NODE-SCREENING TESTS FOR L0-PENALIZED LEAST-SQUARES PROBLEM WITH SUPPLEMENTARY MATERIAL

Théo Guyard[⋆] *Cédric Herzet*[†] *Clément Elvira*[‡]

[⋆] Univ Rennes, INSA Rennes, CNRS, IRMAR-UMR 6625, F-35000 Rennes, France
[†] INRIA Rennes-Bretagne Atlantique, Campus de Beaulieu, 35000 Rennes, France
[‡] SCEE/IETR UMR CNRS 6164, CentraleSupélec, 35510 Cesson Sévigné, France
firstname.lastname@{insa-rennes,inria,centralesupelec}.fr

ABSTRACT

We present a novel screening methodology to safely discard irrelevant nodes within a generic branch-and-bound (BnB) algorithm solving the ℓ_0-penalized least-squares problem. Our contribution is a set of two simple tests to detect sets of feasible vectors that cannot yield optimal solutions. This allows to prune nodes of the BnB exploration tree, thus reducing the overall solution time. One cornerstone of our contribution is a *nesting property* between tests at different nodes that allows to implement screening at low computational cost. Our work leverages the concept of safe screening, well known for sparsity-inducing convex problems, and some recent advances in this field for ℓ_0-penalized regression problems.

Index Terms— Sparse approximation, Mixed-integer problems, Branch and bound, Safe screening.

1. INTRODUCTION

Finding a sparse representation is a fundamental problem in the field of statistics, machine learning and inverse problems. It consists in decomposing some input vector $\mathbf{y} \in \mathbb{R}^m$ as a linear combination of a few columns (dubbed *atoms*) of a dictionary $\mathbf{A} \in \mathbb{R}^{m\times n}$. This task can be addressed by solving

$$\min_{\mathbf{x}\in\mathbb{R}^n} \ \tfrac{1}{2}\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_0 \tag{1}$$

where $\|\mathbf{x}\|_0$ counts the number of nonzero entries in $\mathbf{x}$ and $\lambda > 0$ is a tuning parameter. Unfortunately, problem (1) has proven to be NP-hard in the general case [1, Th. 3]. This has led researchers to develop sub-optimal procedures to approximate its solution and address large-scale problems. Among the most popular, one can mention greedy algorithms and methodologies based on convex relaxation, see *e.g.,* [2, Sec. 3.2 and Ch. 4].

Sub-optimal procedures are unfortunately only guaranteed to solve (1) under restrictive conditions that are rarely met when dealing with highly-correlated dictionaries. On the other hand, there has been recently a surge of interest for methods solving (1) exactly, see [3–7] to name a few. Many approaches leverage the fact that finding

$$\begin{aligned} p^\star \ = \min_{\mathbf{x}\in\mathbb{R}^n} & \tfrac{1}{2}\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_0 \\ \text{s.t. } & \|\mathbf{x}\|_\infty \leq M \end{aligned} \tag{P}$$

is equivalent to solve (1), provided that M is chosen large enough. Interestingly, (P) can be reformulated as a mixed-integer program (MIP) by introducing binary variables encoding the nullity of the entries of $\mathbf{x}$, see *e.g.*, [4]. Besides, (P) has been shown to be solvable for moderate-size problems by both commercial solvers [6] and tailored BnB algorithms [7].

In a recent paper, Atamtürk and Gómez extended the notion of safe screening introduced by El Ghaoui *et al.* in [8] from sparsity-promoting convex problems to non-convex ℓ_0-penalized problems. In particular, they introduced a new methodology that allows to detect (some of) the positions of zero and non-zero entries in the minimizers of a particular ℓ_0-penalized problem [9]. Their methodology is used as a preprocessing step of any algorithmic procedure and allows to reduce the problem dimensionality.

In this paper, we make one step forward in the development of numerical methods addressing large-scale ℓ_0-penalized problems by proposing *node-screening* rules that can be implemented *within* any BnB algorithm. In contrast to [9], we emphasize the existence of a *nesting property* between screening tests at different nodes. This enables to (potentially) prune multiple nodes at any step of the optimization process with marginal cost.

Our exposition is organized as follows. Section 2 gathers the main notations used in the paper. In Section 3, we describe a BnB algorithm tailored to (P). Section 4 presents our new node-screening tests and explains how to implement them efficiently within the BnB process. In Section 5, we assess the performance of our method on synthetic data. All the proofs are postponed to Appendix A.

The research presented in this paper is reproducible. Code and data are available at `https://gitlab.insa-rennes.fr/Theo.Guyard/bnb-screening`.

2. NOTATIONS

We use the following notational conventions throughout the paper. Boldface uppercase (*e.g.*, $\mathbf{A}$) and lowercase (*e.g.*, $\mathbf{x}$) letters respectively represent matrices and vectors. $\mathbf{0}$ denotes the all-zeros vector. Since the dimension will usually be clear from the context, it is omitted in the notation. The ith column of a matrix $\mathbf{A}$ is denoted $\mathbf{a}_i$. Similarly, the ith entry of a vector $\mathbf{x}$ is denoted x_i. Calligraphic letters (*e.g.,* $\mathcal{S}$) are used to denote sets and the notation $|\cdot|$ refers to their cardinality. If $\mathcal{S} \subseteq \{1, \dots, n\}$ and $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}_{\mathcal{S}}$ denotes to the restriction of $\mathbf{x}$ to its elements indexed by $\mathcal{S}$. Similarly, $\mathbf{A}_{\mathcal{S}}$ corresponds to the restriction of $\mathbf{A} \in \mathbb{R}^{m \times n}$ to its columns indexed by $\mathcal{S}$.

