Noname manuscript No.
(will be inserted by the editor)

An Adaptive Primal-Dual Framework for Nonsmooth Convex Minimization

**Quoc Tran-Dinh · Ahmet Alacaoglu ·
Olivier Fercoq · Volkan Cevher**

Received: date / Accepted: date

Abstract We propose a new self-adaptive and double-loop smoothing algorithm to solve composite, nonsmooth, and constrained convex optimization problems. Our algorithm is based on Nesterov's smoothing technique via general Bregman distance functions. It self-adaptively selects the number of iterations in the inner loop to achieve a desired complexity bound without requiring to set the accuracy a priori as in variants of Augmented Lagrangian methods (ALM). We prove $\mathcal{O}\left(\frac{1}{k}\right)$-convergence rate on the last iterate of the outer sequence for both unconstrained and constrained settings in contrast to ergodic rates which are common in ALM as well as alternating direction method-of-multipliers literature. Compared to existing inexact ALM or quadratic penalty methods, our analysis does not rely on the worst-case bounds of the subproblem solved by the inner loop. Therefore, our algorithm can be viewed as a restarting technique applied to the ASGARD method in [60] but with rigorous theoretical guarantees or as an inexact ALM with explicit inner loop termination rules and adaptive parameters. Our algorithm only requires to initialize the parameters once, and automatically updates them during the iteration process without tuning. We illustrate the superiority of our methods via several examples as compared to the state-of-the-art.

Keywords Primal-dual first-order methods · restarting · augmented Lagrangian · self-adaptive method · nonsmooth convex optimization

Mathematics Subject Classification (2000) 90C25 · 90C06 · 90-08

Quoc Tran-Dinh
Department of Statistics and Operations Research, The University of North Carolina at Chapel Hill, 333 Hanes Hall, CB#3260, UNC Chapel Hill, NC 27599-3260.
E-mail: `quoctd@email.unc.edu`

Ahmet Alacaoglu · Volkan Cevher
Laboratory for Information and Inference Systems (LIONS), École Polytechnique Fédérale de Lausanne (EPFL), CH1015-Lausanne, Switzerland.
E-mail: {`ahmet.alacaoglu, volkan.cevher`}`@epfl.ch`

Olivier Fercoq
LTCI, Télécom ParisTech, Université Paris-Saclay, 75634-Paris, France.
E-mail: `olivier.fercoq@telecom-paristech.fr`

1 Introduction

Problem settings: We study the following nonsmooth composite convex minimization template:

$$P^\star := \min_{x\in\mathbb{R}^p}\Big\{P(x) := f(x) + g(Ax)\Big\}, \tag{1}$$

where both $f : \mathbb{R}^p \to \mathbb{R}\cup\{+\infty\}$ and $g : \mathbb{R}^n \to \mathbb{R}\cup\{+\infty\}$ are proper, closed, and nonsmooth convex functions, and $A : \mathbb{R}^p \to \mathbb{R}^n$ is a linear operator.

In addition, we assume that g is Lipschitz continuous when restricted to its domain (see Assumption 1). This assumption covers many important cases:

1. g is Lipschitz continuous on $\mathbb{R}^n$, i.e. there exists $M_g \in (0,+\infty)$ such that $|g(x) - g(y)| \le M_g\|x-y\|$ for any $x, y \in \mathbb{R}^n$.
2. $g := \delta_{b+\mathcal{K}}$, the indicator of $b+\mathcal{K}$, where $b \in \mathbb{R}^n$ is given and $\mathcal{K}$ is a given nonempty, closed, and convex set in $\mathbb{R}^n$. In this case, (1) automatically covers the following constrained convex problem:
$$f^\star := \min_{x\in\mathbb{R}^p}\Big\{f(x) \ \ \text{s.t.} \ \ Ax - b \in \mathcal{K}\Big\}. \tag{2}$$
In particular, if $\mathcal{K} = \{\mathbf{0}^n\}$, then $Ax - b \in \mathcal{K}$ reduces to $Ax = b$.
3. g is a polyhedral function, i.e. its epigraph is a polyhedron.

Our goals: Our goal is to design new adaptive primal-dual methods to solve both (1) and (2) that have low per-iteration complexity cost, i.e., they only require the proximal operators of f and g, and matrix-vector multiplications Ax and $A^\top y$, while having the best-known non-averaging convergence rates.

We require that both problems are convex and satisfy strong duality assumptions. We do not assume that the problems are strongly convex or smooth.

Composite vs. constrained settings: For the general setting (1), under different choices of f and g, it covers a wide range of applications from different fields including compressive sensing, image and signal processing, machine learning, statistics, operations research, and optimal control. Classical and well-known examples such as LASSO, square-root LASSO, support vector machines, image denoising and deblurring, and matrix completion can be cast into (1), see, e.g., [10,18,53,67] for some concrete examples.

For the setting (2), we do not impose any restriction on $\mathcal{K}$. Hence, it covers a large class of constrained problems including equality and inequality constraints. When $\mathcal{K}$ is a given cone (e.g., $\mathbb{R}^n_+$, second-order cone, or symmetric positive semidefinite cone), problem (2) covers also problems with cone constraints such as linear programming, second-order cone, and semidefinite programming. Although the theory for (1) as well as for (2) are well developed, various numerical methods for solving these problems rely on different structure assumptions and do not have a unified analysis: *cf.*, Section 5.

Related works: Under only convexity and zero duality gap assumptions, the state-of-the-art methods for solving (1) include primal-dual first-order methods (PDFOM) [2,13], and augmented Lagrangian-based algorithms [5,34,56]. While PDFOM directly tackles problem (1), the augmented Lagrangian-based framework (ALM) and its variants solve (1) via a constrained reformulation as follows (or using other forms):

$$P^\star := \min_{x\in\mathbb{R}^p, z\in\mathbb{R}^n} \Big\{ P(x,z) := f(x) + g(z) \ \text{ s.t. } \ Ax - z = 0 \Big\}. \tag{3}$$

Alternating direction method of multipliers (ADMM) [25,62] is another (and perhaps the most) successful method to solve (3). ADMM can be viewed as a variant of the ALM framework [10]. In other words, ADMM can be viewed as an approximation to ALM by alternating between x and z to break the computational bottleneck in the primal subproblem. Inexact and linearized variants enhance the scalability of ALM and ADMM for the same problem template [52,68,69].

While ADMM and PDFOM and their variants work really well in practice, their best-known convergence rate is $\mathcal{O}\left(\frac{1}{k}\right)$ under only convexity and zero duality gap assumptions, where k is the iteration counter. Moreover, such a rate is achieved via an ergodic sense (i.e., using an averaging sequence or a weighted averaging sequence) [13,14,20,21,41,42,57].

In stark contrast, empirical evidence shows that averaging sequences in PDFOM and ALM exhibit the theoretical worst case rate $\mathcal{O}\left(\frac{1}{k}\right)$ in practice compared to the last iterate of the algorithm (see Subsection 4.1 for a concrete example), which is superior and often locally linear in many examples.[1] However, for these methods, last iterate generally has convergence guarantees but has much slower rate guarantees, e.g., $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$-rates [21].

Recently, [60] proposed an *accelerated, smoothed gap reduction* (ASGARD) framework to solve nonsmooth convex optimization problems. ASGARD combines acceleration, smoothing, and homotopy techniques to handle both unconstrained and constrained nonsmooth problems, including (1).

One notable feature of ASGARD is a non-ergodic optimal $\mathcal{O}\left(\frac{1}{k}\right)$ rate on the objective residual, and feasibility violation in the constrained settings. Moreover, this method only requires one proximal operator of f, one matrix-vector multiplication, and one adjoint operator per iteration. When f is separable, the algorithm can be naturally parallelized. However, as also noted in [60], ASGARD needs restarting to be competitive with state-of-the-art methods such as ADMM and PDFOM in practice. This is not surprising since empirical evidence [29,32,51,58] has shown that restarting significantly improves the actual convergence rate in practice. While there exists theory to support the restarting strategies in accelerated gradient-type methods, supporting theory of these strategies are not yet investigated in primal-dual methods.

Contributions: In this paper, we introduce an analysis framework for restarting ASGARD and prove the same worst-case $\mathcal{O}\left(\frac{1}{k}\right)$ rate in a non-ergodic sense. While doing so, we identify that restarting ASGARD corresponds to an inexact ALM algorithm in the constrained case. In contrast to existing works on this front, our method has explicit inner-loop termination rules and does

[1] There exist examples showing arbitrarily slow convergence rate of ADMM, see, e.g., [21].

not need to set a horizon (i.e., the maximum number of inner iterations or a predefined inner loop accuracy) for the algorithm.

As a result, we present a method which has the guarantees on the last iterate compared to ALM/ADMM methods and extend the guarantees of ASGARD to the restarting case which significantly improves the practical performance. In addition, we allow general Bregman distances to be used for smoothing and proximal operators in contrast to the original ASGARD scheme. A more thorough discussion and comparison between our method and existing state-of-the-arts is deferred to Section 5 for the sake of presentation.

More concretely, our contributions can be summarized as follows.

(a) We propose a new self-adaptive, double-loop smoothing algorithm to solve nonsmooth convex optimization problems of the form (1). Our algorithm is based on Nesterov's smoothing technique via general Bregman distance functions. It self-adaptively selects the number of iterations in the inner loop to achieve a desired complexity bound without requiring the accuracy a priori as in variants of ALM. Compared to ASGARD [60], it incorporates restarts, updates the dual center, and can work with general Bregman distances instead of only Lipschitz gradient distances.

(b) We prove $\mathcal{O}\left(\frac{1}{k}\right)$-convergence rate on the last iterate of the outer sequence for both unconstrained and constrained settings in contrast to ergodic rates which are common in ALM/ADMM literature. This rate is known to be optimal [46,49,66] under just convexity and strong duality assumptions and when k is not sufficiently large. Compared to existing inexact ALM or quadratic penalty methods such as [43,70], our analysis does not rely on the worst-case bounds of the subproblem solved by the inner loop. Therefore, our algorithm can be viewed as a restarting technique applied to ASGARD but with rigorous theoretical guarantees or as an inexact ALM with explicit inner loop termination rules and adaptive parameters.

(c) As an upshot, we customize our algorithm to solve general constrained problems of the form (2). We prove the same $\mathcal{O}\left(\frac{1}{k}\right)$-convergence rate guarantee on both the objective residual $|f(x^k) - f(x^\star)|$ and the feasibility $\mathrm{dist}_{\mathcal{K}}\left(Ax^k - b\right)$. This rate is given on the last iterate of the outer sequence.

Our algorithm is a primal-dual method, which can solve composite convex problem with linear operators as in Chambolle-Pock's method [13]. It only requires one proximal operator of f and g^*, one matrix-vector multiplication and one adjoint for each iteration. It is parallelizable when f is separable, i.e., $f(x) = \sum_{i=1}^N f_i(x_{[i]})$. Under this structure, our method has more advantages than ADMM and Chambolle-Pock's method. In the algorithm, we provide explicit rules to update all algorithmic parameters. We also note that these updates can be modified to trade-off between the primal or the dual progress. Since the parameters in ASGARD [60] are decreasing making its step-size smaller at each iteration, the restarting schemes developed in this paper reset these parameters to preserve large step-size at each outer-loop iteration.

Paper organization: The rest of this paper is organized as follows. Section 2 recalls some mathematical background and the ASGARD algorithm in [60]. Section 3 presents our main result with algorithm and its convergence guarantee. We study both unconstrained and constrained cases. Section 3.4 shows an extension of our method to three composite objective functions with linearization on potentially smooth terms. In Section 4, we provide seven numerical examples to test our algorithm against state-of-the arts. Section 5 compares our method and existing algorithms in the literature.

