Accelerated Projected Gradient Algorithms for Sparsity Constrained Optimization Problems

Anonymous Author(s)
Affiliation
Address
email

Abstract

We consider the projected gradient algorithm for the nonconvex best subset selection problem that minimizes a given empirical loss function under an ℓ_0-norm constraint. Through decomposing the feasible set of the given sparsity constraint as a finite union of linear subspaces, we present two acceleration schemes with global convergence guarantees, one by same-space extrapolation and the other by subspace identification. The former fully utilizes the problem structure to greatly accelerate the optimization speed with only negligible additional cost. The latter leads to a two-stage meta-algorithm that first uses classical projected gradient iterations to identify the correct subspace containing an optimal solution, and then switches to a highly-efficient smooth optimization method in the identified subspace to attain superlinear convergence. Experiments demonstrate that the proposed accelerated algorithms are magnitudes faster than their non-accelerated counterparts as well as the state of the art.

1 Introduction

We consider the sparsity-constrained optimization problem in $\Re^n$:

$$\min_{w \in A_s} f(w), \tag{1}$$

where f is convex with L-Lipschitz continuous gradient, $s \in \mathbb{N}$, and A_s is the sparsity set given by

$$A_s \coloneqq \{w \in \Re^n : \|w\|_0 \leq s\}, \tag{2}$$

where $\|w\|_0$ denotes the ℓ_0-norm that indicates the number of nonzero components in w. We further assume that f is lower-bounded on A_s.

A classical problem that fits in the framework of (1) is the best subset selection problem in linear regression [4, 15]. Given a response vector $y \in \Re^m$ and a design matrix of explanatory variables $X \in \Re^{m \times n}$, traditional linear regression minimizes a least squares loss function

$$f(w) = \|y - Xw\|^2/2. \tag{3}$$

However, due to either high dimensionality in terms of the number of features n or having significantly fewer instances m than features n (i.e., $m \ll n$), we often seek a linear model that selects only a subset of the explanatory variables that will best predict the outcome y. Towards this goal, we can solve (1) with f given by (3) to fit the training data while simultaneously selecting the best-s features. Indeed, such a sparse linear regression problem is fundamental in many scientific applications, such as high-dimensional statistical learning and signal processing [16]. The loss in (3) can be generalized to the following linear empirical risk to cover various tasks in machine learning beyond regression

$$f(w) = g(Xw), \quad g(z) = \sum\nolimits_{i=1}^{m} g_i(z_i), \tag{4}$$

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

where g is convex. Such a problem structure makes evaluations of the objective and its derivatives highly efficient, and such efficient computation is a key motivation for our algorithm development for (1) in this work.

The discontinuous cardinality constraint in (1) makes the problem difficult to solve. To make the optimization problem easier, a popular approach is to slightly sacrifice the quality of the solution (either not strictly satisfying the sparsity level constraint or the prediction performance is deteriorated) to use continuous surrogate functions for the ℓ_0-norm, which lead to a continuous nonlinear programming problem, where abundant algorithms are at our disposal. For instance, using a convex penalty surrogate such as the ℓ_1-norm in the case of LASSO [26], the problem (1) can be relaxed into a convex (unconstrained) one that can be efficiently solved by many algorithms. Other algorithms based on continuous nonconvex relaxations such as the use of smoothly clipped absolute deviation [11] and the minimax concave penalty [31] regularizers are also popular in scenarios with a higher level of noise and outliers in the data. However, for applications in which enforcing the constraints or getting the best prediction performance is of utmost importance, solving the original problem (1) is inevitable. But unfortunately, methods for this problem is not as well-studied as those for the surrogate problems. Moreover, existing methods are indeed still preliminary and too slow to be useful in large-scale problems often faced in modern machine learning tasks.

In view of the present unsatisfactory status for scenarios that simultaneously involve high-volume data and need to get the best prediction performance, this work proposes efficient algorithms to directly solve (1) with large-scale data. To our knowledge, the only algorithm that directly tackles (1) without the use of surrogates is the well-known projected gradient (PG) algorithm [8, 9].[1] Beck and Teboulle [8] proved linear convergence of the objective value with the least squares loss function (3) for the iterates generated by PG under a scalable restricted isometry property, which also served as their tool to accelerate PG. However, given any problem instance, it is hard, if not computationally impossible, to verify whether the said property holds. On the other hand, Bertsimas et al. [9] established global subsequential convergence to a stationary point for the iterates of PG on (1) without the need for such isometry conditions, and their results are valid for general loss functions f beyond (3). While some theoretical guarantees are known, the practicality of PG for solving (1) remains a big problem in real-world applications as its empirical convergence speed tends to be slow. Recently, "acceleration" approaches for PG on general nonconvex regularized problems have been studied in [19, 27]. While their proposed algorithms are also applicable to (1), the obtained convergence speed for nonconvex problems is not faster than that of PG.