3. BRANCH-AND-BOUND PROCEDURES

Branch-and-bound procedures refer to an algorithmic solution to address MIPs, among others [10]. It consists in implicitly enumerating all feasible solutions and applying pruning rules to discard irrelevant candidates. When particularized to problem (P), it can be interpreted as a search tree where a new decision regarding the nullity of an entry of the variable $\mathbf{x}$ is taken at each node as illustrated in Figure 1a. Formally, we define a node as a triplet $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$ where : *i)* $\mathcal{S}_0$ and $\mathcal{S}_1$ contain the indices of the entries of $\mathbf{x}$ which are forced to be zero and non-zero, respectively; *ii)* $\bar{\mathcal{S}}$ gathers all the entries of $\mathbf{x}$ for which no decision has been taken at this stage of the decision tree.

The BnB algorithm reads as follows: starting from node $\nu = (\emptyset, \emptyset, \{1, \dots, n\})$, the method alternates between processing the current node (*bounding* step) and selecting a new node (*branching* step). The algorithm identifies the global minimum in a finite number of steps with a worst-case complexity equal to an exhaustive search. In practice, its efficiency depends on both the ability to process nodes quickly and the number of nodes processed. We review below specific choices for these two steps tailored to problem (P).

3.1. Bounding step

When processing node $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$, we prospect if any $\mathbf{x}$ with zeros on $\mathcal{S}_0$ and nonzero values on $\mathcal{S}_1$ can yield a global minimizer of (P). To that end, we look for the smallest objective value achieved by feasible candidates with respect to these constraints, *i.e.*, the value of

$$p^\nu = \min_{\mathbf{x} \in \mathbb{R}^n} \tfrac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}_{\bar{\mathcal{S}}}\|_0 + \lambda|\mathcal{S}_1| \quad \text{s.t. } \|\mathbf{x}\|_\infty \leq M \text{ and } \mathbf{x}_{\mathcal{S}_0} = \mathbf{0}. \tag{P^ν}$$

One verifies that $p^\nu \geq p^\star$ since all minimizers of (P^ν) are also feasible for (P). Moreover, equality holds whenever the constraints given by $\mathcal{S}_0$ and $\mathcal{S}_1$ match the configuration of one of the minimizers of (P).

If $p^\nu > p^\star$, ν can be pruned from the tree since no vector $\mathbf{x}$ verifying the constraints defined at this node (and therefore at any sub-nodes) can yield an optimal solution. Unfortunately, the latter pruning rule is of poor practical interest since $p^\star$ is not available. Moreover evaluating p^ν when $\bar{\mathcal{S}} \neq \emptyset$ is also a combinatorial problem. To circumvent these problems, a relaxed version of the rule is devised: let p_l^ν and p_u be respectively lower and upper bounds on p^ν and $p^\star$. Then, a sufficient condition to prune node ν reads $p_l^\nu > p_u$.

To obtain p_l^ν, a standard approach is to consider the following relaxation of (P^ν):

$$p_l^\nu = \min_{\mathbf{x} \in \mathbb{R}^n} \tfrac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \tfrac{\lambda}{M}\|\mathbf{x}_{\bar{\mathcal{S}}}\|_1 + \lambda|\mathcal{S}_1| \quad \text{s.t. } \|\mathbf{x}\|_\infty \leq M \text{ and } \mathbf{x}_{\mathcal{S}_0} = \mathbf{0} \tag{P_l^ν}$$

where the ℓ_0-penalty term has been replaced by an ℓ_1-norm. Interestingly, (P_l^ν) is a constrained LASSO problem [11] and can be solved (to machine precision) in polynomial time by using one of the many methods suitable for this class of problems [12, 13].

The upper bound p_u can be obtained by keeping track of the best known objective value of (P) during the BnB process. More precisely, we construct a feasible solution of (P^ν) at each node to obtain p_u^ν. The best known upper bound is then updated as $p_u \leftarrow \min(p_u, p_u^\nu)$. During the BnB procedure, p_u converges toward $p^\star$ and relaxation are strengthened as new variables are fixed, allowing to prune nodes more efficiently.

3.2. Branching step

If node $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$ has not been pruned during the bounding step, the tree exploration goes on. An index $i \in \bar{\mathcal{S}}$ is selected according to some *branching rule* and two direct sub-nodes are created below ν by imposing either "$x_i = 0$" or "$x_i \neq 0$". Finally, when all nodes have been explored or pruned, the BnB algorithm stops and any candidate yielding the best upper bound p_u is a minimizer of (P).

4. NODE-SCREENING TESTS

In this section, we present our new node-screening tests. They aim at identifying, with marginal cost, nodes of the search tree that *cannot* yield a global minimum of (P).