2 Mathematical tools

We review some key ingredients for the design of our primal-dual methods.

Notation: We denote the norm in primal space $\mathcal{X}$ as $\|\cdot\|_{\mathcal{X}}$ and the norm in dual space $\mathcal{Y}$ as $\|\cdot\|_{\mathcal{Y}}$. Their dual norms are denoted as $\|\cdot\|_{\mathcal{X},*}$ and $\|\cdot\|_{\mathcal{Y},*}$, respectively. Given a positive real number a, $\lfloor a \rfloor$ denotes the largest integer that is less than or equal to a.

For a given nonempty, closed, and convex set $\mathcal{K}$, we denote its indicator function as $\delta_{\mathcal{K}}(x) = 0$, if $x \in \mathcal{K}$, $\delta_{\mathcal{K}}(x) = +\infty$, otherwise; and its support function as $s_{\mathcal{K}}(y) = \sup_{x\in\mathcal{K}}\langle x, y\rangle$. We define the normal cone of $\mathcal{K}$ as $\mathcal{N}_{\mathcal{K}}(x) := \{w \in \mathbb{R}^n \mid \langle w, y - x\rangle \geq 0,\ y \in \mathcal{K}\}$ if $x \in \mathcal{K}$; $\mathcal{N}_{\mathcal{K}}(x) := \emptyset$, otherwise. We also define $\mathcal{K}^o := \{w \in \mathbb{R}^n \mid \langle w, x\rangle \leq 1,\ x \in \mathcal{K}\}$ as the polar set of $\mathcal{K}$. If $\mathcal{K}$ is a convex cone, then $\mathcal{K}^o = -\mathcal{K}^*$, where $\mathcal{K}^* := \{w \in \mathbb{R}^n \mid \langle w, x\rangle \geq 0,\ x \in \mathcal{K}\}$ the dual cone of $\mathcal{K}$.

Given a proper, closed, and convex function f, we use $\mathrm{dom}(f)$ to denote its domain and $\partial f(x)$ to denote its subdifferential at x. When the function is differentiable, we denote its gradient at x as $\nabla f(x)$. The Fenchel conjugate of a function f is defined as $f^*(y) := \sup_x \{\langle x, y\rangle - f(x)\}$. We say that $f : \mathcal{X} \to \mathbb{R}$ has Lipschitz gradient if it satisfies $\|\nabla f(x) - \nabla f(y)\|_{\mathcal{X},*} \leq L_f \|x - y\|_{\mathcal{X}}$, for any $x, y \in \mathcal{X}$. We say that f is Lipschitz continuous on $\mathcal{X}$ with a Lipschitz constant $M_f \in [0, +\infty)$ if $|f(x) - f(y)| \leq M_f\|x - y\|$ for any $x, y \in \mathcal{X}$.

Given a proper, closed, and convex function $f : \mathbb{R}^p \to \mathbb{R}\cup\{+\infty\}$, $\mathrm{prox}_f(x) := \mathrm{argmin}_u \left\{f(u) + (1/2)\|u - x\|^2_{\mathcal{X}}\right\}$ is called the proximal operator of f. We say that f is "proximally tractable" if prox_f can be computed efficiently, e.g., in a closed form, or by a polynomial algorithm. By Moreau's identity, we have $\mathrm{prox}_{\gamma f}(x) + \gamma \mathrm{prox}_{f^*/\gamma}(\gamma^{-1}x) = x$ for any $x \in \mathrm{dom}(f)$.

2.1 Primal-dual formulation

Dual problem and min-max formulation: Associated with the primal problem (1), we also consider the corresponding dual problem:

$$D^\star := \min_{y\in\mathbb{R}^n} \Big\{ D(y) := f^*(-A^\top y) + g^*(y) \Big\}, \tag{4}$$

where f^* and g^* are the Fenchel conjugates of f and g, respectively. Clearly, we can write the pair (1)-(4) in the following min-max saddle point problem:

$$\min_{x\in\mathbb{R}^p} \max_{y\in\mathbb{R}^n} \Big\{ \mathcal{L}(x, y) := f(x) + \langle Ax, y\rangle - g^*(y) \Big\}. \tag{5}$$

Under mild and standard assumptions, this min-max problem is solvable and achieves zero duality gap, i.e., $P^\star + D^\star = 0$. In particular, the dual problem of (2) can be written as follows:

$$D^\star := \min_{y\in\mathbb{R}^n}\left\{D(y) := f^*(-A^Ty) + \langle b, y\rangle + s_{\mathcal{K}}(y)\right\}, \tag{6}$$

where $s_{\mathcal{K}}(y) = \sup_{x\in\mathcal{K}}\langle y, x\rangle$ is the support function of $\mathcal{K}$. Compared to (4), we have $g^*(y) = \langle b, y\rangle + s_{\mathcal{K}}(y) = s_{b+\mathcal{K}}(y)$. Let $\mathcal{X}^\star$ and $\mathcal{Y}^\star$ be the solution sets of the primal problem (1) (or (2)) and dual problem (4) (or (6)), respectively.

Fundamental assumptions: Throughout this paper, we will develop methods for solving (1) and (2). We propose a unified set of assumptions that covers unconstrained and constrained problems.

Assumption 1. We impose the following assumption on (1):
1. The solution set $\mathcal{X}^\star$ of (1) is nonempty.
2. Both f and g are proper, closed, and convex.
3. The function g is Lipschitz continuous its domain: there exists $\widehat{M}_g \in (0, +\infty)$ such that $|g(x) - g(y)| \leq \widehat{M}_g\|x - y\|$ for any $x, y \in \mathrm{dom}(g)$.
4. The Slater condition $\mathrm{ri}\,(\mathrm{dom}(f)) \cap \{x \in \mathbb{R}^p \mid Ax \in \mathrm{ri}\,(\mathrm{dom}(g))\} \neq \emptyset$ holds, where $\mathrm{ri}\,(\mathcal{X})$ is the relative interior of $\mathcal{X}$.

The details of Slater conditions can be found in, e.g., [2]. Assumption 1 covers the case of equality constraint, i.e., $g = \delta_{\{b\}}$, cone constraints, i.e. $g = \delta_{\mathcal{K}}$ as in (2), where $\mathcal{K}$ is a convex cone, Lipschitz continuous functions, i.e. g is globally Lipschitz continuous, and combinations thereof. It guarantees the strong duality of (1) and (4) to hold.

Optimality conditions: Associated with the primal and dual problems (1)-(4), we have the following optimality conditions:

$$0 \in \partial f(x^\star) + A^\top\partial g(Ax^\star) \quad \text{and} \quad 0 \in -A\partial f^*(-A^\top y^\star) + \partial g^*(y^\star).$$

We can write this optimality condition into the following KKT condition:

$$0 \in \partial f(x^\star) + A^\top y^\star \quad \text{and} \quad 0 \in -Ax^\star + \partial g^*(y^\star). \tag{7}$$

For the constrained problem (2) these conditions are written as

$$0 \in \partial f(x^\star) + A^\top y^\star, \quad Ax^\star - b \in \mathcal{K}, \quad \text{and} \quad y^\star \in \mathcal{N}_{\mathcal{K}}\left(Ax^\star - b\right),$$

where $\mathcal{N}_{\mathcal{K}}(\cdot)$ is the normal cone of $\mathcal{K}$ defined above. If $\mathcal{K}$ is a closed, pointed, and convex cone, then $\mathcal{N}_{\mathcal{K}} \equiv -\mathcal{K}^*$ the dual cone of $\mathcal{K}$. In this case, $y^\star \in -\mathcal{K}^*$.

2.2 Bregman distances and generalized proximal operators

In the sequel, we will use Bregman distances for smoothing and computing proximal operators. Therefore, we give basic properties on Bregman distances.

Let $p_{\mathcal{Z}}$ be μ_p-strongly convex, continuous, and differentiable on $\mathcal{Z}$ with the strong convexity $\mu_p = 1$, where $\mathcal{Z} = \mathrm{dom}(p_{\mathcal{Z}})$. We call $p_{\mathcal{Z}}$ a proximity function (or prox-function). We define the Bregman distance induced by $p_{\mathcal{Z}}$ as

$$b_{\mathcal{Z}}(x, y) := p_{\mathcal{Z}}(x) - p_{\mathcal{Z}}(y) - \langle\nabla p_{\mathcal{Z}}(y), x - y\rangle, \quad \forall x, y \in \mathcal{Z}.$$

As special cases, if we choose $p_{\mathcal{Z}}(x) = \frac{1}{2}\|x\|_2^2$, then $b_{\mathcal{Z}}(x,y) = \frac{1}{2}\|x-y\|_2^2$, the standard Euclidean distance square. If we choose $p_{\mathcal{Z}}(x) := \sum_i x_i\ln(x_i)$, the entropy function, then $b_{\mathcal{Z}}(x,y) := \sum_i x_i \ln\left(\frac{x_i}{y_i}\right) - x_i + y_i$, the so-called KL divergence. Moreover, it is obvious that $b_{\mathcal{Z}}(x,y) \geq \frac{1}{2}\|x-y\|_{\mathcal{Z}}^2$ for all $x, y \in \mathcal{Z}$. When a Bregman distance $b_{\mathcal{Z}}$ has Lipschitz continuous gradient, we denote its Lipschitz constant by $L_{b_{\mathcal{Z}}}$. We refer to [16,24,35] for several concrete examples of Bregman divergences.

2.3 Nesterov's smoothing technique

We focus on Nesterov's smoothing technique with general Bregman distances [4,49] as follows. Since g in (1) is possibly nonsmooth, we smooth it by

$$g_\beta(u;\dot{y}) := \max_{y\in\mathcal{Y}}\Big\{\langle u, y\rangle - g^*(y) - \beta b_{\mathcal{Y}}(y,\dot{y})\Big\}, \tag{8}$$

where $\dot{y} \in \mathbb{R}^n$ is a given center point, and $\beta > 0$ is a smoothness parameter. The function $g_\beta(\cdot;\dot{y})$ is convex and smooth, its gradient is given by

$$\nabla g_\beta(u;\dot{y}) = y_\beta^*(u;\dot{y}) = \operatorname*{argmin}_{y\in\mathcal{Y}}\Big\{g^*(y) - \langle u, y\rangle + \beta b_{\mathcal{Y}}(y,\dot{y})\Big\}. \tag{9}$$

Clearly, $\nabla g_\beta(\cdot;\dot{y})$ is Lipschitz continuous with the Lipschitz constant $L_{g_\beta} = \frac{1}{\beta}$. Moreover, we have

$$g_\beta(u;\dot{y}) \leq g(u) \leq g_\beta(u;\dot{y}) + \beta D_{\mathcal{Y}}, \tag{10}$$

where $D_{\mathcal{Y}} := \sup\{b_{\mathcal{Y}}(y,\dot{y}) \mid y \in \operatorname{dom}(g^*)\}$ is the prox-diameter of g^*. Here, $D_{\mathcal{Y}}$ is finite if and only if g is Lipschitz continuous with the Lipschitz constant $L_g := \sqrt{2D_{\mathcal{Y}}}$, i.e., $|g(u) - g(v)| \leq \sqrt{2D_{\mathcal{Y}}}\|u - v\|$ for all $u, v \in \operatorname{dom}(g)$ due to [7, Proposition 4.4.6].

