Submitted to Statistical Science

Sparse regression: Scalable algorithms and empirical performance

Dimitris Bertsimas Jean Pauphilet and Bart Van Parys

Operations Research Center, MIT

Abstract. In this paper, we review state-of-the-art methods for feature selection in statistics with an application-oriented eye. Indeed, sparsity is a valuable property and the profusion of research on the topic might have provided little guidance to practitioners. We demonstrate empirically how noise and correlation impact both the accuracy - the number of correct features selected - and the false detection - the number of incorrect features selected - for five methods: the cardinality-constrained formulation, its Boolean relaxation, ℓ_1 regularization and two methods with non-convex penalties. A cogent feature selection method is expected to exhibit a two-fold convergence, namely the accuracy and false detection rate should converge to 1 and 0 respectively, as the sample size increases. As a result, proper method should recover all and nothing but true features. Empirically, the integer optimization formulation and its Boolean relaxation are the closest to exhibit this two properties consistently in various regimes of noise and correlation. In addition, apart from the discrete optimization approach which requires a substantial, yet often affordable, computational time, all methods terminate in times comparable with the `glmnet` package for Lasso. We released code for methods that were not publicly implemented. Jointly considered, accuracy, false detection and computational time provide a comprehensive assessment of each feature selection method and shed light on alternatives to the Lasso- regularization which are not as popular in practice yet.

Key words and phrases: Feature selection.

1. INTRODUCTION

The identification of important variables in regression is valuable to practitioners and decision makers in settings with large data sets of high dimensionality. Correspondingly, the notion of sparsity, i.e., the ability to make predictions based on a limited number of covariates, has become cardinal in statistics. The so-called cardinality-penalized estimators for instance minimize the trade-off between prediction accuracy and number of input variables. Though computationally expensive,

Operations Research Center, Massachusetts Institute of Technology, 77 Massachusetts Avenue Bldg. E40-111, Cambridge, MA 02139, USA (e-mail: dbertsim@mit.edu; jpauph@mit.edu; vanparys@mit.edu).

they have been considered as a relevant benchmark in high-dimensional statistics. Indeed, these estimators are characterized as the solution of the NP-hard problem

$$(1) \quad \min_{w\in\mathbb{R}^p} \sum_{i=1}^{n} \ell(y_i, w^\top x_i) + \lambda\|w\|_0,$$

where ℓ is an appropriate convex loss function, such as the ones reported in Table 1 (p. 2). The covariates are denoted by the matrix $X \in \mathbb{R}^{n\times p}$, whose rows are the $x_i^\top$'s, and the response data by $Y = (y_1, ..., y_n) \in \mathbb{R}^n$. Here, $\|w\|_0 := |\{j : w_j \neq 0\}|$ denotes the 0-pseudo norm, i.e., the number of non-zero coefficients of w. Alternatively, one can explicitly constrain the number of features used for prediction and solve

$$(2) \quad \min_{w\in\mathbb{R}^p} \sum_{i=1}^{n} \ell(y_i, w^\top x_i) \text{ s.t. } \|w\|_0 \leqslant k,$$

which is likewise an NP-hard optimization problem [37]. For decades, such problems have thus been solved using greedy heuristics, such as step-wise regression, matching pursuits [35], or recursive feature elimination (RFE) [27].

TABLE 1

Relevant loss functions ℓ and their corresponding Fenchel conjugates $\hat{\ell}$ as defined in Theorem 1. The observed data is continuous, $y \in \mathbb{R}$, for regression and categorical, $y \in \{-1, 1\}$, for classification. By convention, $\hat{\ell}$ equals $+\infty$ outside of its domain. The binary entropy function is denoted $H(x) := -x \log x - (1 - x) \log(1 - x)$.

Method	Loss $\ell(y, u)$	Fenchel conjugate $\hat{\ell}(y, \alpha)$
Ordinary Least Square	$\frac{1}{2}(y-u)^2$	$\frac{1}{2}\alpha^2 + y\alpha$
Logistic loss	$\log\left(1 + e^{-yu}\right)$	$-H(-y\alpha)$ for $y\alpha \in [-1, 0]$
1-norm SVM - Hinge loss	$\max(0, 1 - yu)$	$y\alpha$ for $y\alpha \in [-1, 0]$

Consequently, much attention has been directed to convex surrogate estimators which tend to be sparse, while requiring less computational effort. The Lasso estimator, commonly defined as the solution of

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^{n} \ell(y_i, w^\top x_i) + \lambda\|w\|_1,$$

and initially proposed by Tibshirani [42] is widely known and used. Its practical success can be explained by three concurrent ingredients: Efficient numerical algorithms exist [16, 21, 1], off-the-shelf implementations are publicly available [22] and recovery of the true sparsity is theoretically guaranteed under admittedly strong assumptions on the data [44]. However, recent works [41, 18] have pointed out several key deficiencies of the Lasso regressor in its ability to select the true features without including many irrelevant ones as well. In a parallel direction, theoretical work in statistics [43, 45, 24] has identified regimes where Lasso fails to recover the true support even though support recovery is possible from an information theoretic point of view.

Therefore, new research in numerical algorithms for solving the exact formulation (2) directly has flourished. Leveraging recent advances in mixed-integer solvers

[6, 4], Lagrangian relaxation [39] or cutting-plane methods [5, 7], these works have demonstrated significant improvement over existing Lasso-based heuristics. To the best of our knowledge, the exact algorithm proposed by Bertsimas and Van Parys [5], Bertsimas et al. [7] is the most scalable method providing provably optimal solutions to the optimization problem (2), at the expense of potentially significant computational time and the use of a commercial integer optimization solver.

Another line of research has focused on replacing the ℓ_1 norm in the Lasso formulation by other sparsity-inducing penalties which are less sensitive to noise or correlation between features. In particular, non-convex penalties such as smoothly clipped absolute deviation (SCAD) [17] and minimax concave penalty (MCP) [49] have been proposed. Both SCAD and MCP have the so-called oracle property, meaning that they do not require a priori knowledge of the sparsity pattern to achieve an optimal asymptotic convergence rate, which is theoretically appealing. From a computational point of view, coordinate descent algorithms [8] have shown very effective, even though lack of convexity in the objective function hindered their wide adoption in practice.

Convinced that sparsity is an extremely valuable property in high-impact applications where interpretability matters, and conscious that the profusion of research on the matter might have caused confusion and provided little guidance to practitioners, we propose with the present paper a comprehensive treatment of state-of-the-art methods for feature selection in ordinary least square and logistic regression. Our goal is not to provide a theoretical analysis. On the contrary, we selected and evaluated the methods with an eye towards practicality, taking into account both scalability to large data sets and availability of the implementations. In some cases where open-source implementation was not available, we released code on our website, in an attempt to bridge the gap between theoretical advances and practical adoption. Statistical performance of the methods is assessed in terms of Accuracy (A),

$$A(w) := \frac{|\{j : w_j \neq 0,\ w_{true,j} \neq 0\}|}{|\{j : w_{true,j} \neq 0\}|},$$

i.e., the proportion of true features which are selected, and False Discovery Rate (FDR),

$$FDR(w) := \frac{|\{j : w_j \neq 0,\ w_{true,j} = 0\}|}{|\{j : w_j \neq 0\}|},$$

i.e., the proportion of selected features which are not in the true support.

1.1 Outline and contribution

Our key contributions can be summarized as follows:

- We provide a unified treatment of state-of-the-art methods for feature selection in statistics. More precisely, we cover the cardinality-constrained formulation (2), its Boolean relaxation, the Lasso formulation and its derivatives, and the MCP and SCAD penalty. We did not include step-wise regression methods, for they may require a high number of iterations in high dimension and exist in many variants.
- Encouraged by theoretical results obtained for the Boolean relaxation of (2) by Pilanci et al. [39], we propose an efficient sub-gradient algorithm to solve it and provide theoretical rate of convergence of our method.

- We make our code freely available as a `Julia` package named `SubsetSelection`. Our algorithm scales to problems with $n, p = 100,000$ or $n = 10,000$ and $p = 1,000,000$ within minutes, while providing high-quality estimators.
- We compare the performance of all methods on three metrics of crucial interest in practice: accuracy, false detection rate and computational tractability, and in various regimes of noise and correlation.
- More precisely, under the mutual incoherence condition, all methods exhibit a convergence in accuracy, that is the proportion of correct features selected converges to 1 as the sample size n increases, in all regimes of noise and correlation. Yet, on this matter, cardinality-constrained and MCP formulations are the most accurate. As soon as mutual incoherence condition fails to hold, ℓ_1-based estimators are inconsistent with $A < 1$, while non-convex penalties eventually perfectly recover the support.
- In addition, we also observe a convergence in false detection rate, namely the proportion of irrelevant features selected converging to 0 as the sample size n increases, for some but not all methods: The convex integer formulation and its Boolean relaxation are the only methods which demonstrate this behavior, in low noise settings and make the fewest false discoveries in other regimes. In our experiments, Lasso-based estimators return at least 80% of non-significant features. MCP and SCAD have a low but strictly positive false detection rate (around $15-30\%$ in our experiments) as n increases and in all regimes.
- In terms of computational time, the integer optimization approach is unsurprisingly the most expensive option. Nonetheless, the computational cost is only one or two orders of magnitude higher than other alternatives and remains affordable in many real-world problems, even high-dimensional ones. Otherwise, the four remaining codes terminate in time comparable with the `glmnet` implementation of the Lasso, that is within seconds for $n = 1,000$ and $p = 20,000$.

In Section 2, we present each method, its formulation, its theoretical underpinnings and the numerical algorithms proposed to compute it. In each case, we point the reader to appropriate references and open-source implementations. We propose and describe our sub-gradient algorithm for the Boolean relaxation of (2) also in Section 2. Appendix A provides further details on our implementation of the algorithm, its scalability and its applicability to cardinality-penalized estimators (1) as well. In Section 3 (and Appendix B), we compare the methods on synthetic data sets for linear regression. In particular, we observe and discuss the behavior of each method in terms of accuracy, false detection rate and computational time for three families of design matrices and at least three levels noise. In Section 4 (and Appendix C), we apply the methods to classification problems on similar synthetic problems. We also analyze the implications of the feature selection methods in terms of induced sparsity and prediction accuracy on a real data set from genomics.

1.2 Notations

In the rest of the paper, we denote with $\mathbf{e}$ the vector whose components are equal to one. For $q \geqslant 1$, $\|\cdot\|_q$ denotes the ℓ_q norm defined as

$$\|x\|_q = \left(\sum_i |x_i|^q\right)^{1/q}.$$

For any d-dimensional vector x, we denote with $x_{[j]}$ the jth largest component of x. Hence, we have

$$x_{[1]} \geqslant \cdots \geqslant x_{[d]}.$$

2. SPARSE REGRESSION FORMULATIONS

In this section, we introduce the different formulations and algorithms that have been proposed to solve the sparse regression problem. We focus on the cardinality-constrained formulation, its Boolean relaxation, the Lasso and Elastic-Net estimators, the MCP and SCAD penalty.

2.1 Integer optimization formulation

As mentioned in introduction, a natural way to compute sparse regressors is to explicitly constrain the number of non-zero coefficients, i.e., solve

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^n \ell(y_i, w^\top x_i) \text{ s.t. } \|w\|_0 \leqslant k, \tag{2}$$

where ℓ is an appropriate loss function, appropriate in the sense that $\ell(y,\cdot)$ is convex for any y . In this paper, we focus on Ordinary Least Square (OLS), logistic regression and Hinge loss, as presented in Table 1 on page 2. Unfortunately, such a problem is NP-hard [37] and believed to be intractable in practice. The original attempt by Furnival and Wilson [23] using "Leaps and Bounds" scaled to problems with n, p in the $10s$. Thanks to both hardware improvement and advances in mixed-integer optimization solvers, Bertsimas et al. [6], Bertsimas and King [4] successfully used discrete optimization techniques to solve instances with n, p in the $1,000$s within minutes. More recently, Bertsimas and Van Parys [5], Bertsimas et al. [7] proposed a cutting plane approach which scales to data sizes of with n, p in the $100,000$s for ordinary least square and n, p in the $10,000$s for logistic regression. To the best of our knowledge, our approach is the only method which scales to instances of such sizes, while provably solving such an NP-hard problem.

2.1.1 Convex integer formulation Bertsimas and Van Parys [5], Bertsimas et al. [7] consider an ℓ_2-regularized version of the initial formulation (2),

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^n \ell(y_i, w^\top x_i) + \frac{1}{2\gamma}\|w\|_2^2 \text{ s.t. } \|w\|_0 \leqslant k, \tag{3}$$

where $\gamma > 0$ is a regularization coefficient. From a statistical point of view, this extra regularization, referred to as ridge or Tikhonov regularization, is needed to account for correlation between features [30] and mitigate the effect of noise. Indeed, regularization and robustness are two intimately connected properties, as illustrated by Bertsimas and Fertis [3], Xu et al. [47]. In addition, Breiman et al.

[9] proved that subset selection is a very unstable problem and highlighted the stabilizing effect of Ridge regularization. Introducing a binary variable $s \in \{0,1\}^p$ to encode the support of w and using convex duality, problem (3) can be shown equivalent to a convex integer optimization problem as stated in the following theorem.

THEOREM 1. *[7, Theorem 1] For any convex loss function ℓ, problem (3) is equivalent to*

$$\min_{s\in\{0,1\}^p : s^\top \mathbf{e} \leqslant k} \max_{\alpha\in\mathbb{R}^n} f(\alpha, s) := \left(-\sum_{i=1}^n \hat{\ell}(y_i, \alpha_i) - \frac{\gamma}{2} \sum_{j=1}^p s_j \alpha^\top X_j X_j^\top \alpha \right), \tag{4}$$

where $\hat{\ell}(y,\alpha) := \max_{u\in\mathbb{R}} u\alpha - \ell(y,u)$ is the Fenchel conjugate *of the loss function ℓ [see 2, chap. 6.4], as reported in Table 1. In particular, the function f is continuous, linear in s and concave in α.*

In the special case of OLS, the function f is a quadratic function in α

$$f(\alpha, s) = -\frac{1}{2}\|\alpha\|^2 - Y^\top \alpha - \frac{\gamma}{2} \alpha X_s X_s^\top \alpha,$$

where $X_s X_s^\top := \sum_{j=1}^p s_j X_j X_j^\top$. As a result, the inner maximization problem can be solved in closed form: The maximum is attained at $\alpha^\star(s) = -(I_n + \gamma X_s X_s^\top)^{-1} Y$ and the objective value is

$$\max_\alpha f(\alpha, s) = \frac{1}{2} Y^\top (I_n + \gamma X_s X_s^\top)^{-1} Y.$$

2.1.2 Cutting-plane algorithm Denoting

$$c(s) := \max_{\alpha\in\mathbb{R}^n} f(\alpha, s),$$

which is a convex function in s, the cutting-plane algorithm solves the convex integer optimization problem

$$\min_{s\in\{0,1\}^p} \; c(s) \text{ s.t. } s^\top \mathbf{e} \leqslant k,$$

by iteratively tightening a piece-wise linear lower approximation of c. Pseudo-code is given in Algoritm 2.1 (p. 7). Proof of termination and details on implementation can be found in Bertsimas and Van Parys [5] for regression and Bertsimas et al. [7] for classification. This outer-approximation scheme was originally proposed by Duran and Grossmann [15] for general nonlinear mixed-integer optimization problems.

2.1.3 Implementation and publicly available code A naive implementation of Algorithm 2.1 would solve a mixed-integer linear optimization problem at each iteration, which can be as expensive as explicit enumeration of all feasible supports s. Fortunately, with modern solvers such as Gurobi [26] or CPLEX [12], this outer-approximation scheme can be implemented using *lazy constraints*, enabling the use of a single Branch-and-Bound tree for all subproblems.

Algorithm 2.1 Outer-approximation algorithm

Require: $X \in \mathbb{R}^{n\times p}$, $Y \in \mathbb{R}^n$, $k \in \{1, ..., p\}$
  $t \leftarrow 1$
  repeat
    $s^{t+1}, \eta^{t+1} \leftarrow \text{argmin}_{s\in\{0,1\}^p,\eta} \left\{\eta \,:\, \sum_{j=1}^p s_j \leqslant k,\ \eta \geqslant c(s^i) + \nabla c(s^i)^\top (s - s^i),\ \forall i = 1, \ldots, t\right\}$
    $t \leftarrow t + 1$
  until $\eta^t < c(s^t) - \varepsilon$
  return s^t

The algorithm terminates when the incumbent solution is ε-optimal for some fixed tolerance level ε (we chose $\varepsilon = 10^{-4}$ in our simulations). We also need to impose a time limit on the algorithm. Indeed, as often in discrete optimization, the algorithm can quickly find the optimal solution, but spends a lot of the time proving its optimality. In our experiment, we fixed a time limit of 60 seconds for regression and 180 seconds for classification. Such choices were guided by confidence in the quality of the initial solution s_1 we provide to the algorithm (which we will describe in the next section) as well as time needed to compute $c(s)$ and $\nabla c(s)$ for a given support s.

The formulation (3) contains two hyper-parameters, k and γ, to control for the amount of sparsity and regularization respectively. In practice, those parameters need to be tuned using a cross-validation procedure. Since the function to minimize does not depend on k, any piece-wise linear lower approximation of $c(s)$ computed to solve (3) for some value of k can be reused to solve the problem at another sparsity level. In recent work, Kenney et al. [32] proposed a combination of implementation recipes to optimize such search procedures. As for γ, we apply the procedure described in Chu et al. [11], starting with a low value γ_0 (typically scaling as $1/\max_i \|x_i\|^2$) and inflate it by a factor 2 at each iteration.

To bridge the gap between theory and practice, we present a `Julia` code which implements the described cutting-plane algorithm publicly available on GitHub[1]. The code requires a commercial solver like Gurobi or CPLEX and our open-source package `SubsetSelection` for the Lagrangian relaxation, which we introduce in the next section. We also call the open-source library `LIBLINEAR` [19] to efficiently compute $c(s)$ in the case of Hinge and logistic loss.

