Provably Convergent Working Set Algorithm for Non-Convex Regularized Regression
Alain Rakotomamonjy, Rémi Flamary, Gilles Gasso, Joseph Salmon

To cite this version:
Alain Rakotomamonjy, Rémi Flamary, Gilles Gasso, Joseph Salmon. Provably Convergent Working Set Algorithm for Non-Convex Regularized Regression. International Conference on Artificial Intelligence and Statistics, 2022. hal-02875560v4

HAL Id: hal-02875560
https://hal.science/hal-02875560v4
Submitted on 19 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License
Convergent Working Set Algorithm for Lasso with Non-Convex Sparse Regularizers

Alain Rakotomamonjy
LITIS, Univ. de Rouen
Criteo AI Lab, Paris
alain.rakoto@insa-rouen.fr

Rémi Flamary
Univ Côte d’Azur,CNRS , OCA Lagrange
remi.flamary@unice.fr

Gilles Gasso
LITIS, INSA de Rouen
gilles.gasso@insa-rouen.fr

Joseph Salmon
IMAG, Université de Montpellier, CNRS
Montpellier, France
joseph.salmon@umontpellier.fr

Abstract

Non-convex sparse regularizers are common tools for learning with high-dimensional data. For accelerating convergence of a Lasso problem using those regularizers, a working set strategy addresses the optimization problem through an iterative algorithm by gradually incrementing the number of variables to optimize until the identification of the solution support. We propose in this paper the first Lasso working set algorithm for non-convex sparse regularizers with convergence guarantees. The algorithm, named FireWorks, is based on a non-convex reformulation of a recent duality-based approach and leverages on the geometry of the residuals. We provide theoretical guarantees showing that convergence is preserved even when the inner solver is inexact, under sufficient decay of the error across iterations. Experimental results demonstrate strong computational gain when using our working set strategy compared to full problem solvers for both block-coordinate descent or a proximal gradient solver.

1 Introduction

Many real-world learning problems are of (very) high dimension. This is the case for natural language processing problems with very large vocabulary or recommendation problems involving million of items. In such cases, one way of addressing the learning problem is to consider sparsity-inducing penalties. Likewise, when the solution of a learning problem is known to be sparse, using these penalties yield to models that can leverage this prior knowledge. The Lasso \[31\] and the Basis pursuit \[6, 5\] were the first approaches that have employed $\ell_1$-norm penalty for inducing sparsity. The Lasso model has enjoyed large practical successes in the machine learning and signal processing communities \[29, 20, 36\]. Nonetheless, it suffers from theoretical drawbacks (e.g., biased estimates for large coefficients of the model) which can be overcome by considering non-convex sparsity-inducing penalties. These penalties provide continuous approximations of the $\ell_0$-(pseudo)-norm which is the true measure of sparsity. There exists a flurry of different penalties like the Smoothly Clipped Absolute Deviation (SCAD) \[9\], the Log Sum penalty (LSP) \[4\], the capped-$\ell_1$ penalty \[38\], the Minimax Concave Penalty (MCP) \[37\]. We refer the interested reader to \[30\] for a discussion on the pros and cons of such non-convex formulations.

In addition to theoretical statistical analyses, efforts have also been made for developing computationally efficient algorithms for non-convex regularized optimization problems. This includes coordinate descent algorithms \[3\], proximal gradient descent \[15\] or Newton method \[35, 25\]. However, all these methods share one kind of inefficiency in the sense that they spend a similar computational effort for each variable, even when these variables will end up being irrelevant (zero weight) in the final learnt model. In the non-convex setting, few methods have tried to lift this issue. One approach mixes importance sampling and randomized
coordinate descent \cite{10}, while another one seeks to safely screen features that are irrelevant \cite{20}. Working set (also known as active set) strategy aims at focusing computational effort on a subset of relevant variables, making them highly efficient for optimization problem with sparse solutions, provided that the algorithm is able to quickly identify the “relevant” features. In the literature, several works on working set algorithms address this selection issue mostly for convex optimization problems such as the Support Vector Machine problem \cite{24, 13} or the Lasso problem \cite{11, 52, 13, 22}. Working set strategies have been extended to non-convex sparse optimization problems \cite{11, 2} but they are purely heuristic and lack of convergence guarantees.

In this work, inspired by the Blitz algorithm proposed by Johnson and Guestrin \cite{18} (see also \cite{21, 22} for its connection with safe screening rules) we propose a theoretically supported method for selecting a working set in non-convex regularized sparse optimization problems. While Blitz can only be implemented for convex problems, leveraging on primal-dual aspects of the $\ell_1$-regularized problem, we introduce a similar algorithm that exploits the key role of the residual in a sparse regression problem. Our algorithm proposes a method for selecting the variables to integrate into a working set, and provides a theoretical guarantee on objective value decrease. Based on these results, we provide, as far as we know, the first convergence guarantee of working set algorithm in a non-convex Lasso setting and we show that this convergence property is preserved in a realistic inexact setting.

In summary, our contributions are the following: (1) we propose a novel working set algorithm for non-convex regularized regression that selects features to integrate in the model based on a so-called “feasible” residual; (2) we prove that the algorithm enjoys properties such as convergence to a stationary point, even when the inner solver is inexact, under sufficient decay of the error along the iterations; as such, it is the first non-convex working set algorithm with such a theoretical convergence proof. (3) Our experimental results show that our FireWorks algorithm achieves substantial computational gain (that can reach two orders of magnitude) compared to the baseline approaches with proven convergence guarantees and on par with the heuristic working set algorithm of \cite{11}.

Notation We denote as $X \in \mathbb{R}^{n \times d}$ the design matrix. We write vectors of size $d$ or size $n$ in bold e.g., $y \in \mathbb{R}^n$ or $w \in \mathbb{R}^d$. We will consider several sets and they are noted in calligraphic mode. We have set of indices, mostly noted as $\mathcal{A}$, with $\mathcal{A}$ being a subset of indices extracted from $\{1, \ldots, d\}$ and with cardinality noted $|\mathcal{A}|$. Given a set $\mathcal{A}$, $\bar{\mathcal{A}}$ denotes its complement in $\{1, \ldots, d\}$. Set defined by (union of) function level-set will be denoted as $C$, with indices defining the function. Vectors noted as $w_\mathcal{A}$ are of size $|\mathcal{A}|$ and we note $\bar{w}_\mathcal{A} \in \mathbb{R}^d$ for the vector of component $w_j, \bar{w}_\mathcal{A}$ for all $j \in \mathcal{A}$ and 0 elsewhere. Finally, $X_\mathcal{A}$ represents matrix $X$ restricted to columns indexed by $\mathcal{A}$ and we will note $\text{res}(\mathcal{A}) \triangleq y - Xw$ and $\text{res}(w_\mathcal{A}) \triangleq y - X_\mathcal{A}w_\mathcal{A} = y - \bar{w}_\mathcal{A}$.

## 2 Linear regression with non-convex regularizers

We first introduce the non-convex Lasso problem we are interested in as well as its first-order optimality conditions. We emphasize on the form of the optimality conditions which will be key for designing our working set algorithm.

### 2.1 The optimization problem

We consider solving the problem of least-squares regression with a generic penalty of the form

$$
\min_{w \in \mathbb{R}^d} f(w) \triangleq \frac{1}{2} \|y - Xw\|_2^2 + \sum_{j=1}^{d} r_\lambda(|w_j|)
$$

(1)

where $y \in \mathbb{R}^n$ is a target vector, $X = [x_1, \ldots, x_d] \in \mathbb{R}^{n \times d}$ is the design matrix with column-wise features $x_j \in \mathbb{R}^n$, $w$ is the coefficient vector of the model and the map $r_\lambda : \mathbb{R}^d \to \mathbb{R}^d$ is monotonically non-decreasing, concave and differentiable on $[0, +\infty)$ with a regularization parameter $\lambda > 0$. In addition, we assume that $r_\lambda(|\cdot|)$ is a lower semi-continuous function. Note that most penalty functions such as SCAD, MCP or log sum (see their definitions in Table 2 in the supplementary material) satisfy such a property and that for these penalties, $f(\cdot)$ is lower bounded.

