonal are the first k largest eigenvalues of the matrix Q. Moreover, the nonzero coordinates of the optimal variable $x^\star$ in $(\mathcal{P}_k)$ contain the nonzero elements of the optimal solution $z^\star$ in $(\mathcal{M}_k)$.

Thanks to Proposition 3.1, the information regarding the support of the optimal solution $x^\star$ in $(\mathcal{P}_k)$ can effectively be obtained from the support of the optimal solution $z^\star$ in $(\mathcal{M}_k)$. We note that finding the support is indeed the computational bottleneck of program $(\mathcal{P}_k)$. A key feature of the objective function L of program $(\mathcal{M}_k)$ is that when the variables (α, β) is fixed, the minimization over the binary variable z can be solved analytically.

Lemma 3.2 (Closed-form minimizer)**.** *Given any pair (α, β), the minimizer of the function L defined in (3) can be computed as*

$$\arg\min_{\substack{z\in\{0,1\}^n \\ \sum z_j \leq s}} L(z,\alpha,\beta) = \mathbb{1}_n\big(\mathcal{J}(\alpha,\beta)\big), \quad \textit{where} \qquad (4)$$

$$\mathcal{J}(\alpha,\beta) \triangleq \left\{ j \in [n] : \begin{array}{c} j \textit{ is an index of the } s\textit{-largest} \\ \textit{elements of the vector} \\ c + V\sqrt{\Lambda}\alpha + A^\top\beta \end{array} \right\},$$

and $\mathbb{1}_n(\mathcal{J}) \in \{0,1\}^n$ is a binary vector whose coordinates contained in the set $\mathcal{J}$ are ones.

We note that the set of optimal indices defined in $\mathcal{J}(\alpha,\beta)$ may not be unique. In such scenarios, we adopt a deterministic tiebreaker (e.g., a lexicographic rule) to introduce $\mathcal{J}$ as a proper single-valued function. The observation in (4) is the key building block for two scalable optimization algorithms that we will propose to tackle problem $(\mathcal{P}_k)$.

3.1. Alternating best response

The first proposed algorithm can be cast as an attempt to find a saddle point (a Nash equilibrium) of program $(\mathcal{M}_k)$, if it exists. This is similar to the approach discussed in Bertsimas et al. (2020a, Theorem 3.2.3), which concerns the different problem of sparse PCA. Given the inherent nonconvexity of the problem due to the binary variables z, such an equilibrium may not exist. In that case, the algorithm will not converge. Nonetheless, we will also discuss how this approach can still be viewed as a "safe screening" scheme (Atamturk & Gomez, 2020).

Given $z \in \{0,1\}^n$, we define the function

$$\mathrm{BR}(z) \triangleq \arg\max_{\substack{\alpha\in\mathbb{R}^k \\ \beta\in\mathbb{R}^m_+}} L(z,\alpha,\beta). \qquad (5)$$

If the function L in (5) does not have a unique optimizer, in a similar fashion as as $\mathcal{J}$ defined in Lemma 3.2, we deploy a deterministic tiebreaker to properly introduce a single-valued function BR. The function BR is the maximizer of the loss function for a fixed value of z, to which we refer as the "best response". While the objective function L is a jointly concave quadratic function in the variables (α, β),

the description of the optimizer does not necessarily have an explicit description due to the constraint $\beta \geq 0$. However, in the absence of the constraint $Ax \leq b$ in ($\mathcal{P}_k$) (e.g., $A = 0, b = 0$), the function (5) can be explicitly described as

$$\mathrm{BR}(z) = -(I_k/\eta + \sqrt{\Lambda}V^\top \mathrm{diag}(z) V \sqrt{\Lambda})^{-1} \times \sqrt{\Lambda} V^\top \mathrm{diag}(z) c. \tag{6}$$

We note that we slightly abuse the notation as the explicit description (6) is indeed the fist element (α-component) of the original definition (5).