2.2 Lagrangian relaxation

As often in discrete optimization, it is natural to consider the Boolean relaxation of problem (4)

$$\min_{s\in[0,1]^p \,:\, \mathbf{e}^\top s \leqslant k} \max_{\alpha\in\mathbb{R}^n} f(\alpha, s), \tag{5}$$

and study its tightness, as done by Pilanci et al. [39].

2.2.1 Tightness result The above problem is recognized as a convex/concave saddle point problem. According to Sion's minimax theorem [40], the minimization and maximization in (5) can be interchanged. Hence, saddle point solutions $(\bar{\alpha}, \bar{s})$ of (5) should satisfy

$$\bar{\alpha} \in \arg\max_{\alpha\in\mathbb{R}^n} f(\alpha, \bar{s}), \quad \bar{s} \in \arg\min_{s\in[0,1]^p} f(\bar{\alpha}, s) \text{ s.t. } s^\top \mathbf{e} \leqslant k.$$

[1] https://github.com/jeanpauphilet/SubsetSelectionCIO.jl

Since f is a linear function of s, a minimizer of $f(\bar{\alpha}, s)$ can be constructed easily by selecting the k smallest components of the vector $(-\frac{\gamma}{2}\bar{\alpha}^\top X_j X_j^\top \bar{\alpha})_{j=1,\dots,p}$. If those k smallest components are unique, the so constructed binary vector must be equal to $\bar{s}$ and hence the relaxation (5) is tight. In fact, the previous condition is necessary and sufficient as proven by Pilanci et al. [39]:

THEOREM 2. *[39, Proposition 1] The Boolean relaxation (5) is tight if and only if there exists a saddle point $(\bar{\alpha}, \bar{s})$ such that the vector $\bar{\beta} := (\bar{\alpha}^\top X_j X_j^\top \bar{\alpha})_{j=1,\dots,p}$ has unambiguously defined k largest components, i.e., there exists $\lambda \in \mathbb{R}$ such that $\bar{\beta}_{[1]} \geqslant \cdots \geqslant \bar{\beta}_{[k]} > \lambda > \bar{\beta}_{[k+1]} \geqslant \cdots \geqslant \bar{\beta}_{[p]}$.*

This uniqueness condition in Theorem 2 seems often fulfilled in real-world applications. It is satisfied with high probability, for instance, when the covariates X_j are independent [see 39, Theorem 2]. In other words, randomness breaks the complexity of the problem. Similar behavior has already been observed for semi-definite relaxations [31, 48]. Such results have had impact in practice and propelled the advancement of convex proxy based heuristics such as Lasso. Efficient algorithms can be designed to solve the saddle point problem (5) without involving sophisticated discrete optimization tools and provide a high-quality heuristic for approximately solving (4) that could be used as a good warm-start in exact approaches.

2.2.2 Dual sub-gradient algorithm In this section, we propose and describe an algorithm for solving problem (5) efficiently and make our code available as a `Julia` package. Our algorithm is fast and scales to data sets with n, p in the $100,000$s, which is two orders of magnitude larger than the implementation proposed by Pilanci et al. [39].

For a given s, maximizing f over α cannot be done analytically, with the noteworthy exception of ordinary least squares, whereas minimizing over s for a fixed α reduces to sorting the components of $(-\alpha^\top X_j X_j^\top \alpha)_{j=1,\dots,p}$ and selecting the k smallest. We take advantage of this asymmetry by proposing a dual projected sub-gradient algorithm with constant step-size, as described in pseudo-code in Algorithm 2.2. δ denotes the step size in the gradient update and $\mathcal{P}$ the projection operator over the domain of f. At each iteration, the algorithm updates the support s by minimizing $f(\alpha, s)$ with respect to s, α being fixed. Then, the variable α is updated by performing one step of projected sub-gradient ascent with constant step size δ. The denomination "sub-gradient" comes from the fact that at each iteration $\nabla_\alpha f(\alpha^T, s^T)$ is a sub-gradient to the function $\alpha \mapsto \min_s f(\alpha, s)$ at $\alpha = \alpha^T$.

In terms of computational cost, updating α requires $O(n\|s\|_0)$ operations for computing the sub-gradient plus at most $O(n)$ operations for the projection on the feasible domain. The most time-consuming step in Algorithm 2.2 is updating s which requires on average $O(np + p\log p)$ operations.

The final averaging step $\hat{\alpha}_T = \frac{1}{T}\sum_t \alpha_t$ is critical in sub-gradient methods to ensure convergence of the algorithm in terms of optimal value [see 2, chap. 7.5].

THEOREM 3. *[2, chap. 7.5] Assume the sequence of sub-gradients $\{\nabla_\alpha f(\alpha_T, s_T)\}$ is uniformly bounded by some constant $L > 0$, and that the set of saddle point*

Algorithm 2.2 Dual sub-gradient algorithm

$s^0, \alpha^0 \leftarrow$ Initial solution
$T = 0$
repeat
  $s^{T+1} \in \operatorname{argmin}_s f(\alpha^T, s)$
  $\alpha^{T+1} = \mathcal{P}\left(\alpha^T + \delta \nabla_\alpha f(\alpha^T, s^T)\right)$
  $T = T + 1$
until Stop criterion
$\hat{\alpha}_T = \frac{1}{T}\sum_t \alpha^t$
return $\hat{s} = \operatorname{argmin}_s f(\hat{\alpha}_T, s)$

solutions $\bar{A}$ in (5) is non-empty. Then,

$$f\left(\hat{\alpha}_T, \hat{s}\right) \geqslant f(\bar{\alpha}, \bar{s}) - \frac{\delta L^2}{2} - \frac{dist^2(\alpha^0, \bar{A})}{2\delta T},$$

where $dist(\alpha^0, \bar{A})$ denotes the distance of the initial point α^0 to the set of saddle point solutions $\bar{A}$.

As for any sub-gradient strategy, with an optimal choice of step size δ[2], Theorem 3 proves a $O(1/\sqrt{T})$ convergence rate in terms of objective value, which is disappointingly slow. However, in practice, convergence towards the optimal primal solution $\bar{s}$ is more relevant. In that metric, our algorithm performs particularly well as numerical experiments in Sections 3 and 4 demonstrate. The key to our success is that the optimal primal solution is estimated using partial minimization

$$\hat{s} = \arg\min_s f(\hat{\alpha}_T, s),$$

as opposed to averaging

$$\hat{s} = \frac{1}{T}\sum_t s^t,$$

as studied by Nedić and Ozdaglar [38] and commonly implemented for sub-gradient methods. In addition, even though we are solving a relaxation, we are interested in binary vectors s, which can be interpreted as a set of features. To that extend, averaging would not have been a suitable option since the averaged solution is neither binary, nor k-sparse. With the extra cost of computing $c(s^t)$ for all past iterates s^t as well as $c(\hat{s})$, one can also decide to return the support vector with the lowest value. This can only produce a better approximation of $\operatorname{argmin}_{s\in\{0,1\}^p : s^\top \mathbf{e} \leqslant k} c(s)$.

2.2.3 Implementation and open-source package The algorithm terminates after a fixed number of iterations T_{max} which is standard for sub-gradient methods. In our case, however, the quality of the primal variable s should be the key concern. By computing $c(s^t)$ at each iteration and keeping track of the best upper-bound $\min_{t=1,\dots,T} c(s^t)$, one can use the duality gap or the number of consecutive iterations without any improvement on the upper-bound as alternative stopping criteria. Computing $c(s^t)$ increases the cost per iteration, with the hope of terminating the algorithm earlier. By default, our implementation stops after $T_{max} = 200$ iterations or when the duality gap is 10^{-4}.

[2] $\delta = \dfrac{\operatorname{dist}(\alpha^0, \bar{A})}{L\sqrt{T}}$.

The constant step size rule is difficult to implement in practice. Indeed, as seen in Theorem 3, an optimal step size should depend on quantities that are hard do estimate *a priori*, namely L and $\text{dist}^2(\alpha^0, \bar{A})$. In particular for logistic loss, L can be arbitrarily large. Instead, one can use an adaptive stepsize rule such as

$$\delta^T = \frac{\min_{t=1,\dots,T} c(s^t) - \max_{t=1,\dots,T} f(\alpha^t, s^t)}{\|\nabla_\alpha f(\alpha^T, s^T)\|^2}.$$

We implemented such a rule and refer to Bertsekas [2, chap. 7.5] for proofs of convergence and alternative choice.

We apply the same grid-search procedures as for the cutting-plane algorithm in order to cross-validate the hyperparameters k and γ.

We make our code publicly available as a `Julia` package named `SubsetSelection` and source repository can be found on GitHub[3]. Our code implements Algorithm 2.2 for six loss functions including those presented in Table 1. The package consists of one main function, `subsetSelection`, which solves problem (5) for a given value of k. The algorithm can be extended to more loss functions, and cardinality-penalized estimators as well, as described in Appendix A.

2.3 Lasso - ℓ_1 relaxation

Instead of solving the NP-hard problem (1), Tibshirani [42] proposed replacing the non-convex ℓ_0-pseudo norm by the convex ℓ_1-norm which is sparsity-inducing. Indeed, extreme points of the unit ℓ_1 ball $\{x : \|x\|_1 \leqslant 1\}$ are 1-sparse vectors. The resulting formulation

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^n \ell(y_i, w^\top x_i) + \lambda\|w\|_1, \tag{6}$$

is referred to as the Lasso estimator. More broadly, ℓ_1-regularization is now a widely used technique to induce sparsity in a variety of statistical settings [see 28, for an overview]. Its popularity has been supported by an extensive corpus of theoretical results from signal processing and high-dimensional statistics. Since the seminal work of Donoho and Huo [13], assumptions needed for the Lasso estimator to accurately approximate the true sparse signal are pretty well understood. We refer to reader to Candes et al. [10], Meinshausen and Bühlmann [36], Zhao and Yu [50], Wainwright [44] for some of these results. In practice however, those assumptions, namely mutual incoherence and restricted eigenvalues conditions, are quite stringent and hard to verify. In addition, even when the Lasso regressor provably converges to the true sparse regressor in terms of ℓ_2 distance and identifies all the correct features, it also systematically incorporates irrelevant ones, a behavior observed and partially explained by Su et al. [41] and of crucial practical impact.

2.3.1 Elastic-Net formulation The Lasso formulation in its original form (6) involves one hyper-parameter λ only, which controls regularization, i.e., robustness of the estimator against uncertainty in the data [see 47, 3]. At the same time, the ℓ_1 norm in the Lasso formulation (6) is also used for its fortunate but collateral sparsity-inducing property. Robustness and sparsity, though related, are two very

[3] https://github.com/jeanpauphilet/SubsetSelection.jl

distinct properties demanding a separate hyper parameter each. The ElasticNet (ENet) formulation proposed by Zou and Hastie [51]

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^{n} \ell(y_i, w^\top x_i) + \lambda \left[\alpha\|w\|_1 + \tfrac{1-\alpha}{2}\|w\|_2^2\right], \tag{7}$$

addresses the issue by adding an ℓ_2 regularization to the objective. For $\alpha = 1$, problem (7) is equivalent to the Lasso formulation (6), while $\alpha = 0$ corresponds to Ridge regression. Although this extra regularization reduces bias and improves prediction error, it does not significantly improve feature selection, as we will see on numerical experiments in Section 3. In our view, this is due to the fact that ℓ_1-regularization primarily induces robustness of the estimator, through shrinkage of the coefficients [3, 47]. In that perspective, it leads to first-rate out-of-sample predictive performance, even in high-noise regimes [see 29, for extensive experiments]. Nonetheless, the feature selection ability of ℓ_1-regularization ought to be challenged.

2.3.2 Algorithms and implementation For ℓ_1-regularized regression, Least Angle Regression (LARS) [16] is an efficient method for computing an entire path of solutions for various values of the λ parameter, exploiting the fact that the regularization path is piecewise linear. More recently, coordinate descent methods [20, 46, 21] have successfully competed with and surpassed the LARS algorithm, especially in high dimension. Their implementation through the `glmnet` package [22] is publicly available in `R` and many other programming languages. In a different direction, proximal gradient descent methods have also been proposed, and especially the Fast Iterative Shrinkage Thresholding Algorithm (FISTA) proposed by Beck and Teboulle [1].

2.4 Non-convex penalties

Recently, other formulations have been proposed, of the form

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^{n} \ell(y_i, w^\top x_i) + \sum_{j=1}^{p} p_{\lambda,\gamma}(|w_j|), \tag{8}$$

where $p_{\lambda,\gamma}(\cdot)$ is a function parametrized by λ and γ, which control respectively the tradeoff between empirical loss and regularization, and the shape of the function. We will consider two popular choice of penalty functions $p_{\lambda,\gamma}(\cdot)$, which are non-convex and are proved to recover the true support even when mutual incoherence condition fails to hold [33].

2.4.1 Minimax Concave Penalty (MCP) The minimax concave penalty of Zhang [49] is defined on $[0, \infty)$ by

$$p_{\lambda,\gamma}(u) = \lambda \int_0^u \left(1 - \frac{t}{\gamma\lambda}\right)_+ \mathrm{d}t = \begin{cases} \lambda u - \dfrac{u^2}{2\gamma} & \text{if } u \le \gamma\lambda, \\ \dfrac{\gamma\lambda^2}{2} & \text{if } u > \gamma\lambda, \end{cases}$$

for some $\lambda \geqslant 0$ and $\gamma > 1$. The rationale behind the MCP can be explained in the univariate OLS case: In the univariate case, MCP and ℓ_1-regularization lead to the same solution as $\gamma \to \infty$, while the MCP is indeed equivalent to hard-thresholding

when $\gamma = 1$. In other words, in one dimension or under the orthogonal design assumption, the MCP produces the so-called firm-shrinkage estimator introduced by Gao and Bruce [25], which should be understood as a continuous tradeoff between hard- and soft-thresholding.

2.4.2 Smoothly Clipped Absolute Deviation (SCAD) Fan and Li [17] orginally proposed the smoothly clipped absolute deviation penalty, defined on $[0, \infty)$ by

$$p_{\lambda,\gamma}(u) = \begin{cases} \lambda u & \text{if } u \leq \lambda \\ \dfrac{\gamma\lambda u - (u^2 + \lambda^2)/2}{\gamma - 1} & \text{if } \lambda < u \leq \gamma\lambda, \\ \dfrac{\lambda^2(\gamma^2 - 1)}{2(\gamma - 1)} & \text{if } u > \gamma\lambda, \end{cases}$$

for some $\lambda \geqslant 0$ and $\gamma > 2$. The rationale behing the SCAD penalty is similar to the MCP but less straightforward. We refer to Fan and Li [17] for a comparison of SCAD penalty with hard-thresholding and ℓ_1-penalty and to Breheny and Huang [8] for a comparison of SCAD and MCP.

2.4.3 Algorithms and implementation For such non-convex penalties, Zou and Li [52] designed a local linear approximation (LLA) approach where, at each iteration, the penalty function is linearized around the current iterate and the next iterate is obtained by solving the resulting convex optimization problem with linear penalty. Another, more computationally efficient, approach has been proposed by Breheny and Huang [8] and implemented in the open-source `R` package, `ncvreg`. Their algorithm relies on coordinate descent and the fact that the objective function in (8) with OLS loss is convex in any w_j, the other $w_{j'}, j' \neq j$ being fixed. For logistic loss, they locally approximate the loss function by a second-order Taylor expansion at each iteration and use coordinate descent to compute the next iterate.

3. LINEAR REGRESSION ON SYNTHETIC DATA

In this section, we compare the aforementioned methods on synthetic linear regression data where the ground truth is known to be sparse. The convex integer optimization algorithm of Bertsimas and Van Parys [5], Bertsimas et al. [7] (CIO in short) was implemented in `Julia` [34] using the commercial solver Gurobi 7.5.0 [26], interfaced using the optimization package `JuMP` [14]. The Lagrangian relaxation is also implemented in `Julia` and openly available as the `SubsetSelection` package (SS in short). We used the implementation of Lasso/Enet provided by the `glmnet` package [22], available both in `R` and `Julia`. We also compared to MCP and SCAD penalty formulations implemented in the `R` package `ncvreg` [8]. The computational tests were performed on a computer with Xeon @2.3GhZ processors, 1 CPUs, 16GB RAM per CPU.

3.1 Data generation methodology

The synthetic data was generated according to the following methodology: We draw $x_i \sim \mathcal{N}(0_p, \Sigma), i = 1, .., n$ independent realizations from a p-dimensional normal distribution with mean 0_p and covariance matrix Σ. We randomly sample a weight vector $w_{true} \in \{-1, 0, 1\}$ with exactly k_{true} non-zero coefficient. We draw $\varepsilon_i, i = 1, ..., n$, i.i.d. noise components from a normal distribution scaled according to a chosen signal-to-noise ratio $\sqrt{SNR} = \|Xw_{true}\|_2/\|\varepsilon\|_2$. Finally, we

compute $Y = Xw_{true} + \varepsilon$. With this methodology, we are able to generate data sets of arbitrary size (n, p), sparsity k_{true}, correlation structure Σ and level of noise $\sqrt{SNR}$. The signal-to-noise ratio relates to the percentage of variance explained (PVE). Indeed, Hastie et al. [29] showed that

$$PVE = \frac{SNR}{1 + SNR}.$$

Accordingly, we will consider SNR values ranging from 6 ($PVE = 85.7\%$) to 0.05 ($PVE = 4.8\%$).

3.2 Metrics and benchmarks

Statistical performance of the methods is assessed in terms of Accuracy (A),

$$A(w) := \frac{|\{j : w_j \neq 0,\ w_{true,j} \neq 0\}|}{|\{j : w_{true,j} \neq 0\}|},$$

i.e., the proportion of true features which are selected, and False Discovery Rate (FDR),

$$FDR(w) := \frac{|\{j : w_j \neq 0,\ w_{true,j} = 0\}|}{|\{j : w_j \neq 0\}|},$$

i.e., the proportion of selected features which are not in the true support. We refer to the quantities in the numerators as the number of true features (TF) and false features (FF) respectively. One might argue that accuracy as we defined it here is a purely theoretical metric, since on real-world data, the ground truth is unknown and there is no such thing as *true* features. Still, accuracy is the only metric which assesses feature selection only, while derivative measures such as predictive power depend on more factors than the features selected alone. Moreover, accuracy has some practical implications in terms of interpretability and also in terms of predictive power: Common sense and empirical results suggest that better selected features should yield diminished prediction error. To that end, we also compare the performance of the methods in terms of out-of-sample Mean Square Error

$$MSE(w) := \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i^\top w)^2,$$

which will be the metric of interest on real data. Note that the sum can be taken over the observations in the training (in-sample) or test set (out-of-sample).

