Impact of Penalization on Sparse Least-Squares Solutions

Martín Pasche
CentraleSupélec
Rennes, France
martin.pasche@student-cs.fr

Guillaume Di Fatta
CentraleSupélec
Rennes, France
guillaume.difatta@student-cs.fr

Gabin Dieudonne
CentraleSupélec
Rennes, France
gabin.dieudonne@student-cs.fr

Theo Montant
CentraleSupélec
Rennes, France
theo.montant@student-cs.fr

Côme Delecourt
CentraleSupélec
Rennes, France
come.delecourt@student-cs.fr

Louis Huhardeaux
CentraleSupélec
Rennes, France
louis.huhardeaux@student-cs.fr

Clément Elvira
Laboratoire IETR
CentraleSupélec
Rennes, France
clement.elvira@centralesupelec.fr

***Abstract*—We present an innovative approach for analyzing the impact of the penalizer parameter in parsimonious least-squares problems. Our methodology focuses on developing an algorithm to identify parameter intervals where the optimal solution remains unchanged. By leveraging the Fenchel dual formulation of the relaxed problem, we establish a condition for infeasibility, thereby delineating the parameter range where the solution is valid. To demonstrate the effectiveness of the approach, we applied it to a simplified case of the parsimonious least-squares problem using a branch-and-bound algorithm. While the identified interval encompassed the entire parameter space, providing no additional insights, the method shows promise for future applications and alternative implementations.**

***Index Terms*—Sparse representation, Branch-and-Bound algorithm, Fenchel duality, Optimization methods**

I. Introduction

The sparse representation of a vector is a essential problem in many mathematical and computer science fields, such as Artificial Intelligence, signal processing, or statistics. It consists in decomposing some input vector $y \in \mathbb{R}^m$ as a linear combination of a few columns of a dictionary $A \in \mathbb{R}^{m \times n}$. To translate this goal in a mathematical language, we search :

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

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] [2], preventing its resolution by classical exhaustive algorithms in a decent time. Thus, it has been a prolific field of research to find some rewriting of problem (1), to apply well-known algorithms and try to solve it. One of the most used approach consist in rewriting the problem as :

$$P^\star = \min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2}\|\mathbf{y} - A\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_0, \quad \text{s.t.} \quad ||\mathbf{x}||_\infty \leq M \tag{2}$$

It has indeed been proven that problem (2) is equivalent to (1), provided that M is chosen large enough. This relaxation of the problem has the advantage of being solvable with well-known algorithms such as Branch and Bound, which will be detailed later.

In this paper, we will focus our interest on the parameter λ, which measure the influence of the number of nonzero entries in $\mathbf{x}$ in the problem. The basic idea is to know until which point we can tune the parameter λ, without changing the global optimal solution $\mathbf{x}^*$ of the problem. In particular, we will try to develop a specific algorithm, to find $]\lambda_l, \lambda_u[$ intervals, such that $\forall \lambda \in]\lambda_l, \lambda_u[$, $\mathbf{x}^*$ remains the same.

Our exposition is organized as follows. Section 2 provides an overview of the Branch-and-Bound algorithm, outlining its key principles and applications. Section 3 introduces our approach to the problem, detailing the design and functionality of our algorithm. In Section 4, we present and analyze the results obtained from our experiments. Finally, Section 5 concludes the paper, summarizing our findings and discussing potential avenues for future research.

II. Branch-and-bound procedures

As mentioned before, the NP-hard nature of problem (1) precludes the development of direct algorithms for its solution. However, with the (2) relaxation, we can use approaches such as Branch and Bound (BnB) to find the solution. The BnB procedures consists in constructing a search tree for the solution of the problem, in which each node ν will represent a component of vector x, for example x_i, and the decision at this node will be to know if $x_i = 0$ or $x_i \neq 0$. Formally, a node will be represented by three sets : $\nu = (S_0, S_1, \bar{S})$, where S_0 is the set of indexes of zero entries in x at node ν, S_1 is the set of indexes of nonzero entries in x, and $\bar{S}$ is the set of indexes of entries that have not been processed yet.