Proposition 3.3 (Alternating best response)**.** *Consider the set of update rules*

$$\begin{cases} \begin{bmatrix} \alpha_{t+1} \\ \beta_{t+1} \end{bmatrix} & = \mathrm{BR}(z_t) \\ z_{t+1} & = \mathbb{1}_n\big(\mathcal{J}(\alpha_t, \beta_t)\big), \end{cases} \tag{7}$$

where the set $\mathcal{J}$ and the function BR *are defined as in Lemma 3.2 and (5), respectively. Starting from an initialization (z_0, α_0, β_0), algorithm (7) converges after finitely many iterations to a limit cycle. If the set of this period behavior is singleton (i.e., the iterations convergence to a fixed point), then the variable z of the convergence point is the optimal solution of program ($\mathcal{M}_k$).*

Proof. Recall that for any $z \in \{0,1\}^n$ the objective function L in (5) is strongly concave, and that admits a unique maximizer. On the other hand, the number of possible binary variable z is finite and bounded by 2^n. These two observations together then imply that the iterations (7) necessarily yield a period behavior with the cardinality at most 2^n. When the period is one, it then means that the best response algorithm has an equilibrium, implying that the min-max program ($\mathcal{M}_k$) is indeed a minimax game with a Nash equilibrium. Namely, there exist $(\alpha^\star, \beta^\star) \in \mathbb{R}^k \times \mathbb{R}^m_+$ and $z^\star \in \{0,1\}^n$ such that for all $(\alpha, \beta) \in \mathbb{R}^k \times \mathbb{R}^m_+$, and $z \in \{0,1\}^n$, $\sum z_j \leq s$, we have

$$L(z^\star, \alpha, \beta) \leq L(z^\star, \alpha^\star, \beta^\star) \leq L(z, \alpha^\star, \beta^\star)$$

and, by definition, $z^\star$ solves the outer minimization of program ($\mathcal{M}_k$). □

In the iterative scheme (7), evaluating the best response function $\mathrm{BR}(z_t)$ is equivalent to solving a linearly constrained convex quadratic program, which can be done efficiently using commercial solvers such as MOSEK (MOSEK ApS, 2019). In case we have no linear constraints in the form of $Ax \leq b$, we can also use the explicit description (6). Therefore, the algorithm (7) is indeed highly tractable.

Remark 3.4 (Safe screening)**.** *We expect that the periodic behavior anticipated by Proposition 3.3 typically has a periodicity larger than one. In fact, if the min-max characterization in Proposition 3.1 is not interchangeable without suffering from a duality gap, then the periodic behavior does have more than one element. In this setting, one can consider all the indices $\mathcal{J}(\alpha_t, \beta_t)$, where (α_t, β_t) belongs to the period behavior, as potential candidates for the ones of the optimal vector $z^\star$ in ($\mathcal{M}_k$). This selection is indeed in accordance with the safe screening terminology of Atamturk & Gomez (2020).*

3.2. Dual program: a subgradient ascent approach

The second proposed algorithm aims to solve the dual of program ($\mathcal{M}_k$) described via

$$d_k^\star \triangleq \max_{\substack{\alpha \in \mathbb{R}^k \\ \beta \in \mathbb{R}^m_+}} \min_{\substack{z \in \{0,1\}^n \\ \sum z_j \leq s}} L(z, \alpha, \beta), \tag{$\mathcal{D}_k$}$$

where the function L was defined in (3). Thanks to the weak duality, it is obvious that $d_k^\star \leq J_k^\star$. The second approach is essentially the application of the subgradient ascent algorithm to the inner minimal function

$$f(\alpha, \beta) \triangleq \min_{z \in \{0,1\}^n, \sum z_j \leq s} L(z, \alpha, \beta). \tag{8}$$

We note that the continuous relaxation of the binary variable z from $\{0,1\}^n$ to $[0,1]^n$ in (8) does not change anything and the program remains equivalent to the original program ($\mathcal{D}_k$). Also notice that the function f in (8) is concave and piecewise quadratic, jointly in (α, β). This observation allows us to apply the classical subgradient algorithm from the convex optimization literature (Nesterov, 2003, Section 3.2.3).

Proposition 3.5 (Dual program)**.** *Consider the set of update rules defined as*

$$\begin{cases} \alpha_{t+1} = (1 - \frac{1}{2}\kappa_t)\alpha_t - \frac{1}{2}\eta\kappa_t \sqrt{\Lambda} V^\top \mathrm{diag}(z_t) \times \\ \qquad\qquad (c + V\sqrt{\Lambda}\alpha + A^\top \beta) \\ \beta_{t+1} = \max\Big\{0, \beta_t - \kappa_t b - \frac{1}{2}\eta\kappa_t A \mathrm{diag}(z_t) \times \\ \qquad\qquad (c + V\sqrt{\Lambda}\alpha + A^\top \beta)\Big\} \\ z_{t+1} = \mathbb{1}_n\big(\mathcal{J}(\alpha_{t+1}, \beta_{t+1})\big), \end{cases} \tag{9}$$

where $\{\kappa_t\}_t$ is the sequence of step sizes that satisfy the non-summable diminishing rule[2]

$$\lim_{t \to \infty} \kappa_t = 0, \quad \sum_{t=1}^{\infty} \kappa_t = \infty.$$