Practical scalability of the algorithms is assessed in terms of computational time. In order to provide a fair comparison between methods that are not implemented in the same programming language, we report computational time for each algorithm relative to the time needed to compute a Lasso estimator with `glmnet` in the same language and on the same data. For these experiments, we fixed a time limit of 60 seconds for the cutting-plane algorithm and considered 150 iterations of the sub-gradient algorithm for the Boolean relaxation.

3.3 Synthetic data satisfying mutual incoherence condition

We first consider Toeplitz covariance matrix $\Sigma = \left(\rho^{|i-j|}\right)_{i,j}$. Such matrices satisfy the mutual incoherence condition, required by ℓ_1-regularized estimators to be statistically consistent. We compare the performance of the methods in six different regimes of noise and correlation described in Table 2 (p. 14).

TABLE 2
Regimes of noise (SNR) and correlation (ρ) considered in our experiments on regression with Toeplitz covariance matrix

	Low correlation	High correlation
Low noise	$\rho = 0.2$ $SNR = 6$ $p = 20,000$ $k = 100$	$\rho = 0.7$ $SNR = 6$ $p = 20,000$ $k = 100$
Medium noise	$\rho = 0.2$ $SNR = 1$ $p = 10,000$ $k = 50$	$\rho = 0.7$ $SNR = 1$ $p = 10,000$ $k = 50$
High noise	$\rho = 0.2$ $SNR = 0.05$ $p = 2,000$ $k = 10$	$\rho = 0.7$ $SNR = 0.05$ $p = 2,000$ $k = 10$

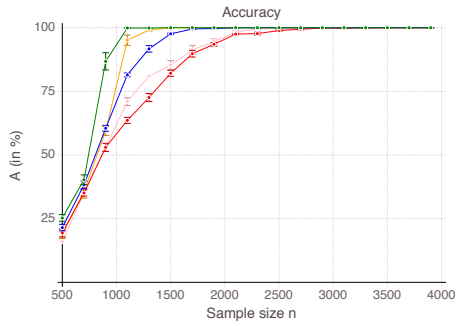
3.3.1 Feature selection with a given support size We first consider the case when the cardinality k of the support to be returned is given and equal to the true sparsity k_{true} for all methods, while all other hyper-parameters are cross-validated on a separate validation set. In this case, accuracy and false discovery rate are complementary. Indeed, in this case

$$|\{j : w_{true,j} \neq 0\}| = |\{j : w_j \neq 0\}| = k_{true},$$

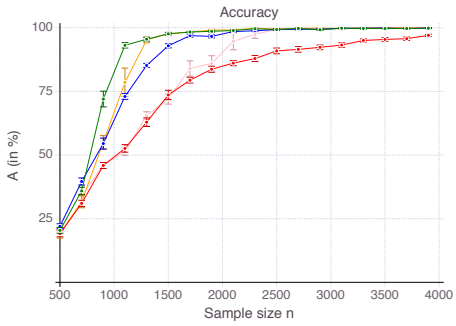
which leads to $A = 1 - FDR$ so that we may consider accuracy by itself.

As shown on Figure 1 (p. 15), all methods converge in terms of accuracy. That is their ability to select correct features as measured by A smoothly converges to 1 with an increasing number of observations $n \to \infty$. Noise in the data has an equalizing effect on all methods, meaning that noise reduces the gap in performance. Indeed, in high-noise regimes, all methods are comparable. On the contrary, correlation is discriminating: High correlation strongly hinders the performance of Lasso/ENet, moderately those of SCAD and very slightly CIO, SS and MCP methods. Among all methods, ℓ_1-regularization is the less accurate, selects fewer correct features than the four other methods and is sensitive to correlation between features. SCAD provides modest improvement over ENet in terms of accuracy, in comparison with CIO, SS and MCP. Unsurprisingly, we observe a gap between the solutions returned by the cutting-plane method and its Boolean relaxation, gap which decreases as noise increases. All things considered, CIO and the MCP penalization are the best performing method in all six regimes, with a fine advantage for CIO.

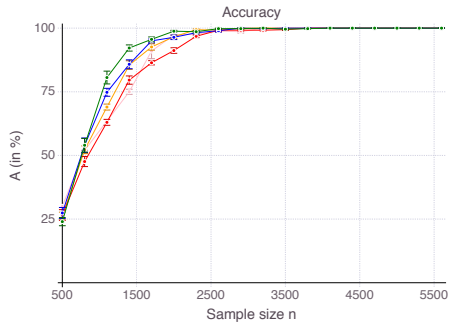
Figure 2 on page 17 reports relative computational time compared to `glmnet` in log scale. It should be kept in mind that we restricted the cutting-plane algorithm to a 60-second time limit and the sub-gradient algorithm to $T_{max} = 200$ iterations. All methods terminate in times of one to two orders of magnitude larger than `glmnet` (seconds for the problem size at hand), contradicting the common belief that ℓ_1-penalization is the *only* tractable alternative to exact subset selection. Computational time for the discrete optimization algorithm CIO and sub-gradient



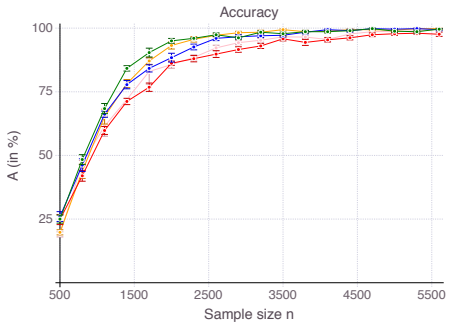
(a) Low noise, low correlation



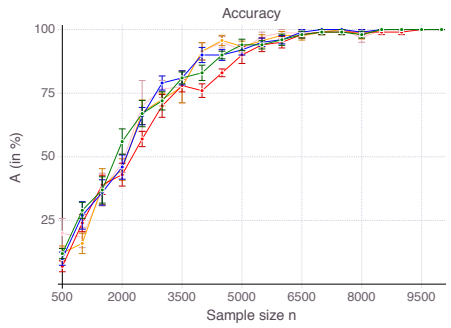
(b) Low noise, high correlation



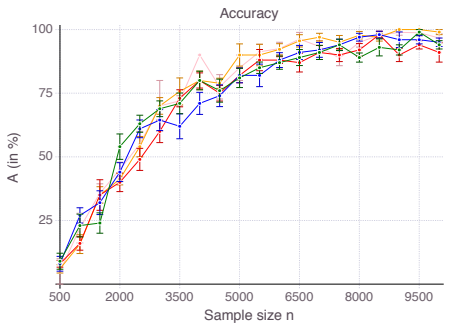
(c) Medium noise, low correlation



(d) Medium noise, high correlation



(e) High noise, low correlation



(f) High noise, high correlation

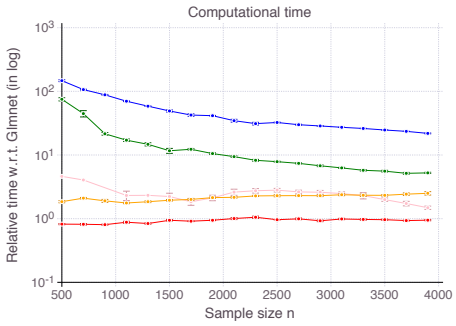
Fig 1: Accuracy as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.

algorithm SS highly depends on the regularization parameter γ. For low γ, which are suited in high noise regimes, the algorithm is extremely fast, while it can take as long as a minute in low noise regimes. This phenomenon explains the relative comparison of SS with `glmnet` in Figure 2. For this is an important practical aspect, we provide detailed experiments regarding computational time in Appendix B.1. As previously mentioned, stopping the algorithm SS after a consecutive number of non-improvements can drastically reduce computational time. Empirically, this strategy did not hinder the quality of the solution in regression settings, but was not as successful in classification setting, so we did not reported its performance. As CIO has a fixed time limit independent of n, the relative gap in terms of computational time with `glmnet` narrows as sample size increases.

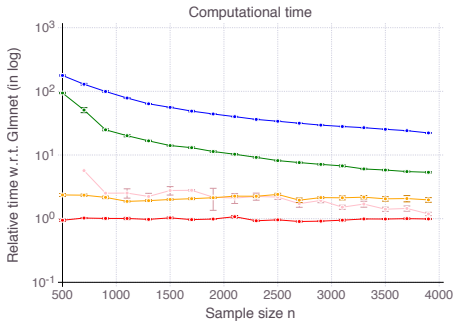
Finally, though a purely theoretic metric, accuracy has some intuitive and practical implications in terms of out-of-sample prediction. To support our claim, Figure 3 (p. 18) represents the out-of-sample MSE for all five methods, as n increases, for the six noise/correlation settings of interest. There is a clear connection between performance in terms of accuracy and in terms of predictive power, with CIO performing the best. Still, good predictive power does not necessarily imply that the features selected are mostly correct. SCAD, for instance, seems to provide a larger improvement over ENet in terms of predictive power than in accuracy. Similarly, SS dominates MCP in terms of out-of-sample MSE, while this is not the case in terms of accuracy.

3.3.2 Feature selection with cross-validated support size We now compare all methods when k_{true} is no longer given and needs to be cross-validated from the data itself.

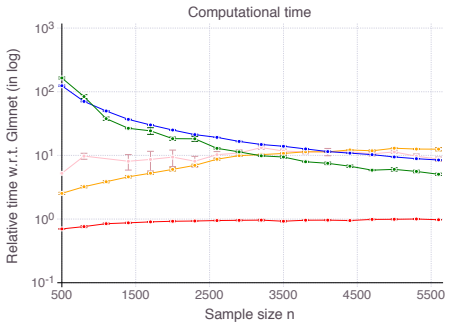
For each value of n, each method fits a model on a training set for various levels of sparsity k, either explicitly or by adjusting the penalization parameter. For each sparsity level k, the resulting classifier incorporates some true and false features. Figure 4 (p. 20) represents the number of true features against the number of false features for all five methods, for a range of sparsity levels k, all other hyper-parameters being tuned so as to minimize MSE on a validation set. To obtain a fair comparison, we used the same range of sparsity levels for all methods. Some methods only indirectly control the sparsity k through a regularization parameter λ and do not guarantee to return *exactly* k features. In these cases, we calibrated λ as precisely as possible and used linear interpolation when we were unable to get the exact value of k we were interested in. From Figure 4, we observe that in low correlation settings, CIO and MCP strictly dominate ENet, SCAD and SS. There is no clear winner between CIO and MCP. When noise is low, CIO tends to make less false discoveries, while the latter is generally more accurate, but the difference between all methods diminishes as noise increases. In high correlation settings, no method clearly dominates. CIO, SS and MCP are better for small support size k, while ENet and SCAD dominate for larger supports. In high noise and high correlation regimes though, Enet and SCAD seem to clearly dominate their competitors. In practice however, one does not have access to "true" features and cannot decide on the value of k based on such ROC curves. As often, we select the value $k^\star$ which minimizes out-of-sample error on a validation set. To this end, Figure 5 (p. 21) visually represents validation MSE as a function of k for all five methods.The vertical black line corresponds to $k = k_{true}$. For each method, $k^\star$ is identified as the minimum of the out-of-sample MSE curve. From Figure 5,



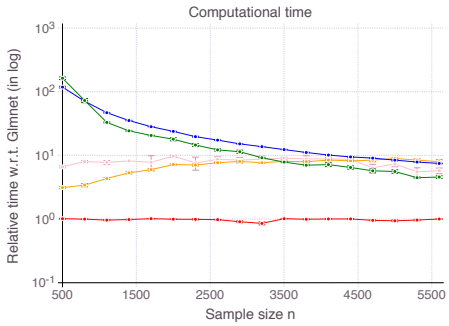
(a) Low noise, low correlation



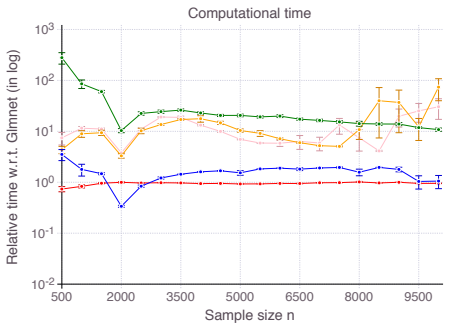
(b) Low noise, high correlation



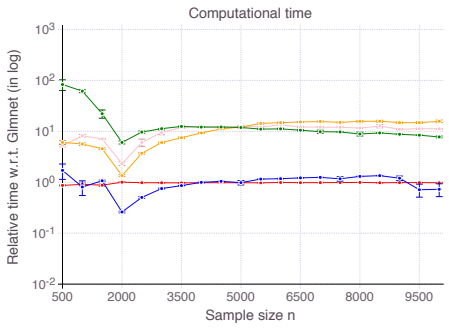
(c) Medium noise, low correlation



(d) Medium noise, high correlation

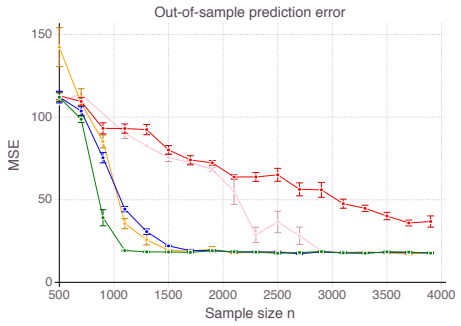


(e) High noise, low correlation

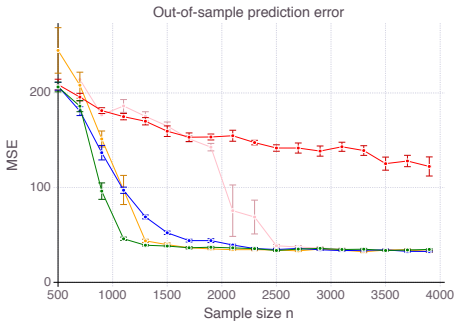


(f) High noise, high correlation

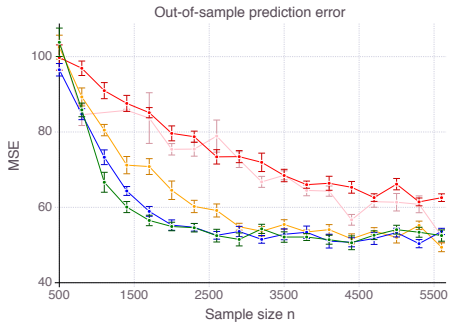
Fig 2: Computational time relative to Lasso with glmnet as n increases, for CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.



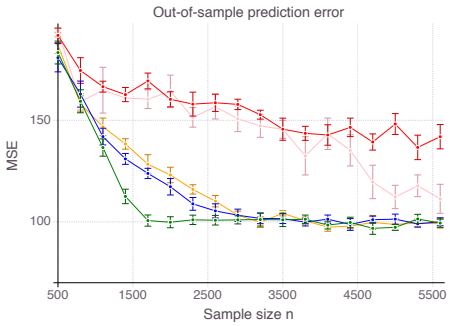
(a) Low noise, low correlation



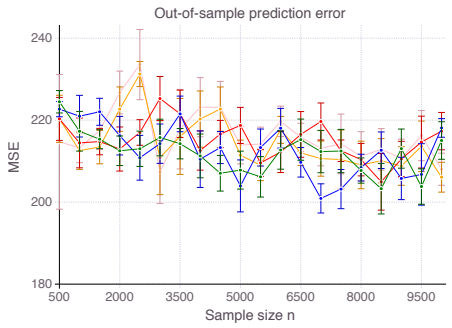
(b) Low noise, high correlation



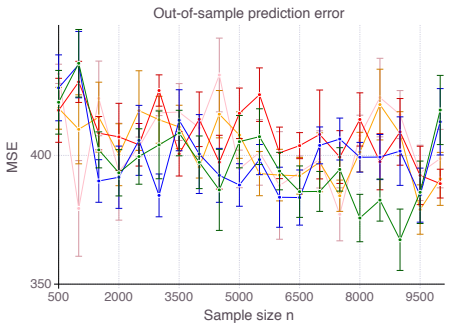
(c) Medium noise, low correlation



(d) Medium noise, high correlation



(e) High noise, low correlation



(f) High noise, high correlation

Fig 3: Out-of-sample mean square error as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.

we can expect the Lasso/ENet and SCAD formulations to select many irrelevant features, while CIO, SS and MCP are relatively close to the true sparsity pattern.

As a result, for every n, each method selects $k^\star$ features, some of which are in the true support, others being irrelevant, as measured by accuracy and false detection rate respectively. Figures 6 (p. 22) and 7 (p. 23) report the results of the cross-validation procedure for increasing n. In terms of accuracy (Figure 6), all five methods are relatively equivalent and demonstrate a clear convergence: $A \to 1$ as $n \to \infty$. The first to achieve perfect accuracy is ENet, followed by SCAD, MCP, CIO and then SS. On false detection rate however (Figure 7), the methods rank in the opposite order. Among all five, CIO and SS achieve the lowest FDR with FDR as low as 0% in low noise settings and around 30% when noise is high. On the contrary, ENet persistently returns around 80% of incorrect features in all regimes of noise and correlation. Concerning MCP and SCAD, in low noise regimes, false detection rate quickly drops as sample size increases. Yet, for large values of n, we observe a strictly positive FDR on average (around 15% in our experiments) and high variance, suggesting that feature selection with these regularizations is pretty unstable. As noise increases, FDR for those methods remains significant (around 50%), with a fine advantage of MCP over SCAD. In our opinion, this is due to the fact that MCP and SCAD, just like Lasso/ENet, do not enforce sparsity explicitly, like CIO or SS do, but rely on regularization to *induce* it.