4.1. Dual properties

The crux of our procedure is a connection between the Fenchel dual problem of (P_l^ν) at two consecutive nodes. Prior to expose this result, let us define for all $\mathbf{u} \in \mathbb{R}^m$ and $i \in \{1, \dots, n\}$ three families of *pivot values* as

$$\begin{aligned} \boldsymbol{\gamma}_i(\mathbf{u}) &\triangleq M(|\mathbf{a}_i^{\mathrm{T}}\mathbf{u}| - \tfrac{\lambda}{M}) \\ \boldsymbol{\gamma}_i^0(\mathbf{u}) &\triangleq M[|\mathbf{a}_i^{\mathrm{T}}\mathbf{u}| - \tfrac{\lambda}{M}]_+ \\ \boldsymbol{\gamma}_i^1(\mathbf{u}) &\triangleq M[\tfrac{\lambda}{M} - |\mathbf{a}_i^{\mathrm{T}}\mathbf{u}|]_+ \end{aligned} \tag{2}$$

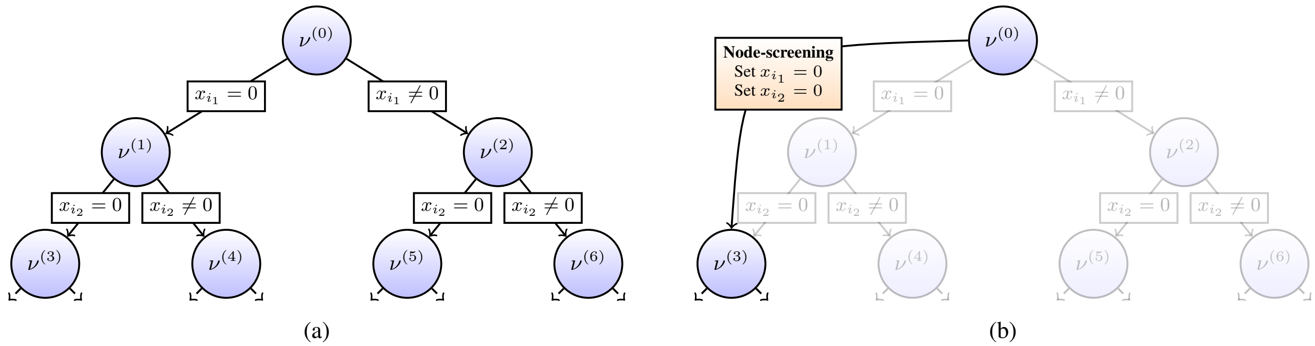

Fig. 1: First nodes of the BnB tree. The root node $\nu^{(0)}$ corresponds to problem (P) and each sub-node corresponds to a sub-problem (P^ν) with different fixed variables. (a) Standard implementation of a BnB where all nodes are processed. (b) Impact of applying node-screening tests on the BnB search tree: at node $\nu^{(0)}$, test (6b) is passed for entries i_1 and i_2; one can thus directly switch to the sub-node including constraints "$x_{i_1} = 0$" and "$x_{i_2} = 0$", namely $\nu^{(3)}$. The shaded nodes need not be explored.

where $[z]_+ \triangleq \max(0, z)$ for all scalars z. We now express the Fenchel dual of (P_l^ν) with respect to these quantities.

Proposition 1. *The Fenchel dual of (P_l^ν) is given by*

$$d_l^\nu = \max_{\mathbf{u}\in\mathbb{R}^m} \mathrm{D}_l^\nu(\mathbf{u}) \triangleq \tfrac{1}{2}\|\mathbf{y}\|_2^2 - \tfrac{1}{2}\|\mathbf{y}-\mathbf{u}\|_2^2 - \textstyle\sum_{i\in\bar{\mathcal{S}}} \boldsymbol{\gamma}_i^0(\mathbf{u}) - \sum_{i\in\mathcal{S}_1} \boldsymbol{\gamma}_i(\mathbf{u}) \tag{D_l^ν}$$

and strong duality holds for (P_l^ν)-(D_l^ν), i.e., $p_l^\nu = d_l^\nu$. Moreover, if $(\mathbf{x}_l^\star, \mathbf{u}_l^\star)$ is a couple of primal-dual solutions, one has

$$\mathbf{u}_l^\star = \mathbf{y} - \mathbf{A}\mathbf{x}_l^\star. \tag{3}$$

Hence, at a given node ν, the dual objective of (P_l^ν) is the sum of a term common to all nodes and some well-chosen pivot values. Interestingly, the dual objective functions between two consecutive nodes only differ from one pivot value as shown in the following result:

Corollary 1. *Let $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$ and $\ell \in \bar{\mathcal{S}}$, then $\forall \mathbf{u} \in \mathbb{R}^m$,*

$$\mathrm{D}_l^{\nu\cup\{x_\ell=0\}}(\mathbf{u}) = \mathrm{D}_l^\nu(\mathbf{u}) + \boldsymbol{\gamma}_\ell^0(\mathbf{u}) \tag{4a}$$

$$\mathrm{D}_l^{\nu\cup\{x_\ell\neq0\}}(\mathbf{u}) = \mathrm{D}_l^\nu(\mathbf{u}) + \boldsymbol{\gamma}_\ell^1(\mathbf{u}) \tag{4b}$$

where $\nu \cup \{x_\ell = 0\}$ and $\nu \cup \{x_\ell \neq 0\}$ denote the two direct *sub-nodes of ν where index ℓ has been swapped from $\bar{\mathcal{S}}$ to $\mathcal{S}_0$ or $\mathcal{S}_1$.*