In this work, we revisit the projected gradient algorithm for solving the general problem (1) and propose two acceleration schemes by leveraging the combinatorial nature of ℓ_0-norm. In particular, we decompose the feasible set A_s as the finite union of s-dimensional linear subspaces, each representing a subset of the coordinates $\{1, \ldots, n\}$, as detailed in (7) of Section 2. Such subspaces are utilized in devising techniques to efficiently accelerate PG. Our first acceleration scheme is based on a same-space extrapolation technique such that we conduct extrapolation only when two consecutive iterates w_{k-1} and w_k lie in the same linear subspace, and the step size for this extrapolation is determined by a spectral initialization combined with backtracking linesearch to ensure sufficient function decrease. This approach is motivated by the observation that for (4), objective and derivatives at the extrapolated point can be inferred efficiently through a linear combination of Xw_{k-1} and Xw_k, making the extrapolation step extremely cheap. The second acceleration technique starts with the plain PG method, and when consecutive iterates stay in the same subspace, it begins to alternate between a full PG step and a truncated Newton step in the subspace to obtain superlinear convergence with extremely low computational cost. Accordingly, our main contributions are as follows:

1. We prove that PG for (1) is globally convergent to a local optimum with a local linear rate, improving upon the sublinear results of Bertsimas et al. [9]. We emphasize that our framework, like [9], is applicable to general loss functions f satisfying the convexity and smoothness requirements, and therefore covers not only the classical sparse regression problem but also many other ones encompassed by the empirical risk minimization (ERM) framework.
2. By decomposing A_s as the union of linear subspaces, we further show that PG is provably capable of identifying a subspace containing a local optimum of (1). By exploiting this property, we propose two acceleration strategies with practical implementation and convergence guarantees

[1] Gotoh et al. [13] proposed an algorithm for a similar optimization problem that minimizes $f(w) + C\|w\|_0$ for some $C > 0$. But whether it is equivalent to (1) is unclear because both problems are nonconvex, and for any prespecified sparsity level s, it is hard to find C that leads to a solution w^* with $\|w^*\|_0 = s$.

for the general problem class (1). Our acceleration provides both computational and theoretical advantages for convergence, and can in particular obtain superlinear convergence.

3. In comparison with existing acceleration methods for nonconvex problems [19, 27], this work provides new acceleration schemes with faster theoretical speeds (see Theorems 3.2 and 3.3), and beyond being applied to the classical PG algorithm, those schemes can also easily be combined with existing accelerated PG approaches to further make them converge even faster.
4. Numerical experiments exemplify the significant improvement in both iterations and running time brought by our acceleration methods, in particular over the projected gradient algorithm by [9] as well as the accelerated proximal gradient method for nonconvex problems proposed by [19].

This work is organized as follows. We review the projected gradient algorithm and prove its local linear convergence and subspace identification for arbitrary smooth loss functions in Section 2. In Section 3, we propose the acceleration schemes devised through decomposing the constraint set in (1) into subspaces of $\Re^n$. Experiments in Section 4 then illustrate the effectiveness of the proposed acceleration techniques, and Section 5 concludes this work. All proofs, details of the experiment settings, and additional experiments are in the appendices.

2 Projected Gradient Algorithm

The *projected gradient algorithm* for solving (1) is given by the iterations

$$w^{k+1} \in T_{\mathrm{PG}}^{\lambda}(w^k) \coloneqq P_{A_s}(w^k - \lambda \nabla f(w^k)), \tag{5}$$

where $P_{A_s}(w)$ denotes the projection of w onto A_s, which is set-valued because of the nonconvexity of A_s. When f is given by (3), global linear convergence of this algorithm under a restricted isometry condition is established in [8]. For a general convex f with L-Lipschitz continuous gradients, that is,

$$\|\nabla f(w) - \nabla f(w')\| \leq L\|w - w'\| \quad \forall w, w' \in \Re^n, \tag{6}$$

the global subsequential convergence of (5) is proved in [9], but neither global nor local rates of convergence is provided. In this section, we present an alternative proof of global convergence and more importantly establish its local linear convergence.

A useful observation that we will utilize in the proofs of our coming convergence results is that the nonconvex set A_s given by (2) can be decomposed as a finite union of subspaces in $\Re^n$:

$$A_s = \bigcup\nolimits_{J \in \mathcal{J}_s} A_J, \quad A_J \coloneqq \operatorname{span}\{e_j : j \in J\}, \quad \mathcal{J}_s \coloneqq \{J \subseteq \{1, 2, \ldots, n\} : |J| = s\}, \tag{7}$$

where e_j is the jth standard unit vector in $\Re^n$.

Theorem 2.1. *Let $\{w^k\}$ be a sequence generated by* (5) *with $\lambda \in (0, L^{-1})$. Then:*

(a) (Subsequential convergence) The sequence $\{f(w^k)\}$ is strictly decreasing, and any accumulation point w^* of $\{w^k\}$ is a stationary point of (1)*, or equivalently $w^* \in P_{A_s}(w^* - \lambda \nabla f(w^*))$.*

(b) (Subspace identification and full convergence) There exists $N \in \mathbb{N}$ such that

$$\{w^k\}_{k=N}^{\infty} \subseteq \bigcup\nolimits_{J \in \mathcal{I}_{w^*}} A_J, \qquad \mathcal{I}_{w^*} \coloneqq \{J \in \mathcal{J}_s : w^* \in A_J\}. \tag{8}$$

whenever $w^k \to w^*$. In particular, if $T_{\mathrm{PG}}^{\lambda}(w^*)$ is a singleton for an accumulation point w^* of $\{w^k\}$, then w^* is a local minimum for (1)*, $w^k \to w^*$, and* (8) *holds.*

(c) (Q-linear convergence) If $T_{\mathrm{PG}}^{\lambda}(w^*)$ is a singleton for an accumulation point w^* and $w \mapsto w - \lambda \nabla f(w)$ is a contraction over A_J for all $J \in \mathcal{I}_{w^*}$, then $\{w^k\}$ converges to w^* at a Q-linear rate. In other words, there is $N_2 \in \mathbb{N}$ and $\gamma \in [0, 1)$ such that

$$\left\|w^{k+1} - w^*\right\| \leq \gamma \left\|w^k - w^*\right\|, \quad \forall k \geq N_2. \tag{9}$$