We consider tools such as Fréchet subdifferentials and limiting-subdifferentials \cite{19, 27, 23} well suited for non-smooth and non-convex optimization, so that a vector $w^*$ belongs to the set of minimizers (not necessarily global) of Problem (1) if following Fermat’s condition holds (see Definition 1.1 and Proposition 1.2 in \cite{19} and Chapter 9 of \cite{28}):

$$
\forall j, x_j^\top (y - Xw^*) \in \partial r_\lambda(|w_j^*|)
$$

(2)

with $\partial r_\lambda(|\cdot|)$ being the Fréchet subdifferential of $r_\lambda(|\cdot|)$, assuming it exists at $w^*$. In particular, this is the case for the MCP, log sum and SCAD penalties presented in Table 2. For the sake of clarity, we present next the optimality conditions for MCP and log sum.
Example 1. For the MCP penalty (see Table 2 for its definition and its subdifferential), it is easy to show that \( \partial r_\lambda([0]) = [-\lambda, \lambda] \). Hence, Fermat’s condition becomes with the residual \( \text{res}(\mathbf{w}^*) \)

\[
\begin{cases}
-x_j^\top \text{res}(\mathbf{w}^*) = 0, & \text{if } |w_j^*| > \lambda \theta \\
-x_j^\top \text{res}(\mathbf{w}^*) + \lambda \text{sign}(w_j^*) = \frac{w_j^*}{\theta}, & \text{if } 0 < |w_j^*| \leq \lambda \theta \\
|\mathbf{x}_j^\top \text{res}(\mathbf{w}^*)| \leq \lambda, & \text{if } w_j^* = 0
\end{cases}
\]  

(3)

Example 2. For the log sum penalty, one can explicitly compute \( \partial r_\lambda([0]) = [-\frac{\lambda}{2}, \frac{\lambda}{2}] \) and leverage the smoothness of \( r_\lambda([w]) \) when \( |w| > 0 \) for computing \( \partial r_\lambda([w]) \). Then, the condition in Equation 2 can be written as:

\[
\begin{cases}
-x_j^\top \text{res}(\mathbf{w}^*) + \lambda \frac{\text{sign}(w_j^*)}{\theta |w_j^*|} = 0, & \text{if } w_j^* \neq 0 \\
|\mathbf{x}_j^\top \text{res}(\mathbf{w}^*)| \leq \frac{\lambda}{\theta}, & \text{if } w_j^* = 0
\end{cases}
\]  

(4)

As we can see, first-order optimality conditions lead to simple equations and inclusions. More interestingly, one can note that regardless of the regularizer, the structure of optimality condition for a weight \( w_j^* = 0 \) depends on the correlation of the feature \( x_j \) with the optimal residual \( \text{res}(\mathbf{w}^*) = \mathbf{y} - \mathbf{X}\mathbf{w}^* \). Hence, these conditions can be used for defining a region in which the optimal residual has to live in.

3 Working set algorithm and analysis

Before presenting the FireWorks algorithm, we first introduce all concepts needed for defining and analyzing our working set algorithm.

3.1 Restricted problem and optimality

Given a set \( \mathcal{A} \) of \( m \) indices belonging to \{1, \ldots, d\}, the problem defined in Equation 5 is the restriction of Problem 1 to the columns of \( \mathbf{X} \) indexed by \( \mathcal{A} \):

\[
\min_{\mathbf{w}_\mathcal{A} \in \mathbb{R}^{|\mathcal{A}|}} \frac{1}{2} \| \mathbf{y} - \mathbf{X}_\mathcal{A} \mathbf{w}_\mathcal{A} \|^2 + \sum_{j=1}^{|\mathcal{A}|} r_\lambda(|w_j, \mathcal{A}|) .
\]

(5)

Naturally, a vector \( \mathbf{w}_\mathcal{A}^* \) minimizing this problem has to satisfy its own optimality condition. However, the next proposition derives a necessary condition for optimality, that will be useful for characterizing whether \( \mathbf{w}_\mathcal{A}^* \) is optimal for the full problem.

Proposition 1. If \( \mathbf{w}_\mathcal{A}^* \) satisfies Fermat’s condition of Problem 1, then for all \( j \in \mathcal{A} \), we have

\[
|x_j^\top (\mathbf{y} - \mathbf{X}_\mathcal{A} \mathbf{w}_\mathcal{A}^*)| \leq r_j^* (0)
\]  

(6)

where \( r_j^* \) is the derivative of \( r_\lambda \).

Now given Proposition 1, we are going to define some sets useful for characterizing candidate stationary points of either Equations 1 or 5. Let us define the function \( h_j : \mathbb{R}^n \rightarrow \mathbb{R} \), for \( j \in \{1, \ldots, d\} \) as \( h_j(\mathbf{a}) = |x_j^\top \mathbf{a} - r_j^* (0) \) and the convex sets \( \mathcal{C}_j \) as the slab

\[
\mathcal{C}_j \triangleq \{ \mathbf{a} \in \mathbb{R}^n : h_j(\mathbf{a}) \leq 0 \}
\]

and \( \mathcal{C}_j^\circ \) as its boundary

\[
\mathcal{C}_j^\circ \triangleq \{ \mathbf{a} \in \mathbb{R}^n : h_j(\mathbf{a}) = 0 \}.
\]

By introducing \( \mathcal{C} = \bigcap_{j=1}^d \mathcal{C}_j \) and \( \mathcal{C}_\mathcal{A} = \bigcap_{j \in \mathcal{A}} \mathcal{C}_j \) the necessary optimality condition defined in Proposition 1 can be written as \( \mathbf{y} - \mathbf{X}_\mathcal{A} \mathbf{w}_\mathcal{A}^* \in \mathcal{C}_\mathcal{A} \). Hence, assuming that \( \mathbf{w}_\mathcal{A}^* \) is a minimizer of its restricted Problem 5, its extension \( \mathbf{w}_\mathcal{A}^\circ \in \mathbb{R}^d \) satisfies Fermat’s condition of the full problem if the following holds

\[
\mathbf{y} - \mathbf{X}\mathbf{w}_\mathcal{A}^\circ \in \mathcal{C}_\mathcal{A}^\circ .
\]

(7)

where \( \mathcal{A} \) is the complement of \( \mathcal{A} \) in \{1, \ldots, d\}. Indeed, since \( \mathbf{w}_\mathcal{A}^* \) is optimal for the restricted problem, Fermat’s condition is already satisfied for all \( j \in \mathcal{A} \). Then, the above condition ensures that \( \forall j \in \mathcal{A} \), we have \( |x_j^\top (\mathbf{y} - \mathbf{X}\mathbf{w}_\mathcal{A}^\circ)| \leq r_j^* (0) \) since, as by definition, \( \tilde{w}_j^* = 0, \forall j \in \mathcal{A} \).