If we choose $b_{\mathcal{Y}}(y,\dot{y}) = \frac{1}{2}\|y-\dot{y}\|_2^2$, then we can write $y_\beta^*(u;\dot{y})$ as:

$$\nabla g_\beta(u;\dot{y}) = \arg\min_{y\in\mathbb{R}^n}\left\{g^*(y) - \langle u, y\rangle + \tfrac{\beta}{2}\|y-\dot{y}\|^2\right\} = \operatorname{prox}_{g^*/\beta}\left(\dot{y} + \tfrac{1}{\beta}u\right). \tag{11}$$

Smoothing techniques are widely used in the literature, including [4,8,9,22,44]. The idea of smoothing is to approximate the original problem (1) by a (partially) smoothed problem. For example, in our setting, we smooth g and consider the following smoothed problem:

$$P_\beta^\star := \min_{x\in\mathbb{R}^p}\Big\{P_\beta(x;\dot{y}) := f(x) + g_\beta(Ax;\dot{y})\Big\}. \tag{12}$$

We define the following generalized proximal operator with Bregman distance $d_{\mathcal{X}}$ induced by a prox-function $q_{\mathcal{X}}$:

$$\mathcal{P}_{\theta f}^{d_{\mathcal{X}}}(u,y) := \operatorname*{argmin}_{v\in\mathcal{X}}\Big\{f(v) + \langle y, v-u\rangle + \tfrac{1}{\theta}d_{\mathcal{X}}(v,u)\Big\}. \tag{13}$$

Given that the Bregman distance $d_{\mathcal{X}}$ is defined in $\mathcal{X}$ and $b_{\mathcal{Y}}$ is defined in $\mathcal{Y}$, we define the following operator norm of A:

$$\|A\| := \max_{x \in \mathbb{R}^p} \left\{ \frac{\|Ax\|_{\mathcal{Y},*}}{\|x\|_{\mathcal{X}}} \right\}. \tag{14}$$

Different from [4,9,22,44,49], our strategy allows one to update the smoothness parameter β gradually at each iteration. Similar work can be found in [8, 48], which are also essentially different from ours as discussed in Section 5.

3 Main results: Self-Adaptive Double-Loop ASGARD

In this section, we develop a self-adaptive double-loop accelerated smoothed primal-dual gap reduction algorithm to solve (1) and (2). We first present the complete algorithm. Next, we provide its convergence analysis. Then, we specify our algorithm to handle the constrained setting (2). Finally, we extend our method to handle (1) with the sum of three objective functions where the third function has Lipschitz gradient.

3.1 The algorithm and its convergence guarantee

Main idea: The proposed algorithm consists of two loops:

– The inner loop performs an accelerated proximal gradient (APG) scheme [61] to solve the smoothed problem (12) for a fixed β, which is different from [60], where β is updated at each iteration. We note that in the constrained case, the smoothed problem (12) is the augmented Lagrangian.
– The outer loop can be considered as a restarting step and simultaneously decreases the smoothness parameter β.

The intuition behind our new strategy lies on the fact that when applied to (12) with a fixed β, APG gets $\mathcal{O}\left(\frac{1}{k^2}\right)$ rate, whereas ASGARD as presented in [60] controls the parameters in such a way that the algorithm gets $\mathcal{O}\left(\frac{1}{k}\right)$ rate throughout its execution. The idea is to take the advantage of the faster rate of APG for the inner loop while carefully adjusting the number of inner iterations and the smoothness parameter to get the same overall $\mathcal{O}\left(\frac{1}{k}\right)$ rate with better practical performance. Our analysis also gives insights on the heuristic restart strategy outlined in [60]. For the sake of presentation and its flexibility for using Bregman distances in proximal operators, we choose Tseng's variant of APG [61]. However, we can replace by another scheme such as FISTA [3]. We adaptively determine the number of inner iterations at each outer iteration. Therefore, there is no need to tune this parameter. The outer loop gradually decreases the smoothness parameter β such that the algorithm is still guaranteed to converge to the true solution of (1) or (2).

The algorithm: The complete algorithm is presented in Algorithm 1.

Algorithm 1 uses APG with **Option 1** at Step 8. This step requires one subproblem in $\tilde{y}^{k+1}$, one prox_f of f, one matrix-vector multiplication Ax and its adjoint $A^\top y$ at each iteration. Hence, Algorithm 1 has the same per-iteration complexity as ASGARD, except for the extra step, Step 14, where we update the dual center $\dot{y}^s$ at each outer loop iteration. In general, the number of outer iterations is small as it is the number of restarting steps. Hence, Step 14 does not significantly increase the overall computational cost of the entire algorithm. Note that $\bar{x}^{k+1}$ computed at Step 9 using **Option 1** is a weighted averaging

Algorithm 1 (Self-Adaptive Double Loop ASGARD Algorithm)

```
1: Initialization:
2: Choose $\beta_0 > 0$, $\omega > 1$, a positive integer $m_0 \geq 1$, $\bar{x}^0 \in \mathbb{R}^p$, and $\dot{y}^0 \in \mathbb{R}^n$.
3: Choose a Bregman distance $b_{\mathcal{Y}}$ for $y$ and $d_{\mathcal{X}}$ for $x$.
4: Set $K_0 \leftarrow 0$, $\hat{x}^0 \leftarrow \bar{x}^0$, and $\tau_0 := 1$.
5: For $s = 0$ to $S_{\max} - 1$, perform:
6:     For $j := 0$ to $m_s - 1$ perform
7:         Set $k \leftarrow K_s + j$.
8:         Update $\begin{cases} \tilde{x}^k \leftarrow (1-\tau_k)\bar{x}^k + \tau_k\hat{x}^k \\ \tilde{y}^{k+1} \leftarrow \operatorname*{argmin}_{y\in\mathcal{Y}} \left\{ g^*(y) - \langle A\tilde{x}^k, y\rangle + \beta_s b_{\mathcal{Y}}(y, \dot{y}^s) \right\} \\ \hat{x}^{k+1} \leftarrow \mathcal{P}^{d_{\mathcal{X}}}_{\gamma_k f}\left(\hat{x}^k, A^\top\tilde{y}^{k+1}\right) \quad \text{with} \quad \gamma_k \leftarrow \frac{\beta_s}{\|A\|^2\tau_k}. \end{cases}$
9:         Update $\bar{x}^{k+1}$ using one of the following two options:
           $\left[\begin{array}{ll} \bar{x}^{k+1} \leftarrow \tilde{x}^k + \tau_k(\hat{x}^{k+1} - \hat{x}^k) & (\textbf{Option 1:} \text{ Averaging step}) \\ \bar{x}^{k+1} \leftarrow \mathcal{P}^{d_{\mathcal{X}}}_{\beta_s f/\|A\|^2}\left(\tilde{x}^k, A^\top\tilde{y}^{k+1}\right) & (\textbf{Option 2:} \text{ Proximal step}). \end{array}\right.$
10:        Update $\tau_k \leftarrow \frac{2}{k - K_s + 2}$.
11:    End For
12:    Update $K_{s+1} \leftarrow K_s + m_s$.
13:    Restart $\bar{x}^{K_{s+1}} \leftarrow \hat{x}^{K_{s+1}} \equiv \hat{x}^{K_s + m_s}$.
14:    Restart $\dot{y}^{s+1} \leftarrow \mathrm{prox}_{\frac{1}{\beta_s}g^*}\left(\dot{y}^s + \frac{1}{\beta_s}A\bar{x}^{K_{s+1}}\right)$.
15:    Restart $\tau_{K_{s+1}} \leftarrow 1$.
16:    Set $m_{s+1} \leftarrow \lfloor \omega(m_s + 1) + 1 \rfloor - 1$.
17:    Set $\beta_{s+1} \leftarrow \frac{\beta_s(m_{s+1}+1)}{\omega\sqrt{m_{s+1}(m_{s+1}+3)}}$
18: End For
```

step. To avoid this averaging, we can choose **Option 2**, which requires an additional generalized proximal operator of f.

We can replace Step 8 of Algorithm 1 by the following FISTA step:

$$\begin{cases} \tilde{y}^{k+1} \leftarrow \operatorname*{argmin}_{y\in\mathcal{Y}} \left\{ g^*(y) - \langle A\tilde{x}^k, y\rangle + \beta_s b_{\mathcal{Y}}(y, \dot{y}^s) \right\} \\ \bar{x}^{k+1} \leftarrow \mathcal{P}^{d_{\mathcal{X}}}_{\gamma_k f}\left(\tilde{x}^k, A^\top\tilde{y}^{k+1}\right) \quad \text{with} \quad \gamma_k \leftarrow \frac{\beta_s}{\|A\|^2} \\ \tilde{x}^{k+1} \leftarrow \bar{x}^{k+1} + \frac{(1-\tau_k)\tau_{k+1}}{\tau_k}(\bar{x}^{k+1} - \bar{x}^k). \end{cases} \tag{15}$$

However, we need to replace the general Bregman distance $d_{\mathcal{X}}$ by an Euclidean distance $d_{\mathcal{X}}(\cdot, \dot{x}) := \frac{1}{2}\|\cdot - \dot{x}\|^2$. The scheme (15) allows us to compute $\bar{x}^k$ through $\mathcal{P}^{d_{\mathcal{X}}}_{\gamma_k f}(\cdot)$ instead of a weighted averaging step as with **Option 1**.

Comparison between ASGARD [60] and Algorithm 1: When designing Algorithm 1, we focused on improving the practical efficiency of ASGARD while retaining the $\mathcal{O}\left(\frac{1}{k}\right)$-worst-case rate on the last primal iterate for the objective residual and feasibility violation. To achieve this goal, we introduced several fundamental changes to the algorithm.

Firstly, ASGARD only works with the Euclidean distance in the primal space while Algorithm 1 works with any Bregman distance.

Secondly, accelerated proximal gradient or (15) serves as a sub-routine from Step 6 to Step 11 in Algorithm 1 when β_k is fixed at β_s, while (15) is intertwined with updates of β_k in ASGARD.

Thirdly, as we discussed earlier, the parameter τ_k in ASGARD is gradually decreased to zero, making its performance to be slow. Algorithm 1 allows one to reset τ back to one at Step 15 making use of larger step-sizes at Steps 8 and 9. We can view Algorithm 1 as applying a multiple stage strategy to ASGARD, where we rerun ASGARD at a new but better initial point at each stage s.

Finally, Algorithm 1 can be the basis for a unified framework, where we replace the inner loop with any other accelerated schemes such as stochastic and coordinate descent variants.

The following lemma provides a key estimate for the optimality condition of (2), whose proof is given in Appendix 6.2.