It is well-known that an optimal solution of (1) is also a stationary point of it [6, Theorem 2.2], and therefore (a) proves the global subsequential convergence of PG to candidate solutions of (1). Consider $z^* \coloneqq w^* - \lambda \nabla f(w^*)$, and let τ be a permutation of $\{1, \ldots, n\}$ such that $z^*_{\tau(1)} \geq z^*_{\tau(2)} \geq \cdots \geq z^*_{\tau(n)}$. The requirement of $T_{\mathrm{PG}}^{\lambda}(w^*)$ being a singleton in Theorem 2.1 (b) then simply means the mild condition of $z^*_{\tau(s)} > z^*_{\tau(s+1)}$, which is almost always true in practice. The requirement for (c) can be fulfilled when f confined to A_J is strongly convex, even if f itself is not. This often holds true

in practice when f is of the form (4) and we restrict s in (1) to be smaller than the number of data instances m, and is thus also mild. The existence of a stationary point can be guaranteed when $\{w^k\}$ is a bounded sequence, often guaranteed when f is coercive on A_J for each $J \in \mathcal{J}_s$.

In comparison to existing results in [9, 1, 10], parts (b) and (c) of Theorem 2.1 are new. In particular, part (b) provides a full convergence result that usually requires stronger regularity assumptions like the Kurdyka-Łojasiewicz (KL) condition [1, 10]. Part (c) gives a local linear convergence for the PG iterates even if the problem is nonconvex, while the rates in [10] requires a KL condition and the rate is measured in the objective value.

The following result further provides rates of convergence of the objective values even without the conventional KL assumption.

Theorem 2.2. *Let $\{w^k\}$ be a sequence generated by* (5) *with $\lambda \in (0, L^{-1})$. If $w^k \to w^*$, such as when $T^\lambda_{\mathrm{PG}}(w^*)$ is a singleton at an accumulation point w^* of* (5)*, then*

$$f(w^k) - f(w^*) = O(k^{-1}). \tag{10}$$

Moreover, under the hypothesis of Theorem 2.1 (c), the objective converges to $f(w^*)$ R-linearly, i.e.,

$$f(w^k) - f(w^*) = O(\exp(-k)). \tag{11}$$

By using Theorem 2.1, we can also easily get rates faster than 10 under a version of the KL condition weaker than those assumed in existing works. In particular, existing analyses require the KL condition to hold in a neighborhood in $\Re^n$ of an accumulation point, but we just need it to hold around w^* within A_J for the restriction $f|_{A_J}$ for each $J \in \mathcal{I}_{w^*}$. These results are postponed to Theorem 3.2 in the next section as the PG method is a special case of our acceleration framework.

3 Accelerated methods

The main focus of this work is the proposal in this section of new techniques with solid convergence guarantees to accelerate the PG algorithm presented in the preceding section. Our techniques fully exploit the subspace identification property described by the inclusion (8), as well as the problem structure of (4) to devise efficient algorithms.

We emphasize that the two acceleration strategies described below can be combined together, and they are also widely applicable such that they can be employed to other existing algorithms for (1) as long as such algorithms have a property similar to (8).

3.1 Acceleration by extrapolation

Traditional extrapolation techniques are found in the realm of convex optimization to accelerate algorithms [7, 22] with guaranteed convergence improvements, but were often only adopted as heuristics in the nonconvex setting, until some recent works showed that theoretical convergence can also be achieved [19, 27]. However, unlike the convex case, these extrapolation strategies for nonconvex problems do not lead to faster convergence speed nor an intuitive reason for doing so. An extrapolation step proceeds by choosing a *positive* stepsize along the direction determined by two consecutive iterates. That is, given two iterates w^{k-1} and w^k, an intermediate point $z^k := w^k + t_k(w^k - w^{k-1})$ for some stepsize $t_k \geq 0$ is first calculated before applying the original algorithmic map (T^λ_{PG} in our case).[2]

Another popular acceleration scheme for gradient algorithms is the spectral approach pioneered by [3]. They take the differences of the gradients and of the iterates in two consecutive iterations to estimate the curvature at the current point, and use it to decide the step size for updating along the reversed gradient direction. It has been shown in [29] that equipping this step size with a backtracking procedure leads to significantly faster convergence for proximal gradient on regularized optimization problems, which includes our PG for (1) as a special case.

To describe our proposed double acceleration procedure that combines extrapolation and spectral techniques, we first observe that all PG iterates lie on A_s, and that A_s can be finitely decomposed as (7). When two consecutive iterates lie on the same convex subspace A_J for some $J \in \mathcal{J}_s$, within these

[2] it is clear that if $t_k \equiv 0$, we reduce back to the original algorithm.

two iterations, we are actually conducting convex optimization. In this case, an extrapolation step within A_J is reasonable because it will not violate the constraint, and acceleration can be expected from the improved rates of accelerated proximal gradient on convex problems in [7, 23]. Judging from Theorem 2.1 (b), the corresponding J is also a candidate index set that belongs to $\mathcal{I}_{w^*}$, so extrapolation within A_J makes further sense. We set $t_k = 0$ to skip the extrapolation step if d^k is not a descent direction for f at w^k. Otherwise, we start from some $\hat{t}_k > 0$ decided by the curvature information of f, and then execute a backtracking linesearch along $d^k := w^k - w^{k-1}$ to set $t_k = \eta^i \hat{t}_k$ for the smallest integer $i \geq 0$ that provides sufficient descent

$$f(w^k + t_k d^k) \leq f(w^k) - \sigma t_k^2 \|d^k\|^2, \tag{12}$$

given parameters $\eta, \sigma \in (0, 1)$. We then apply (5) to $z^k = w^k + t_k d^k$ to obtain w^{k+1}.