Equation 7 provides an easy way to check whether a solution of a restricted problem is a potential candidate for being also a solution to the full problem. For this purpose, we define the distance of a vector \( \mathbf{r} \in \mathbb{R}^n \) to the convex set \( \mathcal{C}_j \) and \( \mathcal{C}_j^\circ \) as

\[
\text{dist}(\mathbf{r}, \mathcal{C}_j) \triangleq \min_{\mathbf{z} \in \mathbb{R}^n} \| \mathbf{z} - \mathbf{r} \|_2 , \text{ s.t. } h_j(\mathbf{z}) \leq 0 ;
\]

and

\[
\text{dist}_2(\mathbf{s}, \mathcal{C}_j^\circ) \triangleq \min_{\mathbf{z} \in \mathbb{R}^n} \| \mathbf{z} - \mathbf{s} \|_2 , \text{ s.t. } h_j(\mathbf{z}) = 0 .
\]

These distances can also be used for defining the most violated optimality condition, a key component of the methods proposed by 11. Indeed, given a set \( \mathcal{A} \), the solution \( \mathbf{w}_\mathcal{A}^* \) of Equation 5 and the associated residual \( \text{res}(\mathbf{w}_\mathcal{A}^*) \), the index \( j^* = \arg \max_{j \in \mathcal{A}} \text{dist}(\text{res}(\mathbf{w}_\mathcal{A}^*), \mathcal{C}_j) \) is the index of the most violated optimality condition among non-active variables for the residual \( \text{res}(\mathbf{w}_\mathcal{A}^*) \).

3.2 Feasible Residual Working Set Algorithm for non-convex Lasso

A working set algorithm for solving Problem 1 consists in sequentially solving a series of restricted problem as defined in Equation 5 with a sequence of working sets \( \mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_k \). The main differences among

\footnote{For \( \ell_1 \)-type convex regularizers \( \mathcal{C} \) is the dual feasible set.}
working set algorithms lie on the way the set is being updated. For instance, the approach of [1], denoted in the experiment as MaxVC, selects the variable with the most violated optimality conditions (as defined above) in the non-active set to be included in the new working set, leading to the algorithm presented in the supplementary material. Flamary et al. [10] followed a similar approach but considered a randomized selection in which the probability of selection is related to \( \text{dist}(\hat{\text{res}}(\tilde{w}_{\mathcal{A}_k}), \mathcal{C}) \).

Our algorithm is inspired by Blitz [18] which is a working set algorithm dedicated to convex constrained optimization problem. But as the problem we address is a non-convex one, we manipulate different mathematical objects that need to be redefined. The procedure is presented in Algorithm 1. It starts by selecting a small subset of indices for instance the ten indices with largest \( |x_j^Ty| \) as initial working set and by choosing a vector \( s_1 \) such that \( s_1 \in \mathcal{C} = \bigcap_{j=1}^n C_j \), for instance setting \( s_1 = 0 \). From this vector \( s_1 \), we will generate a sequence \( \{s_k\} \) that plays a key role in the selection of the features to be integrated in the next restricted model. Then, at iteration \( k \), it solves the restricted problem with the set \( \mathcal{A}_k \) and then by computing the residual \( r_k = \text{res}(\tilde{w}_{\mathcal{A}_k}) \) with \( \tilde{w}_{\mathcal{A}_k} \) the true solution to the restricted problem. As noted in Equation (7), if \( r_k \in \mathcal{C}_{\mathcal{A}_k} \) then the vector \( \tilde{w}_{\mathcal{A}_k} \) is a stationary point of the full problem. If \( r_k \notin \mathcal{C}_{\mathcal{A}_k} \), we need to update the working set \( \mathcal{A}_k \). We first prune \( \mathcal{A}_k \) by removing indices associated to zero weights in \( \tilde{w}_{\mathcal{A}_k} \). Then, in order to add features to the working set, we define \( \mathcal{S}_{k+1} \) as the vector on the segment \( \{s_k, r_k\} \), nearest to \( r_k \) that belongs to \( \mathcal{C} \). Then, the working set is updated by integrating predictors \( j \) whose associated slab \( C_j \) frontier are nearest to \( s_k \). Hence, the index \( j \) is included in the new working set if \( \text{dist}(s_k, C_j^\top) \leq \tau_k \), where \( \tau_k \) is a strictly positive term that defines the number of features to be added to the current working set. In practice, we have chosen \( \tau_k \) so that a fixed number \( n_{\text{added}} \) of features is added to the working set \( \mathcal{A}_k \) at each iteration \( k \).

We provide the following intuition on why this algorithm works in practice. At first, note that by construction \( s_{k+1} \) is a convex combination of two vectors one of which is the residual hence justifies its interpretation as a pseudo-residual. However, the main difference between the \( s_k \)’s and \( r_k \)’s is that the former belongs to \( \mathcal{C} \) and thus to any \( C \) while \( r_k \) belongs to \( \mathcal{C} \) only for a potential \( \tilde{w}_{\mathcal{A}_k} \) optimal for the full problem. Then, when \( \tilde{w}_{\mathcal{A}_k} \) is a stationary point for the restricted problem but not for the full problem, we have \( r_k \in C \) but \( r_k \notin \mathcal{C} \). Hence, \( s_{k+1} \) represents a residual candidate for optimality and slab’s frontiers near this pseudo-residual \( s_{k+1} \) can be interpreted as the slabs associated to features that need to be integrated in the working set (allowing associated weights \( w_j \)’s to be potentially non-zero at the next iteration). This mechanism for selection is shown in Figure 1.

Relation with maximum violated optimality condition algorithm [11]. The mechanism we have proposed for updating the working set is based on the current residual \( r_k \) and a feasible residual \( s_k \). By changing how \( s_{k+1} \) is defined, we can retrieve the algorithm proposed by Boisbunon et al. [11]. Indeed, if we set at Line 5 of Algorithm 1, \( \forall k, s_k = 0 \) and \( s_{k+1} = \alpha_k r_k \), with \( \alpha_k \in [0, 1] \) then \( s_{k+1} \) is a rescaling of the current residual and the scale is chosen so that \( s_{k+1} \in \mathcal{C} \). Using a simple inequality argument, it is straightforward to show that \( \alpha_k = \min(\min_{j \in \mathcal{A}_k} \lambda_j \frac{A^\top_j w_j}{r_{kj}^2}, 1) \) and the minimum in \( j \) occurs for the largest value of \( |x_j^Ty|^\top r_k | \). From the theoretical side, we want to emphasize that Boisbunon et al. [11] do not provide convergence proof of this algorithm. Nonetheless, we conjecture that the polynomial convergence of this algorithm is guaranteed for exact inner solver and when working set is never pruned (removing from the set \( \mathcal{A}_k \) variables which weights are 0 is not allowed).

3.3 Some properties of the algorithm

In this subsection, we analyze some properties of the proposed algorithm. At first, we introduce an alternative optimality condition (whose proof is in the supplementary), based on \( \alpha_k \) for the full problem. Based on this property and some intermediate results, we will
show that the iterates \( \tilde{w}_{A_{k+1}} \) converge towards a stationary point of the full problem.

**Proposition 2.** Given a working set \( A_k \) and \( w_{A_k}^* \) solving the related restricted problem, \( \tilde{w}_{A_k}^* \) is also optimal for the full problem if and only if \( \alpha = 1 \) in Algorithm 1, step 4 (which also means \( s_{k+1} = r_k \)).

Now, we are going to characterize the decrease in objective value obtained between two updates of working sets, assuming that in the update, there is at least one feature that does not satisfy its optimality condition.