4.2. Node-screening tests

We now expose our proposed screening strategy to identify nodes of the tree that provably cannot yield a global minimizer of (P). Let ν be a node and p_u an upper bound on $p^\star$. We have by strong duality between (P_l^ν)-(D_l^ν) that

$$\forall \mathbf{u} \in \mathbb{R}^m, \quad \mathrm{D}_l^\nu(\mathbf{u}) \leq d_l^\nu = p_l^\nu \leq p^\nu. \tag{5}$$

Hence, combining (5) with Corollary 1 leads to the next result:

Proposition 2. *Let $\ell \in \bar{\mathcal{S}}$ be some index. Then, $\forall \mathbf{u} \in \mathbb{R}^m$,*

$$\mathrm{D}_l^\nu(\mathbf{u}) + \boldsymbol{\gamma}_\ell^0(\mathbf{u}) > p_u \implies p^{\nu\cup\{x_\ell=0\}} > p^\star \tag{6a}$$

$$\mathrm{D}_l^\nu(\mathbf{u}) + \boldsymbol{\gamma}_\ell^1(\mathbf{u}) > p_u \implies p^{\nu\cup\{x_\ell\neq0\}} > p^\star. \tag{6b}$$

Stated otherwise, Proposition 2 describes a simple procedure to identify some sub-nodes of ν that cannot yield an optimal solution of (P). In particular, if both (6a) and (6b) pass for a given index, no feasible vector with respect to the constraints defined at ν is a minimizer of (P) (ν can therefore be pruned).

Our proposed procedure differs from the pruning methodology described in Section 3 in several aspects. Performing a test only requires the evaluation of one single inner product. They can therefore be implemented at marginal cost compared to the overall cost of the bounding step. Very importantly, they also inherit from the following *nesting property*:

Corollary 2. *Let $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$ be a node. Then, if test* (6a) *or* (6b) *passes for index $\ell \in \bar{\mathcal{S}}$, then it also passes for any sub-node $\nu' = (\mathcal{S}_0', \mathcal{S}_1', \bar{\mathcal{S}}') \subset \nu$ such that $\ell \in \bar{\mathcal{S}}'$.*

In particular, Corollary 2 has the following consequence. Assume that two distinct indexes ℓ and ℓ' pass test (6a) at node ν. Then index ℓ' will also pass (6a) at node $\nu \cup \{x_\ell \neq 0\}$. The same consequence holds for test (6b) and node $\nu \cup \{x_\ell = 0\}$. In other words, if several node-screening tests pass at a given node, one can *simultaneously* prune several sub-nodes of ν, as illustrated in Figure 1b. This is in contrast with the bounding procedure described in Section 3 which has to process each sub-node individually before applying any potential pruning operation.

4.3. Implementation considerations

Implementing node-screening tests described by Proposition 2 requires the knowledge of an upper bound p_u of $p^\star$ and a suitable $\mathbf{u} \in \mathbb{R}^m$, *i.e.,* as close as possible to the maximizer of (D_l^ν). The value of p_u can be obtained at no additional cost since the standard implementation of BnB already requires storing such valid upper bounds on $p^\star$. To obtain a relevant candidate for $\mathbf{u}$, we suggest the following procedure: assuming the method used to solve (P_l^ν) generates a sequence of iterates $\{\mathbf{x}^{(t)}\}_{t\in\mathbb{N}}$ that converges to a global minimizer, we set

$$\forall\, t \in \mathbb{N}, \quad \mathbf{u}^{(t)} = \mathbf{y} - \mathbf{A}\mathbf{x}^{(t)}. \tag{7}$$

This choice enjoys two desirable properties. First, the sequence $\{\mathbf{u}^{(t)}\}_{t\in\mathbb{N}}$ converges toward a maximizer of (D_l^ν) as a consequence of the optimality condition (3). Second, both $\mathbf{u}^{(t)}$ and $\mathbf{A}^{\mathrm{T}}\mathbf{u}^{(t)}$ are already evaluated by most solvers as they correspond to the residual error and the (negative) gradient of the least-squares term of the objective function, respectively. Thus, the computational cost of evaluating the screening tests at all undecided entries is marginal compared to the cost needed to address (P_l^ν). The latter choice for $\mathbf{u}^{(t)}$ suggests performing node-screening tests *during* the bounding step. Hence, when some entries passes the test at node ν, the BnB algorithm immediately switches to some sub-node, without solving (P_l^ν) at high precision.

5. NUMERICAL RESULTS

We finally report simulation results illustrating the relevance of the node-screening methodology on two different setups.