Lemma 3.1. *Suppose that $b_{\mathcal{Y}}$ has an $L_{b_{\mathcal{Y}}}$-Lipschitz gradient, $L_{b_{\mathcal{Y}}} \in (0, +\infty]$ and that g is $\widehat{M}_g$-Lipschitz on its domain. Let $(x^\star, y^\star)$ be a saddle point of the Lagrange function of* (1) *and*

$$S_\beta(\bar{x}, \dot{y}) := \max_{y\in\mathbb{R}^n}\Big\{f(\bar{x}) + \langle y, A\bar{x}\rangle - g^*(y) - \beta b_{\mathcal{Y}}(y, \dot{y}) - f(x^\star) - g(Ax^\star)\Big\}. \tag{16}$$

Let $\beta_b := \beta L_{b_{\mathcal{Y}}}$, $\bar{y}^*_\beta = y^*_\beta(A\bar{x}, \dot{y})$, and $\bar{z}$ a projection of $A\bar{x}$ onto $\mathrm{dom}(g)$*. If either $L_{b_{\mathcal{Y}}} < +\infty$ or $D_{\mathcal{Y}} < +\infty$, then we have*

$$\begin{cases} f(\bar{x}) + g(\bar{z}) - P^\star & \geq -\|y^\star\|_{\mathcal{Y}}\,\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}, \mathrm{dom}(g)\right)\\ f(\bar{x}) + g(\bar{z}) - P^\star & \leq S_\beta(\bar{x}, \dot{y})\\ & \quad + \beta\min\Big\{D_{\mathcal{Y}}, (2\widehat{M}_g + \|\dot{y}\|_{\mathcal{Y}})\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta, \dot{y})\|_{\mathcal{Y},*}\Big\}\\ \mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}, \mathrm{dom}(g)\right) & \leq \beta\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta, \dot{y})\|_{\mathcal{Y},*}\\ & \leq \beta_b\Big[\|y^\star - \dot{y}\|_{\mathcal{Y}} + \sqrt{\|y^\star - \dot{y}\|^2_{\mathcal{Y}} + \frac{2}{\beta_b}S_\beta(\bar{x}, \dot{y})}\Big]. \end{cases} \tag{17}$$

Now, we are ready to state the main convergence result in the following theorem, whose proof is given in Appendix 7.

Theorem 3.1. *Assume that Assumption 1 holds. Let $\left\{\bar{x}^{K_s}\right\}$ be the sequence generated by Algorithm 1 and $\bar{z}^{K_s}$ be the projection of $A\bar{x}^{K_s}$ onto* $\mathrm{dom}(g)$*. If* *either $L_{b_{\mathcal{Y}}} < +\infty$ or $D_{\mathcal{Y}} < +\infty$, then we have*

$$\begin{cases} f(\bar{x}^{K_s}) + g(\bar{z}^{K_s}) - P^\star \geq -\|y^\star\|_{\mathcal{Y}}\ \mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s}, \mathrm{dom}(g)\right) \\ f(\bar{x}^{K_s}) + g(\bar{z}^{K_s}) - P^\star \leq \dfrac{\omega\kappa_0 R_0^2}{\rho_0\left[(\omega-1)K_s + \kappa_0\right]} + \dfrac{\beta_0\omega\kappa_0 C_0^\star}{(\omega-1)K_s+\kappa_0} \\ \mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s}, \mathrm{dom}(g)\right) \leq \dfrac{\beta_0 L_{b_{\mathcal{Y}}}\omega\kappa_0(\sqrt{2}+2)R_0}{\rho_0[(\omega-1)K_s+\kappa_0]}, \end{cases} \tag{18}$$

where $y^\star$ is any dual solution of (4)*, and*

$$\begin{cases} \rho_0 := \beta_0\left(1 - \frac{1}{(\omega-1)m_0}\right) \\ \kappa_0 := m_0 + \frac{\omega}{\omega-1} \\ R_0 := \left[\frac{4\|A\|^2}{(m_0+1)^2} d_{\mathcal{X}}(x^\star, \bar{x}^0) + \frac{\beta_0^2 m_0(m_0+3)}{(m_0+1)^2} b_{\mathcal{Y}}(y^\star, \dot{y}^0)\right]^{1/2} \\ C_0^\star := \min\left\{D_{\mathcal{Y}}, (\sqrt{2}+2)\frac{L_{b_{\mathcal{Y}}} R_0}{\rho_0}\left[2\widehat{M}_g + \|y^\star\|_{\mathcal{Y}} + \frac{\sqrt{2}R_0}{\rho_0}\right]\right\}. \end{cases}$$

Consequently, Algorithm 1 achieves an $\mathcal{O}\big(\frac{1}{K_s}\big)$ convergence rate in a ***non-ergodic sense****: the objective satisfies $|f(\bar{x}^{K_s}) + g(\bar{z}^{K_s}) - P^\star| \leq \mathcal{O}\big(\frac{1}{K_s}\big)$ and the constraints satisfy* $\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s}, \mathrm{dom}(g)\right) \leq \mathcal{O}\big(\frac{1}{K_s}\big)$.

Remark 3.1. We make a few remarks about Theorem 3.1.
- The smoothness parameter β is only updated at the outer loop but with a geometric rate, depending on the factor parameter ω. We can select different ω to observe its performance in particular applications.
- The convergence rate is given at the last iterate instead of the averaged sequence as often seen in other primal-dual methods [13,52,57,68].
- In the case where g is Lipschitz continuous ($D_{\mathcal{Y}} < +\infty$), we can choose any dual prox-function. Indeed, as $\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s}, \mathrm{dom}(g)\right) = 0$, we do not need the third inequality and we can choose $b_{\mathcal{Y}}$ such that $L_{b_{\mathcal{Y}}} = +\infty$. Moreover, $\bar{z}^{K_s} = A\bar{x}^{K_s}$. Hence, $f(\bar{x}^{K_s}) + g(\bar{z}^{K_s}) = f(\bar{x}^{K_s}) + g(A\bar{x}^{K_s}) = P(\bar{x}^{K_s})$. See Corollary 3.2 below.
- When $\mathcal{Y}$ is unbounded, i.e. $D_{\mathcal{Y}} = +\infty$ (for instance when there are constraints), we need to choose a smooth prox-function for the dual problem. We develop this case in detail in Subsection 3.2.
- Evaluating the projection of a vector onto $\mathrm{dom}(g)$ is usually a simple task when prox_g has an explicit form.
- The convergence rate depends on both the prox-distance between $\bar{x}^0$ to $x^\star$ and $\dot{y}^0$ to $y^\star$.

Remark 3.2. By using (42) in our analysis, we can show that the approximated objective sequence $\left\{f(\bar{x}^{K_s}) + g(\bar{z}^{K_s})\right\}$ converges to $P^\star$ at the rate of $\mathcal{O}\left(\frac{1}{k}\right)$ for any $k \geq 1$ instead of $k = K_s$ at the outer loop only.
- If we use the averaging step of APG, then $\bar{x}^k$ is computed via a weighted averaging step of the inner loop.

– However, if we use the proximal step in APG or the FISTA scheme (15), then $\bar{x}^k$ is computed through the generalized proximal operator $\mathcal{P}^{d_{\mathcal{X}}}_{\beta_s f/\|A\|^2}$. This rate is fully non-ergodic for both inner and outer loops.

3.2 Application to constrained convex optimization

In this subsection, we specify Algorithm 1 to solve the constrained problem (2). First, we choose the Bregman distance used in smoothing for the dual variables to have Lipschitz gradient. Under this condition, we have

$$b_{\mathcal{Y}}(y,\dot{y}) \le \frac{L_{b_{\mathcal{Y}}}}{2}\|y-\dot{y}\|^2_{\mathcal{Y}}. \tag{19}$$

Let us define $g(Ax) := \delta_{\mathcal{K}}(Ax-b)$ the indicator function of $\mathcal{K}$, where the set $\mathcal{K}$ is such that Slater's condition in Assumption 1 holds. Then, we can write

$$g(Ax) := \sup_{y\in\mathbb{R}^n}\left\{\langle Ax-b, y\rangle - s_{\mathcal{K}}(y)\right\}, \tag{20}$$

where $s_{\mathcal{K}}(y) := \sup_{u\in\mathcal{K}}\langle y,u\rangle$ is the support function of $\mathcal{K}$. In this case, the smooth function $g_\beta(Ax;\dot{y})$ becomes

$$g_\beta(Ax;\dot{y}) := \max_{y\in\mathbb{R}^n}\Big\{\langle Ax-b,y\rangle - s_{\mathcal{K}}(y) - \beta b_{\mathcal{Y}}(y,\dot{y})\Big\}. \tag{21}$$

Example 3.1. *Assume that we choose $b_{\mathcal{Y}}(x,\dot{x}) = \frac{1}{2}\|x-\dot{x}\|_2^2$. Then*

$$g_\beta(Ax;\dot{y}) = \frac{1}{2\beta}\mathrm{dist}_{\mathcal{K}}\left(Ax-b+\beta\dot{y}\right)^2 - \frac{\beta}{2}\|\dot{y}\|^2. \tag{22}$$

Moreover, the solution $y^*_\beta(Ax;\dot{y})$ of the maximization problem in (21) is

$$y^*_\beta(Ax;\dot{y}) = \dot{y} + \tfrac{1}{\beta}\left(Ax-b-\mathrm{proj}_{\mathcal{K}}\left(Ax-b+\beta\dot{y}\right)\right), \tag{23}$$

where $\mathrm{proj}_{\mathcal{K}}(\cdot)$ denotes the projection onto $\mathcal{K}$.

In particular, if $\mathcal{K}$ is a cone, then $y^*_\beta(Ax;\dot{y}) = \mathrm{proj}_{-\mathcal{K}^*}\Big(\dot{y}+\frac{1}{\beta}(Ax-b)\Big)$, where $\mathcal{K}^*$ is the dual cone of $\mathcal{K}$. The dual step for computing $\tilde{y}^k$ at the second line of Step 8 of Algorithm 1 becomes

$$\begin{aligned}\tilde{y}^{k+1} &\leftarrow \dot{y}^s + \tfrac{1}{\beta_s}\left(A\tilde{x}^k - b - \mathrm{proj}_{\mathcal{K}}\left(A\tilde{x}^k - b + \beta_s\dot{y}^s\right)\right)\\ &= \tfrac{1}{\beta_s}\mathrm{proj}_{-\mathcal{K}^*}\left(A\tilde{x}^k - b + \beta_s\dot{y}^s\right).\end{aligned} \tag{24}$$

In this case, we can apply Theorem 3.1 to (2) with $g(\bar{z}^{K_{s+1}}) = 0$, $D_{\mathcal{Y}} = +\infty$, and $\widehat{M}_g = 0$ to obtain the following corollary.

Corollary 3.1. *Assume that Assumption 1 holds and that $g(z) := \delta_{\mathcal{K}}(z-b)$. Let us choose $b_{\mathcal{Y}}$ such that $L_{b_{\mathcal{Y}}} < +\infty$. Let $\left\{\bar{x}^{K_s}\right\}$ be the sequence generated by Algorithm 1 and $\bar{z}^{K_s}$ be the projection of $A\bar{x}^{K_s}$ onto* $\mathrm{dom}(g)$. *Then, we have*

$$\begin{cases} f(\bar{x}^{K_s}) - P^\star \ge -\|y^\star\|_{\mathcal{Y}}\,\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s},\mathrm{dom}(g)\right)\\ f(\bar{x}^{K_s}) - P^\star \le \dfrac{\omega\kappa_0}{(\omega-1)K_s+\kappa_0}\left[\dfrac{R_0^2}{\rho_0} + (\sqrt{2}+2)\dfrac{\beta_0 L_{b_{\mathcal{Y}}}R_0}{\rho_0}\big(\,\|y^\star\|_{\mathcal{Y}} + \dfrac{\sqrt{2}R_0}{\rho_0}\big)\right]\\ \mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s},\mathrm{dom}(g)\right) \le \dfrac{\beta_0 L_{b_{\mathcal{Y}}}\omega\kappa_0(\sqrt{2}+2)R_0}{\rho_0[(\omega-1)K_s+\kappa_0]}, \end{cases}$$

where $y^\star$ is any dual solution of (6)*, and ρ_0, κ_0 and R_0 are defined as in Theorem 3.1.*

3.3 Application to Lipschitz convex optimization

If g is globally Lipschitz on $\mathbb{R}^n$, then we recover the framework of [49]. In this case, $\mathrm{dom}(g) = \mathbb{R}^n$ and $\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s}, \mathrm{dom}(g)\right) = 0$. Moreover, $D_{\mathcal{Y}} < +\infty$. Theorem 3.1 simplifies as follows.