For the spectral initialization $\hat{t}_k$ for accelerating the convergence, instead of directly using approaches of [3, 29] that takes the reversed gradient as the update direction, we need to devise a different mechanism as our direction d^k is not directly related to the gradient. We observe that for the stepsize

$$\alpha_k := \left\langle s^k, s^k \right\rangle / \left\langle s^k, r^k \right\rangle, \quad s^k := w^k - w^{k-1}, \quad r^k := \nabla f(w^k) - \nabla f(w^{k-1}) \tag{13}$$

used in [3], the final update $-\alpha_k \nabla f(w^k)$ is actually the minimizer of the following subproblem

$$\min_{d \in \Re^n} \quad \left\langle \nabla f(w^k), d \right\rangle + \|d\|^2 / (2\alpha_k). \tag{14}$$

By juxtaposing the above quadratic problem and the upper bound provided by the descent lemma [5, Lemma 5.7], we can view α_k^{-1} as an estimate of the local Lipschitz parameter that could be much smaller than L but still guarantee descent of the objective. We thus follow this idea to decide $\hat{t}_k$ using such curvature estimate and the descent lemma by

$$\hat{t}_k = \arg\min_{t \geq 0} \ \left\langle \nabla f(w^k), t d^k \right\rangle + \left\| t d^k \right\|^2 / (2\alpha_k) \quad \Leftrightarrow \quad \hat{t}_k = -\left\langle \alpha_k \nabla f(w^k), d^k \right\rangle / \left\| d^k \right\|^2. \tag{15}$$

Another interpretation of (13) is that $\alpha_k^{-1} I$ also serves as an estimate of $\nabla^2 f(w^k)$,[3] and the objective in (14) is a low-cost approximation of the second-order Taylor expansion of f. However, we notice that for problems in the form of (4) and with $d^k \in A_J$, the exact second-order Taylor expansion

$$f(w^k + t d^k) \approx f(w^k) + t \left\langle \nabla f(w^k), d^k \right\rangle + t^2 \left\langle \nabla^2 f(w^k) d^k, d^k \right\rangle / 2 \tag{16}$$

can be calculated efficiently. In particular, for (4) and any $d^k \in A_J$, we get from $X d^k = X_{:,J} d_J^k$:

$$\begin{aligned} \nabla f(w^k)^\top d^k &= \nabla g\left((X w^k)\right)^\top \left(X_{:,J} d_J^k\right), \\ \left\langle \nabla^2 f(w^k) d^k, d^k \right\rangle &= \left\langle (X_{:,J} d_J^k), \nabla^2 g\left((X w^k)\right) (X_{:,J} d_J^k) \right\rangle, \end{aligned} \tag{17}$$

which can be calculated in $O(ms)$ time by computing $X_{:,J} d_J^k$ first. This $O(ms)$ cost is much cheaper than the $O(mn)$ one for evaluating the full gradient of f needed in the PG step, so our extrapolation plus spectral techniques has only negligible cost. Moreover, for our case of $d^k = w^k - w^{k-1}$, we can further reduce the cost of calculate $X_{:,J} d_J^k$ and thus (17) to $O(m)$ by recycling intermediate computational results needed in evaluating $f(w^k)$ through $X_{:,J} d_J^k = X w^k - X w^{k-1}$. With such tricks for efficient computation, we therefore consider the more accurate approximation to let $\hat{t}_k$ be the scalar that minimizes the quadratic function on the right-hand side of (16) for problems in the form (4). That is, we use

$$\hat{t}_k := -\left\langle \nabla f(w^k), d^k \right\rangle / \left\langle \nabla^2 f(w^k) d^k, d^k \right\rangle. \tag{18}$$

Finally, for both (18) and (15), we safeguard $\hat{t}_k$ by

$$\hat{t}_k \leftarrow P_{[c_k \alpha_{\min}, c_k \alpha_{\max}]}\left(\hat{t}_k\right) \tag{19}$$

for some fixed $\alpha_{\max} \geq \alpha_{\min} > 0$, where

$$c_k := \left\| (\nabla f(w^k))_J \right\| / (\zeta_k \left\| d^k \right\|), \quad \zeta_k := -\left\langle d^k, \nabla f(w^k) \right\rangle / (\left\| d^k \right\| \left\| (\nabla f(w^k))_J \right\|) \in (0, 1]. \tag{20}$$

We also note that the low cost of evaluating $X d^k$ is also the key to making the backtracking in (12) practical, as each $f(w^k + \eta^i \hat{t}_k d^k)$ can be calculated in $O(m)$ time through linear combinations of $X w^k$ and $X d^k$. The above procedure is summarized in Algorithm 1, whose global convergence is guaranteed by Theorem 3.1, and whose full convergence and convergence rates under a relaxed KL condition are provided in Theorem 3.2.

[3] As ∇f is Lipschitz continuous, it is differentiable almost everywhere. Here, we denote by $\nabla^2 f(w^k)$ a generalized Hessian of f at w, which is well-defined for f with Lipschitz continuous gradient [14].