**Proposition 3.** Assume that \( \|X\|_2 > 0 \) and \( w_{A_k}^* \) and \( w_{A_{k+1}}^* \) are respectively the solutions of the restricted problem with the working set \( A_k \) and \( A_{k+1} \), with \( A_{k+1} = \{j_1, \ldots, j_{\text{new added}}\} \cup A_k \), such that there exists at least one \( j_i \) with \( \text{dist}(r_k, C_{j_i}) > 0 \). As we note \( r_k \triangleq \text{res}(w_{A_k}^*) \), the following inequality holds for all \( j_i \) such that \( \text{dist}(r_k, C_{j_i}) > 0 \)

\[
\| \tilde{w}_{A_{k+1}}^* - \tilde{w}_{A_k}^* \|_2 \geq \frac{1}{\|X\|_2} \text{dist}(r_k, C_{j_i}) \, .
\]

**Proof.** We have the following inequalities

\[
\| r_{k+1} - r_k \|_2 = \| X(\tilde{w}_{A_{k+1}}^* - \tilde{w}_{A_k}^*) \|_2 \leq \| X \|_2 \| \tilde{w}_{A_{k+1}}^* - \tilde{w}_{A_k}^* \|_2 \, .
\]

Now recall that \( r_k \notin C_{A_{k+1}} \), since \( \exists j_i : \text{dist}(r_k, C_{j_i}) > 0 \), while \( r_{k+1} \in C_{A_{k+1}} \), as \( w_{A_{k+1}}^* \) has been optimized over \( A_{k+1} \). As such, for all \( j_i \) : \( \text{dist}(r_{k+1}, C_{j_i}) > 0 \), we also have \( h_j(r_{k+1}) \leq 0 \). Now by definition of dist \( (r_k, C_{j_i}) \) either \( r_{k+1} \) is the minimizer of the distance optimization problem, hence \( \text{dist}(r_k, C_{j_i}) = \| r_{k+1} - r_k \|_2 \) or \( \text{dist}(r_k, C_{j_i}) \leq \| r_{k+1} - r_k \|_2 \). Plugging this latter inequality in (8) concludes the proof.

Given the right hand side of the equation in Proposition 3, we now show that the distance of the residual \( r_k \) at step \( k \) to a set \( C_j \), defined by a feature \( j \) that is not yet in the active set, is lower bounded by a term depending on the parameter \( r_{k-1} \) which governs the number of features that has been added to the active set at step \( k - 1 \).

**Lemma 1.** At step \( k \geq 2 \), consider a set \( C_j \) such that \( h_j(r_k) > 0 \) and \( h_j(s_k) < 0 \), then

\[
\text{dist}(r_k, C_j) \geq \frac{1 - \alpha_k}{\alpha_k} r_{k-1} \, .
\]

The proof of this lemma is available in the supplementary material. From the above Proposition 3 and Lemma 1 we can ensure that the sequence \( \{\tilde{w}_{A_k}\} \) produced by Algorithm 1 converges towards a stationary point under mild conditions on the inner solver.

**Theorem 1.** Suppose that for each step \( k \), the algorithm solving the inner problem ensures a decrease in the objective value in the form

\[
f(\tilde{w}_{A_{k+1}}^*) - f(\tilde{w}_{A_k}^*) \leq -\gamma_k \| \tilde{w}_{A_{k+1}}^* - \tilde{w}_{A_k}^* \|_2^2 \, .
\]

with \( \forall k, \gamma_k \geq \gamma > 0 \). For the inner solver, we also impose that when solving the problem with set \( A_{k+1} \), the inner solver is warm-started with \( w_{A_k}^* \). Assume also that \( \|X\|_2 > 0 \), \( r_k \geq \zeta > 0 \) and \( h_j \) satisfies assumption in Lemma 1 then the sequence of \( \alpha_k \) produced by Algorithm 1 converges towards 1 and \( \forall j, \lim_{k \to \infty} \| x_j^* r_{k+1} \| \leq r_{k+1}^j(0) \).

The above theorem ensures convergence to a stationary point under some conditions on the inner solver and on the \( \gamma_k \)’s which needs to be lower bounded by...
\( \gamma > 0 \). Several algorithms may satisfy this assumption. For instance, any first-order iterative algorithm which selects its step size as \( t_k \) based on line search criterion of the form \( \nabla k \cdot f(w_{k+1}) \leq \nabla k \cdot f(w_k) - \frac{\gamma}{2} t_k \| w_{k+1} - w_k \|^2 \), where \( \sigma \) is a constant in the interval \((0, 1)\), provides such a guarantee. This is the case of the generalized proximal algorithm of Gong et al. [15] [Section 2.3.2] or proximal Newton approaches [23], assuming that \( f \) is differentiable with gradient Lipschitz and \( r_\lambda (\cdot) \) admits a proximal operator. Since non-convex block coordinate descent algorithms [8] can also be interpreted as proximal algorithm, they also satisfy this sufficient decrease condition under the same assumptions than proximal approaches.

Another important condition for convergence is based on the parameter \( r_k \). We note the lower bound \( \tau \) can be set to any arbitrary small positive value. At a non-optimal for the full problem \( w_{Ak} \), and as small as this lower bound is, the set \( \{ j : \text{dist}(s_{k+1}, C_j^w) \leq \tau \} \) always contains at least the index \( j \) that makes \( \alpha_k \) maximal and corresponds to the \( j \) such that \( \text{dist}(s_{k+1}, C_j^w) = 0 \) (see Line 4 of the algorithm). This would correspond to updating the working set by one element at each iteration.

The above theorem states about the convergence of the working set strategy. We want to emphasize here that the convergence rate of the whole algorithm [11] (working set + inner solver) depends on the convergence rate of the inner solver. For instance, if we consider as an inner solver the proximal algorithm of Gong et al. [15], then the convergence rate for each inner problem is of the form \( C \cdot \frac{\| w_0 - w^* \|}{k} \) where \( C \) is a constant depending on the inner problem, \( T \) the total number of iterations for that solver, and \( w_0 \) the initial point when solving that problem. Since Algorithm 1 runs this inner solver several times, the convergence rate is still in \( O(n) \) but with a different constant. The gain in computation time achieved by using a working set strategy comes from the fact that each inner solver involves far fewer variables than the full problem dimensionality \( d \) and thus gradients are cheaper to compute.

**Inexact inner solver** One key point when considering a meta-solver like Blitz [13] or a working set algorithm is that for some approaches, theoretical properties hold only when the solution of the inner solver is exact. This is for instance the case for the SimpleSVM algorithm of Vishwanathan et al. [34] or the active set algorithm proposed by Boisbunon et al. [1]. The convergence of these approaches are based on non-cyclicity of the working set selection (prohibiting pruning) and thus on the ability of solving exactly the inner problem. For the approach we propose, we show next that the distance between two consecutive inexact solutions of the inner problem is still lower bounded.

**Proposition 4.** Let \( w_{Ak}^* \) and \( w_{Ak+1}^* \) the approximate solutions of the inner problem with respectively the working sets \( A_k \) and \( A_{k+1} \), as defined in Proposition 3. Assume that \( w_{Ak+1}^* \) has been obtained through a tolerance of \( \xi_{k+1} \leq r_k \) of its Fermat’s condition (e.g., for the log sum penalty, Equation (4) are satisfied up to \( \xi_{k+1} \)), then the following inequality holds:

\[
\| w_{Ak+1}^* - w_{Ak}^* \|^2 \geq \frac{1}{\| X \|^2} \text{dist}(r_k, C_j) - \xi_{k+1}.
\]

**Proof.** First note that if \( w_{Ak+1}^* \) is such that \( r_{k+1} \in C_{Ak+1} \), then we are in the same condition than in Proposition 3 and the same proof applies. Let us assume then that \( r_{k+1} \not\in C_{Ak+1} \) and \( \text{dist}(r_{k+1}, C_j) \leq \xi_{k+1} \). Define as \( u \) the point in \( C_j \) that defines the distance of \( r_k \) to \( C_j \) and as \( p \) the point that minimizes the distance between \( r_{k+1} \) and the segment \( [u, r_k] \). Then, owing to simple geometrical arguments and orthogonality we have:

\[
\| r_{k+1} - r_k \|^2 = \| r_{k+1} - p \|^2 + \| p - r_k \|^2.
\]

Thus \( \| r_{k+1} - r_k \| \geq \| r_k - p \| \). Now, because \( p \) belongs to the segment defined by \( u \) and \( r_k \), we have:

\[
\| r_{k+1} - r_k \| \geq \| r_k - u \| - \| u - p \| \geq \text{dist}(r_k, C_j) - \xi_{k+1}
\]

where the last inequality comes from the fact that \( \| u - p \| = \text{dist}(r_{k+1}, C_j) \leq \xi_{k+1} \). Plugging this inequality into Equation (8) completes the proof. \( \square \)

Note that the above lower bound is meaningful only if the tolerance \( \xi_{k+1} \) is smaller than the distance of the residual to the set \( C_j \). This is a reasonable assumption to be made since we expect \( r_k \) to violate \( C_j \). Now, we can derive condition of convergence towards a stationary point of the full problem.