Corollary 3.2. *Assume that Assumption 1 holds and g is M_g-globally Lipschitz continuous. Let $\left\{\bar{x}^{K_s}\right\}$ be the sequence generated by Algorithm 1. Then*

$$0 \le f(\bar{x}^{K_s}) + g(A\bar{x}^{K_s}) - P^\star \le \frac{\omega\kappa_0}{(\omega-1)K_s + \kappa_0}\left[\frac{R_0^2}{\rho_0} + \beta_0 D_{\mathcal{Y}}\right], \tag{25}$$

where ρ_0, κ_0 and R_0 are defined as in Theorem 3.1.

3.4 Extension to composite case with three objective terms

It is straightforward to apply Algorithm 1 in the presence of a smooth term in the objective. The problem template we focus on in this section is

$$F^\star := \min_{x\in\mathbb{R}^p}\left\{F(x) := f(x) + g(Ax) + h(x)\right\}, \tag{26}$$

where f and g are as described in Assumption 1 and h is a differentiable function with L_h-Lipschitz gradient. In this case, only Step 8 in Algoritm 1 needs to be modified as follows (see also in [50]):

$$\hat{x}^{k+1} \leftarrow \mathcal{P}^{d_{\mathcal{X}}}_{\gamma_k f}\left(\hat{x}^k, \nabla h(\tilde{x}^k) + A^\top \tilde{y}^{k+1}\right) \quad \text{with} \quad \gamma_k \leftarrow \tfrac{\beta_s}{\tau_k(\|A\|^2 + \beta_s L_h)}.$$

Note that this modification only changes the analysis of the inner loop as in [50] which does not affect our analysis of the outer loop. In addition, using L_h in the stepsize is not restrictive. When the Lipschitz constant is not known, line search strategies can be employed, see [50] for more details. The convergence of this variant is still guaranteed by Theorem 3.1 but the quantity R_0^2 will depend on L_h. We omit the details of this result here for succinctness.

4 Numerical experiments

We will test standard ASGARD [60,50], ASGARD with restart [60,50] and standard Chambolle-Pock's algorithm [13] on the following problems. Note that when there is a smooth term in the objective, we use the version of Chambolle-Pock which linearizes the smooth term, which is also known in the literature as Vu-Condat's algorithm [63,19]. We only compare with HOPS [70] in the first example because it does not apply to Basis pursuit, sparse subspace clustering, linear programming and Markowitz's portfolio optimization problems due to the unboundedness of the dual domain. For the ℓ_1-SVM example, we observed it to be extremely slow and difficult to tune for different datasets. In all the experiments, we have used the standard $b_{\mathcal{Y}}(y_1, y_2) = \frac{1}{2}\|y_1 - y_2\|_2^2$

and $d_{\mathcal{X}}(x_1, x_2) = \frac{1}{2}\|x_1 - x_2\|_2^2$ for smoothing and computing the proximal operators for fair comparison with other methods which do not allow Bregman distances. In the sequel, we refer to our algorithm as ASGARD-DL, and we only run Option 1 at Step 9. In some cases, we also compare with ADMM and its variants.

The parameters are set as follows. For Chambolle-Pock's method, we set its step-sizes $\sigma := \frac{1}{\|A\|}$ and $\tau := \frac{0.9999}{\|A\|^2\sigma}$, where A is the linear operator in (1). For ASGARD-DL, we choose $\omega := 1.2$ and $m_0 := 6$ which gives us comparable performance. For restarting ASGARD, we set the restarting frequency to be $s = 10$ in all experiments.

4.1 Convergence guarantees: Ergodic vs. Non-ergodic

ALM, ADMM and Chambolle-Pock methods have the convergence rate guarantees in an ergodic sense. That is, they have the rate guarantees only on the averaged iterate sequence. In contrast, our guarantees are for the last iterate of the algorithm. To illustrate the importance between these two, we consider two synthetic problems in this section. The first one is a square root LASSO problem widely studied in the literature, which is given by:

$$F^\star := \min_{x \in \mathbb{R}^p} \Big\{ F(x) := \|Ax - b\|_2 + \lambda\|x\|_1 \Big\},$$

where $A \in \mathbb{R}^{n \times p}$ is generated using a Gaussian distribution and is normalized such that column norms are equal to 1. Given a groundtruth vector $x^\natural$, we generate the observations as $b = Ax^\natural + \sigma\mathbf{n}$, where $\mathbf{n}$ is a noise vector generated by a standard Gaussian distribution and $\sigma = 0.01$. We set $\lambda = 0.03$ which is tuned to get a good recovery of $x^\natural$.

In this experiment, we test the ergodic and non-ergodic variants of Linearized ADMM (in the sense that the augmented term in the Lagrangian is linearized) [30] and Chambolle-Pock's algorithm [13], as well as the primal dual homotopy smoothing method HOPS [70]. The methods in [30,13] have convergence guarantees for their last iterates, however, their rate guarantees only apply to the averaged sequence. Moreover, they are very successful to solve this type of problems as can be seen from the literature. The behavior of the algorithms is given in Figure 1.

As can be seen in Figure 1, last iterates of Linearized ADMM and Chambolle Pock's algorithms seem to have the best performance. However, the averaged iterates for which the methods have the rate guarantees shows the slowest convergence behavior. Our method has the same rate as restarted ASGARD which does not have any convergence guarantees.

As can be observed in Figure 1, the performance of HOPS shows too much fluctuation with different datasets. We suspect that the reason is that HOPS requires 3 separate parameters which are very difficult to tune, so a parameter set that is working well for one dataset performs very poorly for another. Note that we are using the same set of parameters for different datasets for all the algorithms. HOPS requires knowing $\epsilon_0 \geq F(x^0) - F^\star$ which we bound using the fact that $F^\star \geq 0$ and we set $\epsilon_0 = F(x^0)$. The second parameter is the

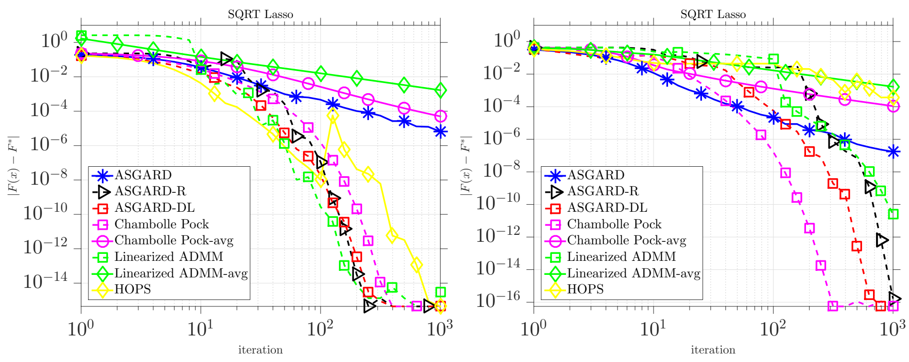

Fig. 1 Performance of 5 algorithms for solving square root LASSO problem. Left: $\sigma = 0.1, \lambda = 0.04$, Right: $\sigma = 0.01, \lambda = 0.03$

rate at which they decrease the smoothness parameter, which is similar to the ω parameter in our algorithm. The last parameter is the number of inner iterations they need to run which is constant across the run of the method. For Algorithm 1 in contrast, we only require setting the initial parameters for the inner iteration and smoothness parameter and we use homotopy to set them for further iterations. As we illustrate, we have obtained a similar performance across different datasets with the same set of parameters for our method.

To illustrate the behavior of the last iterates of Linearized ADMM and Chambolle Pock's algorithm, we consider a degenerate linear program which is also studied in [60]:

$$\min_{x\in\mathbb{R}^p} \left\{ h(x) := 2x_p \ \middle|\ \sum_{k=1}^{p-1} x_k = 1, \quad x_p - \sum_{k=1}^{p-1} x_k = 0 \quad (2 \le j \le n), \quad x_p \ge 0 \right\}.$$

The second inequality is repeated $n-1$ times which causes the problem to be degenerate. We define the linear constraint as

$$Ax := \left[\sum_{k=1}^{p-1} x_k, \quad x_p - \sum_{k=1}^{p-1} x_k, \cdots, \quad x_p - \sum_{k=1}^{p-1} x_k \right]^\top.$$

We have $b := (1, 0, \cdots, 0)^\top \in \mathbb{R}^n$. We map the problem to our template in (26) as $f(x) := \delta_{\{x_p \ge 0\}}(x_p)$, $g(x) := \delta_{\{b\}}(Ax)$, and $h(x) := 2x_p$. For this problem, we pick $p = 10$ and $n = 200$.

In addition to Linearized ADMM and Chambolle-Pock's algorithm, we also include linearized ALM [30] to solve this example. The result of this test is given in Figure 2, where $F(x) = h(x)$.

As can be seen from Figure 2, Linearized ADMM, Linearized ALM and Chambolle-Pock's algorithm can get extremely slow where our algorithm and ASGARD with restart makes progress and converges to optimal value with a very high accuracy, and beyond the theoretical rate guarantee.

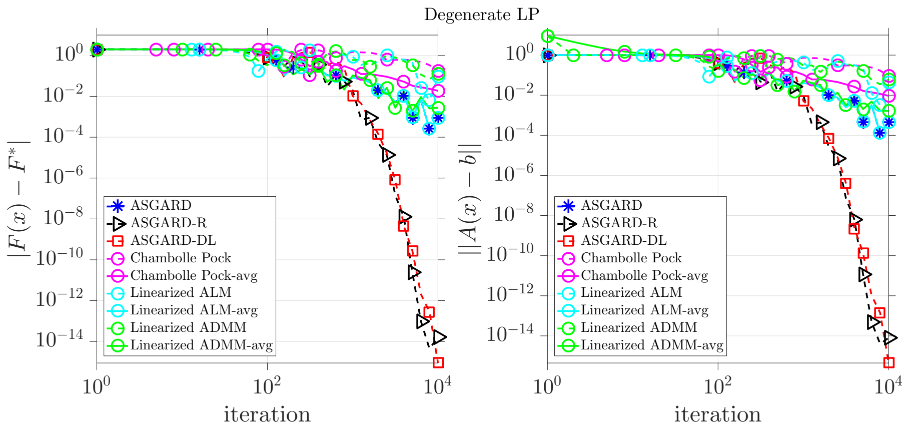

Fig. 2 Performance of 6 algorithms for solving the degenerate linear program.

4.2 Basis Pursuit for recovering Bag-of-Words of text documents

We first consider a basis pursuit problem which is used in signal/image processing, statistics, and machine learning [17,23,12]:

$$\min_{x\in\mathbb{R}^p} \left\{ F(x) := \|x\|_1 \ \mid \ Ax = b \right\}, \tag{27}$$

where $A \in \mathbb{R}^{n\times p}$ and $b \in \mathbb{R}^n$. This problem clearly fits into our template (1) by mapping $f(\cdot) = \|\cdot\|_1$ and $g(\cdot) = \delta_{\{b\}}(\cdot)$. It is also a special case of (2) with $\mathcal{K} = \{\mathbf{0}\}$. Proximal operators of both terms are given in a closed form.