5.1. Experimental setups

For each trial, we generate a new realization of $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{y} \in \mathbb{R}^m$ as follows. In the *Gaussian* setup, we set $(m, n) = (500, 1000)$ and the entries of $\mathbf{A}$ are i.i.d. realizations of a normal distribution. In the *Tœplitz* setup, we set $(m, n) = (500, 300)$ and the first column of $\mathbf{A}$ contains a sampled sinc function . The other columns are obtained by shifting the entries in a row-wise fashion so that $\mathbf{A}$ inherits from a Tœplitz structure. In both setups, the columns of $\mathbf{A}$ are normalized to one. To obtain $\mathbf{y}$, we first sample a k-sparse vector $\mathbf{x}^0 \in \mathbb{R}^n$ with uniformly distributed non-zero entries. Each non-zero entry is set to $s(1 + |a|)$, where $s \in \{-1, +1\}$ is a random sign and a is a realization of the normal distribution. In our experiment, we chose $k \in \{5, 7, 9\}$. Finally, the observation is constructed as $\mathbf{y} = \mathbf{A}\mathbf{x}^0 + \boldsymbol{\epsilon}$ where the entries of $\boldsymbol{\epsilon}$ are i.i.d. realizations of a centered Gaussian distribution with standard deviation $\sigma = \|\mathbf{A}\mathbf{x}^0\|_2/\sqrt{10m}$. Such a design leads to a SNR of 10dB in average [14]. We tune λ statistically as in [15], *i.e.*, by setting $\lambda = 2\sigma^2 \log(n/k-1)$. Finally, we empirically set $M = 1.5\|\mathbf{A}^{\mathrm{T}}\mathbf{y}\|_\infty$ as advised in [7].

We compare three methods that address (P): *i)* `MIP`, that uses CPLEX [6], a state-of-the-art commercial MIP solver; *ii)* `BnB`, the algorithm presented in [7], which is (up to our knowledge) the fastest BnB algorithm tailored for (P); *iii)* `BnB+scr` which corresponds to `BnB` enhanced with the node-screening methodology presented in Section 4. Note that both `BnB` and `BnB+scr` leverage an efficient implementation of the Active-Set algorithm [16, Sec. 16.5] to solve (P_l^ν) and the branching rule described in [7, Sec. 2.2].

Experiments are run on a MacOS with an i7 CPU, clocked at 2.2 GHz and with 8Go of RAM. We restrict calculations on a single core to avoid bias due to parallelization capabilities. The Academic Version 20.1 of CPLEX is used and both `BnB` and `BnB+scr` are implemented in Julia v1.5 [17]. Results are averaged over 100 instances of problem (P).

5.2. Method comparison

Table 1 compares the average number of nodes treated (*i.e.,* the number of relaxations (P_l^ν) solved at machine precision) and the solution time for the three methods and all simulation setups. One observes that `BnB+scr` outperforms the two other methods on all scenarii and all figures of merit. We also note that, compared to `BnB`, the reduction in the solution time is more significant than the reduction in the number of processed nodes. A thorough examination of our results indicates that the bounding step is performed all the faster as many variables are fixed to zero in (P_l^ν). Our node-screening methodology allows to reach quickly nodes where the bounding step is performed with a lower computational cost.

As a final remark, we mention that `MIP` relies on an efficient C++ implementation while `BnB` and `BnB+scr` are implemented in Julia. As C++ usually runs faster than Julia, there is still room for improvements in the comparison of `BnB` and `BnB+scr` with `MIP`.

		`MIP`			`BnB`			`BnB+scr`		
	k	Nds	T	F	Nds	T	F	Nds	T	F
Gaussian	5	96	25.9	0	70	1.5	0	**56**	**0.7**	**0**
	7	292	60.8	0	180	5.1	0	**152**	**3.0**	**0**
	9	781	102.6	10	483	15.6	0	**412**	**9.8**	**0**
Tœplitz	5	1,424	10.2	0	965	6.4	0	**725**	**4.2**	**0**
	7	17,647	106.5	0	10,461	79.3	0	**7,881**	**52.2**	**0**
	9	80,694	353.4	50	47,828	346.4	48	**41,166**	**267.0**	**40**

Table 1: Number of nodes explored (Nds), solution time in seconds (T) and number of instances not solved within 1,000 seconds (F).

6. CONCLUSION

In this paper, we presented a novel node-screening methodology aiming at accelerating the resolution of the ℓ_0-penalized least-squares problem with a branch-and-bound solver. Our contribution leverages a nesting property between the node-screening tests at two consecutive nodes. Our method leads to significant improvements in terms of number of nodes explored and resolution time on two simulated datasets.

A. PROOFS

This appendix gathers the proofs of the results derived in Section 4. We refer to [7] for proofs regarding Section 3.

A.1. Proof of Proposition 1

Our proof of Proposition 1 leverages the Fenchel conjugate of a function f and denoted f^*. In particular, we first state the following technical lemma whose proof is postponed to the end of the section

Lemma 1. *Define for all* $\mathbf{z} \in \mathbb{R}^m$ *and* $\mathbf{x} \in \mathbb{R}^n$

$$f_1(\mathbf{z}) \triangleq \tfrac{1}{2}\|\mathbf{y} - \mathbf{z}\|_2^2 + \varepsilon \tag{8a}$$

$$f_2(\mathbf{x}) \triangleq \tfrac{\lambda}{M}\|\mathbf{x}_{\mathcal{S}}\|_1 + \eta_{\left\{\mathbf{x}' :\, \mathbf{x}'_{\mathcal{S}'} = \mathbf{0}\right\}}(\mathbf{x}) + \eta_{\mathcal{B}_\infty(M)}(\mathbf{x}) \tag{8b}$$