**Corollary 1.** Under the assumption of Theorem 4 and assuming that the sequence of tolerances \( \xi_k \) is such that \( \sum_k \xi_k < \infty \), then Algorithm 4 produces a sequence of iterates that converges towards a stationary point.

The proof follows the same steps as for Theorem 1 with the addition that sequence \( \{ \xi_k \} \) is convergent and thus has been omitted. Note that the assumption of convergent sum of errors is a common assumption, notably in the proximal algorithm literature [2, 33] and it helps guaranteeing convergence towards exact stationary point instead of an approximate convergence.

### 4 Numerical Experiments

**Set-up** We now present some numerical studies showing the computational gain achieved by our approach. Our main baselines are algorithms that also
Table 1: Running time in seconds of different algorithms on different problems. In the first column, we reported data, the tolerance on the stopping criterion and the constant K such that λ = K max_j |x_j| y for (the larger the K, the sparser w^*). The small Toy dataset has n = 100, d = 1000 and p = 30; the large one has n = 1000, d = 5000, p = 500. For each inner solver, we bold the most efficient algorithm. The symbol "–" denotes that the algorithm did not finish one iteration in 24 hours and the 0.0 standard as a mean convergence guarantees, we have conjectured that only one iteration were terminated after 48 hours. The number in parenthesis is the number of non-zero weights in w^*_A.

<table>
<thead>
<tr>
<th>Data and Setting</th>
<th>MM prox</th>
<th>GIST</th>
<th>MaxVC Gist</th>
<th>FireWorks Gist</th>
<th>MM BCD</th>
<th>BCD</th>
<th>MaxVC-BCD</th>
<th>FireWorks BCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toy small - 1.00e-03 - 0.0 - 4.12e-04 (34)</td>
<td>0.81e-02 (44)</td>
<td>0.28e-02 (34)</td>
<td>0.28e-02 (34)</td>
<td>0.28e-02 (34)</td>
<td>3.4e-01 (34)</td>
<td>0.02e-01 (34)</td>
<td>0.02e-01 (34)</td>
<td>4.4e-01 (34)</td>
</tr>
<tr>
<td>Toy small - 1.00e-05 - 0.0 - 1.5e-04 (34)</td>
<td>1.45e-04 (34)</td>
<td>0.63e-04 (34)</td>
<td>0.63e-04 (34)</td>
<td>0.63e-04 (34)</td>
<td>3.4e-04 (34)</td>
<td>0.63e-04 (34)</td>
<td>0.63e-04 (34)</td>
<td>3.4e-04 (34)</td>
</tr>
<tr>
<td>Toy small - 1.00e-03 - 0.01</td>
<td>1.11e-02 (71)</td>
<td>6.1e-02 (71)</td>
<td>1.6e-02 (71)</td>
<td>1.6e-02 (71)</td>
<td>8.3e-01 (71)</td>
<td>7.3e-02 (71)</td>
<td>1.6e-02 (71)</td>
<td>8.3e-02 (71)</td>
</tr>
<tr>
<td>Toy small - 1.00e-05 - 0.01</td>
<td>1.2e-02 (56)</td>
<td>1.1e-02 (56)</td>
<td>1.1e-02 (56)</td>
<td>1.1e-02 (56)</td>
<td>8.2e-02 (56)</td>
<td>8.2e-02 (56)</td>
<td>8.2e-02 (56)</td>
<td>8.2e-02 (56)</td>
</tr>
<tr>
<td>Toy large - 1.00e-03 - 0.07</td>
<td>4.14e-01 (365)</td>
<td>2.6e-01 (365)</td>
<td>5.8e-01 (365)</td>
<td>5.8e-01 (365)</td>
<td>8.2e-01 (365)</td>
<td>8.2e-01 (365)</td>
<td>8.2e-01 (365)</td>
<td>8.2e-01 (365)</td>
</tr>
<tr>
<td>Toy large - 1.00e-05 - 0.07</td>
<td>5.5e-01 (371)</td>
<td>3.8e-01 (371)</td>
<td>3.1e-01 (371)</td>
<td>3.1e-01 (371)</td>
<td>1.4e-00 (371)</td>
<td>1.4e-00 (371)</td>
<td>1.4e-00 (371)</td>
<td>1.4e-00 (371)</td>
</tr>
<tr>
<td>Toy large - 1.00e-03 - 0.0 - 5.69e+04 (786)</td>
<td>5.1e+04 (786)</td>
<td>6.5e+04 (786)</td>
<td>3.4e+04 (786)</td>
<td>3.4e+04 (786)</td>
<td>5.4e+04 (786)</td>
<td>5.4e+04 (786)</td>
<td>5.4e+04 (786)</td>
<td>5.4e+04 (786)</td>
</tr>
<tr>
<td>Toy large - 1.00e-05 - 0.01</td>
<td>-</td>
<td>5.83e+04 (759)</td>
<td>1.02e+05 (759)</td>
<td>6.09e+04 (759)</td>
<td>5.83e+04 (759)</td>
<td>5.83e+04 (759)</td>
<td>5.83e+04 (759)</td>
<td>5.83e+04 (759)</td>
</tr>
</tbody>
</table>

For all approaches, we leverage the closed-form proximal operator available for several (non-convex) regularizers. For our experiments, we have used the log-sum penalty which has an hyperparameter θ that has been set to 1. For all algorithms, the stopping criterion is based on the tolerance (either 10^-3 or 10^-5 ) over Fermat’s optimality condition given in Equation 2. The used performance measure for comparing all algorithms is the CPU running time. For all problems, we have set τ_k adaptively (by sorting as described in Algorithm 1 line 7) so as to add the same fixed number n_added of features into the working set of our FireWorks algorithm and for MaxVC. Results are averaged over 5 different runs.

Toy problem - Here, the regression matrix X ∈ R^n×d is drawn uniformly from a standard Gaussian distribution (zero-mean unit variance). For given n, d and a number p of active variables, the true coefficient vector w^true is obtained as follows. The p non-zero positions are chosen randomly, and their values are drawn from a zero-mean unit variance Gaussian distribution, to which we added ±0.1 according to sign(w^true). Finally, the target vector is obtained as y = Xw^true + e where e is a zero-mean Gaussian noise with standard deviation σ = 0.01. For these problems, we have arbitrarily set n_added = 30 and extra experiments in the appendix illustrates the impact of this choice. Table 1 presents the running time for different algorithms to reach convergence under various settings. We note that our FireWorks algorithm is faster than the genuine inner solver and (at least on par) with the MaxVC approach especially in setting where λ is properly tuned with respect to the number of variables, ie when the solution is not too sparse. Note that the MM+Blitz approaches...
is performing worse than all other methods in almost all settings. We explain this gain by the working set framework and the ability to prune the working set, which size is therefore not monotonically increasing.