Thus the BnB algorithm begins at node $\nu = (\emptyset, \emptyset, \{1, \ldots, n\})$, and continues by alternating between processing the current node (bounding step), and choosing the next node to process (branching step). Indeed, not to list all the intermediate solutions of the problem, we will only explore the branch of the tree, that have "good chances" to

give us the final solution. Theses steps will be detailed later. With this method, we can find the global minimum in a finite number of step, with a complexity at worst-case that is the exhaustive research.

A. Bounding step

When processing node $\nu = (S_0, S_1, \bar{S})$, we search for $x \in \mathbb{R}^n$, such that, $\forall i \in S_0, x_i = 0$, $\forall i \in S_1, x_i \neq 0$, and x minimize problem (2). Thus we can rewrite the problem as :

$$\begin{aligned} P^\nu = \min_{\mathbf{x}\in\mathbb{R}^n} \tfrac{1}{2}||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 + \lambda||\mathbf{x}_{\overline{S}}||_0 + \lambda|S_1| \\ \text{s.t. } ||\mathbf{x}||_\infty \leq M \text{ and } \mathbf{x}_{S_0} = \mathbf{0} \end{aligned} \tag{3}$$

Since all minimizers of (3) are also minimizers of (2), we have : $P^\nu \geq P^*$. This gives us a first condition to explore or not a node. Indeed if $P^\nu > P^*$, we can prune the node and branch, because none of the x vectors respecting the condition at ν can be optimal, that means that nones of the ν that follows in the same branch arrive the optimal value. Nevertheless, in practice we do not know P^* and computing P^ν with $\bar{S} \neq \emptyset$ is also costly. To overstep these issues, we will try to compute a lower bound P_l^ν of P^ν and an upper bound P^u of P^*, by doing this, we would still have the pruning condition : $P_l^\nu > P^u$.

To compute P_l^ν, we use the property : $||x_{\bar{S}}||_1 \leq M||x_{\bar{S}}||_0$, since $||x||_\infty \leq M$, then we can write:

$$\begin{aligned} P_l^\nu = \min_{\mathbf{x}\in\mathbb{R}^n} \tfrac{1}{2}||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 + \tfrac{\lambda}{M}||\mathbf{x}_{\overline{S}}||_1 + \lambda|S_1| \\ \text{s.t. } ||\mathbf{x}||_\infty \leq M \text{ and } \mathbf{x}_{S_0} = \mathbf{0} \end{aligned} \tag{4}$$

The upper bound P^u can be obtained by keeping in memory, the best possible candidate of (2) during the algorithm. More precisely, we construct a feasible solution of (3) at each node to obtain a potentially better upper bound. The best known upper bound is then updated as:

$$P^u := min(P^u, P^\nu) \tag{5}$$

During the BnB procedure, P^u converges toward P^*.

B. Branching step

If the node $\nu = (S_0, S_1, \bar{S})$ is still explorable in the tree, i.e. has not been pruned, the exploration goes on, and we select $i \in \bar{S}$, and create two sub-nodes corresponding to the decisions $x_i = 0$ or $x_i \neq 0$. We continue like that until all nodes have been explored or pruned, and we consider then that the p^u computed, is a minimum of (2).