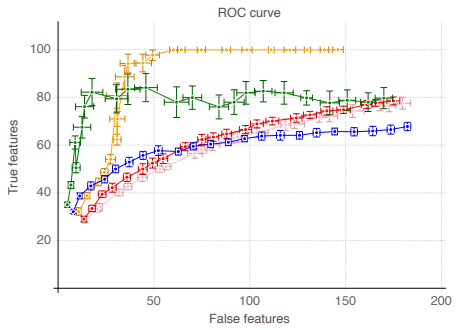
3.4 Synthetic data *not* satisfying mutual incoherence condition

We now consider a "hard" correlation structure, i.e., a setting where the standard Lasso estimator is inconsistent. Fix p, k_{true} and a scalar $\theta \in \left(\frac{1}{k_{true}}, \frac{1}{\sqrt{k_{true}}}\right)$[4] Define Σ as a matrix with 1's on the diagonal, θ's in the first k_{true} positions of the $(k_{true}+1)$th row and column, and 0's everywhere else. Such a matrix does not satisfy mutual incoherence [see 33, Appendix F.2. for a proof]. As opposed to the previous setting, we fix $w_{true} = \left(\frac{1}{\sqrt{k_{true}}}, \dots, \frac{1}{\sqrt{k_{true}}}, 0, \dots, 0\right)$, and compute noisy signals, for increasing noise levels (see Table 3 p. 19). In this setting, the ℓ_1-penalty result in an estimator that puts nonzero weight on the $(k+1)$th coordinate, while MCP and SCAD penalties eventually recover the true support [33].

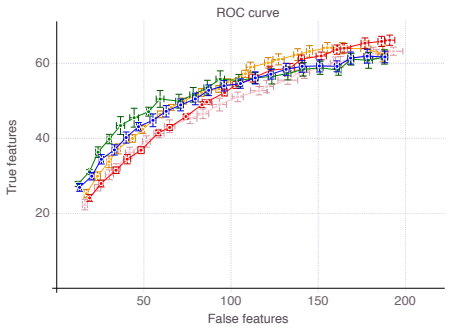
TABLE 3
Regimes of noise (SNR) considered in our experiments on regression

Low noise	$SNR = 6$ $p = 20,000$ $k = 100$
Medium noise	$SNR = 1$ $p = 10,000$ $k = 50$
High noise	$SNR = 0.05$ $p = 2,000$ $k = 10$

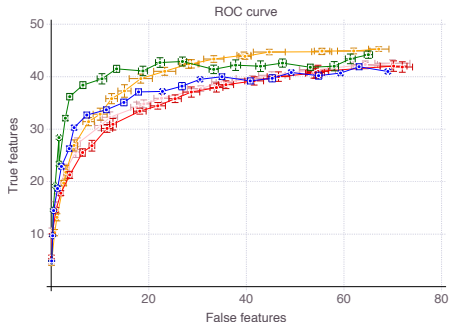
[4] in our experiment we take $\theta = \frac{1}{2k_{true}} + \frac{1}{2\sqrt{k_{true}}}$.



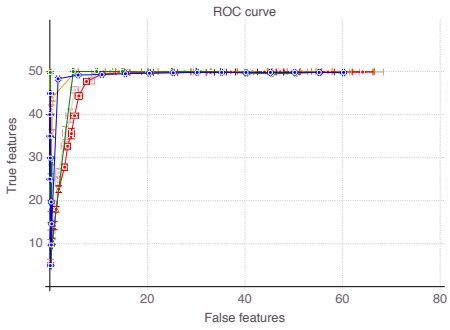
(a) Low noise, low ρ, $n = 900$



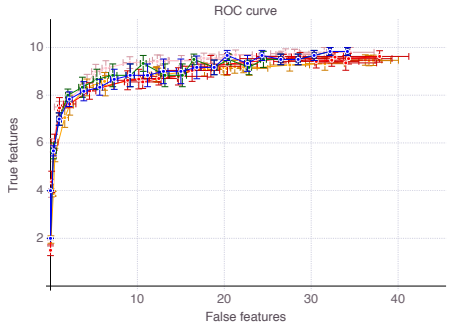
(b) Low noise, high ρ, $n = 900$



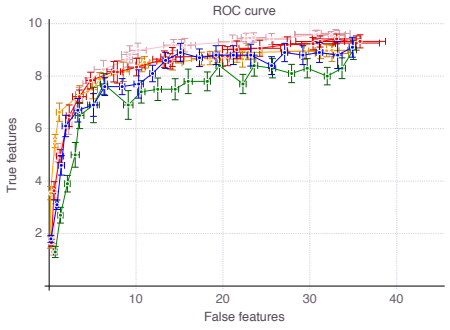
(c) Medium noise, low ρ, $n = 1,100$



(d) Medium noise, high ρ, $n = 1,100$

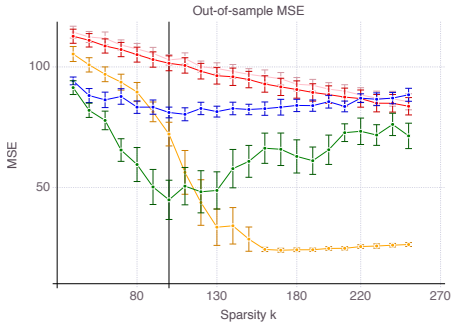


(e) High noise, low ρ, $n = 3,500$

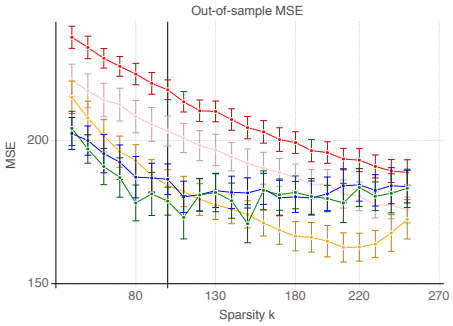


(f) High noise, high ρ, $n = 3,500$

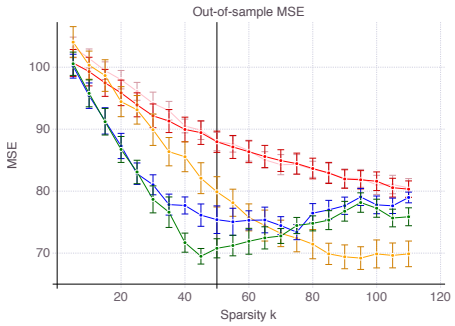
Fig 4: Number of true features TF vs. number of false features FF for the CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets with a fixed n.



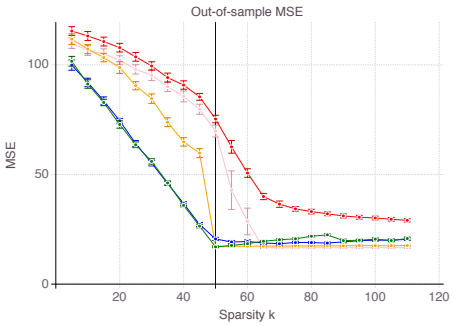
(a) Low noise, low ρ, $n = 900$



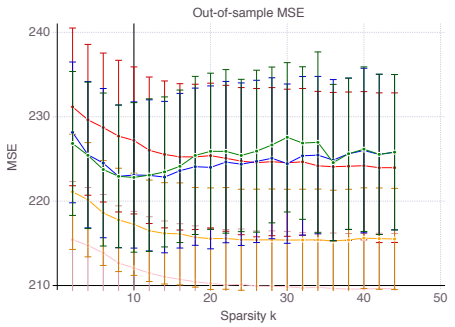
(b) Low noise, high ρ, $n = 900$



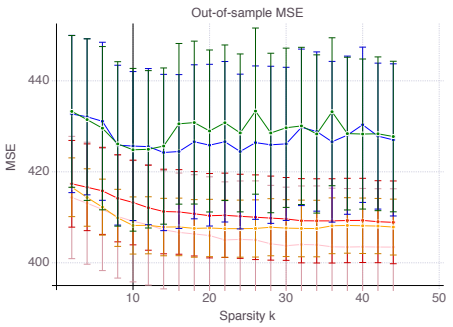
(c) Medium noise, low ρ, $n = 1,100$



(d) Medium noise, high ρ, $n = 1,100$

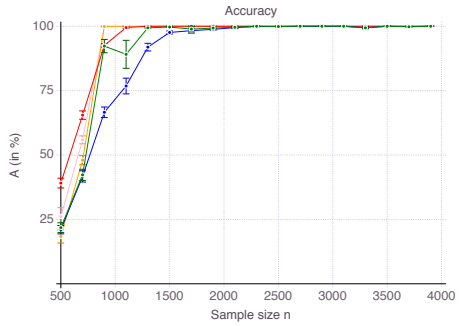


(e) High noise, low ρ, $n = 3,500$

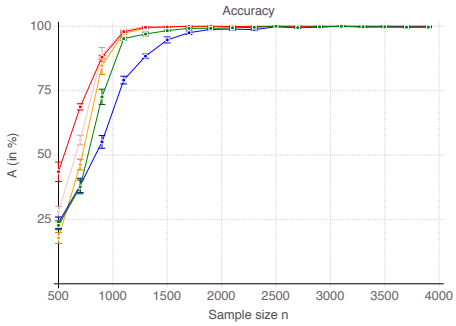


(f) High noise, high ρ, $n = 3,500$

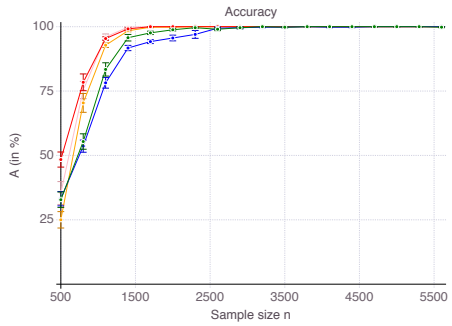
Fig 5: Out-of-sample mean square error as k increases, for the CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets with a fixed n.



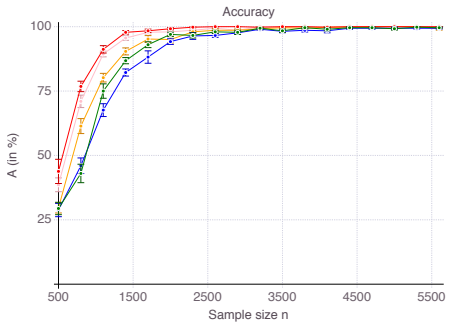
(a) Low noise, low correlation



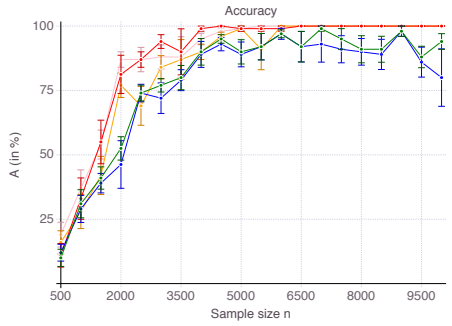
(b) Low noise, high correlation



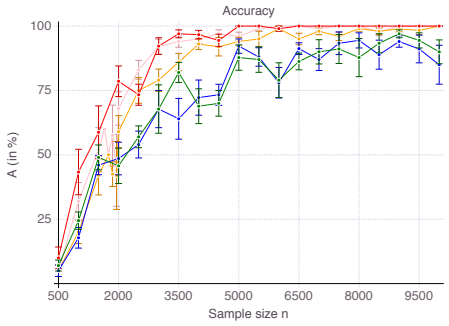
(c) Medium noise, low correlation



(d) Medium noise, high correlation

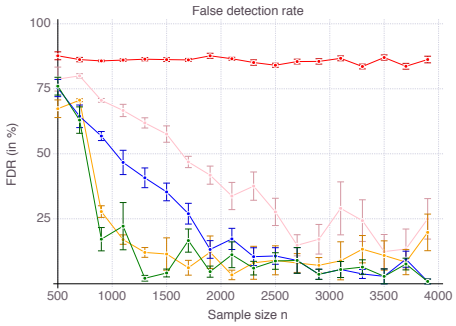


(e) High noise, low correlation

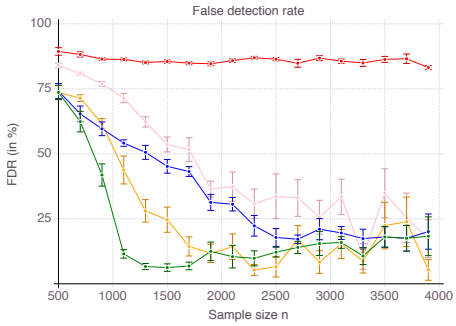


(f) High noise, high correlation

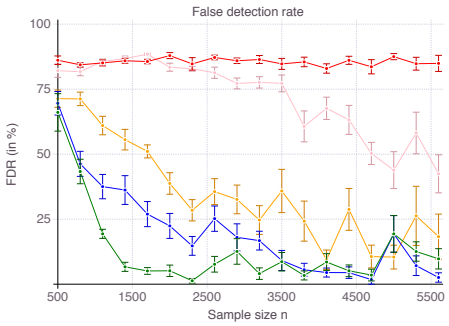
Fig 6: Accuracy A as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.



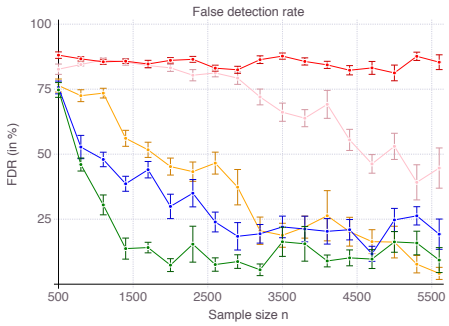
(a) Low noise, low correlation



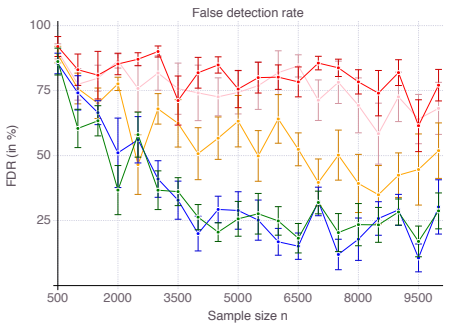
(b) Low noise, high correlation



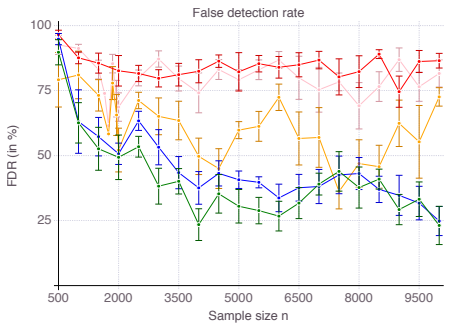
(c) Medium noise, low correlation



(d) Medium noise, high correlation



(e) High noise, low correlation



(f) High noise, high correlation

Fig 7: False detection rate FDR as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.

3.4.1 Feature selection with a given support size We first consider the case when the cardinality k of the support to be returned is given and equal to the true sparsity k_{true}. In this setting, ℓ_1-estimators are expected to always return at least 1 incorrect feature, while MCP and SCAD will provably recover the entire support [33].

As shown on Figure 8 (p. 25), we observe empirically what theory dictates: The accuracy of ENet reaches a threshold strictly lower than 1. Non-convex penalties MCP and SCAD, on the other hand, see their accuracy converging to 1 as n increases. Cardinality-constrained estimators CIO and SS, which are also non-convex, behave similarly, although no theory like Loh and Wainwright [33] exists, to the best of our knowledge. As far as accuracy is concerned (left panel), CIO dominates all other methods. Interestingly, while ENet is the least accurate in the limit $n \to +\infty$, it is sometimes more accurate than non-convex penalties for smaller values of n.

We report computational time in Appendix B.2.1.

3.4.2 Feature selection with cross-validated support size Behavior of the methods when k_{true} is no longer given and needs to be cross-validated from the data itself is very similar to the case where Σ satisfies the mutual incoherence condition. To avoid redundancies, we report those results in Appendix B.2.2.

3.5 Real-world design matrix $\boldsymbol{X}$

To illustrate the implications of feature selection on real-world applications, we consider an example from genomics. We collected data from The Cancer Genome Atlas Research Network[5] on $n = 1,145$ lung cancer patients. The data set consists of $p = 14,858$ gene expression data for each patient. We discarded genes for which information was only partially recorded so there is no missing data. We used this data as our design matrix $X \in \mathbb{R}^{n \times p}$ and generated synthetic noisy outputs Y, for 10 uniformely log-spaced values of SNR, as in Hastie et al. [29] (Table 4 p. 24). We held 15% of patients in a test set (171 patients). We used the remaining 974

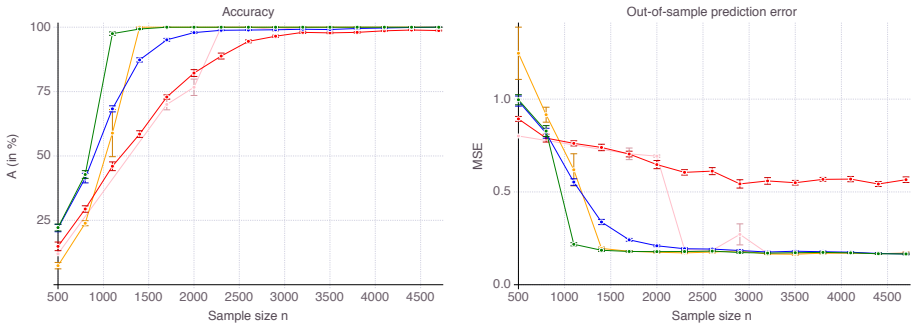
TABLE 4
Regimes of noise (SNR) considered in our regression experiments on the Cancer data set

SNR	0.05	0.09	0.14	0.25	0.42	0.71	1.22	2.07	3.52	6
PVE	0.05	0.08	0.12	0.20	0.30	0.42	0.55	0.67	0.78	0.86

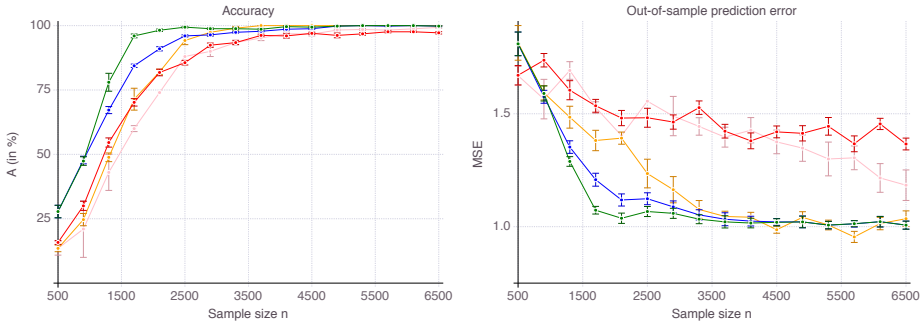
patients as a training and validation set. For each algorithm, we computed models with various degrees of sparsity and regularization on the training set, evaluated them on the validation set and took the most accurate model. Figure 9 (p. 26) represents the accuracy and false detection rate of the resulting regressor, for all methods, as SNR increases. ENet ranks the highest both in terms of number of true and false features. On the contrary, MCP is the least accurate, while making fewer incorrect guesses. CIO, SS, SCAD demonstrate in-between performance. Nevertheless, differences in feature selection does not translate into significant differences into predictive power in this case (see Figure 20 in Appendix B.3 page 43).