Algorithm 1: Accelerated projected gradient algorithm by extrapolation (APG)

Given an initial vector $w^0 \in \Re^n$ and parameters $\eta, \sigma \in (0,1)$, $\alpha_{\max} \geq \alpha_{\min} > 0$, $\lambda \in (0, 1/L)$.
for $k = 0, 1, 2, \ldots$ **do**
  if $k > 0$*;* w^{k-1} *and* w^k *activate the same* A_J*; and* $\langle w^k - w^{k-1}, \nabla f(w^k) \rangle < 0$ **then**
    $d^k \leftarrow w^k - w^{k-1}$, and compute $\hat{t}_k$ from (19) with either (15) or (18)
    for $i = 0, 1, \ldots$ **do**
      $t_k \leftarrow \eta^i \hat{t}_k$
      if (12) *is satisfied* **then** $z^k \leftarrow w^k + t_k d^k$, and break
  else $z^k \leftarrow w^k$
  $w^{k+1} \leftarrow T^{\lambda}_{\mathrm{PG}}(z^k)$

Theorem 3.1. *Under the hypotheses of Theorem 2.1, any accumulation point of a sequence generated by Algorithm 1 is a stationary point.*

Theorem 3.2. *Consider either* (5) *with* $\lambda \in (0, L^{-1})$ *or Algorithm 1 with* $\lambda \in (0, L^{-1})$*,* $\eta, \sigma \in (0,1)$*, and* $\alpha_{\max} \geq \alpha_{\min} > 0$*, and suppose that there is an accumulation point* w^* *of the iterates at which* $T^{\lambda}_{\mathrm{PG}}(w^*)$ *is a singleton. Assume that the following relaxed Kurdyka-Łojasiewicz condition holds: there exists neighborhood* $U \subset \Re^n$ *of* w^**,* $\theta \in (0, 1]$*, and* $\kappa > 0$ *such that for every* $J \in \mathcal{I}_{w^*}$*,*

$$\left(f(w) - f(w^*)\right)^{\theta} \leq \kappa \|(\nabla f(w))_J\|, \quad \forall w \in A_J \cap U, \tag{21}$$

Then $w^k \to w^*$*. Moreover, if* n_k *denotes the number of successful extrapolation steps in the first* k *iterations of Algorithm 1, then the following rates hold:*

(a) If $\theta \in (0, 1/2)$*:* $f(w^k) - f(w^*) = O((k + n_k)^{-1/(1-2\theta)})$.
(b) If $\theta \in [1/2, 1]$*:* $f(w^k) - f(w^*) = O(\exp(-(k + n_k)))$.

We note that the additional mild assumption of $T^{\lambda}_{\mathrm{PG}}(w^*)$ being a singleton is simply for the relaxed KL condition in (21), and even without it, we can still get similar rates by using the conventional KL condition.

There are several advantages of the proposed extrapolation strategy over existing ones in [19, 27]. The most obvious one is the faster rates in Theorem 3.2 over PG such that each successful extrapolation step in our method contributes to the convergence speed, while existing methods only provide the same convergence speed as PG. Next, existing strategies only use prespecified step sizes without information from the given problem nor the current progress, and they only restrict such step sizes to be within $[0, 1]$. Our method, on the other hand, fully takes advantage of the function curvature and can allow for arbitrarily large step sizes to better decrease the objective. In fact, we often observe $t_k \gg 1$ in our numerical experiments. Moreover, our acceleration techniques utilize the nature of (7) and (4) to obtain very efficient implementation for ERM problems such that the per-iteration cost of Algorithm 1 is almost the same as that of PG, while the approach of [19] requires evaluating f and ∇f at two points per iteration, and thus has twice the per-iteration cost.

3.2 Subspace Identification

In line with the above discussion, we interpret (8) as a theoretical property guaranteeing that the iterates of the projected gradient algorithm (5) will eventually identify the subspaces A_J that contain a candidate solution w^* after a finite number of iterations. Consequently, the task of minimizing f over the nonconvex set A_s can be reduced to a convex optimization problem of minimizing f over A_J. Motivated by this, we present a two-stage algorithm described in Algorithm 2 that switches to a high-order method for smooth convex optimization after a candidate piece A_J is identified to obtain even faster convergence. Since ∇f is assumed to be Lipschitz continuous, the generalized Hessian of f exists everywhere [14], so we may employ a semismooth Newton (SSN) method [25] with backtracking linesearch to get a faster convergence speed with low cost (details in Appendix A). In particular, we reduce the computation costs by considering the restriction of f on the subspace A_J by treating the coordinates not in J as non-variables so that the problem considered is indeed smooth and convex. As we cannot know a priori whether I_{w^*} is indeed identified, we adopt the approach implemented in [18, 20, 17] to consider it identified when w^k activates the same A_J for long enough consecutive iterations. To further safeguard that we are not optimizing over a wrong subspace, we

Algorithm 2: Accelerated projected gradient algorithm by subspace identification (PG+)

Given an initial vector $w^0 \in \Re^n$ and $S, t \in \mathbb{N}$. Set Unchanged $\leftarrow 0$.
for $k = 0, 1, 2, \ldots$ **do**
  if $k > 0$, *and* w^{k-1} *and* w^k *activate the same component of* A_s **then**
    Let $J \in \mathcal{J}_s$ correspond to the activated component
    Unchanged $\leftarrow$ Unchanged $+1$
  else Unchanged $\leftarrow 0$
  if *Unchanged* $\geq S$ **then**
    $y^k \leftarrow P_{A_J}(w^k)$ and use t steps of SSN described in Appendix A, starting from y^k, to find z^k that approximately minimizes $f|_{A_J}$
    if *SSN fails* **then** $z^k \leftarrow w^k$ and Unchanged $\leftarrow 0$.
  else $z^k \leftarrow w^k$
  $w^{k+1} \leftarrow T^{\lambda}_{\mathrm{PG}}(z^k)$

also incorporate the idea of [28, 2, 20, 17] to periodically alternate to a PG step (5) after switching to the SSN stage. A detailed description of this two-stage algorithm is in Algorithm 2.

In the following theorem, we show that superlinear convergence can be obtained for Algorithm 2 even if we take only one SSN step every time between two steps of (5), using a simplified setting of twice-differentiability. For our description, we need to introduce some additional notations. Given any $w \in A_J$, we use $f_J(w_J) := f(w)$ to denote the function of considering only the coordinates of w in J as variables and treating the remaining as constant zeros.