III. Penalizer interval searching algorithm

Even though, direct access to P^* is unavailable, we have seen that yet less efficient algorithms provide a workaround. Additionally, the screening method can enhance the algorithm's efficiency. To accomplish this, we need to define the Fenchel dual, presented in Equation (6) :

$$\begin{aligned} d_l^\nu &= \max_{\mathbf{u}\in\mathbb{R}^m} D_l^\nu(\mathbf{u}) \\ &\triangleq \max_{\mathbf{u}\in\mathbb{R}^m} \tfrac{1}{2}||\mathbf{y}||_2^2 - \tfrac{1}{2}||\mathbf{y} - \mathbf{u}||_2^2 \\ &\quad - \sum_{i\in\overline{S}} M[|\mathbf{a}_i^T\mathbf{u}| - \tfrac{\lambda}{M}]_+ - \sum_{i\in S_1} M(|\mathbf{a}_i^T\mathbf{u}| - \tfrac{\lambda}{M}) \end{aligned} \tag{6}$$

(Notation consideration: $\mathbf{a}_i$ is the i-th column of $\mathbf{A}$, and $[z]_+ \triangleq \max(0, z)$.) Due to strong duality, Equation (6) holds.

$$\forall \mathbf{u} \in \mathbb{R}^m, \quad D_l^\nu(\mathbf{u}) \leq d_l^\nu = P_l^\nu \leq P^\nu \tag{7}$$

Equation (7) leads to the decision criteria, which we will employ to determine the range of possible parameters λ without altering the solution.

Decision 1. If $\exists \mathbf{u} \in \mathbb{R}^m$ such that $P^* < D_l^\nu(\mathbf{u})$, then ν doesn't have the global optimal solution.

The first approach is going to be the most naive, since we are going to consider that tuning the parameter λ doesn't change the solution vector $\mathbf{x}^*$ and $\mathbf{u}^*$.

The second consideration is that we are going to check the decision 1 in every node of the main tree. To do so, we are going to develop the decision 1 for node ν. Let $\mathbf{x}^* \in \mathbb{R}^n, \mathbf{u}^* \in \mathbb{R}^m$ given by the optimal solution of the parameter $\lambda^* \in \mathbb{R}^+$.

$$\begin{aligned} P^* &< D_l^\nu \\ \tfrac{1}{2}||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 + \lambda^*||\mathbf{x}||_0 &< \tfrac{1}{2}||\mathbf{y}||_2^2 - \tfrac{1}{2}||\mathbf{y} - \mathbf{u}||_2^2 \\ &\quad + \lambda^*|S_1| - \sum_{i\in S_1} M|\mathbf{a}_i^T\mathbf{u}| \\ &\quad - \sum_{i\in\overline{S}} [M|\mathbf{a}_i^T\mathbf{u}| - \lambda^*]_+ \end{aligned} \tag{8}$$

For handling the constant terms, we will do:

$$\begin{aligned} \Omega = \tfrac{1}{2}||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 - \tfrac{1}{2}||\mathbf{y}||_2^2 \\ + \tfrac{1}{2}||\mathbf{y} - \mathbf{u}||_2^2 + \sum_{i\in S_1} M|\mathbf{a}_i^T\mathbf{u}| \end{aligned} \tag{9}$$

Even knowing that $\mathbf{u}^* = \mathbf{y} - \mathbf{A}\mathbf{x}^*$, we cannot know a priori if Ω is non-negative, i.e., $\Omega \geq 0$. With the constant (9) in the equation ((8)), we have:

$$\Omega < \lambda^*|S_1| - \lambda^*||\mathbf{x}||_0 - \sum_{i\in\overline{S}} [M|\mathbf{a}_i^T\mathbf{u}| - \lambda^*]_+ \tag{10}$$

From what can be seen, the $[x]_+$ expression can't be solved directly, that is why we are going to develop a procedure to tackle this inconvenience.

For the next part, we are going to manipulate the equation ((10)) for different intervals of feasible λ parameters.