We apply this model to text processing. In [1], the authors proposed using basis pursuit formulation to obtain bag-of-words representation from the unigram embedding representation of a text. The setting can be briefly described as the following: For any word w, there exists a word vector $v_w \in \mathbb{R}^n$. For a given text document $\{w_1, \cdots, w_T\}$, one defines the unigram embedding as $\sum_{i=1}^T v_{w_i}$. It is easy to see that unigram embeddings can be written as a linear system Ax where $A \in \mathbb{R}^{n\times p}$ contains v_{w_i} in the i^{th} column and $x \in \mathbb{R}^p$ is the bag-of-words vector which counts the number of occurances of words in a text. This application is considered in text processing applications to obtain the original text document given the unigram embeddings [65].

For this experiment, we have used the movie review dataset of [40]. We have selected 4 different documents and computed the unigram embeddings using pre-trained word embeddings from `GloVe` [54] with $n = 50$ as the dimension of the word vectors and restricted the vocabulary size to $p = 10,000$ for getting faster results with all algorithms.

We have applied 4 methods to solve (27) for 4 different documents. Here, the parameter β_0 in ASGARD, ASGARD-restart, and ASGARD-DL is set to $\beta_0 := 10\|A\|$. Note that this choice is not optimal, but give us reasonable results in all test. The results are compiled in Figure 3.

As we can observe from Figure 3, our new algorithm works quite well and is comparable with state-of-the-art methods for low accuracy. It outperforms them if we run the algorithms long enough to get more accurate solutions

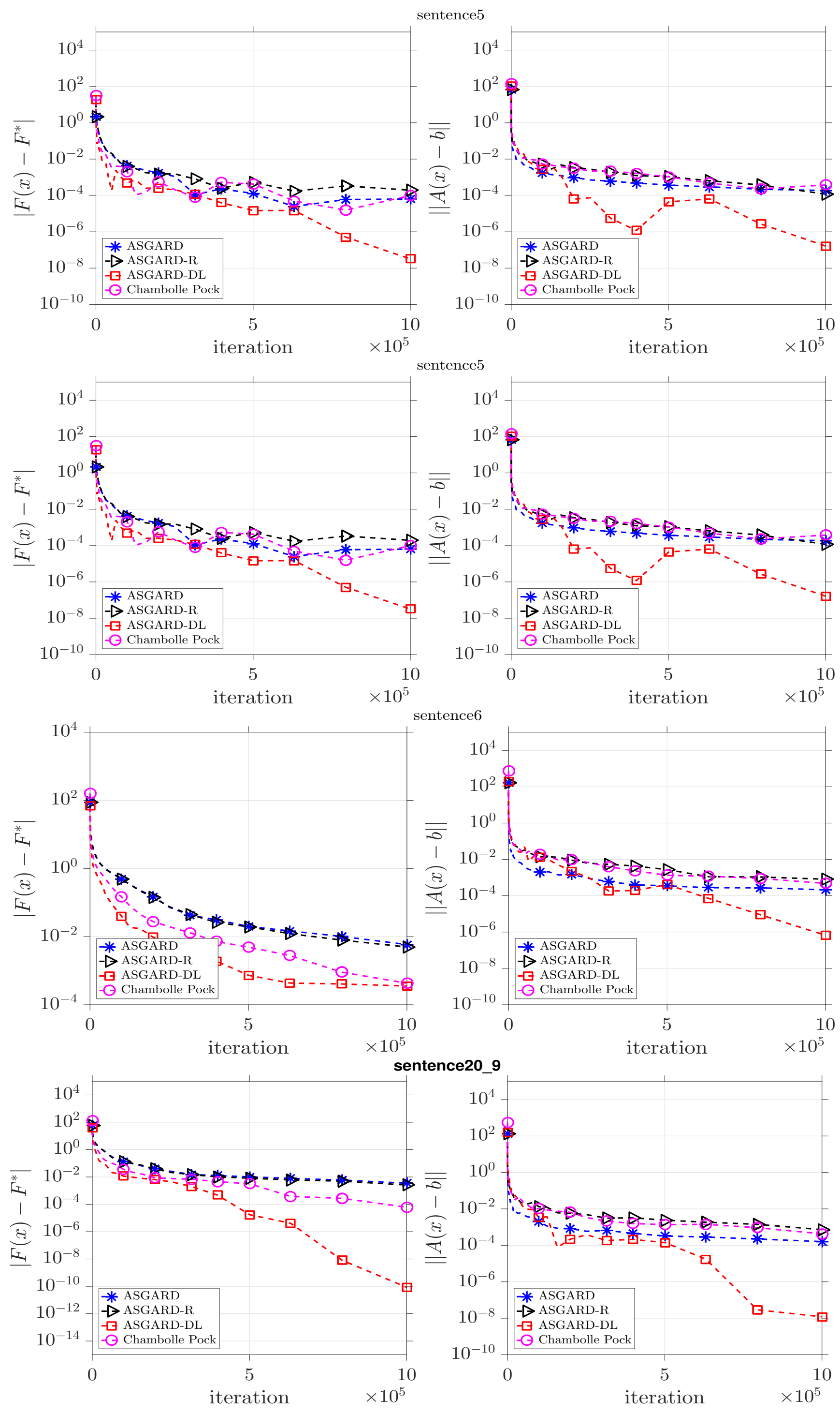

Fig. 3 Performance of 4 algorithms for solving basis pursuit for 4 text documents.

than $\varepsilon = 10^{-5}$ both in objective residual and feasibility. Note that (27) is fully nonsmooth, and A is non-orthogonal. If we apply ADMM to solve (27), then it requires to solve a general convex subproblem, or a linear system, which has higher per-iteration complexity than four methods we used in this example.

4.3 The ℓ_1-Regularized Least Absolute Deviation Problem (LAD)

Our second example is the ℓ_1-regularized least absolute deviation regression problem, also known as LAD-Lasso in the literature. It is known that when the noise has a heavy tailed distribution such as Laplace distribution, LAD-Lasso is more robust to the outliers [64]. The optimization model of this problem is

$$\min_{x\in\mathbb{R}^p}\left\{F(x) := \|Ax-b\|_1 + \lambda\|x\|_1\right\},$$

where $A \in \mathbb{R}^{n\times p}$ is generated according to a normal distribution and the noise $\mathbf{n} \in \mathbb{R}^n$ is generated by Laplace$(0,1)$ distribution. We generate an observed vector $b := Ax^\natural + \sigma\mathbf{n}$, where $\sigma := 0.1$ and $x^\natural$ is a s-sparse vector of ground-truth. We choose $\lambda := 1/n$ for the regularization parameter, which gives us a good recovery of $x^\natural$.

This problem fits to our template by setting $f(\cdot) := \lambda\|\cdot\|_1$ and $g(\cdot) = \|\cdot - b\|_1$. We set β_0 in ASGARD, ASGARD-restart, and ASGARD-DL as $\beta_0 := 100\|A\|$. We generated three problem instances of the size $n := 340r$, $p := 1000r$, $s := 100r$, where s is the sparsity level, and $r = 1, 2, 3$ for the first, second and third instances, respectively. We present the results of this example in Figure 4.

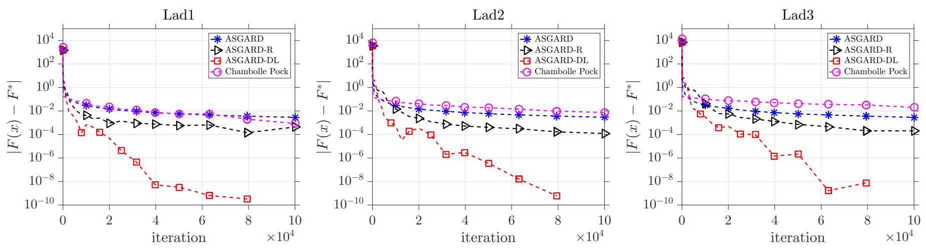

Fig. 4 Performance of 4 algorithms for LAD-Lasso problem in 3 different realization of varying problem size.

As we can see from Figure 4 that, with the same per-iteration complexity, our method significantly outperforms the other algorithms after accuracy 10^{-4}. It beats other algorithms after a couple of hundred iterations and continues to decrease the objective values. Although this problem is fully nonsmooth, heuristic restart such as in ASGARD still improves the performance of the non-restart one, but does not significant outperform.

4.4 Support Vector Machines

Our next example is the following primal support vector machines (SVM) problem in binary classification. Instead of classical models, we consider the following ℓ_1-regularized nonsmooth hinge loss as proposed in [71]:

$$\min_{x\in\mathbb{R}^p}\left\{F(x) := \frac{1}{n}\sum_{i=1}^{n}\max\left\{0, 1 - b_i\langle a_i, x\rangle\right\} + \lambda\|x\|_1\right\}, \tag{28}$$

where $a_i \in \mathbb{R}^p$ are the feature vectors and $b_i \in \{-1, +1\}$ are the labels for $i = 1, \cdots, n$. We can cast (28) into our template by setting $f(\cdot) := \lambda\|\cdot\|_1$ and

$$g(Ax) = \frac{1}{n}\sum_{i=1}^{n}\max\{0, 1 - b_i\langle a_i, x\rangle\} = \max_{y\in[0,1]^n}\langle y, Ax + \frac{1}{n}\mathbb{1}\rangle,$$

where $A := -\frac{1}{n}\left[b_1a_1, b_2a_2, \cdots, b_na_n\right]^\top$ and $\mathbb{1}$ is a vector of all ones. Clearly, the proximal operator of g is simply a projection onto $[0, 1]^n$.

We use 10 different datasets from `libsvm` [15] to test four different algorithms. The initial value β_0 in ASGARD, ASGARD-restart, and ASGARD-DL is set to $\beta_0 := 0.1\|A\|$. But for `covtype` dataset, we used $\beta_0 := 0.01\|A\|$. The details about the datasets are given in Table 1. We test 4 algorithms on these

Table 1 Datasets used for classification.

Data set	Training size	Number of features
w1a	2,477	300
w2a	3,470	300
w3a	4,912	300
w4a	7,366	300
w5a	9,888	300
w6a	17,188	300
w7a	24,692	300
w8a	49,749	300
rcv1	20,242	47,236
covtype	581,012	54

ten datasets. The results of the first 8 problems are given in Figure 5, and the results of the two last problems are in Figure 6.

We again observe that Algorithm 1 significantly outperform the other methods. Since these algorithms have the same per-iteration complexity, it is sufficient to compare them in terms of iteration numbers. Although all the algorithms have $\mathcal{O}\left(\frac{1}{k}\right)$-worst-case convergence rate, due to its double-loop, Algorithm 1 performs much better than the others, especially for high accurate solutions. This is not surprise. The double-loop allows Algorithm 1 to use large stepsize by frequently restarting τ_k and β_k, while ASGARD gradually decreases these parameters to zero, and Chambolle-Pock's method fixes the step-size. Note that the $\mathcal{O}\left(\frac{1}{k}\right)$ rate of Chambolle-Pock's method is achieved via the averaging sequence, which is often much slower than the last iteration as we showed in Figures 5 and 6.