Theorem 3.3. *Assume that the conditions of Theorem 2.1 (b) hold with $w^* \in A_s$; f is twice-differentiable around a neighborhood U of w^* with $\nabla^2 f_J$ Lipschitz continuous in U and $\nabla^2 f_J(w^*)$ positive definite for all $J \in \mathcal{I}_{w^*}$; and starting after $k \geq N$ and $P_{A_s}(w^k) \subset U$, we conduct t Newton steps between every two steps of* (5) *for $t \geq 1$:*

$$w^{k,0} \in P_{A_s}(w^k), \begin{cases} J & \in \mathcal{I}_{w^{k,0}}, \\ w_i^{k,j+1} & = 0, \quad \forall i \notin J, \quad j = 1, \ldots, t-1, \\ w_J^{k,j+1} & = w_J^{k,j} - \nabla^2 f_J(w_J^{k,j})^{-1} \nabla f_J(w_J^{k,j}), \end{cases} \quad w^{k+1} \in T^{\lambda}_{\mathrm{PG}}(w^{k,t}). \tag{22}$$

Then $w^k \to w^*$ at a Q-quadratic rate.

In practice, the linear system for obtaining the SSN step is only solved inexactly via a (preconditioned) conjugate gradient (PCG) method, and with suitable stopping conditions for PCG and proper algorithmic modifications such as those in [30, 21], superlinear convergence can still be obtained easily. Interested readers are referred to Appendix A for a more detailed description of our implementation.

4 Experiments

In this section, we conduct numerical experiments to demonstrate the accelerated techniques presented in Section 3. We employ Algorithm 1 (APG) to accelerate PG, and further accelerate APG by incorporating subspace identification described in Algorithm 2, which we denote by APG+.[4] For f in (1), we consider both least square in (3) and logistic regression

$$f(w) = \sum_{i=1}^{m} \log\left(1 + \exp\left(-y_i x_i^\top w\right)\right) + \mu \|w\|^2/2, \tag{23}$$

where $(x_i, y_i) \in \Re^n \times \{-1, 1\}$, $i = 1, \ldots, m$, are the training instances, and $\mu > 0$ is a small regularization parameter added to make the logistic loss coercive.

The algorithms are implemented in MATLAB and tested with public datasets listed in Table 1.[5] For the datasets news20 and webspam that do not come with a test set, we manually do a $80/20$ split to obtain a test set. To fit the practical scenario of using (1), we specifically selected high-dimensional datasets with n larger than m. Source code of our implementations is attached in the supplementary materials.

[4] That is, if $Unchanged < S$ in Algorithm 2, we calculate z^k as in Algorithm 1

[5] Downloaded from `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

We conduct experiments with various s to widely test the performance under different scenarios. In particular, we consider $s \in \{\lceil 0.01m \rceil, \lceil 0.05m \rceil, \lceil 0.1m \rceil\}$ on all data except for the largest dataset webspam, for which we set $s \in \{\lceil 0.001m \rceil, \lceil 0.005m \rceil, \lceil 0.01m \rceil\}$.

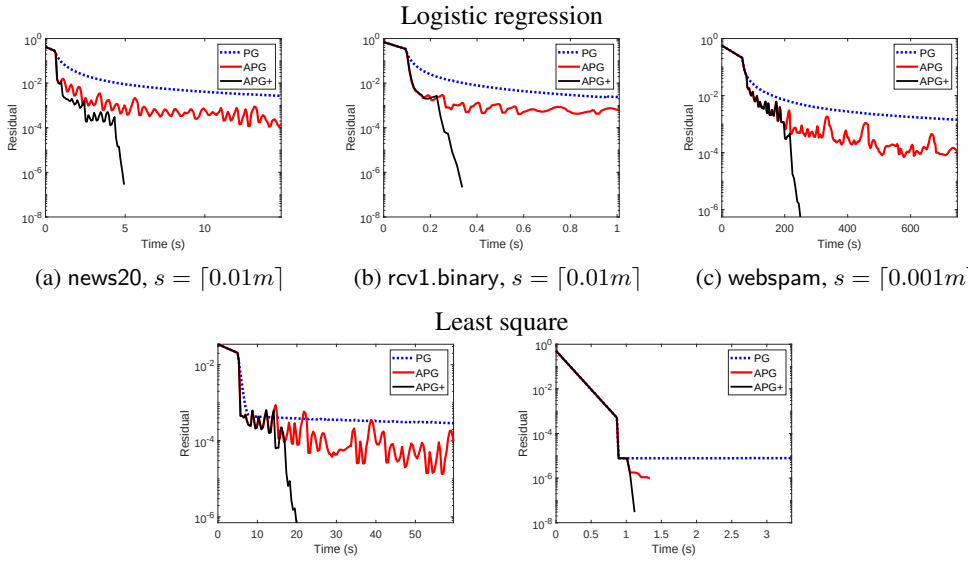
Table 1: Data statistics.

Dataset	Loss	#training instances (m)	#features (n)	#test instances
news20	(23)	15,997	1,355,191	3,999
rcv1.binary	(23)	20,242	47,236	677,399
webspam	(23)	280,000	16,609,143	70,000
E2006-log1p	(3)	16,087	4,272,227	3,308
E2006-tfidf	(3)	16,087	150,360	3,308

All the algorithms are initialized from $w^0 = 0$ and terminated when the first-order optimality condition

$$\text{Residual}(w) := \|w - P_{A_s}(w - \lambda \nabla f(w))\| / (1 + \|w\| + \lambda \|\nabla f(w)\|) < \epsilon \tag{24}$$

is met for some given $\epsilon > 0$. The results of the experiment with the smallest s are summarized in Figure 1, and results of the other two settings of s are in the Appendix C.