Without loss of generality, we are going to assume that $|\overline{S}| \leq k$ and $\overline{S} = \{1, \ldots, k\}$, and at the same time, the array of $\{M|\mathbf{a}_i^T\mathbf{u}|\}_{i=1}^k$ is sorted in a decreasing way, i.e. :

$$\Rightarrow M|\mathbf{a}_1^T\mathbf{u}| \geq M|\mathbf{a}_2^T\mathbf{u}| \geq \cdots \geq M|\mathbf{a}_k^T\mathbf{u}|$$

Since we know that for λ^* the equation ((10)) is fulfilled, we use it as the start line. We search

$$j_0 = \arg\max_{j \in \overline{S}} M|\mathbf{a}_j^T \mathbf{u}| \geq \lambda^* \tag{11}$$

Then, $\forall \lambda \in [M|\mathbf{a}_{j_0+1}^T \mathbf{u}|, M|\mathbf{a}_{j_0}^T \mathbf{u}|]$, we have:

$$\sum_{i \in \overline{S}} [M|\mathbf{a}_i^T \mathbf{u}| - \lambda]_+ = \sum_{i=1}^{j_0} M|\mathbf{a}_i^T \mathbf{u}| - \lambda \cdot j_0 \tag{12}$$

Summing up ((12)) with ((10)), we can say that there $\exists\ \lambda \in [M|\mathbf{a}_{j_0+1}^T \mathbf{u}|, M|\mathbf{a}_{j_0}^T \mathbf{u}|]$

$$\Omega + \sum_{i=1}^{j_0} M|\mathbf{a}_i^T \mathbf{u}| < \lambda(|S_1| - ||\mathbf{x}||_0 + j_0) \tag{13}$$

We want to isolate the term λ, however, we have to pay attention to the term $|S_1| - ||\mathbf{x}||_0 + j_0$, since it can be either positive or negative. Lets suppose that it is positive, $|S_1| - ||\mathbf{x}||_0 + j_0 \geq 0$, then, we have:

$$\frac{\Omega + \sum_{i=1}^{j_0} M|\mathbf{a}_i^T \mathbf{u}|}{|S_1| - ||\mathbf{x}||_0 + j_0} < \lambda \tag{14}$$

Then, if there exists $\lambda' \in [M|\mathbf{a}_{j_0+1}^T \mathbf{u}|, M|\mathbf{a}_{j_0}^T \mathbf{u}|]$ such that (14) is not fulfilled, then we have an upper bound for the parameter without changing the solution, which would be:

$$\lambda_u^\nu = \frac{\Omega + \sum_{i=1}^{j_0} M|\mathbf{a}_i^T \mathbf{u}|}{|S_1| - ||\mathbf{x}||_0 + j_0} \tag{15}$$

In the case that this doesn't happen, we check for the interval with higher values. Again, we check if there is $\lambda' \in [M|\mathbf{a}_{j_0}^T \mathbf{u}|, M|\mathbf{a}_{j_0-1}^T \mathbf{u}|]$ leading to:

$$\Omega + \sum_{i=1}^{j_0-1} M|\mathbf{a}_i^T \mathbf{u}| < \lambda(|S_1| - ||\mathbf{x}||_0 + j_0 - 1) \tag{16}$$

If there is any upper or lower bound, which depends on the sign of $|S_1| - ||\mathbf{x}||_0 + j_0 - 1$. This process is repeated going to the higher intervals or the lowers, until λ_u^ν and λ_l^ν are found (there might be a case where we won't be able to find an upper or lower bound, in those cases we take ∞ and $-\infty$ respectively), which will give us the interval $[\lambda_l^\nu, \lambda_u^\nu]$ for possible parameters which don't change the solution for the node ν.

Completing this procedure for all the nodes of the main tree, will give us $\{\lambda_u^\nu\}_\nu$ and $\{\lambda_l^\nu\}_\nu$, and from this list, we want the following:

$$\lambda_u = \min_\nu \lambda_u^\nu \ , \qquad \lambda_l = \max_\nu \lambda_l^\nu \tag{17}$$

With this terms, we have finished our first approach to determine a possible interval $[\lambda_l, \lambda_u]$for the λ parameter in which the solution remains unchanged.