Then, algorithm (9) converges to the optimal value d_k^* of problem ($\mathcal{D}_k$), i.e.,

$$d_k^\star = \lim_{t \to \infty} f(\alpha_t, \beta_t).$$

Moreover, if the variable z_t also converges, then the convergent binary variable is the solution to program ($\mathcal{M}_k$)

[2] For instance, $\kappa_t = a/\sqrt{t}$ for a constant $a \in \mathbb{R}_+$.

and the duality gap between $(\mathcal{M}_k)$ *and* $(\mathcal{D}_k)$ *is zero, i.e.,* $d_k^\star = J_k^\star$.

Proof. Using standard results in variational analysis (Rockafellar & Wets, 2009), a subgradient of the function $f(\alpha, \beta)$ can be computed as

$$\frac{\partial f}{\partial \alpha} = -\frac{1}{2}\alpha - \frac{\eta}{2}\mathrm{diag}(\sqrt{\lambda})V^\top \mathrm{diag}(z_{\alpha,\beta}) \times (c + V\mathrm{diag}(\sqrt{\lambda})\alpha + A^\top\beta),$$
$$\frac{\partial f}{\partial \beta} = -b - \frac{\eta}{2}A\mathrm{diag}(z_{\alpha,\beta})(c + V\mathrm{diag}(\sqrt{\lambda})\alpha + A^\top\beta),$$

where $z_{\alpha,\beta} = \mathbb{1}_n\big(\mathcal{J}(\alpha,\beta)\big)$ is the optimizer of the objective function $L(., \alpha, \beta)$; see also (4). The subgradient algorithm updates the dual variables α_t, β_t by the following rule

$$\begin{bmatrix}\alpha_{t+1}\\ \beta_{t+1}\end{bmatrix} = \begin{bmatrix}\alpha_t\\ \beta_t\end{bmatrix} + \kappa_t \begin{bmatrix}\frac{\partial}{\partial\alpha} f(\alpha_t, \beta_t)\\ \frac{\partial}{\partial\beta} f(\alpha_t, \beta_t)\end{bmatrix},$$

where κ_t is the learning rate or stepsize. The computational complexity of the subgradient algorithm is well-known for concave and Lipschitz-continuous objective functions (Boyd et al., 2003; Nesterov, 2003). In the remainder of the proof, we first verify that the classical results are applicable here too. To continue, let $\lambda = (\alpha, \beta)$ for short. Note that $z \in \{0,1\}^n$ which bounds the number of quadratic regions of f to 2^n which, in turn, implies that any level set of f is bounded. Since f is also concave and thus continuous, we conclude that f is in fact Lipschitz-continuous in any fixed level set of f. At iteration t, say $f(\lambda_t) = f_{z_t}(\lambda_t) := L(z_t, \alpha_t, \beta_t)$. This also means $g_t = \nabla f_{z_t}(\lambda_t)$ is a subgradient of f at λ_t. If t is large enough, then step size κ_t is small enough, and therefore the algorithm update $\lambda_{t+1} = \lambda_t + \kappa_t g_t$ increases the value of f_{z_t}. Hence, $f(\lambda_t) = f_{z_t}(\lambda_t) < f_{z_t}(\lambda_{t+1}) \leq f(\lambda_{t+1})$. Therefore, the subgradient algorithm remains afterwards within a level set of f, specified by $\{\lambda : f(\lambda) \geq f(\lambda_t)\}$ when t is sufficiently large. Now recall that f is Lipschitz-continuous in any fixed level set of f. So, we can apply the result of Boyd et al. (2003) for all sufficiently large $t \in \mathbb{N}$.

Concerning the second part of the assertion, suppose that the variable z_t also converges to a binary variable $z^\star$. Since the feasible set of the variable z is finite, the convergence assumption effectively implies that for all sufficiently large $t \in \mathbb{N}$, we have constant $z_t = z^\star$ in (9). As such, the dual algorithm (9) essentially reduces to

$$\begin{bmatrix}\alpha_{t+1}\\ \beta_{t+1}\end{bmatrix} = \begin{bmatrix}\alpha_t\\ \beta_t\end{bmatrix} + \kappa_t \begin{bmatrix}\frac{\partial}{\partial\alpha} L(z^\star, \alpha_t, \beta_t)\\ \frac{\partial}{\partial\beta} L(z^\star, \alpha_t, \beta_t)\end{bmatrix}.$$