Then, for all $\mathbf{u} \in \mathbb{R}^m$ *and* $\mathbf{v} \in \mathbb{R}^n$

$$f_1^*(\mathbf{u}) = \tfrac{1}{2}(\|\mathbf{y} + \mathbf{u}\|_2^2 - \|\mathbf{y}\|_2^2) - \varepsilon \tag{9a}$$

$$f_2^*(\mathbf{v}) = \sum_{i \in \mathcal{S}} M[|v_i| - \tfrac{\lambda}{M}]_+ + M\|\mathbf{v}_{\mathcal{S}''}\|_1. \tag{9b}$$

In the latter result, ε *is a scalar,* $(\mathcal{S}, \mathcal{S}', \mathcal{S}'')$ *is a partition of* $\{1, \ldots, n\}$*,* $\mathcal{B}_\infty(M)$ *denotes the* ℓ_∞*-ball of* $\mathbb{R}^n$ *centered at* $\mathbf{0}$ *with radius* M*,* $\eta_{\mathcal{A}}$ *denotes the indicator function of set* $\mathcal{A} \subseteq \mathbb{R}^n$ *defined for all* $\mathbf{x} \in \mathbb{R}^n$ *by* $\eta_{\mathcal{A}}(\mathbf{x}) = 0$ *if* $\mathbf{x} \in \mathcal{A}$ *and* $+\infty$ *otherwise.*

We now use Lemma 1 to prove : *a)* the formulation of the dual problem (D_l^ν), *b)* the strong-duality relation linking (P_l^ν)-(D_l^ν) and *c)* the relation (3). In the sequel, we let $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$ be a given node.

a) Dual problem (D_l^ν)**.** Using the functions defined in Lemma 1 with $\varepsilon = \lambda|\mathcal{S}_1|$, $\mathcal{S} = \bar{\mathcal{S}}$, $\mathcal{S}' = \mathcal{S}_0$ and $\mathcal{S}'' = \mathcal{S}_1$, the relaxed problem (P_l^ν) can be rewritten as

$$\min_{\mathbf{x} \in \mathbb{R}^n} \; f_1(\mathbf{A}\mathbf{x}) + f_2(\mathbf{x}). \tag{10}$$

Then, the Fenchel dual of (10) is given by (see [18, Definition 15.19][1])

$$\max_{\mathbf{u} \in \mathbb{R}^m} \; -f_1^*(-\mathbf{u}) - f_2^*(\mathbf{A}^{\mathrm{T}}\mathbf{u}). \tag{11}$$

One concludes the proof by remarking that for all $\mathbf{u} \in \mathbb{R}^m$,

$$\begin{aligned} f_2^*(\mathbf{A}^{\mathrm{T}}\mathbf{u}) - \varepsilon &= \sum_{i \in \bar{\mathcal{S}}} \boldsymbol{\gamma}_i^0(\mathbf{u}) + M\left\|\mathbf{A}_{\mathcal{S}_1}^{\mathrm{T}}\mathbf{u}\right\|_1 - \tfrac{M}{M}\lambda|\mathcal{S}_1| \\ &= \sum_{i \in \bar{\mathcal{S}}} \boldsymbol{\gamma}_i^0(\mathbf{u}) + \sum_{i \in \mathcal{S}_1} \boldsymbol{\gamma}_i(\mathbf{u}). \end{aligned}$$

b) Strong duality. Strong duality holds as a consequence of [18, Proposition 15.24]. More particularly, one easily verifies that the condition in item *vii)* is fulfilled since the functions f_1 and f_2 are proper and continuous.

c) Relation (3)**.** Let $(\mathbf{x}_l^\nu, \mathbf{u}^\nu)$ be a couple of primal-dual solution of (P_l^ν)-(D_l^ν). Since strong duality holds, we have by item *ii)* of [18, Theorem 19.1] that[2]

$$-\mathbf{u}^\nu \in \partial f_1(\mathbf{A}\mathbf{x}_l^\nu) \tag{12}$$

where $\partial f_1(\mathbf{A}\mathbf{x}_l^\nu)$ denotes the sub-differential of f_1 evaluated at $\mathbf{A}\mathbf{x}_l^\nu$. One finally obtains (1) by noticing that f_1 is differentiable at $\mathbf{A}\mathbf{x}_l^\nu$ so that the sub-differential reduces to the singleton $\{\nabla f_1(\mathbf{A}\mathbf{x}_l^\nu)\}$.

A.2. Proof of Lemma 1.

The Fenchel conjugate of f_1 evaluated at $\mathbf{u} \in \mathbb{R}^m$ is defined as

$$f_1^*(\mathbf{u}) = \max_{\mathbf{z} \in \mathbb{R}^m} \; \mathbf{u}^{\mathrm{T}}\mathbf{z} - f_1(\mathbf{z}). \tag{13}$$

One easily sees that (13) is an unconstrained concave optimization problem whose objective function is differentiable. We then obtain (8a) by noticing that the maximum is attained at $\mathbf{z}^\star = \mathbf{y} + \mathbf{u}$.