As previously mentioned, these results are the conclusion of a cross-validation procedure to find the right value of k. In Figure 10 (p. 27), we represent the

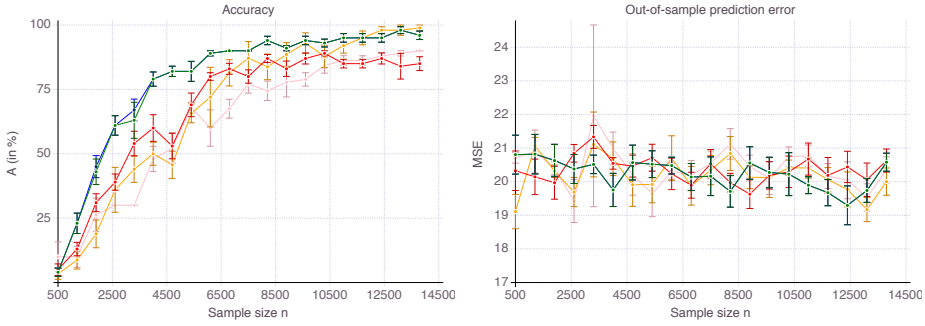
[5] http://cancergenome.nih.gov



(a) Low noise

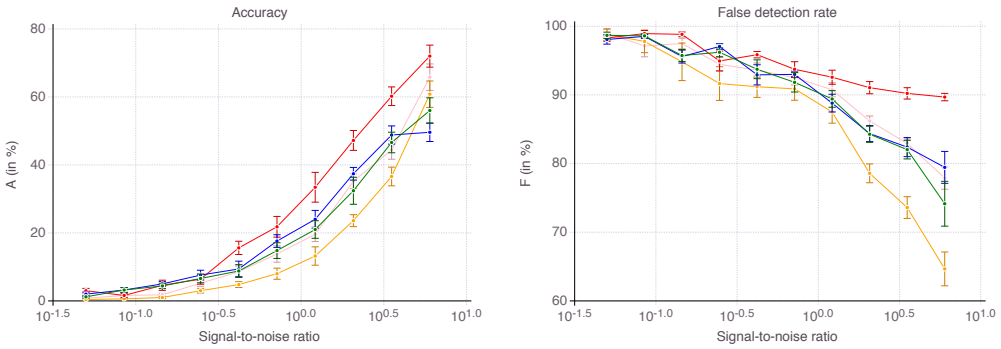


(b) Medium noise



(c) High noise

Fig 8: Accuracy (left panel) and out-of-sample mean square error (right panel) as n increases, for the CIO (in green), SS (in blue with $T_{max} = 150$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets with $n = 500, ..., 3700$, $p = 20,000$, $k_{true} = 100$.



(a) Accuracy (b) False detection rate

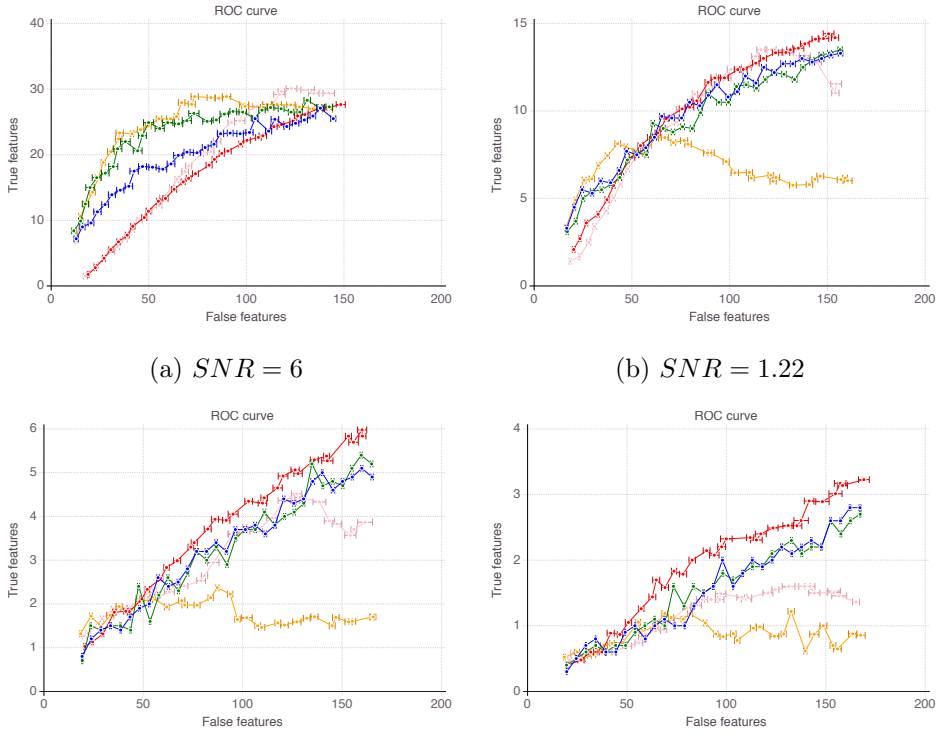
Fig 9: Accuracy and false detection as SNR increases, for the CIO (in green), SS (in blue with $T_{max} = 150$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets with $SNR = 0.05, ..., 6$, $k_{true} = 50$.

ROC curve corresponding to four of the ten regimes of noise. For low noise, ENet is dominated by SCAD, SS, MCP and CIO. As noise increases however, ENet gradually improves and even dominates all methods in very noisy regimes. These ROC curves are of little interest in practice, where true features are unknown - and potentially do not even exist. They raise, in our view, interesting research questions about the cross-validation procedure and its ability to efficiently select the "best" model.

3.6 Summary and guidelines

In this section, we compared five feature selection methods in regression, in various regimes of noise and correlation, and under different design matrices. Based on those extensive experiments, we can make the following observations:

- As far as accuracy is concerned, non-convex methods should be preferred over ℓ_1-regularization for they provide better feature selection, even in the absence of the mutual incoherence condition. In particular, MCP, the cutting-plane algorithm for the cardinality-constrained formulation, and its Boolean relaxation have been particularly effective in our experiments.
- In terms of false detection rate, cardinality-constrained formulations improve substantially over ENet and SCAD, and moderately over MCP.
- Computational time might still be the limiting factor in the use of such methods in practice. To that matter, publicly available software, such as the `ncvreg` package for SCAD and MCP estimators and our package `SubsetSelection` for the Boolean relaxation, should be advertised to practitioners since they compete with `glmnet`, which remains the gold standard for tractability. For time can be a crucial bottleneck in practice, we provide detailed experiments regarding computational time and scalability of the algorithms in Appendix B.1.
- In practice, we should recommend using a combination of all these methods: Lasso or ENet can be used as first feature screening/dimension reduction step, to be followed by a more computationally expensive non-convex feature selection method if time permits.



(a) $SNR = 6$ (b) $SNR = 1.22$

(c) $SNR = 0.25$ (d) $SNR = 0.05$

Fig 10: True features against false features, for the CIO (in green), SS (in blue with $T_{max} = 150$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.

- While Lasso/ENet performs poorly in low noise settings, its competes and sometimes dominates other methods as noise increases. This observation supports the view that ℓ_1-regularization is, first and foremost, a robustness story [3, 47]: Through shrinkage of the coefficients, the ℓ_1 penalty reduces variance in the estimator and improves out-of-sample accuracy, especially in presence of noise. Experiments by Hastie et al. [29] even suggested that Lasso outperforms cardinality-constrained estimators in high noise regimes. Our experiments suggest that their observations are still valid but less obvious as soon as the best subset selection estimator is regularized as well (with an ℓ_2 penalty in our case).

4. SYNTHETIC AND REAL-WORLD CLASSIFICATION PROBLEMS

In this section, we compare the five methods included in our study on classification problems. For implementation considerations, we use CIO and SS with the Hinge loss and ENet, MCP and SCAD with the logistic loss.

4.1 Methodology and metrics

Synthetic data is generated according to the same methodology as for regression, except that we now compute the signal Y according to

$$Y = \text{sign}(Xw_{true} + \varepsilon),$$

instead of $Y = Xw_{true} + \varepsilon$ previously.

On synthetic data, feature selection is assessed in terms of accuracy A and false detection rate FDR as in the previous section. Prediction accuracy, on the other hand, is assessed in terms of Area Under the Curve (AUC). The AUC corresponds to the area under the receiver operating characteristic curve, which represents true positive rate against false positive rate. The AUC ranges from 0.5 (for a completely random classifier) to 1. This area also corresponds to the probability that a randomly chosen positive example is correctly ranked with higher suspicion than a randomly chosen negative example. Correspondingly, $1 - AUC$ is a common measure of prediction error for real-world data.

4.2 Synthetic data satisfying mutual incoherence condition

In this section, we consider consider Toeplitz covariance matrix $\Sigma = \left(\rho^{|i-j|}\right)_{i,j}$, which satisfy mutual incoherence condition. We compare the performance of the methods in six different regimes of noise and correlation described in Table 5 (p. 29).

4.2.1 Feature selection with a given support size We first conducted experiments where the cardinality k of the support to be returned is given and equal to the true sparsity k_{true} for all methods. We report the results in Appendix C.1.1.

4.2.2 Feature selection with cross-validated support size We now compare the methods on cases where the support size needs to be cross-validated from data.

For every n, each method selects $k^\star$ features, some of which are in the true support, others being irrelevant, as measured by accuracy and false detection rate respectively. Figures 11 (p. 30) and 12 (p. 31) report the results of the cross-validation procedure for increasing n. In terms of accuracy (Figure 11), all methods increase in accuracy as n increases, although CIO and SS converge significantly

TABLE 5
Regimes of noise (SNR) and correlation (ρ) considered in our experiments on regression with Toeplitz covariance matrix

	Low correlation	High correlation
Low noise	$\rho = 0.2$ $SNR = 6$ $p = 10,000$ $k = 100$	$\rho = 0.7$ $SNR = 6$ $p = 10,000$ $k = 100$
Medium noise	$\rho = 0.2$ $SNR = 1$ $p = 5,000$ $k = 50$	$\rho = 0.7$ $SNR = 1$ $p = 5,000$ $k = 50$
High noise	$\rho = 0.2$ $SNR = 0.05$ $p = 1,000$ $k = 10$	$\rho = 0.7$ $SNR = 0.05$ $p = 1,000$ $k = 10$

slower than ENet, MCP and SCAD. However, this lower accuracy comes with the benefit of a strictly lower false detection rate (Figure 12).

4.3 Synthetic data *not* satisfying mutual incoherence condition

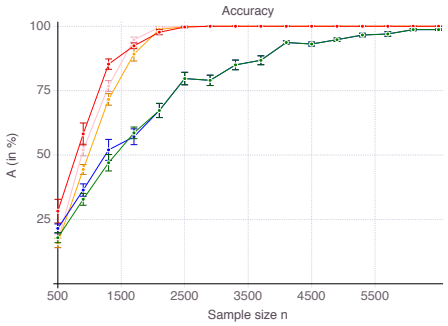
As for regression, we now consider the covariance matrix that does not satisfy mutual incoherence [33], in three regimes of noise (see Table 6 p. 29). We consider

TABLE 6
Regimes of noise (SNR) considered in our experiments on regression

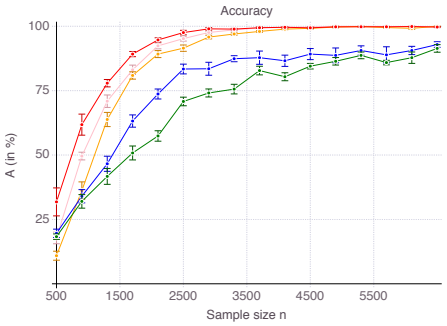
Low noise	$SNR = 6$ $p = 10,000$ $k = 100$
Medium noise	$SNR = 1$ $p = 5,000$ $k = 50$
High noise	$SNR = 0.05$ $p = 1,000$ $k = 10$

the case when the cardinality k of the support to be returned is given and equal to the true sparsity k_{true}.

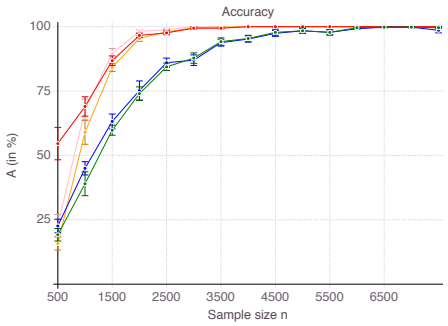
Results are shown on Figure 13 (p. 32) and corroborate our previous observations in the case of regression: the accuracy of ENet reaches a threshold strictly lower than 1. Non-convex penalties MCP and SCAD, on the other hand, will see their accuracy converging to 1, yet for a fixed n there are not necessarily more accurate than ENet. Cardinality-constrained estimators CIO and SS dominate all other methods, with a clear edge for CIO.



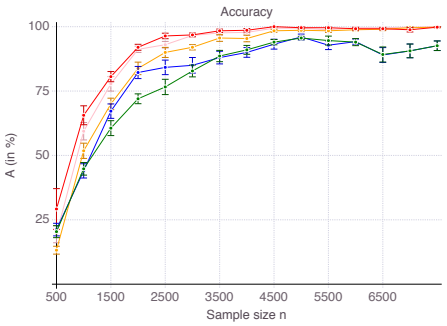
(a) Low noise, low correlation



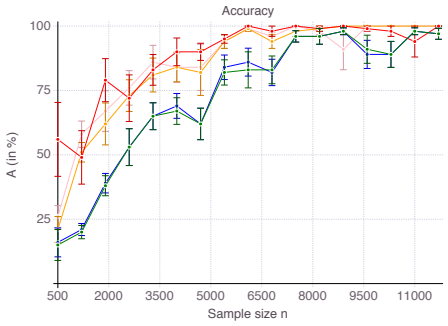
(b) Low noise, high correlation



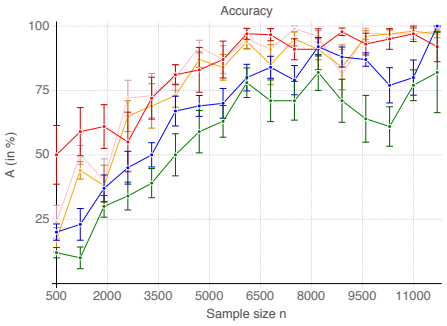
(c) Medium noise, low correlation



(d) Medium noise, high correlation

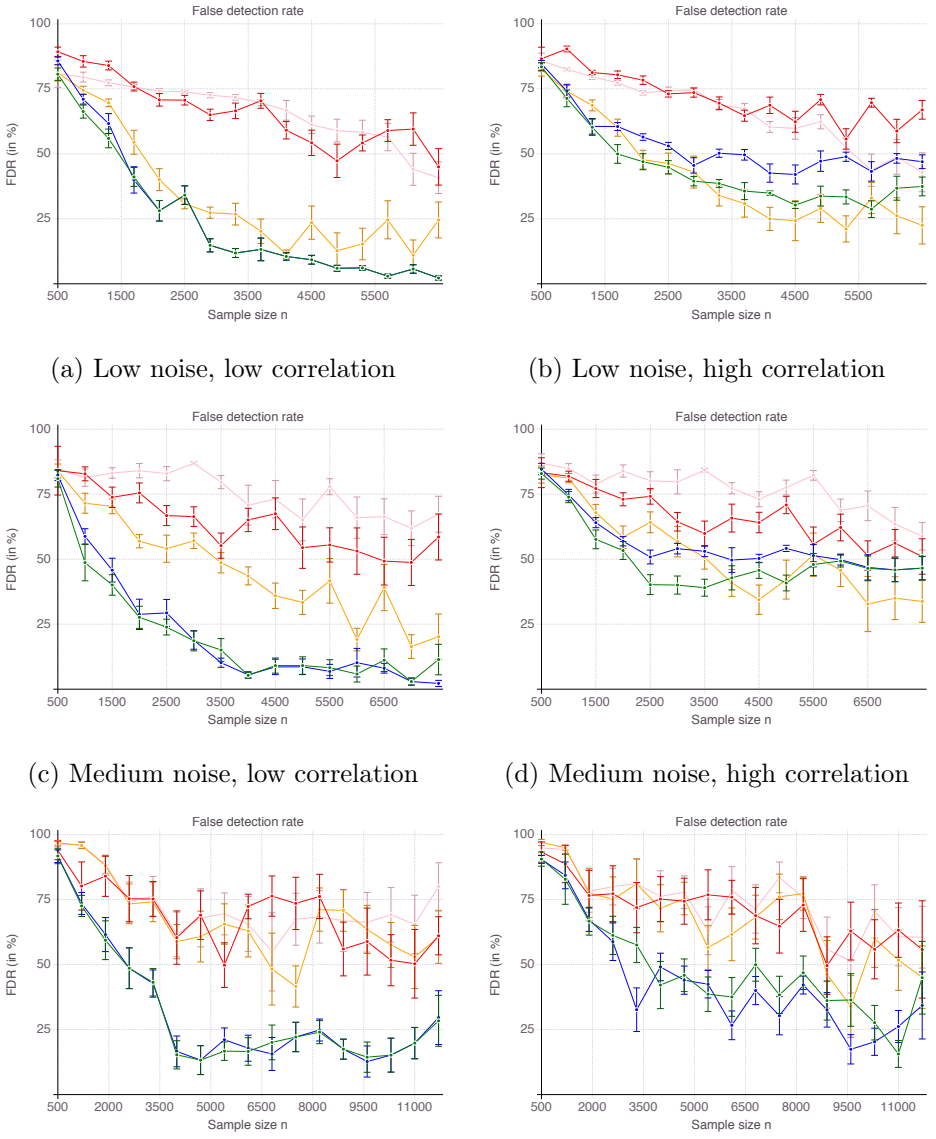


(e) High noise, low correlation



(f) High noise, high correlation

Fig 11: Accuracy A as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$) with Hinge loss, ENet (in red), MCP (in orange), SCAD (in pink) with logistic loss. We average results over 10 data sets.