The above argument allows us to also interpret the algorithm (9) as the subgradient ascent algorithm for the quadratic concave mapping $(\alpha, \beta) \mapsto L(z^\star, \alpha, \beta)$. This observation yields

$$\lim_{t\to\infty} L(z^\star, \alpha_t, \beta_t) = \max_{\alpha\in\mathbb{R}^k,\, \beta\in\mathbb{R}^m_+} L(z^\star, \alpha, \beta).$$

Thanks to the above equality, one can inspect that

$$J_k^\star \geq d_k^\star = \lim_{t\to\infty} L(z^\star, \alpha_t, \beta_t) = \max_{\substack{\alpha\in\mathbb{R}^k\\ \beta\in\mathbb{R}^m_+}} L(z^\star, \alpha, \beta) \geq J_k^\star,$$

where the first inequality is due to the weak duality between programs $(\mathcal{M}_k)$ and $(\mathcal{D}_k)$, and the last inequality follows from the definition of the optimal value of $(\mathcal{M}_k)$. Since both sides of the above inequalities are $J_k^\star$, all the middle terms coincide. Thus, it concludes that $z^\star$ solves program $(\mathcal{M}_k)$ and the zero duality gap holds, i.e., $J_k^\star = d_k^\star$. □

Similar to the best response algorithm in Proposition 3.3, we expect that in the long-run the duality algorithm (9) exhibits a period behavior over a number of $z_t \in \{0,1\}^n$. In this light, one can also consider the coordinates of ones elements of z_t as a safe screening suggestion (cf. Remark 3.4).

Remark 3.6 (Computational complexity)**.** *Formulating* $(\mathcal{P}_k)$ *requires a PCA decomposition with a (crude) time complexity of* $\mathcal{O}(n^3)$*, and sorting for operation* (4) *takes* $\mathcal{O}(n \log n)$*. In addition, the best response algorithm in* (7) *has a complexity of* $\mathcal{O}(k^3 + nk)$*. Thus, overall the best response method has a complexity of* $\mathcal{O}(n^3 + t(k^3 + nk + n\log n))$*, where* t *are the number of iterations. The dual program algorithm* (9) *requires algebraic operations with complexity* $\mathcal{O}(nk)$*, sand that the overall complexity of the dual program is* $\mathcal{O}(n^3 + t(nk + n\log n))$.

3.3. Post-processing

Suppose, without any loss of generality, that either the best response algorithm (7) or the dual program algorithm (9) terminates after T iterations. By collecting the incumbent solutions $z_{T-p}, \ldots, z_T$ in the variable z over the last p iterations, we can form the unique indices

$$Z = z_{T-p} \mid z_{T-p+1} \mid \cdots \mid z_T \in \{0,1\}^n,$$

where $\mid$ represents the componentwise OR operator. Intuitively, the binary value of Z_i indicates if at least one of the last p incumbent solutions has the i-th element being non-zero. The vector Z thus represents the indices of x that are likely to be non-zero in the optimal solution of problem $(\mathcal{P})$. We now utilize the binary vector Z as an input to resolve the reduced problem

$$\begin{array}{ll}\min & \langle c, x\rangle + \langle x, Qx\rangle + \frac{1}{\eta}\|x\|_2^2\\ \text{s.t.} & x \in \mathbb{R}^n,\\ & Ax \leq b,\ \|x\|_0 \leq s,\ |x| \leq MZ,\end{array} \qquad (\mathcal{P}_Z)$$

where M is the big-M constant. If $Z_i = 0$, then the last constraint of ($\mathcal{P}_Z$) implies that $x_i = 0$ and a preprocessing step can remove this redundant component in x. As a consequence, the effective dimension of the variable x in ($\mathcal{P}_Z$) is upper bounded by the number of non-zero elements in Z, which is essentially $\|Z\|_0$. It is likely that Z has many elements that are 0, thus $\|Z\|_0 \ll n$ and problem ($\mathcal{P}_Z$) is easier to solve compared to ($\mathcal{P}$). In general, $\|Z\|_0 > s$, so ($\mathcal{P}_Z$) remains a binary quadratic optimization problem. In the optimistic case when $\|Z\|_0 = s$, then the cardinality constraint $\|x\|_0 \leq s$ becomes redundant and ($\mathcal{P}_Z$) reduces to a quadratic program.

In practice, the magnitude of M may affect the run time, and a tight value of M can significantly improve the numerical stability and reduce the solution time. We follow the suggestion from Hazimeh et al. (2020) to compute M as follows. Given the terminal solution z_T from either the best response algorithm (7) or the dual program algorithm (9), we solve problem ($\mathcal{P}_Z$) with the input Z being replaced by z_T to get x_T. As z_T satisfies $\|z_T\|_0 = s$, this problem reduces to a quadratic program. To ensure that the big-M formulation adds no binding constraints we assign $M = 4\|x_T\|_\infty$.