Let $\mathbf{v} \in \mathbb{R}^n$. By definition of f_2^* and using the fact that $f_2(\mathbf{x}) = +\infty$ for all $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{x}_{\mathcal{S}'} \neq \mathbf{0}$, We have

$$\begin{aligned} f_2^*(\mathbf{v}) &= \max_{\left\{\mathbf{x} \in \mathbb{R}^n :\, \|\mathbf{x}\|_\infty \leq M\right\}} \mathbf{v}^{\mathrm{T}}\mathbf{x} - \tfrac{\lambda}{M}\|\mathbf{x}_{\mathcal{S}}\|_1 \\ &= \sum_{i \in \mathcal{S}} \max_{\{x \in \mathbb{R} :\, |x| \leq M\}} v_i x_i - \tfrac{\lambda}{M}|x_i| \\ &\quad + \max_{\left\{\mathbf{x}_{\mathcal{S}''} \in \mathbb{R}^{|\mathcal{S}''|} :\, \|\mathbf{x}_{\mathcal{S}''}\|_\infty \leq M\right\}} \mathbf{v}_{\mathcal{S}''}^{\mathrm{T}}\mathbf{x}_{\mathcal{S}''}. \end{aligned}$$

where the optimization problem can be split since it is linear and $\mathcal{S}, \mathcal{S}''$ have non empty intersection by definition. Let $i \in \mathcal{S}$ and consider the problem

$$x_i^\star \in \mathop{\arg\max}_{\{x \in \mathbb{R} :\, |x| \leq M\}} \; v_i x_i - \tfrac{\lambda}{M}|x_i|. \tag{14}$$

We let the reader check that $x_i^\star$ defined by

$$x_i^\star = \begin{cases} 0 & \text{if } |v_i| - \frac{\lambda}{M} \leq 0 \\ \mathrm{sign}(v_i)M & \text{otherwise} \end{cases} \tag{15}$$

is the maximizer of (14). One finally expresses the maximum as a sum of pivot values by injecting the value of $x_i^\star$ in the objective function.

[1] Note that, unlike [18, Definition 15.19], we use the formulation of the dual problem that involves a maximization problem.

[2] Again, we note that (12) involves the opposite of $\mathbf{u}^\nu$ since the authors consider the reformulation of the dual as a minimization problem.

Second, see that

$$\begin{aligned}
&\max_{\{\mathbf{x}_{\mathcal{S}''}\in\mathbb{R}^{|\mathcal{S}''|}:\ \|\mathbf{x}_{\mathcal{S}''}\|_\infty\le M\}} \mathbf{v}_{\mathcal{S}''}^{\mathrm{T}}\mathbf{x}_{\mathcal{S}''}\\
&\quad= M\times \max_{\{\mathbf{x}_{\mathcal{S}''}\in\mathbb{R}^{|\mathcal{S}''|}:\ \|\mathbf{x}_{\mathcal{S}''}\|_\infty\le 1\}} \mathbf{v}_{\mathcal{S}''}^{\mathrm{T}}\mathbf{x}_{\mathcal{S}''}\\
&\quad= M\|\mathbf{v}_{\mathcal{S}''}\|_1
\end{aligned}$$

where the last equality holds since one recognizes the Fenchel dual of the indicator function of the unit-ball with respect to the ℓ_∞-norm [18, item *iv)* in Example 13.3] in the penultimate line.

A.3. Proof of Corollary 1

Let $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$ be a node and i be some element of $\bar{\mathcal{S}}$ such that $\nu \cup \{x_i = 0\}$ defines a child node. Using (D_l^ν), we have for all $\mathbf{u} \in \mathbb{R}^m$

$$\begin{aligned}
\mathrm{D}_l^{\nu\cup\{x_i=0\}}(\mathbf{u}) - \mathrm{D}_l^{\nu}(\mathbf{u}) &= \sum_{i'\in\bar{\mathcal{S}}} \boldsymbol{\gamma}_{i'}^0(\mathbf{u}) - \sum_{i'\in\bar{\mathcal{S}}\setminus\{i\}} \boldsymbol{\gamma}_{i'}^0(\mathbf{u})\\
&= \boldsymbol{\gamma}_i^0(\mathbf{u}).
\end{aligned}$$

Similar calculations lead for all $\mathbf{u} \in \mathbb{R}^m$ to

$$\begin{aligned}
\mathrm{D}_l^{\nu\cup\{x_i\neq0\}}(\mathbf{u}) - \mathrm{D}_l^{\nu}(\mathbf{u}) &= \boldsymbol{\gamma}_i^0(\mathbf{u}) - \boldsymbol{\gamma}_i(\mathbf{u})\\
&= \boldsymbol{\gamma}_i^1(\mathbf{u})
\end{aligned}$$

where the last line uses the relation $[u]_+ - [-u]_+ = u$ that hold for all scalar u.

A.4. Proof of Corollary 2

Let $\nu = (\mathcal{S}_0, \mathcal{S}_1, \bar{\mathcal{S}})$ be a node. We first state the following lemma which can easily be proved by induction using (4a) and (4b).