4.5 Markowitz Portfolio Optimization

We consider a classical example from Markowitz portfolio optimization [11]. The setting we consider here aims at maximizing the expected return for a

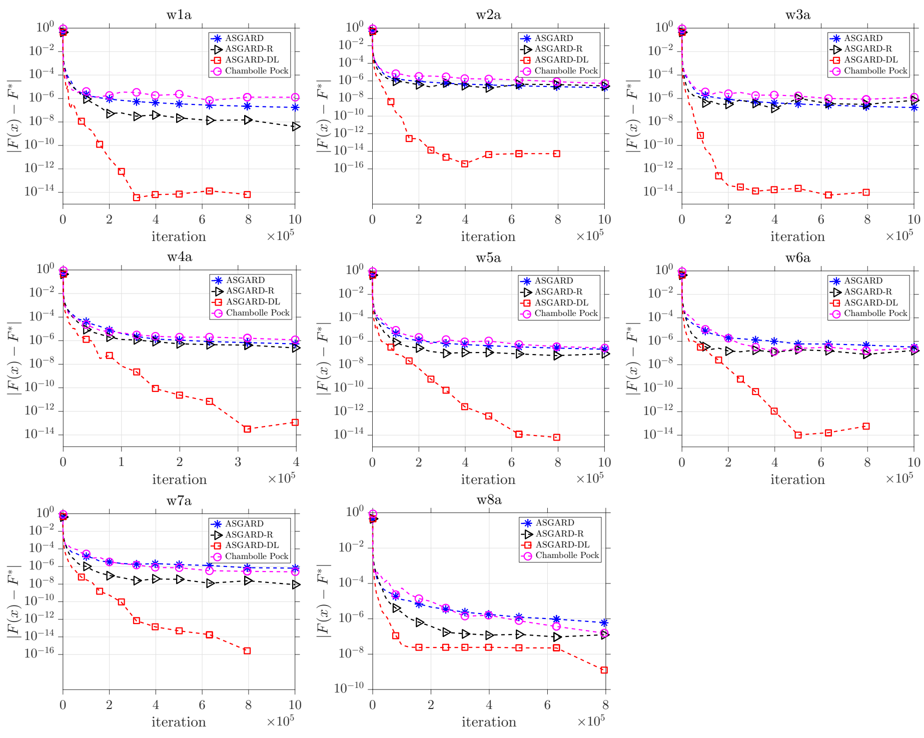

Fig. 5 Performance of 4 algorithms for the ℓ_1-regularized SVM problem on $\{\texttt{w1a}, \cdots, \texttt{w8a}\}$.

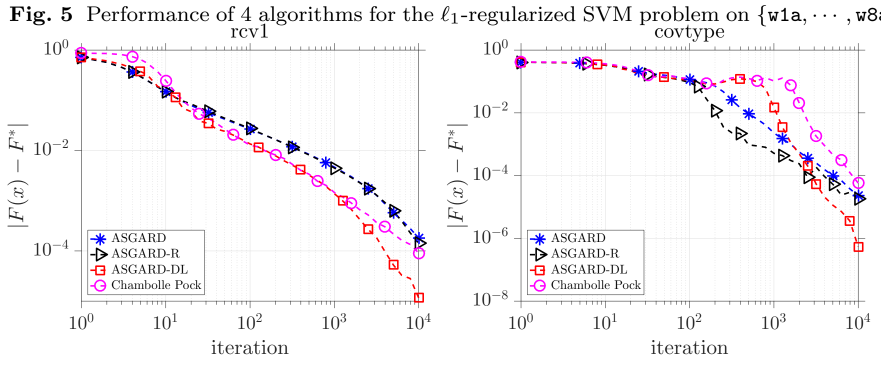

Fig. 6 Performance of 4 algorithms for the ℓ_1-regularized SVM problem on $\{\texttt{rcv1}, \texttt{covtype}\}$.

given risk level. Assume that we are given a vector $\rho \in \mathbb{R}^n$, where ρ is composed of expected returns from n assets. This problem can be formulated as

$$\max_{x \in \mathbb{R}^p} \left\{ \rho^\top x \;\mid\; x \in \triangle,\; \mathbb{E}\left[|(a_i - \rho)^\top x|^2 \right] \le \epsilon \right\}, \tag{29}$$

For our setting, we use empirical sample average instead of the expectation and convert the problem to a minimization problem by negating the objective:

$$\min_{x \in \mathbb{R}^p} \left\{ -\langle \rho, x \rangle \;\mid\; x \in \triangle,\; \tfrac{1}{p}\|Ax\|_2^2 \le \epsilon \right\}, \tag{30}$$

where $A = [(a_1 - \rho), (a_2 - \rho), \dots, (a_n - \rho)]^\top$. We map this problem to our template (26) by mapping $f(\cdot) := \delta_{\triangle}(\cdot)$, $g(\cdot) := \delta_{\{\|\cdot\|_2 \le \sqrt{p\epsilon}\}}(\cdot)$, and $h(x) := -\langle \rho, x \rangle$. One key step of primal-dual algorithms is computing the projection onto an ℓ_2-norm ball and on a simplex. Here, the complexity of simplex projection is $\mathcal{O}(p \log p)$.

As before, we apply 4 algorithms to solve (29). We use 4 datasets that are also considered in [6]. The details about the datasets are given in Table 2.

Table 2 Portfolio optimization datasets and parameters of algorithms.

The size of datasets				Parameters used in 4 algorithms.					
Datasets	n	p	ϵ in (30)	β_0	RF	ω	m_s	τ	σ
DJIA	507	30	0.002	$\|A\|$	10	1.1	11	$\frac{1}{\|A\|}$	$\frac{1}{\|A\|}$
NYSE	5651	36	0.02	$100\|A\|$	10	1.1	11	$\frac{1}{\|A\|}$	$\frac{1}{\|A\|}$
SP500	1276	25	0.02	$100\|A\|$	10	1.2	6	$\frac{1}{\|A\|}$	$\frac{1}{\|A\|}$
TSE	1258	88	0.002	$100\|A\|$	10	1.1	11	$\frac{1}{\|A\|}$	$\frac{1}{\|A\|}$

We summarized the parameters that we used for these algorithms in Table 2, where β_0 is common to ASGARD, ASGARD-restart, our algorithm, restart frequency (RF) is specific to ASGARD-restart, ω and m_s are specific to our algorithm and τ and σ are specific to Chambolle-Pock's algorithm.

We have tested 4 algorithms on 4 real datasets and the results are compiled in Figure 7. As can be seen, except for SP500 dataset, Algorithm 1 significantly outperforms the other methods and shows a much faster practical performance than $\mathcal{O}\left(\frac{1}{k}\right)$ guarantee. For SP500 dataset, ASGARD-restart algorithm shows a comparable performance to our method. However, as discussed in [60], the effect of restarting to ASGARD method is not understood theoretically. Our algorithm theoretically preserves the best-known $\mathcal{O}\left(\frac{1}{k}\right)$ guarantee while performing as fast as, and most of the times faster than the heuristic restarting ASGARD method.

4.6 Sparse Subspace Clustering

In the last example, we consider the following sparse subspace clustering problem which has broad applications in machine learning, computer vision and image processing. This problem is studied extensively in the literature [27, 26, 55]. In this problem setting, we assume that there exist n points $\{x_1, x_2, \cdots, x_n\} \in \mathbb{R}^p$ lying in the union of subspaces in $\mathbb{R}^p$. We form a matrix $X \in \mathbb{R}^{p \times n}$ by stacking $\{x_1, x_2, \cdots, x_n\}$ as the columns. With this notation, each point can be represented as

$$x_j = Xc_j + e_j, \quad \text{s.t.} \quad [c_j]_j = 0 \quad \text{and} \quad \mathbb{1}^\top c_j = 1.$$

where $c_j \in \mathbb{R}^n$ represents the coefficients to represent point $x_j \in \mathbb{R}^p$ as an affine combination of other points, $e_j \in \mathbb{R}^p$ is the representation error and $\mathbb{1} \in \mathbb{R}^n$ is a vector of 1's.

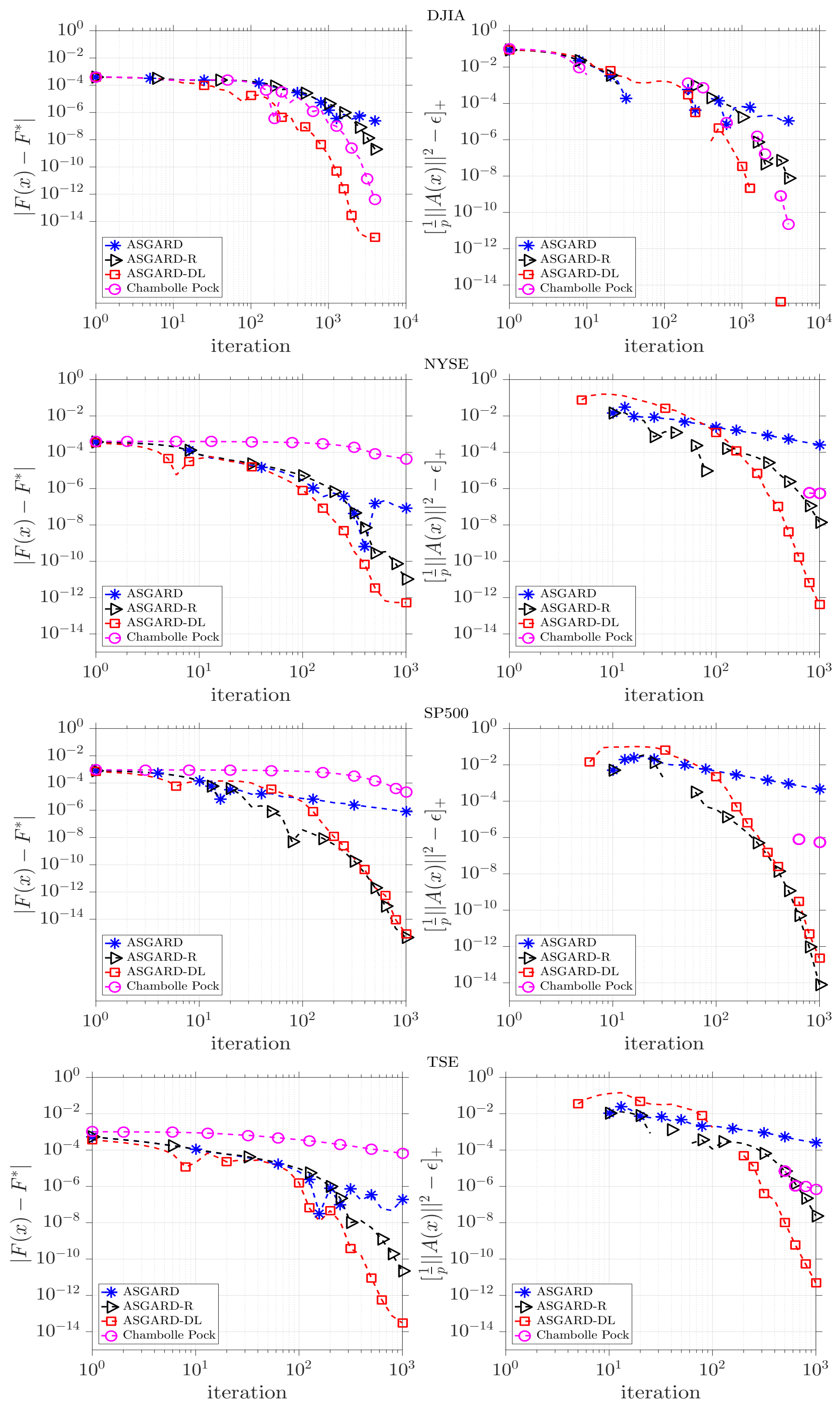

Fig. 7 Performance of 4 algorithms for Markowitz portfolio optimization problem on 4 real datasets.