(a) news20, $s = \lceil 0.01m \rceil$ (b) rcv1.binary, $s = \lceil 0.01m \rceil$ (c) webspam, $s = \lceil 0.001m \rceil$

(d) E2006-log1p, $s = \lceil 0.01m \rceil$ (e) E2006-tfidf, $s = \lceil 0.01m \rceil$

Figure 1: Experiment on sparse regularized logistic regression and least square regression. We present time versus the residual defined in (24).

Evidently, the extrapolation procedure in APG provides a significant improvement in the running time compared with the base algorithm PG, and further incorporating subspace identification as in APG+ results to a very fast algorithm that outperforms PG and APG by magnitudes. Since the per-iteration cost of PG and APG are almost the same as argued in Section 3, we note that the convergence of APG in terms of iterations is also superior to that of PG.

We next compare APG and APG+ with the extrapolated PG algorithm of Li and Lin [19] that we implement in MATLAB and denote by PG-LL, which is a state-of-the-art approach for nonconvex regularized optimization and thus suitable for (1). We report the required time and number of gradient evaluations (which is the main computation at each iteration) for the algorithms to drive (24) below $\epsilon = 10^{-6}$. For PG, APG, and APG+, one gradient evaluation is needed per iteration, so the number of gradient evaluations is equivalent to the iteration count. For PG-LL, two gradient evaluations are needed per iteration, so its cost is twice of other methods. We also report the prediction performance on the test data, and we in particular use the test accuracy for (23) and the mean-squared error for (3). Results for the two smaller s are in Table 2 while that for the largest s is in Appendix C. It is clear from the results in Table 2 that APG outperforms PG-LL for most of the test instances considered, while APG+ is magnitudes faster than PG-LL. When we equip PG-LL with our acceleration techniques by replacing T^{λ}_{PG} in Algorithms 1 and 2 with the algorithmic map defining PG-LL, we can further speed

Table 2: Comparison of algorithms for (1) to meet (24) with $\epsilon < 10^{-6}$, with (23) and (3) and with sparsity levels $s_1 = \lceil 0.01m \rceil$ and $s_2 = \lceil 0.05m \rceil$ for all datasets except webspam where $s_1 = \lceil 0.001m \rceil$ and $s_2 = \lceil 0.005m \rceil$. CPU: CPU time in seconds. GE: number of gradient evaluations. In one iteration, PG, APG, and APG+ needs one gradient evaluation , while PG-LL and PG-LL+ needs two. CG: number of Hessian-vector products in the PCG procedure for obtaining SSN steps. PA: prediction accuracy (for (23)). MSE: mean-squared error (for (3)). Time with $*$ indicates that the algorithm is terminated after running 10000 iterations without satisfying (24).

Dataset	Method	s_1				s_2			
		CPU	GE	CG	PA	CPU	GE	CG	PA
news20	PG	$*$738.7	10000	0	0.877	$*$728.9	10000	0	0.935
	APG	151.7	1583	0	0.877	758.3	8428	0	0.923
	APG+	**5.0**	52	63	0.853	**16.1**	171	67	0.923
	PG-LL	366.7	4682	0	0.873	$*$1494.4	20000	0	0.922
	APG-LL+	6.6	152	88	0.854	29.2	417	89	0.920
rcv1.binary	PG	$*$58.4	10000	0	0.937	$*$72.7	10000	0	0.951
	APG	12.6	1120	0	0.935	82.4	6372	0	0.934
	APG+	**0.3**	21	42	0.931	**2.4**	192	138	0.940
	PG-LL	22.2	3638	0	0.935	72.1	8738	0	0.929
	APG-LL+	0.6	99	49	0.930	4.9	626	236	0.939
webspam	PG	$*$18660.1	10000	0	0.964	$*$30776.2	10000	0	0.978
	APG	19683.4	7682	0	0.981	7722.4	2008	0	0.991
	APG+	**248.3**	75	88	0.969	**695.4**	164	57	0.991
	PG-LL	9001.3	4720	0	0.972	10163.5	3098	0	0.990
	APG-LL+	447.3	264	92	0.965	837.3	294	90	0.992
		CPU	GE	CG	MSE	CPU	GE	CG	MSE
E2006-log1p	PG	$*$2998.6	10000	0	0.167	$*$3644.1	10000	0	0.161
	APG	270.6	669	0	0.136	811.8	1757	0	0.133
	APG+	**19.5**	40	49	0.141	**105.6**	222	124	0.132
	PG-LL	$*$6049.8	20000	0	0.132	2696.0	7086	0	0.132
	APG-LL+	41.2	142	38	0.142	107.5	326	100	0.138
E2006-tfidf	PG	$*$242.7	10000	0	0.152	$*$666.9	10000	0	0.152
	APG	1.3	14	0	0.154	3.3	33	0	0.153
	APG+	**1.3**	8	6	0.141	**3.3**	31	7	0.139
	PG-LL	110.6	4440	0	0.152	304.8	4558	0	0.151
	APG-LL+	1.7	34	6	0.141	3.7	47	7	0.139

up PG-LL greatly as shown under the name APG-LL+. We do not observe a method that consistently possesses the best prediction performance, as this is mainly affected by which local optima is found, while no algorithm is able to find the best local optima among all candidates. With no prediction performance degradation, we see that APG+ and APG-LL+ reduce the time needed to solve (1) to a level significantly lower than that of the state of the art.