(a) Low noise, low correlation

(b) Low noise, high correlation

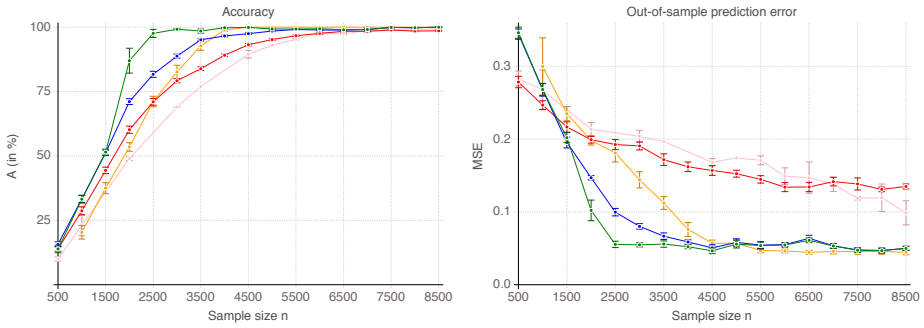
(c) Medium noise, low correlation

(d) Medium noise, high correlation

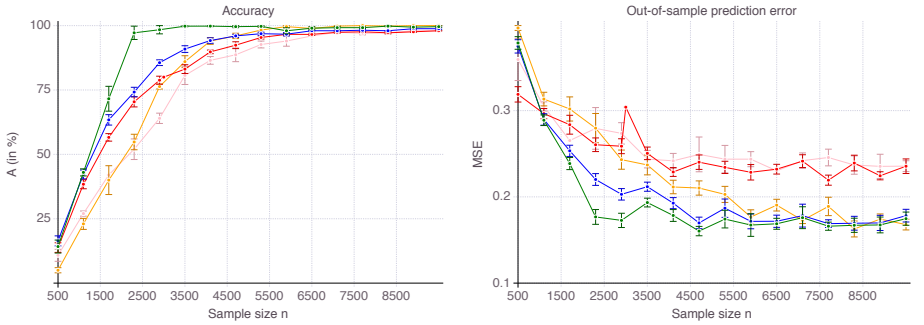
(e) High noise, low correlation

(f) High noise, high correlation

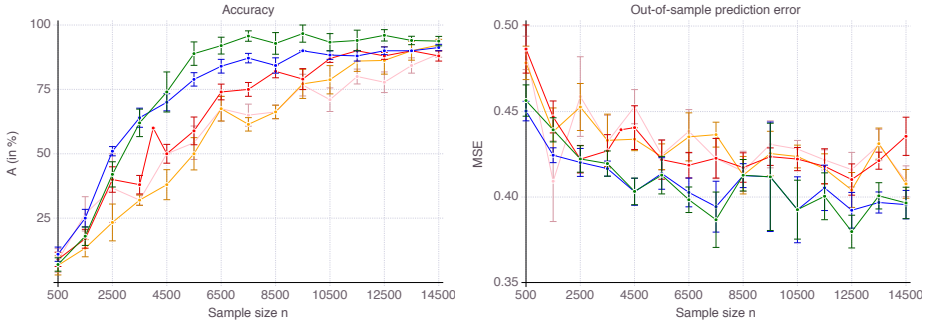
Fig 12: False detection rate FDR as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$) with Hinge loss, ENet (in red), MCP (in orange), SCAD (in pink) with logistic loss. We average results over 10 data sets.



(a) Low noise



(b) Medium noise



(c) High noise

Fig 13: Accuracy (left panel) and out-of-sample mean square error (right panel) as n increases, for the CIO (in green), SS (in blue with $T_{max} = 150$) with Hinge loss, ENet (in red), MCP (in orange), SCAD (in pink) with logistic loss. We average results over 10 data sets with $n = 500, ..., 3700$, $p = 20,000$, $k_{true} = 100$.

4.4 Real-world design matrix X

To illustrate the implications of feature selection on real-world applications, we re-consider the example from genomics we introduced in the previous section. Our $n = 1,145$ lung cancer patients naturally divide themselves into two groups, corresponding to different tumor types. In our sample for instance, 594 patients (51.9%) suffered from Adenocarcinoma while the remaining 551 patients (48.1%) suffered from Squamous Cell Carcinoma, making our data amenable to a binary classification task. Our goal is to identify a genetic signature for each tumor type, which only involves a limited number of genes, to better understand the disease or narrow the search for potential treatment for instance. We held 15% of patients from both groups in a test set (171 patients). We used the remaining 974 patients as a training and validation set. For each algorithm, we computed models with various degrees of sparsity on the training set, evaluated them on the validation set and took the most accurate model. Table 7 reports the induced sparsity $k^\star$ and out-of-sample accuracy in terms of AUC on the test set for each models. Results correspond to the median values obtained over ten different training/validation splits. Compared to the regression cases, we now have a real-world design matrix X and real world signals Y as well.

TABLE 7
Median of the results on Lung Cancer data, over 10 *different training/validation set splits.*

Method	Sparsity $k^\star$	Out-of-sample AUC
Exact sparse	87.5	0.9798
Boolean relaxation	65	0.9821
Lasso	114	0.9814
ENet	398.5	0.9806
MCP	39	0.9741
SCAD	79.5	0.9752

The first conclusion to be drawn from our results is that the all the feature selection methods considered in this paper, including the convex integer optimization formulation, scale to sizes encountered in real-world impactful applications. MCP and SCAD provide the sparsest classifiers with median sparsity of 39 and 79.5 respectively. At the same time, they achieve the lowest prediction accuracy, which questions the relevance of the genes selected by those methods. The ℓ_1-based formulations, Lasso and ENet, reach an AUC above 0.98 with 114 and 398.5 genes respectively. In comparison, CIO and SS have similar accuracy while selecting respectively only 87 and 65 genes.

5. CONCLUSION

In this paper, we provided a unified treatment of methods for feature selection in statistics. We focused on five methods: the NP-hard cardinality-constrained formulation (3), its Boolean relaxation, ℓ_1-regularized estimators (Lasso and Elastic-Net) and two non-convex penalties, namely the smoothly clipped absolute deviation (SCAD) and minimax concave penalty (MCP).

In terms of statistical performance, we compared the methods based on two metrics: accuracy and false detection rate. A reasonable feature selection method should exhibit a two-fold convergence property: the accuracy and false detection rate should converge to 1 and 0 respectively, as the sample size increases. Jointly

observed, these two properties ensure the method selects all true features and nothing but true features.

Most of the literature on feature selection so far has focused solely on accuracy, from both a theoretical and empirical point of view. Indeed, on that matter, our observations match existing theoretical results. When mutual incoherence condition is satisfied, all five methods attain perfect accuracy, irrespective of the noise and correlation level. As soon as mutual incoherence fails to hold however, ℓ_1-regularized estimators do not recover all true features, while non-convex formulations do. In all our experiments, Lasso-based formulations are the least accurate and sensitive to correlation between features, while cardinality-constrained formulation and the MCP non-convex estimator are the most accurate. As far as accuracy is concerned, we observe a clear distinction between convex and non-convex penalties, which echoes in our opinion the distinction between robustness and sparsity. Robustness is the property of an estimator to demonstrate good out-of-sample predictive performance, even in noisy settings, and convex regularization techniques are known to produce robust estimators [3, 47]. When it comes to sparsity however, non-convex penalties are theoretically more appealing, for they do not require stringent assumptions on the data [33]. Because both properties should deserve attention, we believe - and observe - that the best approaches are those combining a convex and a non-convex component. The ℓ_1-regularization on its own is not sufficient to produce reliably accurate feature selection.

In real-world applications, false detection rate is at least as important as accuracy. We were able to observe a zero false detection rate for Lasso-based formulations only under the mutual incoherence condition and in low noise settings where $SNR \to \infty$. Otherwise, false detection rate remains strictly positive and stabilizes above 80% (we observed this behavior as early as for $SNR \leqslant 25$). False detection rate for non-convex penalties MCP and SCAD quickly drops as n increases, but remains strictly positive (around $15-30\%$) and highly volatile, even for large sample sizes. The exact sparse formulation is the only method in our study which clearly outperforms all other methods, in all settings, and both for regression and classification, with the lowest false detection rate. Its Boolean relaxation demonstrates a similar behavior but less acute, especially in classification. In our opinion, such an observation speaks in favor of formulations that explicitly constrain the number of features instead of using regularization to induce sparsity. In practice, one could use Lasso or non-convex penalties as a good feature screening method, that is to discard irrelevant features and reduce the dimensionality of the problem. Nonetheless, in order to select relevant features only, we highly recommend the use of cardinality-constrained formulation or its relaxation, depending on available computing resources. Table 8 (p. 35) summarizes the advantages and disadvantages we observed for each method.

Those observations would be of little use if the best performing method were neither scalable nor available to practitioners. To that end, we released the code of a cutting-plane algorithm which solves the exact formulation (3) in minutes for n and p in the $10,000$s. Though computationally expensive, this method requires only one to two orders of magnitude more time than other methods. We believe this additional computational cost is affordable in many applications and justified by the resulting improved statistical performance. For more time-sensitive applications, its Boolean relaxation provides a high-quality approximation. We proposed a scalable

sub-gradient algorithm to solve it and released our code in the `Julia` package `SubsetSelection`, which can compete with the `glmnet` implementation of Lasso in terms of computational time, while returning statistically more relevant features. With `SubsetSelection`, we hope to bring to the community an easy-to-use and generic feature selection tool, which addresses deficiencies of ℓ_1-penalization but scales to high-dimensional data sets.

TABLE 8
Summary of the advantages and disadvantages of each method.

Method	Pros and Cons
Exact sparse	(+) Very good A/FDR. Convergence robust to noise/correlation. (−) Commercial solver and extra computational time.
Boolean relaxation	(+) Good A/FDR. Convergence robust to noise/correlation. (−) Heuristic.
Lasso/ENet	(+) Whole regularization path at no extra cost. (−) FDR very sensitive to noise. A very sensitive to correlation.
MCP	(+) Excellent A. (−) Unstable FDR.
SCAD	(+) Very good A. (−) Unstable FDR. A sensitive to correlation.

APPENDIX A: EXTENSION OF THE SUB-GRADIENT ALGORITHM AND IMPLEMENTATION

From a theoretical point of view, the boolean relaxation of the sparse learning problem is often tight, especially in the presence of randomness or noise in the data, so that feature selection can be expressed as a saddle point problem with continuous variables only. This min-max formulation is easier to solve numerically than the original mixed-integer optimization problem (3) because the original combinatorial structure vanishes. Our proposed algorithm benefits from both tightness of the Boolean relaxation and integrality of optimal solutions. In this section, we present the `Julia` package `SubsetSelection` which competes with the `glmnet` implementation of Lasso in terms of computational time, while returning statistically more relevant features, in terms of accuracy but more significantly in terms of false discovery rate. With `SubsetSelection`, we hope to bring to the community an easy-to-use and generic feature selection tool, which addresses deficiencies of ℓ_1-penalization but scales just as well to high-dimensional data sets.

A.1 Cardinality-constrained formulation

In this paper, we have proposed a sub-gradient algorithm for solving the following boolean relaxation

$$\min_{s\in[0,1]^p\,:\,\mathbf{e}^\top s\leqslant k}\ \max_{\alpha\in\mathbb{R}^n}\ f(\alpha,s),$$

where

$$f(\alpha,s) := -\sum_{i=1}^{n}\hat{\ell}(y_i,\alpha_i) - \frac{\gamma}{2}\sum_{j=1}^{p} s_j\alpha^\top X_j X_j^\top\alpha$$

is a linear function in s and concave function in α. The function f depends on the loss function ℓ through its Fenchel conjugate $\hat{\ell}$. In this paper, we mainly focused

on OLS and logistic loss but the same methodology could be applied to any convex loss function. Indeed, the package `SubsetSelection` supports all loss functions presented in Table 9.

TABLE 9

Supported loss functions ℓ and their corresponding Fenchel conjugates $\hat{\ell}$ as defined in Theorem 1. The observed data $y \in \mathbb{R}$ for regression and $y \in \{-1, 1\}$ for classification. By convention, $\hat{\ell}$ equals $+\infty$ outside of its domain. The binary entropy function is denoted as $H(x) := -x \log x - (1-x)\log(1-x)$.

Method	Loss $\ell(y, u)$	Fenchel conjugate $\hat{\ell}(y, \alpha)$
Logistic loss	$\log\left(1 + e^{-yu}\right)$	$-H(-y\alpha)$ for $y\alpha \in [-1, 0]$
1-norm SVM - Hinge loss	$\max(0, 1-yu)$	$y\alpha$ for $y\alpha \in [-1, 0]$
2-norm SVM	$\frac{1}{2}\max(0, 1-yu)^2$	$\frac{1}{2}\alpha^2 + y\alpha$ for $y\alpha \leqslant 0$
Least Square Regression	$\frac{1}{2}(y-u)^2$	$\frac{1}{2}\alpha^2 + y\alpha$
1-norm SVR	$(\lvert y-u\rvert - \varepsilon)_+$	$y\alpha + \varepsilon\lvert\alpha\rvert$ for $\lvert\alpha\rvert \leqslant 1$
2-norm SVR	$\frac{1}{2}(\lvert y-u\rvert - \varepsilon)_+^2$	$\frac{1}{2}\alpha^2 + y\alpha + \varepsilon\lvert\alpha\rvert$

At each iteration, the algorithm updates the variable α by performing one step of projected gradient ascent with step size δ, and updates the support s by minimizing $f(\alpha, s)$ with respect to s, α being fixed. Since s satisfies $s \in [0, 1]^p$, $s^\top \mathbf{e} \leqslant k$, and f is linear in s, this partial minimization boils down to sorting the components of $(-\alpha^\top X_j X_j^\top \alpha)_{j=1,\dots,p}$ and selecting the k smallest. Pseudo-code is given in Algorithm 2.2.

A.2 Scalability

As experiments in Sections 3 and 4 demonstrated, our proposed algorithm and implementation provides an excellent approximation for the solution of the discrete optimization problem (3), while terminating in times comparable with coordinate descent for Lasso estimators for low values of γ. Table 10 reports some computational time of `SubsetSelection` for data sets with various values of n, p and k. The algorithm scales to data sets with $(n, p) = (10^5, 10^5)$s or $(10^4, 10^6)$s within a few minutes. More comparison on computational time are given in Appendix B.1.

TABLE 10

Computational time of SS with $T_{max} = 200$ for data sets with large values of n and p, $\gamma = 2\,p/k/\max_i \|x_i\|^2/n$. Due to the dimensionality of the data, computations where performed on 1 CPU with 250GB of memory. We provide the average computational time (and the standard deviation) over 10 experiments.

Loss function ℓ	n	p	k	time (in s)
Least Squares	10,000	100,000	100	12.90 (0.45)
Least Squares	50,000	100,000	100	28.45 (1.83)
Least Squares	10,000	500,000	100	33.00 (1.86)
Least Squares	10,000	500,000	500	43.00 (0.54)
Hinge Loss	10,000	100,000	100	37.26 (0.14)
Hinge Loss	50,000	100,000	100	160.73 (0.28)
Hinge Loss	10,000	500,000	100	157.09 (1.18)
Hinge Loss	10,000	500,000	500	59.74 (0.08)

A.3 Extension to cardinality-penalized formulation

Our proposed approach naturally extends to cardinality-penalized estimators as well, where the 0-pseudonorm is added as a penalization term instead of an explicit constraint. Let us consider the ℓ_2-regularized optimization problem

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^{n} \ell(y_i, w^\top x_i) + \frac{1}{2\gamma}\|w\|_2^2 + \lambda\|w\|_0, \tag{9}$$

which corresponds to the estimator (1) in the unregularized limit $\gamma \to +\infty$. Similarly introducing a binary variable s encoding the support of w, we get that the previous problem (9) is equivalent to

$$\min_{s\in\{0,1\}^p} \max_{\alpha\in\mathbb{R}^n} f(\alpha, s) + \lambda \mathbf{e}^\top s. \tag{10}$$

The new saddle point function $s \mapsto f(\alpha, s) + \lambda \mathbf{e}^\top s$ is still linear in s and concave in α. As before, its boolean relaxation

$$\min_{s\in[0,1]^p} \max_{\alpha\in\mathbb{R}^n} f(\alpha, s) + \lambda \mathbf{e}^\top s \tag{11}$$

is tight if the minimizer of $f(\alpha, s) + \lambda \mathbf{e}^\top s$ with respect to s is unique. More precisely, we prove an almost verbatim analogue of Theorem 2.