Lastly, we emphasize that the solution z_t in the subgradient ascent algorithm (9) does not fluctuate significantly from one iteration to another. Thus, for the dual program approach, we need to set a periodic value p which is sufficiently large in order to recuperate meaningful signals on the indices. The best response method using the update (7), on the contrary, requires a smaller number of period p.

4. Numerical Experiments

We benchmark different approaches to solve problem ($\mathcal{P}$) in the sparse linear regression setting. All experiments are run on a laptop with Intel(R) Core(TM) i7-8750 CPU and 16GB RAM using MATLAB 2020b. The optimization problems are modeled using YALMIP (Löfberg, 2004), as the interface for the mixed-integer solver (MOSEK ApS, 2019). Codes are available at: `https://github.com/RVreugdenhil/sparseQP`

We compare our algorithms against four state-of-the-art approaches. They include two screening methods: the safe `screening` method of Atamturk & Gomez (2020) and the `warm start` method in Bertsimas & van Parys (2017), and two direct optimization approaches: the Algorithm 7 in Beck & Hallak (2015) (denoted `BH Alg 7`) and the method of Yuan et al. (2020) (denoted `KDD`).[3] The best response alternation in Section 3.1 is referred to as `BR`, and the dual program approach in Section 3.2 is referred to as `DP`.

[3] Available at `https://yuangzh.github.io/`

4.1. Empirical results for synthetic data

We generate the covariate $\omega \in \mathbb{R}^n$ and the univariate response $\xi \in \mathbb{R}$ using the linear model

$$\xi = x_{\text{true}}^\top \omega + \epsilon$$

following the similar setup in Bertsimas & van Parys (2017). The unobserved true vector $x_{\text{true}} \in \mathbb{R}^n$ has s-nonzero components at indices selected uniformly at random, without replacement. The nonzero components in x_{true} are selected uniformly at random from $\{\pm 1\}$. Moreover, the covariate ω is independently generated from a Gaussian distribution $\mathcal{N}(0, \Sigma)$, where Σ is parametrized by the correlation coefficient ρ as $\Sigma_{i,j} \triangleq \rho^{|i-j|}$ for all $i, j \in [n]$ and $0 \leq \rho \leq 1$. The noise ϵ is independently generated from a normal distribution $\mathcal{N}(0, \sigma^2)$ with

$$\sigma^2 = \frac{\text{var}(x_{\text{true}}^\top \omega)}{\text{SNR}} = \frac{x_{\text{true}}^\top \Sigma x_{\text{true}}}{\text{SNR}},$$

where SNR is a chosen signal-to-noise ratio (Xie & Deng, 2020).

To avoid a complicated terminating criterion, we run the `BR` method for $T_{\texttt{BR}} = 20$ iterations, and we run the `DP` method for $T_{\texttt{DP}} = 500$ iterations. We empirically observe that `BR` converges by $T_{\texttt{BR}} = 20$ on the synthetic data. The stepsize constant for `DP` (see Footnote 2) is set to $a = 4 \times 10^{-3}$. Regarding the post-processing step in Section 3.3, we fix $p_{\texttt{BR}} = 6$ and $p_{\texttt{DP}} = 50$ as the number of terminating solutions that are used to estimate the support Z. For the experiment on the synthetic data, the big-M constant is set to 4, because $\|x_{\text{true}}\|_\infty = 1$.

Our first experiment studies the impact of the regularization parameter η on the performance of `BR` and `DP` in terms of safe screening. To this end, we fix $N = 1000, n = 1000, s = 10, \rho = 0.5$ and SNR $= 6$ to generate the data.

The screening capacity of `BR` and `DP` is measured by the sparsity of the input parameter Z in problem ($\mathcal{P}_Z$), this reduced size is measured by $\|Z\|_0$. A similar quantity can be computed for `screening`. We choose the dimension of subspace $k = 400$ to ensure that our the principal component approximation generates a good quality solution to the original problem. Table 1 reports the screening capacity and we can observe that `screening` effectively reduces the dimension for small values of η, which is in agreement with the empirical results reported in Atamturk & Gomez (2020). However, `screening` performs less convincingly for $\eta \geq 1$. Our methods `BR` and `DP` perform more consistently over the whole range of η: they can reduce ($\mathcal{P}_Z$) to a quadratic program for 59% and 81% of all instances respectively.