IV. Results and validation

Our approach and algorithm have been presented, we now need to test them in practice by implementing them in a python simulation. This code is available at [3]. First we started by recreating the Branch and Bound algorithm, using Python solver library for the optimisation problems such as the least-square problem. To validate our code, we have used the Julia code of Théo Guyard, a CentraleSupélec PhD student, whose Branch and Bound implementation can be found at [4]. By running our two simulations on the same entries data, we could validate that our implementation is correct

Once our Branch and Bound algorithm is correct, we could start to implement our method to find λ intervals. To test it, we worked on a simple case, considering $A = Id_n$ and $\mathbf{y} \in \mathbb{R}^n$ given, we can get the expression:

$$\begin{aligned}\min_{\mathbf{x} \in \mathbb{R}^n} \tfrac{1}{2}||\mathbf{y} - A\mathbf{x}||_2^2 + \lambda||\mathbf{x}||_0 = \min_{\mathbf{x} \in \mathbb{R}^n} \tfrac{1}{2}||\mathbf{y}||_2^2 + \tfrac{1}{2}||\mathbf{x}||_2^2 \\ - \mathbf{y}^T\mathbf{x} + \lambda||\mathbf{x}||_0\end{aligned} \tag{18}$$

We can decompose the norm $||\cdot||_0$ by dimensions:

$$||\mathbf{x}||_0 = ||\mathbf{x}_1||_0 + ||\mathbf{x}_2||_0 + \cdots + ||\mathbf{x}_n||_0 \tag{19}$$

With this, the minimization problem in (18) can be separated by their dimensions, since they are independent between each other. Therefore, we get the expression (20).

$$\min_{x_i \in \mathbb{R}} \tfrac{1}{2}x_i^2 + \tfrac{1}{2}y_i^2 - y_i xi + \lambda||x_i||_0 \tag{20}$$

To get the solution, it only require to study the problem by cases, for $x_i = 0$ or $x_i \neq 0$, we have:

$$x_i = \begin{cases} y_i & \lambda < \frac{1}{2}y_i^2 \\ 0 & \text{other cases} \end{cases} \tag{21}$$

Unfortunately, after some tests, we found out that our code gave us $]-\infty; \infty[$ with this simple case as an entry parameter. Thus, we can arrive to two conclusions, the first case would be that there are some bugs in our implementation or this method is flawed for the objective of our study. This is nothing to be surprised of, since this technique is highly sensitive to the big M and that it relies in the pruning decision 1, which is harder to be fulfilled, since in the case where ν has the solution between its branches, then $D_l^\nu \leq P_l^\nu \leq P^*$ is true.

V. Conclusion

Our project proposed to study an innovative approach, that aimed at investigating the impact of the parameter λ in the context of l_0-penalized least-squares problems. We have so studied the problem, and the Branch and Bound algorithm used to solve it, and created a simple theoretical method to find some λ intervals, in which the global solution $\mathbf{x}^*$ of the problem does not change. Unfortunately, because of a lack of time, our code implementation of this method does not work yet but is, according to us, a good start point to continue researches on this topic, however, we doubt that this method can provide any new useful information.

ACKNOWLEDGMENT

Acknowledgements to all the people providing guidance and supervising the pôle project at Rennes from CentraleSupélec.

REFERENCES

[1] X. Chen, D. Ge, Z. Wang, and Y. Ye, "Complexity of unconstrained $l_2 - l_p$ minimization," *Mathematical Programming*, vol. 143, no. 1, pp. 371–383, Feb 2014. [Online]. Available: https://doi.org/10.1007/s10107-012-0613-0

[2] T. Guyard, C. Herzet, and C. Elvira, "Node-screening tests for the l0-penalized least-squares problem," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 5448–5452.

[3] M. Pasche and G. D. Fatta, "L0-Penalized-lambda-search," https://github.com/mapasche/L0-Penalized-lambda-search, accessed: [27/11/2024].

[4] T. Guyard, "Un solveur efficace pour la résolution de problèmes parcimonieux avec pénalité l0," in *24ème édition du congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, 2023.