Real data We have reported comparisons on three real datasets. The first one is the Leukemia dataset [13] which has a dense regression matrix with $n = 72$ and $d = 7129$. We have also considered sparse problem such as newsgroups dataset in which we have kept only 3 categories (religion, atheism and graphics) resulting in $n = 1441$, $d = 26488$ and 5 categories comp leading to $n = 4891$, $d = 94414$ (see the supplemental for details). For these two problems, we have respectively 223173 and 676247 non-zeros elements in the related design matrix $X$. We have also used a large-scale dataset which is a subset of the Criteo Kaggle dataset composed of 2M samples and 1M features, with about 78M non-zero elements in $X$. For Leukemia, we have $n_{\text{added}} = 30$ at each iteration, whereas we have added 300 and 1000 features respectively for the newsgroup and Criteo problem. Figure 2 presents an example of how objective value and maximum constraint violation (measured as $\max_j (|x_j^T r_k| - r_j'(0))$) evolve during the optimization process for the two newsgroup datasets. We see in these examples that both MaxVC and FireWorks algorithms achieve approximately the same objective value whereas our FireWorks approach converges faster. Quantitative results are reported in the bottom part of Table 1. At first, we can note that the convex relaxation approach using MM and Blitz is always more efficient than the baseline non-convex methods using either BCD or GIST. Moreover, the table also shows that using FireWorks leads to a speedup of at least one order of magnitude compared to the baseline algorithm and the MM approach. For large $\lambda$ leading to sparse solutions, MaxVC is the most efficient approach on Leukemia, while for large-scale datasets newsgroup-3 and newsgroup-5, FireWorks is substantially faster than all competitors. For Criteo, only the BCD working set algorithms are able to terminate in reasonable time and FireWorks is more efficient than MaxVC. Again the MM+Blitz approach is performing worse than the two non-convex active set algorithms and fails to converge in a reasonable time for large datasets.

Running-time on a grid-search. We report here the sum of running time (in seconds) of FireWorks and MaxVC for solving the problem with 10 different values of $\lambda$ varying from 0.06$\lambda_{\text{max}}$ to 0.01$\lambda_{\text{max}}$ with $\lambda_{\text{max}} = \max_j (|x_j^T y|)$ on the Leukemia dataset. For a tolerance of $10^{-5}$, we have for GIST as inner solver, MaxVC runs in 24.3s and Fireworks takes 18.8s while for BCD as inner solver, we have for MaxVC 91s and for Fireworks 53.5s. Hence, in this context, the running time of our approach is still better than the most efficient competitor. We have similar results for the (small) toy problem.

Additional experiments in supplementary. Since our main metric for comparing our algorithm to competitors is its computational running time, as a sanity check, we have also evaluated the quality of the estimate $\hat{w}$. For instance, for the toy problem we have measured whether our approach is able to recover the support of the true vector $w^\text{true}$. Our results show that there is no approach that outperforms the others under other metrics. This makes clear that the gain in running time of FireWorks is not at the expense of worse estimate. We have also reported some studies that analyze the impact of the parameter $n_{\text{added}}$ (and the related $r_k$) and of pruning on the running time of our algorithm FireWorks and on MaxVC. According to our results, the 1% rule seems to be a good heuristic for both algorithms and across the range of parameters, FireWorks is as efficient as MaxVC.

5 Conclusion

We have introduced in this paper a working set based meta-algorithm for non-convex regularized regression. By generalizing the concept of primal-dual approach in a non-convex setting, we were able to derive a novel rule for updating the features optimized by an iterative incremental algorithm. From a theoretical point of view, we showed convergence of the algorithm, even when the inner problem is not solved exactly but up to a certain tolerance. This is in contrast with the classical maximal violating optimality condition algorithms approach whose convergence requires the exact resolution of each inner problem. Our experimental results show the computational gain achieved for a given solver when applied directly on the full variables or within our working set algorithm. The main limitation of our work is that our provably convergent method is not always as efficient as heuristic ones.

Broader and potential negative impact We expect this work to benefit research and applications related to large scale sparse learning problems. Since the work is a methodological work and as such it is hard to see any foreseeable societal consequences without precise applications. The computational gain from our algorithm can be interesting for practitioners from a computational (and financial) perspective but it can also be counterbalanced by the potential use on larger dataset that this can also bring.
Figure 2: Example of evolution of the objective value and the maximum violation constraint on the 0-valued weights. The tolerance on the inner problem is set to $10^{-6}$; (most-left) performance on Neusgroup-3; (most-right) performance on Neusgroup-5.

Acknowledgments

This work benefited from the support of the project Chaire AI RAIMO, 3IA Côte d’Azur Investments ANR-19-P3IA-0002 of the French National Research Agency (ANR) and was performed using computing resources of CRIANN (Normandy, France). This research was produced within the framework of Energy4Climate Interdisciplinary Center (E4C) of IP Paris and Ecole des Ponts ParisTech. This research was supported by 3rd Programme d’Investissements d’Avenir ANR-18-EUR-0006-02. This action benefited from the support of the Chair ”Challenging Technology for Responsible Energy” led by l’X – Ecole polytechnique and the Fondation de l’Ecole polytechnique, sponsored by TOTAL.

References


A Supplementary material
Provably Convergent Working Set Algorithm for Non-Convex Regularized Regression

A.1 Maximum-Violating Optimality Condition Working Set Algorithm

The maximum-violating constraint algorithm is a simple algorithm that solves at each iteration a subproblem with a subset of variables and then add some others that violate the most the statement \( y - X \hat{w}_{A_k} \in C_{A_k} \), where “the most” is evaluated in term of distance to each set \( C_j \), with \( j \in A_k \). Hence, at each iteration, we compute all these distances, sort them in descending order and add to the current working set, the \( n_k \) variables that yield to the largest distances. The algorithm is presented below.

**Algorithm 2 Maximum Violating Constraints Algorithm**

**Input:** \( \{X, y\} \), \( A_1 \) active set, \( n_k \) number of variables to add at iteration \( k \), initial vector \( \hat{w}_{A_0} \)

**Output:** \( \hat{w}_{A_k} \)

1: for \( k = 1, 2, \ldots \) do
2: \( w_{A_k} = \arg \min_{\omega \in A_k} \frac{1}{2} \|y - X_{A_k} \omega \|^2 + \sum_{j \in A_k} r(\|w_j\|) \) - warm-start solver with \( w_{A_{k-1}} \)
3: \( r_k = y - X_{A_k} \hat{w}_{A_k} \) - current residual
4: \( v = \arg\text{sort}(r_k, C_j) \) in descending order
5: \( A_{k+1} = v[1 : n_k] \cup A_k \) - update working set by adding the \( n_k \) most violating variables
6: end for
7: Build \( \hat{w}_{A_k} \)

A.2 Proof of Proposition 1

**Proposition 1.** If \( \omega^*_A \) satisfies Fermat's condition of Problem \( \mathcal{P} \), then for all \( j \in \mathcal{A} \), we have

\[
|\nabla_j \omega^*_A| \leq r'_k(0) \leq 0
\]

where \( r'_k \) is the derivative of \( r_k \).

**Proof.** At first, note that since the function \( r_k \) is monotone and concave, then its derivative is positive and non-increasing. Hence \( \forall w \geq 0, r'_k(w) \leq r'_k(0) \). Now, for \( j \in \{i \in \mathcal{A} : \omega^*_{i,A} = 0\} \), the inequality in Equation 10 naturally comes from Fermat’s condition in Equation 2. When \( j \in \{i \in \mathcal{A} : \omega^*_{i,A} \neq 0\} \), we have \( \nabla_j \omega^*_A = r'_k(\|\omega^*_{i,A}\|) \). Taking the absolute value of this equation and plugging in the inequality of the derivatives concludes the proof.

\[\]

A.3 Proof of Proposition 2

**Proposition 2.** Given a working set \( \mathcal{A}_k \) and \( \omega^*_{A_k} \) solving the related restricted problem, \( \bar{w}^*_{A_k} \) is also optimal for the full problem if and only if \( \alpha = 1 \) (which also means \( s_{k+1} = r_k \)).