We also study the performance of our methods in a setting with $\rho \geq 0.7$. We measure the quality of the estimator x by

Table 1. Effective problem size measured by $\|Z\|_0$ for varying η, averaged over 25 replications. Lower is better. Values are rounded to nearest integer, asterisks denote that $\|Z\|_0 = s$ on all instances.

	DP $k = 400$	BR $k = 400$	screening
$\eta = 10^2$	69	20	1000
$\eta = 10$	$10^\star$	20	1000
$\eta = 1$	10	10	809
$\eta = 10^{-1}$	$10^\star$	10	463
$\eta = 10^{-2}$	$10^\star$	$10^\star$	45
$\eta = 10^{-3}$	$10^\star$	$10^\star$	10

the mean squared error on the data

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} \|\xi_i - x^\top \omega_i\|_2^2.$$

We fix the regularisation term $\eta = 10$. Setting η to a lower value would cause unwanted shrinkage of the estimator x, which increases the MSE: instances where $\eta = 0.1$ reported a MSE for all methods of at least 5 times larger than that of the MSE using $\eta = 10$. As seen in Table 1 for this specific η the screening method does not reduce the problem size, so we compare to the MSE generated by warm start.

We observe in Table 2 that DP outperforms the other methods in terms of MSE with highly correlative data ($\rho \geq 0.8$). A possible explanation for why DP outperforms the warm start is that the eigenvalue decomposition can convert highly correlative features to independent principal components (Liu et al., 2003).

Table 2. MSE over different ρ averaged over 25 independent replications. Lower is better.

	DP $k = 400$	BR $k = 400$	warm start	BH Alg 7	KDD
$\rho = 0.7$	1.829	1.829	1.829	1.897	1.829
$\rho = 0.8$	1.863	1.969	1.988	2.114	1.866
$\rho = 0.9$	1.895	3.666	3.714	2.570	1.984

4.2. Empirical results for real data

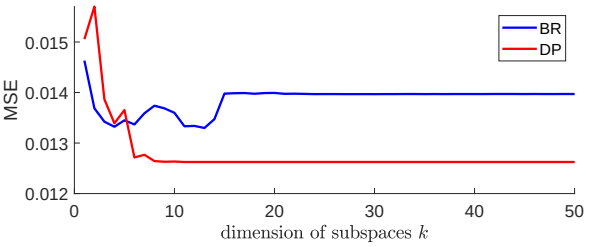
We benchmark different methods using real data sets (Dua & Graff, 2017); the details are listed in Table 3. We preprocess the data by normalizing each covariate and target response independently to values in the range $[0, 1]$. For each independent replication, we randomly sample 70% of the data as training data and 30% as test data. The training set and the test set have N_{train} and N_{test} samples, respectively.

The number of iterations for the BR and DP are chosen to be $T_{\text{BR}} = 40$ and $T_{\text{DP}} = 5000$, respectively. The number of terminating solutions that are used to estimate the support Z in the postprocessing step is fixed to $p_{\text{BR}} = 10$ and $p_{\text{DP}} = 100$ for the two methods. The stepsize constant (see Footnote 2) is set $a = 2 \times 10^{-3}$. Moreover, we fix the big-M constant using the procedure described in Section 3.3.

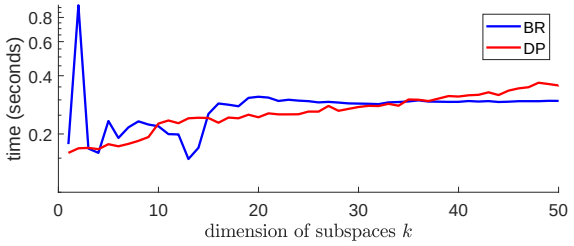
Table 3. List of UCI datasets used for experiments alongside their feature size n, total sample size $\bar{N}$ and the dimension of subspace $\hat{k}$, specified earlier in Figure 2.

Name	n	$\bar{N}$	$\hat{k}$
Facebook (FB)	52	199,030	14
OnlineNews (CR)	58	39,644	20
SuperConductivity (SC)	81	21,263	9
Crime (CR)	100	1,994	53
UJIndoor (UJ)	465	19,937	49

For the real datasets, the number of samples $\bar{N}$ is sufficiently bigger than the dimension n, thus we set the ridge regularization parameter to $\eta = \sqrt{N_{\text{train}}}$ so that the effect of regularization diminishes as $\bar{N}$ increases. We choose the sparsity level $s = 10$, similar to Section 4.1, and set the time limit for MOSEK to 300 seconds.