THEOREM 4. *The boolean relaxation* (11) *is tight if there exists a saddle point* $(\bar\alpha, \bar s)$ *such that the vector* $(\lambda - \frac{\gamma}{2}\bar\alpha^\top X_j X_j^\top \bar\alpha)_{j=1,\dots,p}$ *has non-zero entries.*

PROOF. The saddle-point problem (11) is a continuous convex/concave minimax problem and Slater's condition is satisfied so strong duality holds. Therefore, any saddle-point $(\bar\alpha, \bar s)$ must satisfy

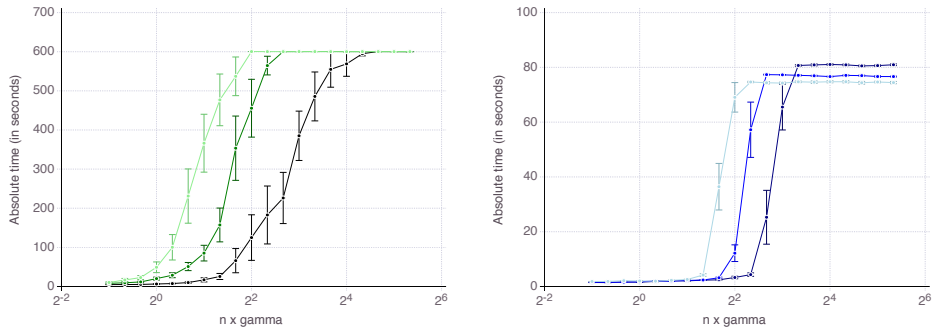
$$\bar\alpha \in \arg\max_{\alpha\in\mathbb{R}^n} f(\alpha, \bar s) + \lambda \mathbf{e}^\top s, \quad \bar s \in \arg\min_{s\in\in[0,1]^p} f(\bar\alpha, s) + \lambda \mathbf{e}^\top s.$$

If there exists $\bar\alpha$ such that $(\lambda - \frac{\gamma}{2}\bar\alpha^\top X_j X_j^\top \bar\alpha)_{j=1,\dots,p}$ has non-zero entries, then there is a unique $\bar s \in \arg\min_{s\in\in[0,1]^p} f(\bar\alpha, s) + \lambda \mathbf{e}^\top s$. In particular, this minimizer is binary and the relaxation is tight. □

This theoretical result suggests that the Lagrangian relaxation (11) can provide a good approximation for the combinatorial problem (9) in many cases. The same sub-gradient strategy as the one described in Algorithm 2.2 can be used to solve (11), with a slightly different partial minimization step: Now, minimizing $f(\alpha, s) + \lambda s^\top \mathbf{e}$ with respect $s \in [0,1]^p$ for a fixed α boils down to computing the components of $(\lambda - \gamma/2\alpha^\top X_j X_j^\top \alpha)_{j=1,\dots,p}$ and selecting the negative ones, which requires $O(np)$ operations. This strategy is also implemented in the package `SubsetSelection`.

APPENDIX B: NUMERICAL EXPERIMENTS FOR REGRESSION - SUPPLEMENTARY MATERIAL

In this section, we provide additional material for the simulations conducted in Section 3 on regression examples.



(a) CIO (b) SS

Fig 14: Absolute computational time as $n \times \gamma$ increases, for CIO (left panel), SS (right panel), with OLS loss. We fixed $p = 5,000$, $k_{true} = 50$, $SNR = 1$, $\rho = .5$ and $k = k_{true}$, and averaged results over 10 data sets. We report results for $n = 500$, $n = 1,000$ and $n = 2,000$ (from light to dark)

B.1 Synthetic data satisfying mutual incoherence condition

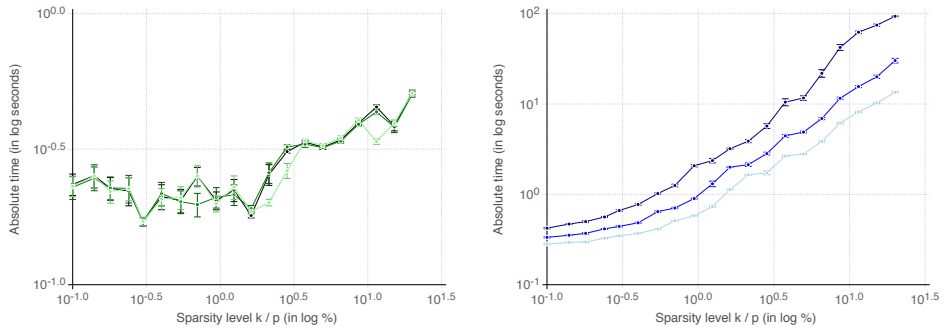
We first consider the case where the design matrix $X \sim \mathcal{N}(0, \Sigma)$, with Σ a Toeplitz matrix. In particular, Σ satisfies the so-called mutual incoherence condition required by Lasso estimators to be statistically consistent. In this setting, we provide more details about the computational time comparison between algorithms for sparse regression presented in Sections 3.

B.1.1 Impact of the hyper-parameters k and γ The discrete convex optimization formulation (3) and its Boolean relaxation (5) involve two hyper-parameters: the ridge penalty γ and the sparsity level k.

Intuition suggests that computational time would increase with γ. Indeed, when $\gamma \to 0$, $w^\star = 0$ is an obvious optimal solution, while for $\gamma \to +\infty$ the problem can become ill-conditioned. We generate 10 problems with $p = 5,000$, $k_{true} = 50$, $SNR = 1$ and $\rho = .5$ and various sample sizes n, fix $k = k_{true}$ and report absolute computational time as $n \times \gamma$ increases in Figure 14 (p. 38). For small values of γ, both methods terminate extremely fast - within 10 seconds for CIO and in less than 2 seconds for SS. As γ increases, computational time sharply increases. For CIO, we capped computational time to 600 seconds. For SS, we limited the number of iterations to $T_{max} = 200$. Regarding the sparsity k, the size of the feasible space $\{s \in \{0,1\}^p \ : \ s^\top \mathbf{e} \leqslant k\}$ grows as p^k. Empirically, we observe (Figure 15 p. 39) that computational time increases at most polynomially with k.

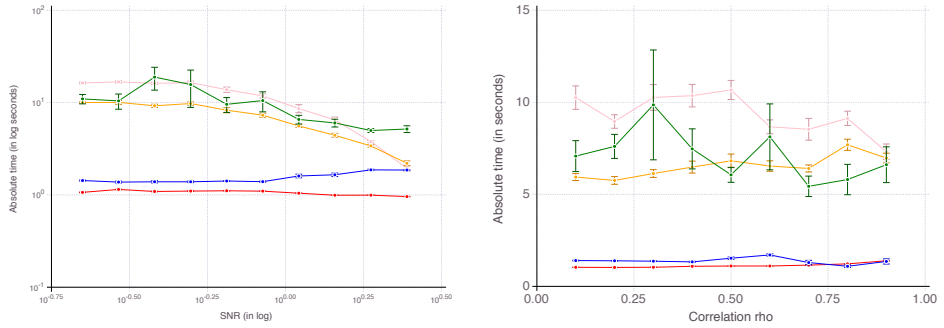
B.1.2 Impact of the signal-to-noise ratio, sample size n and problem size p As more signal becomes available, the feature selection problem should become easier and computational time should decrease. Indeed, in Figure 16 (p. 39), we observe that low SNR generally increases computational time for all methods (left panel). The correlation parameter ρ (right panel), however, does not seem to have a strong impact on computational time. In our opinion, with $SNR = 1$, the effect of correlation on computational time is second order compared to the impact of noise.

Figure 17 (p. 40) represents computational for increasing p, n/p being fixed



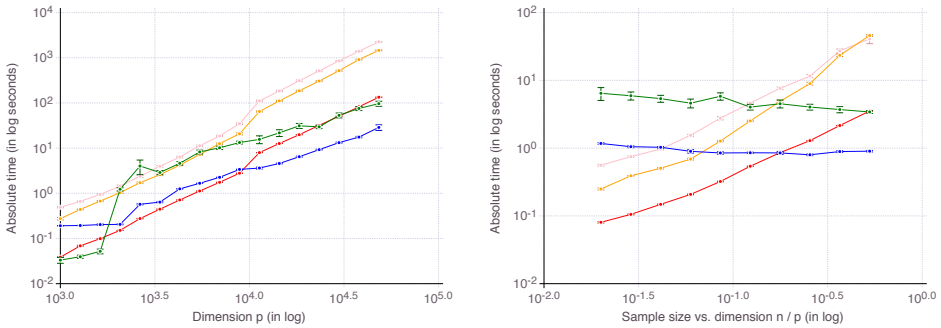
(a) CIO (b) SS

Fig 15: Absolute computational time as k increases from 5 to $1,000$, for CIO (left panel), SS (right panel), with OLS loss. We fixed $p = 5,000$, $k_{true} = 50$, $SNR = 1$, $\rho = .5$ and $n = 1,000$, and averaged results over 10 data sets. We report results for $\gamma = 2^i \gamma_0$ with $i = 0, 2, 4$ (from light to dark) and $\gamma_0 = \dfrac{p}{n\, k_{true}\, \max_i \|x_i\|^2}$.



(a) SNR, for $\rho = .5$ (b) Correlation ρ, for $SNR = 1$

Fig 16: Absolute computational time as signal-to-noise or correlation increases, for CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We fixed $p = 5,000$, $k_{true} = 50$, and $n = 1,000$, and averaged results over 10 data sets. We report results of CIO and SS with $k = k_{true}$ and $\gamma = \dfrac{p}{2n\, k_{true}\, \max_i \|x_i\|^2}$.



(a) p, for $n/p = 1/5$ (b) Sample size n/p, for $p = 5,000$

Fig 17: Absolute computational time as dimension p or sample size n increases, for CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We fixed $p = 5,000$, $k_{true} = 50$ and averaged results over 10 data sets. We report results of CIO and SS with $k = k_{true}$ and $\gamma = \dfrac{p}{2n\, k_{true}\, \max_i \|x_i\|^2}$.

and for increasing n/p, p being fixed. As shown, all methods scale similarly with p (almost linearly), while CIO and SS are less sensitive to n/p than their competitors.

B.2 Synthetic data *not* satisfying mutual incoherence condition

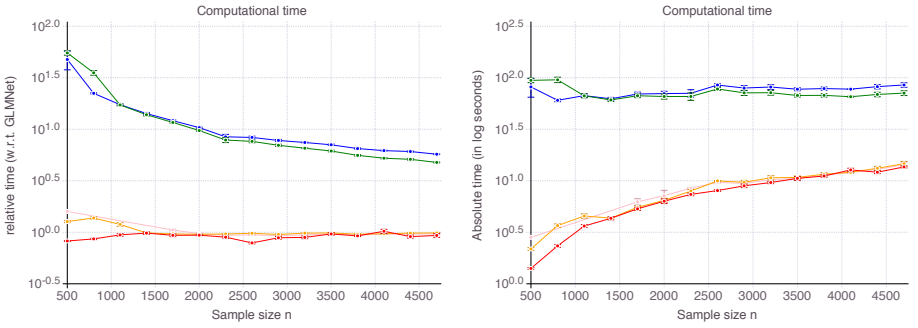
We now consider a covariance matrix Σ, which does not satisfy mutual incoherence, as proved in [33].

B.2.1 Feature selection with a given support size Figure 18 on page 41 reports relative compared to `glmnet` (left panel) and absolute (right panel) computational time in log scale. As for the case where mutual incoherence is satisfied, all methods terminates within a 10-100 factor with respect to `glmnet`.

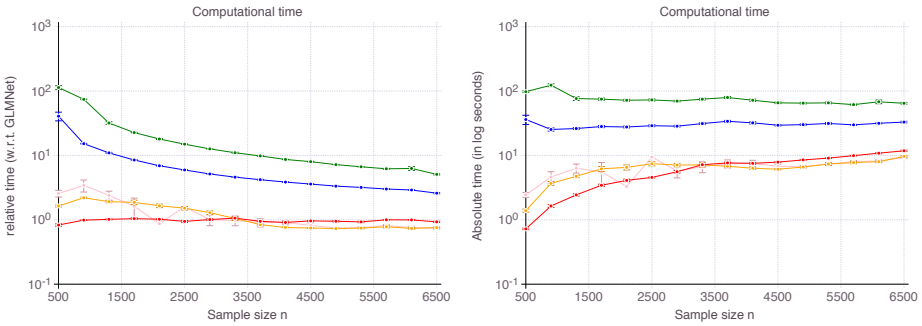
B.2.2 Feature selection with cross-validated support size We compare all methods when k_{true} is no longer given and needs to be cross-validated from the data itself. Figure 19 on page 42 reports the results of the cross-validation procedure for increasing n. In terms of accuracy (left panel), all four methods are relatively equivalent and demonstrate a clear convergence: $A \to 1$ as $n \to \infty$. On false detection rate however (right panel), behaviors vary among methods. Cardinality-constrained estimators achieve the lowest false detection rate ($0 - 30\%$), followed by MCP ($10 - 60\%$), SCAD ($20 - 70\%$) and then ENet (c.80%). In case of ENet, this behavior was expected, for ℓ_1-estimators are provably inconsistent, so that FDR must be positive when $A = 1$.

B.3 Real-world design matrix X

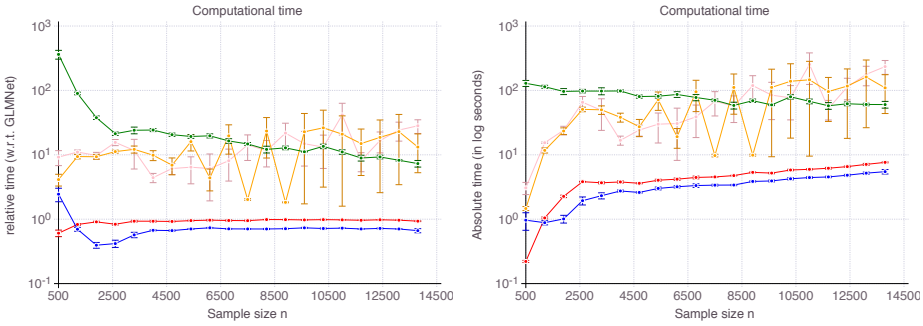
In this section, we consider a real-world design matrix X and generated synthetic noisy signals Y for 10 levels of noise. Figure 20 (p. 43) represents the out-of-sample MSE of all five methods as SNR increases. As mentioned, the difference in MSE between methods is far less acute than the difference observed in terms of accuracy and false detection.



(a) Low noise

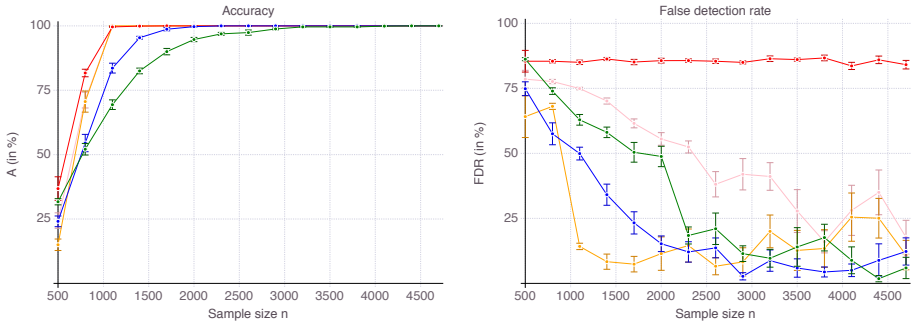


(b) Medium noise

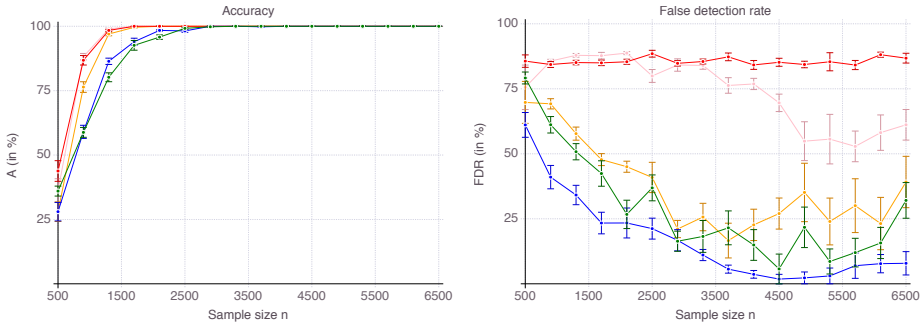


(c) High noise

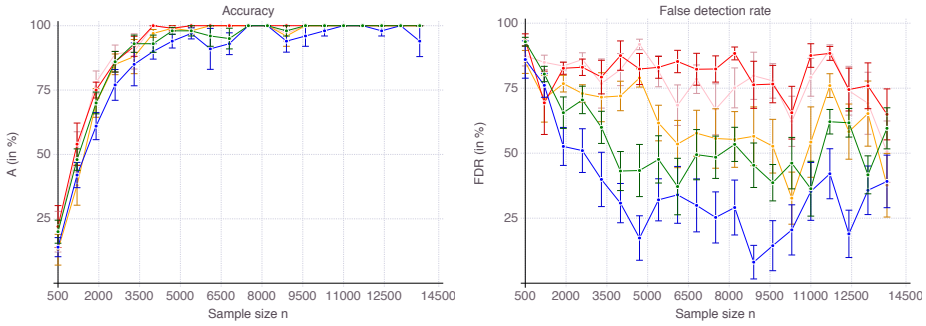
Fig 18: Relative (left panel) and absolute (right panel) computational timeas n increases, for CIO (in green), SS (in blue with $T_{max} = 200$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.



(a) Low noise



(b) Medium noise



(c) High noise

Fig 19: Accuracy A (left panel) and false detection rate FDR (right panel) as n increases, for the CIO (in green), SS (in blue with $T_{max} = 150$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets.

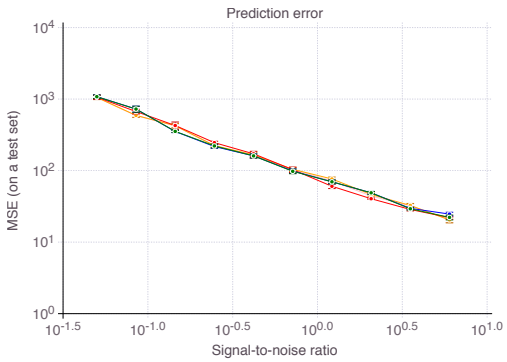


Fig 20: Out-of-sample MSE as SNR increases, for the CIO (in green), SS (in blue with $T_{max} = 150$), ENet (in red), MCP (in orange), SCAD (in pink) with OLS loss. We average results over 10 data sets with $SNR = 0.05, ..., 6$, $k_{true} = 50$.