**Proof.** Assume that \( \omega^*_{A_k} \) and \( \bar{w}^*_{A_k} \) are optimal respectively for the restricted and the full problem.

Let us show that in this case \( \alpha_k = 1 \). Since \( \bar{w}^*_{A_k} \) is optimal for the full problem, we have \( \forall j \in \mathcal{A}_k, |x_j^\top(y - X \bar{w}^*_{A_k})| \leq r'_k(0) \). And thus we have the following equivalent statement

\[
y - X \bar{w}^*_{A_k} \in C \iff y - X_{A_k} \omega^*_{A_k} \in C \iff r_k \in C
\]

and thus \( \alpha_k = 1 \).

Now assume that \( \alpha_k = 1 \) and let us show that \( \bar{w}^*_{A_k} \) is optimal for the full problem. Since \( \alpha_k = 1 \), we have \( s_{k+1} = r_k \) and thus \( r_k \in C \). The latter means that \( \forall j \in \mathcal{A}_k, |x_j^\top(y - X_{A_k} \omega^*_{A_k})| \leq r'_k(0) \) and thus \( \forall j \in \mathcal{A}_k, |x_j^\top(y - X \bar{w}^*_{A_k})| \leq r'_k(0) \). Given this last property and the definition of \( \bar{w}^*_{A_k} \) based on \( \omega^*_{A_k} \), we can conclude that \( \bar{w}^*_{A_k} \) is optimal for the full problem.

\[\]

A.4 Proof of Lemma 1

The proof follows similar steps as those given by John- son and Guestrin [18].

**Lemma 1.** At step \( k \geq 2 \), consider a constraint \( C_j \) such that \( h_j(r_k) > 0 \) and \( h_j(s_k) < 0 \) then

\[
\text{dist}(r_k, C_j) \geq \frac{1 - \alpha_k}{\alpha_k \alpha_k} r_{k-1} . \quad (11)
\]

**Proof.** Denote as \( j \) the index of the function \( h_j \) such that \( h_j(r_k) > 0 \) and \( h_j(s_k) < 0 \). Let’s \( z_k \in \{z \in \mathbb{R}^n : h_j(z) = 0\} \). The following equality holds

\[
\text{dist}(r_k, C_j) = \|z_k - r_k\| = \|z_k - \frac{1}{\alpha_k}(s_{k+1} - (1 - \alpha_k)s_k)\| = \|z_k - \frac{1}{\alpha_k} s_{k+1} + \frac{1 - \alpha_k}{\alpha_k} s_k\| = \|z_k + \frac{1 - \alpha_k}{\alpha_k} s_{k+1} - s_k\| = \frac{1 - \alpha_k}{\alpha_k} \|z_k - s_k\| = \frac{1 - \alpha_k}{\alpha_k} \|s_k + \frac{1 - \alpha_k}{\alpha_k} s_{k+1} - s_k\| \quad (12)
\]

Note that because \( h_j(r_k) > 0 \) and \( h_j(s_k) < 0 \), \( \alpha_k \neq 0 \) since \( h_j \) is a continuous function. By construction, we have \( h_j(z_k) = 0 \) as \( z_k \) is a minimizer of the distance and \( h_j(s_{k+1}) = 0 \) as we have chosen \( j \) as the
Now as of the construction of $A$ with $j \notin A$, since $S$ do not lead to a convex combination of $z$ and $s_{k+1}$ hence, we have $h_j(\frac{-\alpha_k}{1-\alpha_k}z_k + \frac{1}{1-\alpha_k}s_{k+1}) \geq 0$. On the other hand by construction, we have $s_k \in C_j$. Furthermore, we have $\text{dist}(s_k, C_j) \geq \tau_{k-1}$. Indeed, since $h_j(r_k) > 0$, we have $j \notin A_k$ as by construction $r_k \in C_{A_k}$ (w1 has been optimized over $A_k$). Because $j \notin A_k$ means that $\text{dist}(s_k, C_j) \geq \tau_{k-1}$, by definition of the construction of $A_k$ in Algorithm 1.

Now as $h_j(\frac{-\alpha_k}{1-\alpha_k}z_k + \frac{1}{1-\alpha_k}s_{k+1}) \geq 0$ and $\text{dist}(s_k, C_j^c) \geq \tau_{k-1}$, the norm in the above equation (12) is lower bounded by $\tau_k$ and we have

$$\text{dist}(r_k, C_j) \geq 1 - \frac{\alpha_k}{\alpha_k} \tau_{k-1}.$$ \hfill $\square$

### A.5 Proof of Theorem 1

**Theorem 1.** Suppose that for each step $k$, the algorithm solving the inner problem ensures a decrease in the objective value in the form

$$f(\tilde{w}_{A_k}^*) - f(w_{A_k}^*) \leq -\gamma_k \|w_{A_k}^* - \tilde{w}_{A_k}^*\|^2.$$  

with $\forall k$, $\gamma_k \geq \gamma > 0$. For the inner solver, we also impose that when solving the problem with set $A_{k+1}$, the inner solver is warm-started with $w_{A_k}^*$. Assume also that $\|X\|_2 > 0$, $\tau_k \geq \tau > 0$ and $h_j$ satisfies assumption in Lemma 1 then the sequence of $\alpha_k$ produced by Algorithm 1 converges towards 1 and $\forall j$, $\lim_{k \to \infty} |x_j|^T r_k | \leq r_j(0)$.  

**Proof.** Before going into details, note that pruning $w_{A_k}^*$ before warm-starting does not affect $f(\tilde{w}_{A_k}^*)$, and thus the proof still holds for that situation. Using results in Proposition 3 and Lemma 1 and the above assumption, we have, for $k \geq 2$,

$$f(\tilde{w}_{A_{k+1}}^*) - f(\tilde{w}_{A_k}^*) \leq -\gamma_k \|w_{A_k}^* - \tilde{w}_{A_k}^*\|^2.$$  

This means that

$$\sum_{i=2}^{k} \sum_{i=2}^{\tau_{k-1}} \gamma_j \left(1 - \frac{\alpha_k}{\alpha_k}\right)^2 \tau_{k-1}^2 \leq f(\tilde{w}_{A_{k+1}}^*) - f(\tilde{w}_{A_{k-1}}^*).$$  

Since $f$ is bounded from below, the right hand side is less than some positive constant, hence $\sum_{i=2}^{\tau_{k-1}} \gamma_j \left(1 - \frac{\alpha_k}{\alpha_k}\right)^2 \tau_{k-1}^2 < \infty$. Since the latter sum is bounded, it implies that $\gamma_j \left(1 - \frac{\alpha_k}{\alpha_k}\right)^2 \tau_{k-1} \to 0$ as $\tau \to \infty$, and as $\gamma_j \geq \gamma > 0$, $\tau_k \geq \tau > 0$, we have $\lim_{k \to \infty} \alpha_k = 1$. Now using the definition of $s_{k+1}$, we have $\forall j$, $x_j^T r_k = \frac{1}{\alpha_k} x_j^T s_{k+1} - \frac{1}{\alpha_k} x_j^T s_k$. Then, taking the absolute value, triangle inequality, using the fact that $\forall k$, $s_k \in C$ and taking the limit concludes the proof. \hfill $\square$

### A.6 Experimental analysis

#### A.6.1 Data

The toy dataset has been built from scratch and can be reproduced from the companion code of the paper. The Leukemia dataset we have used is available at [https://web.stanford.edu/~hastie/CASI_files/DATA/leukemia.html](https://web.stanford.edu/~hastie/CASI_files/DATA/leukemia.html). The Newsgroup dataset is part of the Sklearn dataset package. The 3 categories is composed of the topic : talk.religion.misc, comp.graphics and alt.atheism. The 5 categories is composed by comp.graphics, comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware, comp.windows.x. We have used the natural default train split as proposed by sklearn [24] and the features are based on TF-IDF representation (using the tfidf function of sklearn) keeping default parameters.
A.6.2 Comparing on other metrics

The main contribution of our work is to propose a working set algorithm for sparse non-convex regression problem with theoretical guarantees of convergence. We have shown that the main benefit of this algorithm is its computational efficiency.