(a) in-sample Mean Squared Error



(b) computational time

Figure 4. Effects of the dimension of subspaces k on performance of BR and DP on the Superconductivity dataset. Results are averaged over 50 independent train-test splits.

Figure 4 illustrates that the MSE of DP monotonically decreases with the dimension of subspace for $k < 8$ and plateaus for $k \geq 8$. The MSE of BR is non monotonic and converges when $k \geq 15$ even though the minimum is achieved at $k = 13$. We define $k^\star$ as the (minimal) dimension of subspace k corresponding to the lowest MSE. For

Table 4. In-sample MSE on real datasets, averaged over 50 independent train-test splits. Lowest error for each case is highlighted in grey.

	DP $k = 40$	DP $k = \hat{k}$	BR $k = 40$	BR $k = \hat{k}$	warm start	screening	BH Alg 7	KDD
(FB)	3.039×10^{-4}	3.040×10^{-4}	3.036×10^{-4}	$\mathbf{3.034\times10^{-4}}$	out of memory	3.036×10^{-4}	3.220×10^{-4}	3.436×10^{-4}
(ON)	$\mathbf{1.912\times10^{-4}}$	1.913×10^{-4}	1.914×10^{-4}	1.914×10^{-4}	1.914×10^{-4}	1.914×10^{-4}	1.921×10^{-4}	1.922×10^{-4}
(SC)	1.262×10^{-2}	1.263×10^{-2}	1.396×10^{-2}	1.368×10^{-2}	1.326×10^{-2}	$\mathbf{1.256\times10^{-2}}$	1.455×10^{-2}	1.470×10^{-2}
(CR)	2.775×10^{-2}	2.775×10^{-2}	2.807×10^{-2}	2.801×10^{-2}	2.800×10^{-2}	$\mathbf{2.760\times10^{-2}}$	3.072×10^{-2}	3.063×10^{-2}
(UJ)	$\mathbf{2.118\times10^{-2}}$	2.294×10^{-2}	2.673×10^{-2}	2.678×10^{-2}	2.440×10^{-2}	2.267×10^{-2}	3.804×10^{-2}	3.066×10^{-2}

the (SC) dataset, $k^\star_{\texttt{BR}} = 13$ and $k^\star_{\texttt{DP}} = 15$. Coincidentally, we observe that $k^\star$ is close to the value $\hat{k}$ reported in Table 3. This observation also persists empirically for the other datasets.

Table 4 shows that DP delivers a lower in-sample MSE than BR in 4 out of 5 datasets, and DP also has a lower in-sample MSE than the warm start, BH Alg 7 and KDD for all datasets. The warm start method runs out of memory for the (FB) dataset because it requires storing and computing based on a kernel matrix $K = [\omega_i^\top \omega_j]_{i,j}$ of dimension $N \times N$. This is in stark contrast to our proposed approach that computes only a matrix Q of dimension $n \times n$, and then further reduce the computational burden by truncating the SVD of Q. The memory usage of our method is hence not sensitive to the number of samples N.

The screening method outperforms on the (SC) and (CR) dataset, however a careful examination of Table 5 shows that screening does minimal reduction effects for these two datasets. The result of screening in Table 4 on the (SC) and (CR) datasets is essentially the results obtained by applying the MOSEK solver to the original problem (reaching a time limit of 300 seconds).

Table 5 also shows that our DP and BR methods can effectively reduce the number of effective variables. Our methods deliver a solution $x^\star$ in around 1 second for all datasets, including the time spent on computing the eigen-decomposition of Q and the solution time for solving ($\mathcal{P}_Z$) using MOSEK.

Table 5. Reduced problem size over different data rounded average over 50 independent train-test splits. Lower is better.

	DP $k = \hat{k}$	BR $k = \hat{k}$	screening
(FB)	11	20	49
(ON)	15	20	57
(SC)	11	20	77
(CR)	12	20	100
(UJ)	16	20	465

Remark 4.1 (Choice between DP and BR)**.** *We have no theoretical or consistent numerical justification in favor of one of the proposed algorithms DP or BR. However, Algorithm BR in Proposition 3.3 typically converges faster (Fig. A in supplementary) while Algorithm DP offers a better solution (Fig. 4a and Tables 2, 5). We thus suggest DP and BR as complementary approaches to solve the problem.*

5. Acknowledgement

We would like to thank the meta-reviewer and four anonymous reviewers for their constructive comments that helped improve the presentation of this paper. This research is partially supported by the ERC grant TRUST-949796.

References

Aguilera, R., Delgado, R., Dolz, D., and Aguero, J. Quadratic MPC with ℓ_0-input constraint. *IFAC Proceedings Volumes*, 19:10888–10893, 2014.