APPENDIX C: NUMERICAL EXPERIMENTS FOR CLASSIFICATION - SUPPLEMENTARY MATERIAL

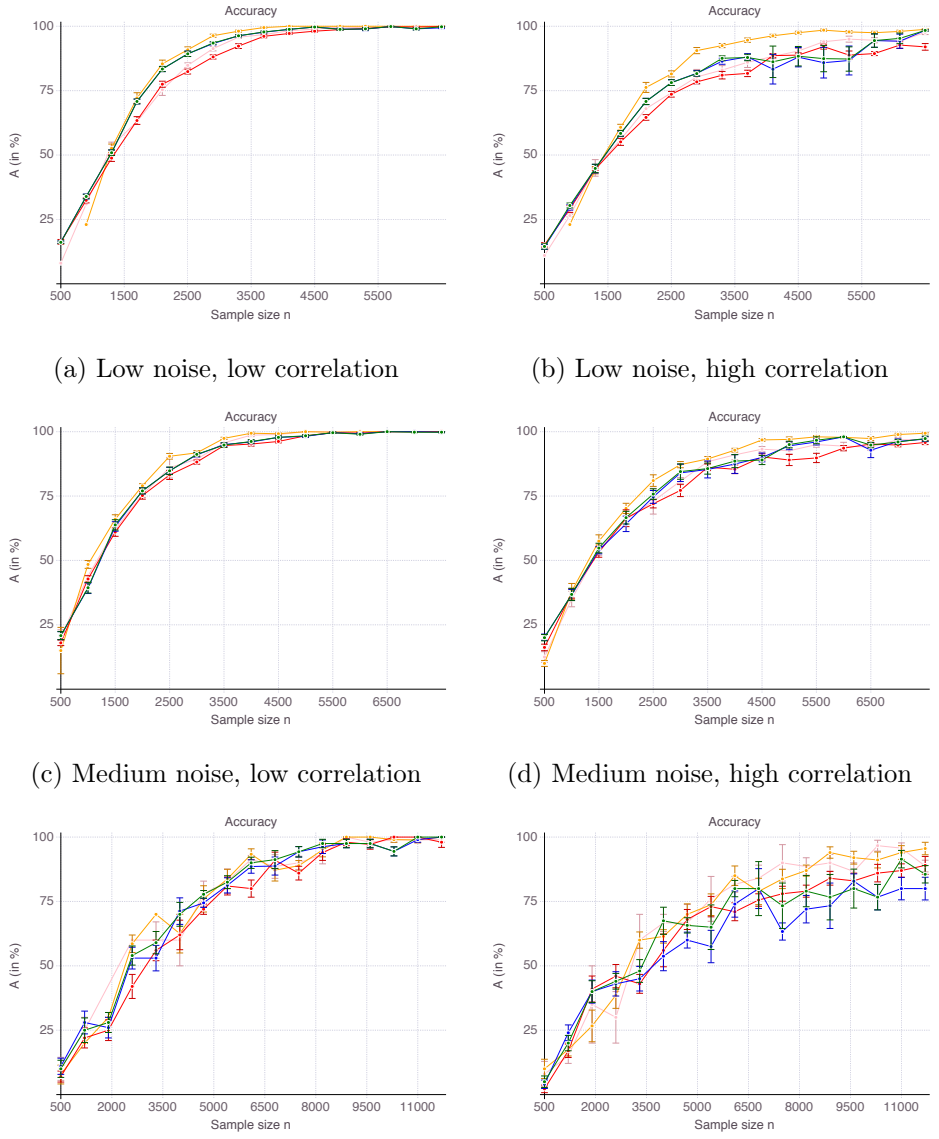
C.1 Synthetic data satisfying mutual incoherence condition

C.1.1 Feature selection with a given support size We first consider the case when the cardinality k of the support to be returned is given and equal to the true sparsity k_{true} for all methods.

As shown on Figure 21 (p. 44), all methods converge in terms of accuracy. That is their ability to select correct features as measured by A smoothly converges to 1 with an increasing number of observations $n \to \infty$. Compared to regression, the difference in accuracy between methods is much narrower. MCP now slightly dominates all methods, including CIO and SS. The suboptimality gap between the discrete optimization method and its Boolean relaxation appears to be much smaller as well and the two methods perform almost identically.

Figure 22 on page 45 reports relative computational time compared to `glmnet` in log scale. It should be kept in mind that we restricted the cutting-plane algorithm to a 180-second time limit and the sub-gradient algorithm to $T_{max} = 200$ iterations. `glmnet` is still the fastest method in general, but it should be emphasized that other methods terminate in times at most two orders of magnitude larger, which is often an affordable price to pay in practice. Combined with results in accuracy from Figure 21, such an observation speaks in favor of a wider use of cardinality-constrained or non-convex formulations in data analysis practice. As previously mentioned, for the sub-gradient algorithm, using an additional stopping criterion would drastically cut computational time (by a factor 2 at least) but would also deteriorate the quality of the solution significantly for such classification problems.

Figure 23 (p. 46) represents the out-of-sample error $1 - AUC$ for all five methods, as n increases, for the six noise/correlation settings of interest. There is a clear connection between performance in terms of accuracy and in terms of predictive power, with CIO performing the best. Still, better predictive power does not necessarily imply that the features selected are more accurate. As we have seen for instance, MCP often demonstrates the highest accuracy, yet not the highest AUC.



(a) Low noise, low correlation

(b) Low noise, high correlation

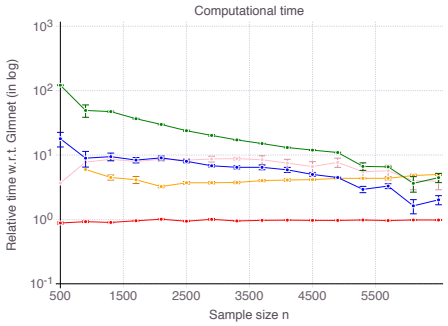
(c) Medium noise, low correlation

(d) Medium noise, high correlation

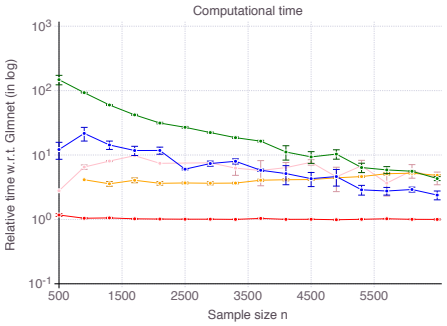
(e) High noise, low correlation

(f) High noise, high correlation

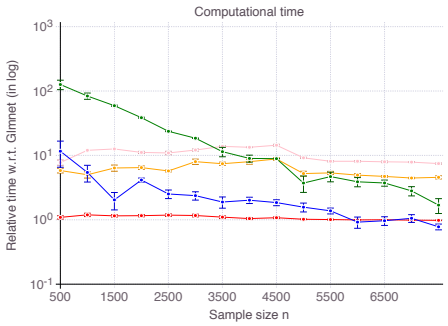
Fig 21: Accuracy as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$) with Hinge loss , ENet (in red), MCP (in orange), SCAD (in pink) with logistic loss. We average results over 10 data sets.



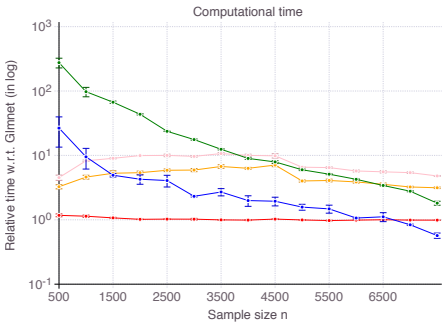
(a) Low noise, low correlation



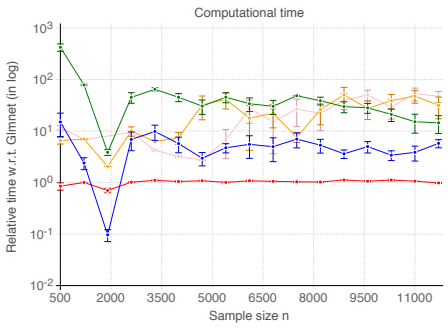
(b) Low noise, high correlation



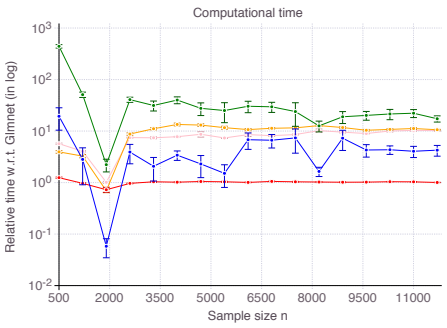
(c) Medium noise, low correlation



(d) Medium noise, high correlation

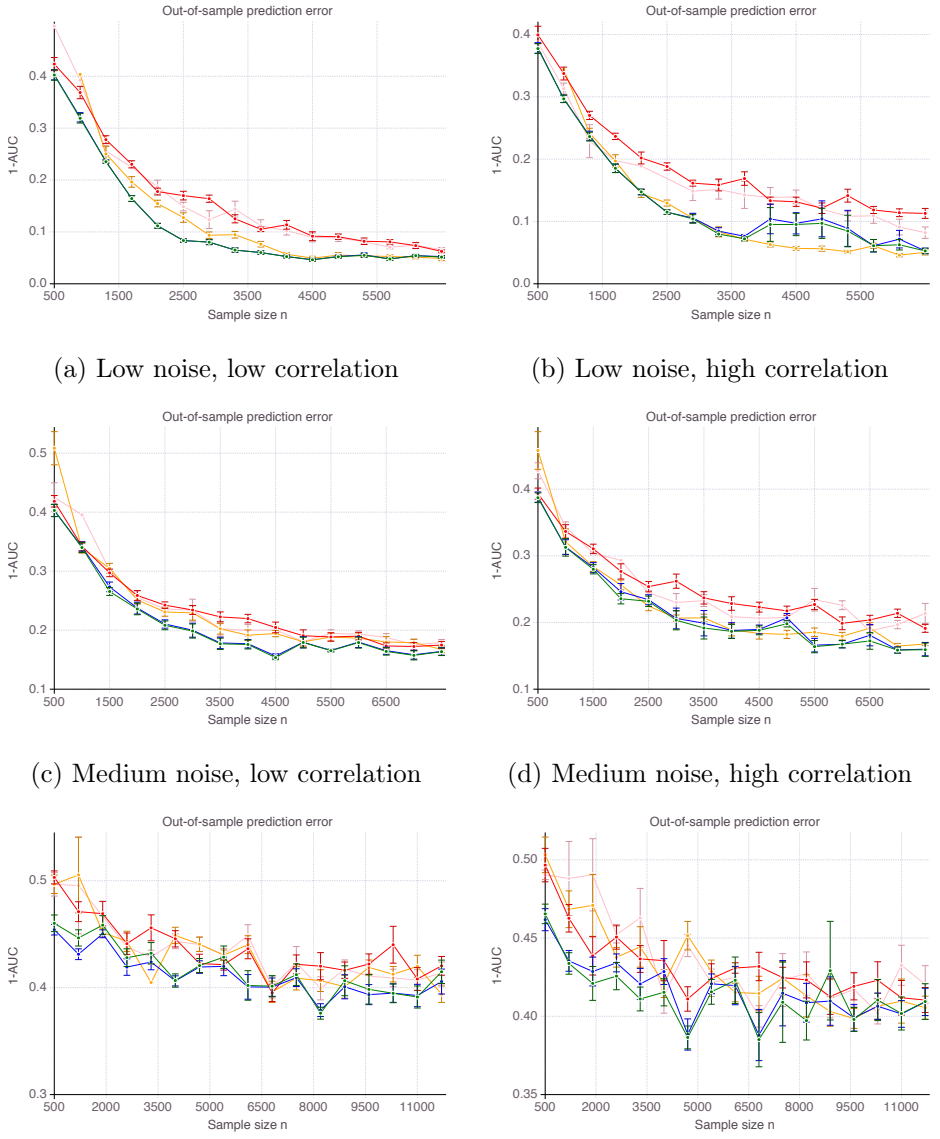


(e) High noise, low correlation



(f) High noise, high correlation

Fig 22: Computational time relative to Lasso with glmnet as n increases, for CIO (in green), SS (in blue with $T_{max} = 200$) with Hinge loss, ENet (in red), MCP (in orange), SCAD (in pink) with logistic loss. We average results over 10 data sets.



(a) Low noise, low correlation

(b) Low noise, high correlation

(c) Medium noise, low correlation

(d) Medium noise, high correlation

(e) High noise, low correlation

(f) High noise, high correlation

Fig 23: Out-of-sample $1 - AUC$ as n increases, for the CIO (in green), SS (in blue with $T_{max} = 200$) with Hinge loss, ENet (in red), MCP (in orange), SCAD (in pink) with logistic loss. We average results over 10 data sets.

REFERENCES

[1] 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.

[2] D. P. Bertsekas. *Nonlinear programming, 3rd edition*. Athena scientific Belmont, 2016.

[3] D. Bertsimas and A. Fertis. On the equivalence of robust optimization and regularization in statistics. Technical report, Massachusetts Institute of Technology, 2009. Working paper.

[4] D. Bertsimas and A. King. Logistic regression: From art to science. 2017. Statistical Science.

[5] D. Bertsimas and B. Van Parys. Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *under revision to Annals of Statistics*, 2017.

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

[7] D. Bertsimas, J. Pauphilet, and B. Van Parys. Sparse classification: a discrete optimization perspective. *under revision*, 2017.

[8] P. Breheny and J. Huang. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Annals of Applied Statistics*, 5(1):232–253, 2011.

[9] L. Breiman et al. Heuristics of instability and stabilization in model selection. *The annals of statistics*, 24(6):2350–2383, 1996.

[10] E. J. Candes, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

[11] B.-Y. Chu, C.-H. Ho, C.-H. Tsai, C.-Y. Lin, and C.-J. Lin. Warm start for parameter selection of linear classifiers. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 149–158. ACM, 2015.

[12] IBM ILOG CPLEX. V12. 1: User's manual for cplex. *International Business Machines Corporation*, 46(53):157, 2009.

[13] D. L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, 47(7):2845–2862, 2001.

[14] A. Dunning, J. Huchette, and M. Lubin. Jump: A modeling language for mathematical optimization. *arXiv preprint arXiv:1508.01982*, 2015.

[15] M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming*, 36(3):307–339, 1986.

[16] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[17] 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, 2001.

[18] J. Fan and R. Song. Sure independence screening in generalized linear models with np-dimensionality. *The Annals of Statistics*, 38(6):3567–3604, 2010.

[19] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, C.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008.

[20] J. Friedman, T. Hastie, H. Höfling, R. Tibshirani, et al. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

[21] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[22] J. Friedman, T. Hastie, and R. Tibshirani. GLMNet: Lasso and elastic-net regularized generalized linear models. r package version 1.9–5, 2013.

[23] G. M. Furnival and R. W. Wilson. Regressions by leaps and bounds. *Technometrics*, 16(4): 499–511, 1974.

[24] D. Gamarnik and I. Zadik. High-dimensional regression with binary coefficients. estimating squared error and a phase transition. *arXiv preprint arXiv:1701.04455*, 2017.

[25] H.-Y. Gao and A. G. Bruce. Waveshrink with firm shrinkage. *Statistica Sinica*, pages 855–874, 1997.

[26] I Gurobi Optimization. Gurobi optimizer reference manual; 2016. *URL http://www. gurobi. com*, 2016.

[27] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422, 2002.

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

[29] T. Hastie, R. Tibshirani, and R. J. Tibshirani. Extended comparisons of best subset selection,

forward stepwise selection, and the lasso. *arXiv preprint arXiv:1707.08692*, 2017.
[30] A. E. Hoerl and E. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.
[31] T. Iguchi, D. G. Mixon, J. Peterson, and S. Villar. On the tightness of an sdp relaxation of k-means. *arXiv preprint arXiv:1505.04778*, 2015.
[32] Ana Kenney, Francesca Chiaromonte, and Giovanni Felici. Efficient and effective l_0 feature selection. *arXiv preprint arXiv:1808.02526*, 2018.
[33] P.-L. Loh and M. J. Wainwright. Support recovery without incoherence: A case for nonconvex regularization. *The Annals of Statistics*, 45(6):2455–2482, 2017.
[34] M. Lubin and I. Dunning. Computing in operations research using Julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.
[35] S. G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41(12):3397–3415, 1993.
[36] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, pages 1436–1462, 2006.
[37] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.
[38] A. Nedić and A. Ozdaglar. Approximate primal solutions and rate analysis for dual subgradient methods. *SIAM Journal on Optimization*, 19(4):1757–1780, 2009.
[39] M. Pilanci, M. J. Wainwright, and L. El Ghaoui. Sparse learning via boolean relaxations. *Mathematical Programming*, 151(1):63–87, 2015.
[40] M. Sion. On general minimax theorems. *Pacific J. Math*, 8(1):171–176, 1958.
[41] W. Su, M. Bogdan, and E. Candes. False discoveries occur early on the lasso path. *arXiv preprint arXiv:1511.01957*, 2015.
[42] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Ser. B*, 58:267–288, 1996.
[43] M. J. Wainwright. Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting. *IEEE Transactions on Information Theory*, 55(12):5728–5741, 2009.
[44] M. J. Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (Lasso). *IEEE Transactions on Information Theory*, 55 (5):2183–2202, 2009.
[45] W. Wang, M. J. Wainwright, and K. Ramchandran. Information-theoretic limits on sparse signal recovery: Dense versus sparse measurement matrices. *IEEE Transactions on Information Theory*, 56(6):2967–2979, 2010.
[46] T. T. Wu and K. Lange. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, pages 224–244, 2008.
[47] H. Xu, C. Caramanis, and S. Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10(Jul):1485–1510, 2009.
[48] Y. Ye. Data randomness makes optimization problems easier to solve? 2016.
[49] C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942, 2010.
[50] P. Zhao and B. Yu. On model selection consistency of lasso. *Journal of Machine learning research*, 7(Nov):2541–2563, 2006.
[51] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.
[52] H. Zou and R. Li. One-step sparse estimates in nonconcave penalized likelihood models. *Annals of statistics*, 36(4):1509, 2008.