We report below some results on other metrics. We want to show that there is no approach outperforming the others. For the Large toy problem, we report the objective value (white background, top) and support recovery F-measure (in percent) (blue background, middle). For the Leukemia dataset, once feature selection has been performed, we report the classification accuracy in percent, (averaged over 5 trials) of a linear SVM trained on the non-zero features of a part of the dataset (50/22 sample splits). Remind that for Leukemia, there is a computational gain of more than 30 between GIST and Fireworks GIST.

A.6.3 On the effect of the number of features to add

In working set algorithms, the number of features to add \( n_{added} \) to the working set at each iteration can be considered as an hyperparameter. Usually, one adds one feature at each iteration but it is not clear whether it is an optimal choice. In the results we reported in Table 1 for the toy problems we fixed \( n_{added} = 30 \). We report in Figure 3 the running time (averaged over 5 runs) we obtain for the Large toy problem (which has 5000 features and 500 informative ones), with respects to that parameter \( n_{added} \). Note that we have reported the performance of MaxVC, a version of MaxVC with pruning (feature with zero weights are removed from \( A_k \)) and our FireWorks using a BCD algorithm as an inner solver.

We remark that for most configurations, adding 1 feature at a time is not optimal and a better choice is to add between 20 to 40 features at a time. When comparing the performance of the different algorithms, as we anticipated, FireWorks is mostly as efficient as MaxVC and its variants. However, we want to emphasize again that MaxVC and its variants are algorithms without convergence proofs, and thus we believe that FireWorks achieves the best compromise between theoretical supported and practical efficiency.
Figure 3: Running time of MaxVC, Max VC with pruning, and our FireWorks on the Large toy problem. The four panels vary in the choice of $K$ in the regularization parameter expressed as $\lambda = K \max_j |x_j^T y|$ and in the tolerance $t$ on the stopping criterion. (top-left) $K = 0.07$ and $t = 1e^{-3}$ (top-right) $K = 0.07$ and $t = 1e^{-5}$. (bottom-left) $K = 0.01$ and $t = 1e^{-3}$ (bottom-right) $K = 0.01$ and $t = 1e^{-5}$. 
<table>
<thead>
<tr>
<th>Data - tol - K</th>
<th>MM prox</th>
<th>GIST</th>
<th>MaxVC GIST</th>
<th>FireWorks GIST</th>
<th>MM BCD</th>
<th>BCD</th>
<th>MaxVC BCD</th>
<th>FireWorks BCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toy large - 1.00e-04 - 0.07</td>
<td>75.8±4.8</td>
<td>76.5±8.4</td>
<td>76.5±8.4</td>
<td>76.5±8.6</td>
<td>75.6±0.0</td>
<td>76.5±8.5</td>
<td>76.5±8.4</td>
<td>76.5±8.6</td>
</tr>
<tr>
<td>Toy large - 1.00e-05 - 0.07</td>
<td>-</td>
<td>76.5±8.4</td>
<td>76.5±8.5</td>
<td>76.5±8.6</td>
<td>75.6±0.0</td>
<td>76.5±8.5</td>
<td>76.5±8.5</td>
<td>76.5±8.6</td>
</tr>
<tr>
<td>Toy large - 1.00e-03 - 0.01</td>
<td>11.5±0.9</td>
<td>11.5±1.4</td>
<td>11.6±1.4</td>
<td>11.5±1.4</td>
<td>11.5±0.0</td>
<td>11.5±1.4</td>
<td>11.6±1.4</td>
<td>11.5±1.4</td>
</tr>
<tr>
<td>Toy large - 1.00e-05 - 0.01</td>
<td>-</td>
<td>11.5±1.4</td>
<td>11.5±1.4</td>
<td>11.5±1.4</td>
<td>11.5±0.0</td>
<td>11.5±1.4</td>
<td>11.5±1.4</td>
<td>11.5±1.4</td>
</tr>
<tr>
<td>Toy large - 1.00e-03 - 0.07</td>
<td>43.6±2.9</td>
<td>44.4±2.9</td>
<td>43.7±3.9</td>
<td>44.2±3.5</td>
<td>43.1±0.0</td>
<td>44.2±2.7</td>
<td>43.6±3.4</td>
<td>44.4±3.6</td>
</tr>
<tr>
<td>Toy large - 1.00e-05 - 0.07</td>
<td>-</td>
<td>44.4±2.9</td>
<td>42.8±4.2</td>
<td>43.9±3.2</td>
<td>43.6±0.0</td>
<td>43.9±2.6</td>
<td>42.8±4.2</td>
<td>43.9±3.2</td>
</tr>
<tr>
<td>Toy large - 1.00e-03 - 0.01</td>
<td>39.1±2.3</td>
<td>39.1±1.1</td>
<td>38.3±1.7</td>
<td>39.3±1.3</td>
<td>37.4±0.0</td>
<td>38.4±1.9</td>
<td>38.4±1.9</td>
<td>39.4±1.2</td>
</tr>
<tr>
<td>Toy large - 1.00e-05 - 0.01</td>
<td>-</td>
<td>39.4±1.7</td>
<td>39.2±1.5</td>
<td>39.8±1.7</td>
<td>38.9±0.0</td>
<td>38.7±1.7</td>
<td>39.0±1.2</td>
<td>39.1±2.1</td>
</tr>
<tr>
<td>Leukemia - 1.00e-03 - 0.07</td>
<td>90.00±5.3</td>
<td>91.82±3.4</td>
<td>90.00±5.3</td>
<td>90.91±6.4</td>
<td>90.00±5.3</td>
<td>88.18±6.2</td>
<td>90.91±6.4</td>
<td>90.91±6.4</td>
</tr>
<tr>
<td>Leukemia - 1.00e-05 - 0.07</td>
<td>86.36±6.4</td>
<td>91.82±3.4</td>
<td>89.09±4.6</td>
<td>91.82±5.3</td>
<td>87.27±6.0</td>
<td>89.09±6.8</td>
<td>90.91±6.4</td>
<td>90.00±7.8</td>
</tr>
<tr>
<td>Leukemia - 1.00e-03 - 0.01</td>
<td>95.45±4.1</td>
<td>96.36±3.4</td>
<td>95.45±2.9</td>
<td>95.45±4.1</td>
<td>95.45±4.1</td>
<td>92.73±4.6</td>
<td>92.73±4.6</td>
<td>97.27±2.2</td>
</tr>
<tr>
<td>Leukemia - 1.00e-05 - 0.01</td>
<td>96.59±3.8</td>
<td>96.96±3.4</td>
<td>94.55±3.4</td>
<td>93.64±2.2</td>
<td>95.45±4.1</td>
<td>92.73±5.5</td>
<td>94.55±3.4</td>
<td>93.64±2.2</td>
</tr>
</tbody>
</table>