Atamturk, A. and Gomez, A. Safe screening rules for ℓ_0-regression from perspective relaxations. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 421–430, 2020.

Baye, M. R. and Parker, D. F. Combining ridge and principal component regression:a money demand illustration. *Communications in Statistics - Theory and Methods*, 13 (2):197–205, 1984.

Beck, A. and Eldar, Y. Sparsity constrained nonlinear optimization: Optimality conditions and algorithms. *SIAM Journal on Optimization*, 23:1480–1509, 2012.

Beck, A. and Hallak, N. On the minimization over sparse symmetric sets: Projections, optimality conditions, and algorithms. *Mathematics of Operations Research*, 41: 196–223, 2015.

Bertsimas, D. and Cory-Wright, R. A scalable algorithm for sparse portfolio selection. *arXiv preprint arXiv:1811.00138*, 2020.

Bertsimas, D. and van Parys, B. Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *The Annals of Statistics*, 48:300–323, 2017.

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

Bertsimas, D., Copenhaver, M. S., and Mazumder, R. The trimmed Lasso: Sparsity and robustness. *arXiv preprint arXiv:1409.8033*, 2017.

Bertsimas, D., Cory-Wright, R., and Pauphilet, J. Solving large-scale sparse PCA to certifiable (near) optimality. *arXiv preprint arXiv:2005.05195*, 2020a.

Bertsimas, D., Cory-Wright, R., and Pauphilet, J. A unified approach to mixed-integer optimization: Nonlinear formulations and scalable algorithms. *arXiv preprint arXiv:1907.02109*, 2020b.

Boyd, S., Xiao, L., and Mutapcic, A. Subgradient methods notes, Stanford University. web.stanford.edu/class/ee392o/subgrad_method.pdf, 2003.

Dedieu, A., Hazimeh, H., and Mazumder, R. Learning sparse classifiers: Continuous and mixed integer optimization perspectives. *arXiv preprint arXiv:2001.06471*, 2020.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Ghaoui, L. E. and Lebret, H. Robust solutions to least-squares problems with uncertain data. *SIAM Journal on Matrix Analysis and Applications*, 18(4):1035–1064, 1997.

Gomez, A. and Prokopyev, O. A mixed-integer fractional optimization approach to best subset selection. *Optimization-online*, 2018.

Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer, 2 edition, 2009.

Hastie, T., Tibshirani, R., and Tibshirani, R. J. Extended comparisons of best subset selection, forward stepwise selection, and the lasso. *arXiv preprint arXiv:1707.08692*, 2017.

Hazimeh, H., Mazumder, R., and Saab, A. Sparse regression at scale: Branch-and-bound rooted in first-order optimization. *arXiv preprint arXiv:2004.06152*, 2020.

Hoerl, A. E. and Kennard, R. W. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

Liu, R., Kuang, J., Gong, Q., and Hou, X. Principal component regression analysis with SPSS. *Computer Methods and Programs in Biomedicine*, 71(2):141 – 147, 2003.

Löfberg, J. YALMIP: A toolbox for modeling and optimization in MATLAB. In *IEEE International Conference on Robotics and Automation*, pp. 284–289, 2004.

Mazumder, R., Radchenko, P., and Dedieu, A. Subset selection with shrinkage: Sparse linear modeling when the SNR is low. *arXiv preprint arXiv:1708.03288*, 2020.

Miller, A. *Subset Selection in Regression*. CRC Press, 2002.

MOSEK ApS. *The MOSEK optimization toolbox. Version 9.2.*, 2019. URL https://docs.mosek.com/9.2/cmdtools/index.html.

Natarajan, B. K. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995.

Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2003.

Næs, T. and Martens, H. Principal component regression in NIR analysis: Viewpoints, background details and selection of components. *Journal of Chemometrics*, 2(2): 155–167, 1988.

Ribeiro, M. T., Singh, S., and Guestrin, C. Why should I trust you? Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144, 2016.

Rockafellar, R. T. and Wets, R. J.-B. *Variational Analysis*. Springer, 2009.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B (Methodological)*, 58(1):267–288, 1996.

Udell, M. and Townsend, A. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.

Xie, W. and Deng, X. Scalable algorithms for the sparse ridge regression. *arXiv preprint arXiv:1806.03756*, 2020.

Yuan, G., Shen, L., and Zheng, W.-S. A block decomposition algorithm for sparse optimization. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 275–285, 2020.

Zou, H. and Hastie, T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B (Methodological)*, 67:301–320, 2005.