5 Conclusions

In this work, we revisited the projected gradient algorithm for solving ℓ_0-norm constrained optimization problems. Through a natural decomposition of the constraint set into subspaces and the proven ability of the projected gradient method to identify a subspace that contains a solution, we further proposed effective acceleration schemes with provable convergence speed improvements. Experiments showed that our acceleration strategies improve significantly both the convergence speed and the running time of the original projected gradient algorithm, and outperform the state of the art for ℓ_0-norm constrained problems by a huge margin. We plan to extend our analysis and algorithm to the setting of a nonconvex objective in the near future.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We only considered the case of f convex, and we mentioned in the conclusions that we plan to extend our analysis to nonconvex f.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] This work has no negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] In the supplemental materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] There is no randomness in our experiments or algorithms.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] The website of the source of data used in the experiments are provided.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Our codes are provided in the supplementary materials.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] The data used are publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] The public data used do not contain such information/content.

References

[1] Hédy Attouch, Jérôme Bolte, and Benar Fux Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods. *Mathematical Programming*, 137(1):91–129, 2013. 4, 20

[2] Gilles Bareilles, Franck Iutzeler, and Jérôme Malick. Newton acceleration on manifolds identified by proximal-gradient methods. Technical report. arXiv:2012.12936. 7, 21

[3] Jonathan Barzilai and Jonathan M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148, 1988. 4, 5

[4] E. M. L. Beale, M. G. Kendall, and D. W. Mann. The discarding of variables in multivariate analysis. *Biometrika*, 54(3-4):357–366, 1967. 1

[5] Amir Beck. *First-Order Methods in Optimization*. SIAM - Society for Industrial and Applied Mathematics, Philadelphia, PA, United States, 2017. 5, 15, 18

[6] Amir Beck and Yonina C. Eldar. Sparsity constrained nonlinear optimization: optimality conditions and algorithms. *SIAM Journal on Optimization*, 23(3):1480–1509, 2013. 3

[7] Amir Beck and Marc Teboulle. A fast iterative shrinkage thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. 4, 5

[8] Amir Beck and Marc Teboulle. A linearly convergent algorithm for solving a class of nonconvex/affine feasibility problems. In H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, volume 49 of *Springer Optimization and Its Applications*, pages 33–48. Springer, New York, NY, 2011. 2, 3

[9] D. Bertsimas, Angela King, and R. Mazumder. Best subset selection via a modern optimization lens. *Annals of Statistics*, 44(2):813–852, 2016. 2, 3, 4

[10] Jérôme Bolte, Shoham Sabach, Marc Teboulle, and Yakov Vaisbourd. First order methods beyond convexity and lipschitz gradient continuity with applications to quadratic inverse problems. *SIAM Journal on Optimization*, 28(3):2131–2151, 2018. 4

[11] Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001. 2

[12] Leonardo Galli and Chih-Jen Lin. A study on truncated newton methods for linear classification. *IEEE Transactions on Neural Networks and Learning Systems*, 2021. 14

[13] Jun-ya Gotoh, Akiko Takeda, and Katsuya Tono. DC formulations and algorithms for sparse optimization problems. *Mathematical Programming*, 169(1):141–176, 2018. 2

[14] Jean-Baptiste Hiriart-Urruty, Jean-Jacques Strodiot, and V Hien Nguyen. Generalized Hessian matrix and second-order optimality conditions for problems with $C^{1,1}$ data. *Applied Mathematics & Optimization*, 11(1):43–56, 1984. 5, 6, 13

[15] Ronald R. Hocking and R. N. Leslie. Selection of the best subset in regression analysis. *Technometrics*, 9(4):531–540, 1967. 1

[16] Prateek Jain and Purushottam Kar. Non-convex optimization for machine learning. *Foundations and Trends in Machine Learning*, 10(3–4):142–363, 2017. 1

[17] Ching-pei Lee. Accelerating inexact successive quadratic approximation for regularized optimization through manifold identification. Technical report, 2020. arXiv:2012.02522. 6, 7

[18] Sangkyun Lee and Stephen J. Wright. Manifold identification in dual averaging for regularized stochastic online learning. *Journal of Machine Learning Research*, 13:1705–1744, 2012. 6

[19] Huan Li and Zhouchen Lin. Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems*, volume 28, 2015. 2, 3, 4, 6, 8, 20

[20] Yu-Sheng Li, Wei-Lin Chiang, and Ching-pei Lee. Manifold identification for ultimately communication-efficient distributed optimization. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 6, 7

[21] Boris S. Mordukhovich, Xiaoming Yuan, Shangzhi Zeng, and Jin Zhang. A globally convergent proximal Newton-type method in nonsmooth convex optimization. *Mathematical Programming*, 2022. Online first. 7

[22] Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983. 4

[23] Yurii E. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013. 5

[24] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, 2e edition, 2006. 13, 21

[25] Liqun Qi and Jie Sun. A nonsmooth version of Newton's method. *Mathematical programming*, 58(1-3):353–367, 1993. 6, 13

[26] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B*, 58(1):267–288, 1996. 2

[27] Bo Wen, Xiaojun Chen, and Ting Kei Pong. A proximal difference-of-convex algorithm with extrapolation. *Computational Optimization and Applications*, 69:297–324, 2018. 2, 3, 4, 6

[28] Stephen J. Wright. Accelerated block-coordinate relaxation for regularized optimization. *SIAM Journal on Optimization*, 22(1):159–186, 2012. 7

[29] Stephen J. Wright, Robert D. Nowak, and Mário A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009. 4, 5

[30] Man-Chung Yue, Zirui Zhou, and Anthony Man-Cho So. A family of inexact SQA methods for non-smooth convex minimization with provable convergence guarantees based on the Luo–Tseng error bound property. *Mathematical Programming*, 174(1-2):327–358, 2019. 7, 13

[31] Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942, 2010. 2