Lemma 2. *Let $\nu' = (\mathcal{S}_0', \mathcal{S}_1', \bar{\mathcal{S}}')$ be a child node of ν. Then, for all $\mathbf{u} \in \mathbb{R}^m$*

$$\mathrm{D}_l^{\nu'}(\mathbf{u}) = \mathrm{D}_l^{\nu}(\mathbf{u}) + \sum_{i\in\mathcal{S}_0'\setminus\mathcal{S}_0} \boldsymbol{\gamma}_i^0(\mathbf{u}) + \sum_{i\in\mathcal{S}_1'\setminus\mathcal{S}_0} \boldsymbol{\gamma}_i^1(\mathbf{u}). \quad (16)$$

We now prove Corollary 2. Let $\ell \in \bar{\mathcal{S}}$ be some index such that test (6a) or (6b) passes at node ν. To easier our exposition, we only prove the result for test (6a) as the same rationale can be used for test (6b).

Hence, assume that test (6a) passes for index ℓ at node ν, *i.e.,* that

$$\mathrm{D}_l^{\nu}(\mathbf{u}) + \boldsymbol{\gamma}_\ell^1(\mathbf{u}) > p_u. \quad (17)$$

Let $\nu' = (\mathcal{S}_0', \mathcal{S}_1', \bar{\mathcal{S}}')$ be a child node of ν such that $\ell \notin \bar{\mathcal{S}}'$. Using Lemma 2, we have

$$\mathrm{D}_l^{\nu'}(\mathbf{u}) = \mathrm{D}_l^{\nu}(\mathbf{u}) + \sum_{i\in\mathcal{S}_0'\setminus\mathcal{S}_0} \boldsymbol{\gamma}_i^0(\mathbf{u}) + \sum_{i\in\mathcal{S}_1'\setminus\mathcal{S}_0} \boldsymbol{\gamma}_i^1(\mathbf{u}). \quad (18)$$

Since the pivots values $\{\boldsymbol{\gamma}_i^0(\mathbf{u})\}_{i=1}^n$ $\{\boldsymbol{\gamma}_i^1(\mathbf{u})\}_{i=1}^n$ are all nonnegative by construction, we necessarily have $\mathrm{D}_l^{\nu'}(\mathbf{u}) \geq \mathrm{D}_l^{\nu}(\mathbf{u})$. Hence the test (6a) also passes for index ℓ at node ν'.

References

[1] Xiaojun Chen, Dongdong Ge, Zizhuo Wang, and Yinyu Ye, "Complexity of unconstrained $\ell_2 - \ell_p$ minimization," *Mathematical Programming*, vol. 143, no. 1-2, pp. 371–383, November 2014.

[2] Simon Foucart and Holger Rauhut, *A Mathematical Introduction to Compressive Sensing*, Springer New York, 2013.

[3] Ryuhei Miyashiro and Yuichi Takano, "Subset selection by Mallows' C_p: A mixed integer programming approach," *Expert Systems with Applications*, vol. 42, no. 1, pp. 325–331, Jan. 2015.

[4] Sébastien Bourguignon, Jordan Ninin, Hervé Carfantan, and Marcel Mongeau, "Exact sparse approximation problems via mixed-integer programming: Formulations and computational performance," *IEEE Transactions on Signal Processing*, vol. 64, no. 6, pp. 1405–1419, 2015.

[5] Dimitris Bertsimas, Angela King, and Rahul Mazumder, "Best subset selection via a modern optimization lens," *The Annals of Statistics*, vol. 44, no. 2, Apr. 2016.

[6] CPLEX User's Manual, "Ibm ilog cplex optimization studio," *Version*, vol. 12, pp. 1987–2018, 1987.

[7] Ramzi Ben Mhenni, Sébastien Bourguignon, Marcel Mongeau, Jordan Ninin, and Hervé Carfantan, "Sparse branch and bound for exact optimization of ℓ_0-norm penalized least squares," in *ICASSP*. IEEE, 2020, pp. 5735–5739.

[8] Laurent El Ghaoui, Vivian Viallon, and Tarek Rabbani, "Safe feature elimination for the lasso and sparse supervised learning problems," 2010.

[9] Alper Atamturk and Andrés Gómez, "Safe screening rules for l0-regression from perspective relaxations," in *International conference on machine learning*. PMLR, 2020, pp. 421–430.

[10] Eugene L Lawler and David E Wood, "Branch-and-bound methods: A survey," *Operations research*, vol. 14, no. 4, pp. 699–719, 1966.

[11] Brian R Gaines, Juhyun Kim, and Hua Zhou, "Algorithms for fitting the constrained lasso," *Journal of Computational and Graphical Statistics*, vol. 27, no. 4, pp. 861–871, 2018.

[12] Neal Parikh and Stephen Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[13] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng, "Efficient sparse coding algorithms," in *Advances in neural information processing systems*, 2007, pp. 801–808.

[14] Don H Johnson, "Signal-to-noise ratio," *Scholarpedia*, vol. 1, no. 12, pp. 2088, 2006.

[15] Charles Soussen, Jérôme Idier, David Brie, and Junbo Duan, "From Bernoulli-Gaussian deconvolution to sparse signal restoration," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4572–4584, 2011.

[16] Stephen Wright, Jorge Nocedal, et al., "Numerical optimization," *Springer Science*, vol. 35, no. 67-68, pp. 7, 1999.

[17] Jeff Bezanson, Stefan Karpinski, Viral B Shah, and Alan Edelman, "Julia: A fast dynamic language for technical computing," 2012.

[18] Heinz H. Bauschke and Patrick L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, Springer International Publishing, 2017.