This formulation can be represented compactly by stacking c_j to the j^{th} column of matrix C as follows:

$$X = CX \quad \text{s.t.} \quad \mathrm{diag}(C) = 0,\ C^\top \mathbb{1} = \mathbb{1}. \tag{31}$$

The optimization problem that we will tackle in this subsection is referred to as an SSC-Lasso problem in the literature, and is written as

$$\min_{C \in \mathbb{R}^{n\times n}} \Big\{ \|C\|_1 + \tfrac{\lambda}{2} \|X - CX\|^2 \ \mid\ \mathrm{diag}(C) = 0,\ C^\top \mathbb{1} = \mathbb{1} \Big\}. \tag{32}$$

In [26] and [27], ADMM is used to solve (32) and recently, [55] proposed an efficient implementation of ADMM and an application of standard accelerated proximal scheme to this setting. One drawback of applying accelerated proximal schemes to (32) is the evaluation of the proximal operator of an ℓ_1-norm over the linear constraint $C^\top \mathbb{1} = \mathbb{1}$. This requires additional computation cost of $\log(n)pn^2$. We fit (32) into our template (26) by defining $f(\cdot) := \|\cdot\|_1 + \delta_{\{\mathrm{diag}(\cdot)=0\}}(\cdot)$, $g(\cdot) := \delta_{\{\langle\cdot,\mathbb{1}\rangle=\mathbb{1}\}}(\cdot)$, and $h(\cdot) = \frac{\lambda}{2}\|X - X(\cdot)\|^2$. If we apply Algorithm 1 to solve this reformulation, then no extra computation cost is incurred as in accelerated proximal gradient methods.

We use a classic benchmark Extended Yale B dataset [31] to test the sparse subspace clustering problem (32). This dataset contains face pictures of 38 individuals taken under 64 different environmental conditions. As previous works, we use downsampled images of size 48×42 pixels which correspond to $p = 2016$. We ran experiments with ADMM, TFOCS, and our method ASGARD-DL. We note that our method includes tuning parameters similar to ADMM. We use $\beta_0 := \sqrt{\|M\|}$, where $M(C) = C^\top \mathbb{1}$. We randomly selected $m = 2, 3, 5$ clusters and ran 3 trials for each case. We have used the implementation of ADMM [27,26] and TFOCS [55] provided by the authors of [55, 27,26]. For fair comparison, since per iteration cost of our method is smaller, we ran ADMM and TFOCS for 500 iterations and our method 2000 iterations and saved their runtimes. Then, we determined the minimum of the runtimes and fetch the results for all the methods corresponding to the same time. We used objective value and clustering error as comparison measures as [55].

We can see from Table 3 that our method consistently outperforms other methods in terms of objective values, and has similar performance in terms of the clustering error. We present our algorithm as another candidate for solving the classical sparse subspace clustering problem with a lower per iteration cost than previous approaches ADMM and TFOCS and similar performance.

4.7 **Obtaining moderate and high accuracy solutions**

Our aim is to show that our ASGARD-DL, Algorithm 1, can reasonably achieve moderate ($\varepsilon = 10^{-4}$) and high ($\varepsilon = 10^{-6}$) accuracies for some challenging nonsmooth problems. This is not often the case in first-order primal-dual methods. We check the KKT conditions in (7) evaluated at the iterates of the algorithm. More precisely, since (7) is equivalent to $\mathrm{prox}_{\gamma f}(x^\star - \gamma A^\top y^\star) = 0$ and $\mathrm{prox}_{\beta g^*}(y^\star + \beta A x^\star) = 0$ for any $\gamma > 0$ and $\beta > 0$, we

Table 3 Comparison of 3 methods on the SSC-Lasso problem with $m = 2, 3, 5$ clusters and 3 independent trials of each.

Problem	ADMM	TFOCS	ASGARD-DL
$(n = 2)$-objective-trial 1	236.5653	226.4371	225.7578
$(n = 2)$-Clustering error-trial 1	0.0312	0.0391	0.0391
$(n = 2)$-objective-trial 2	200.3710	192.2985	191.5177
$(n = 2)$-Clustering error-trial 2	0.0234	0.0469	0.0469
$(n = 2)$-objective-trial 3	197.2510	188.7655	188.1555
$(n = 2)$-Clustering error-trial 3	0.0703	0.0938	0.0938
$(n = 3)$-objective-trial 1	329.9188	320.9690	319.5887
$(n = 3)$-Clustering error-trial 1	0.0156	0.0156	0.0312
$(n = 3)$-objective-trial 2	341.1980	330.6395	329.5704
$(n = 3)$-Clustering error-trial 2	0.0729	0.0677	0.0677
$(n = 3)$-objective-trial 3	398.8778	389.3963	388.0739
$(n = 3)$-Clustering error-trial 3	0.4375	0.3594	0.3646
$(n = 5)$-objective-trial 1	549.8250	530.0340	526.1905
$(n = 5)$-Clustering error-trial 1	0.1625	0.1156	0.0906
$(n = 5)$-objective-trial 2	482.8483	467.0535	461.6563
$(n = 5)$-Clustering error-trial 2	0.2188	0.1125	0.1562
$(n = 5)$-objective-trial 3	1029.5459	1017.7089	1025.6752
$(n = 5)$-Clustering error-trial 3	0.3156	0.3469	0.3156

compute the distance $\gamma_k\|\hat{x}^{k+1} - \hat{x}^k\|$ according to Step 8 in Algorithm 1 that presents an approximation of $\mathrm{prox}_{\gamma f}(x^\star - \gamma A^\top y^\star) = 0$. Similarly, we compute $\beta_s\|\tilde{y}_{k+1} - \dot{y}^s\|$ according to Step 8 in Algorithm 1 that presents an approximation of $\mathrm{prox}_{\beta g^*}(y^\star + \beta A x^\star) = 0$. In our experiments, we used the condition $\max\left\{\gamma_k\|\hat{x}^{k+1} - \hat{x}^k\|, \beta_s\|\tilde{y}_{k+1} - \dot{y}^s\|\right\} \le \varepsilon$ to terminate the algorithm. A similar condition was also used in ASGARD and CP. Our experiment was run on a MacBook Pro. Laptop with 2.7 GHz Intel Core i5 and 16GB memory.

Table 4 The computational time and total number of iterations of three algorithms for solving (27) to achieve 10^{-4} and 10^{-6} accuracy level on the KKT condition (7).

	Computational Time (seconds)			The total number of iterations		
Problem	ASGARD-R	ASGARD-DL	CP	ASGARD-R	ASGARD-DL	CP
KKT error $\le 10^{-4}$						
S5	44.052	27.576	8.437	144630	93475	29125
S6	55.277	57.820	21.938	160230	202740	78545
S19_10	52.692	51.464	37.392	171060	172485	126800
S20_9	41.750	36.175	20.129	159870	117125	68110
KKT error $\le 10^{-6}$						
S5	-	122.766	240.996	-	419005	830005
S6	-	108.458	-	-	380905	-
S19_10	-	104.169	-	-	353620	-
S20_9	-	89.566	287.387	-	304560	976325

Firstly, we tested three algorithms: ASGARD-R, our ASGARD-DL, and CP on 4 real datasets of the basis pursuit problem (27) as an instance of the constrained setting (2). We set the maximum number of iterations in all algorithms at 10^6. To be fair, we also carefully tuned the step-size σ of the CP for one problem and used that value for 4 datasets. Since ASGARD could not achieve 10^{-4} accuracy after 10^6 iterations, we did not report the results. The final results are reported in Table 4, where "-" shows that the algorithm could not reach the desired accuracy after 10^6 iterations. From Table 4, we observed that CP can reach 10^{-4} accuracy faster than ASGARD-DL. ASGARD-R can also reach 10^{-4} accuracy, but it is slower than ASGARD-DL. However, with a higher accuracy, i.e., 10^{-6}, ASGARD-DL can reach this accuracy in 4 datasets, while ASGARD-R fails in all cases and CP fails with 2/4 datasets. Moreover, the number of iterations in the two successful cases in CP is much higher than that of ASGARD-DL.

Next, we tested four algorithms: ASGARD, ASGARD-R, our ASGARD-DL, and CP on 10 real datasets of the support vector machine problem (28) as an instance of the composite form (1). We used the same setting as in the previous test, and the results are reported in Table 5.

Table 5 The computational time and total number of iterations of three algorithms for solving (28) to achieve 10^{-4} and 10^{-6} accuracy level on the KKT condition (7).

	Computational Time (seconds)				The total number of iterations			
Prob.	ASGARD	ASGARD-R	ASGARD-DL	CP	ASGARD	ASGARD-R	ASGARD-DL	CP
KKT error $\leq 10^{-4}$								
w1a	18.880	0.231	0.453	0.705	133440	1575	4335	6445
w2a	20.524	0.305	0.505	1.360	116095	1590	3480	9715
w3a	17.369	0.283	0.391	1.403	100320	1410	2440	10065
w4a	60.305	0.317	0.833	1.284	261780	1245	4145	6990
w5a	79.265	0.409	1.029	4.085	288290	1350	4060	17640
w6a	208.476	0.599	1.803	4.731	491465	1215	4430	12525
w7a	269.159	0.736	2.250	5.652	430840	1005	3635	8955
w8a	715.254	1.199	4.513	8.389	486135	690	3045	5470
rcv1	-	3.777	7.660	8.417	100000	210	1085	1355
covtype	1304.755	26.321	30.378	42.574	58015	1170	1380	1870
KKT error $\leq 10^{-6}$								
w1a	102.919	8.403	5.496	6.942	785915	65280	54915	69325
w2a	101.371	6.464	5.244	20.529	598780	36945	37290	149995
w3a	175.270	4.888	2.532	13.587	990150	26070	16380	93940
w4a	-	11.362	6.110	28.481	-	48345	30900	156100
w5a	-	14.867	10.130	31.317	-	51795	41005	136775
w6a	-	15.817	10.298	79.060	-	35715	26030	207400
w7a	-	27.216	13.700	187.070	-	41205	23035	246765
w8a	-	84.382	40.811	453.838	-	53730	27930	290575
rcv1	-	82.550	80.690	159.554	-	17910	18935	34185
covtype	-	433.162	1247.115	571.864	-	19170	56465	27255

In this test, ASGARD fails to reach 10^{-4} accuracy in one problem: rcv1, and 10^{-6} accuracy in the 7 last problems, while other three methods all achieve these accuracy levels. This is not surprise since ASGARD has $\mathcal{O}\left(\frac{1}{k}\right)$-convergence rate. For 10^{-4} accuracy, ASGARD-R seems to work well and slightly outperformed ASGARD-DL and CP. However, for higher accuracy, 10^{-6}, ASGARD-R becomes slower than ASGARD-DL. ASGARD-DL outperforms CP in both cases: 10^{-4} and 10^{-6} accuracies. Note that the KKT condition (7) tested in this experiment measures the maximum error of the primal and dual optimality conditions at the same time, while in our previous experiments, we only focused on the primal objective residual for the composite form (1), and both the primal objective residual and the primal feasibility for the constrained setting (2), but not on the dual optimality condition.

5 Further discussion and comparison with previous work

Theory and numerical methods for solving (1) and (2) are well-studied in the literature. Due to such a large proportion of solution methods, we only focus on some recent works that are the most related to our method developed in this paper. We briefly survey these results to highlight the similarities and differences with our work.

In [49], Nesterov proposed combining smoothing technique and accelerated gradient methods to obtain $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$-iteration complexity to obtain an ε-approximate solution to (1). However, this method requires ε to be predefined, and both primal and dual domains are bounded. In addition, the step-size of the underlying gradient-type scheme is proportional to ε, which is often small. This leads to a poor performance in early iterations. In [48], Nesterov introduced an excessive gap technique to develop new algorithms that allow the smoothness parameter to be adaptively updated. Nevertheless, these methods still require both primal and dual domains to be bounded, and one additional proximal operator for every two iterations.

In [13,14], A. Chambolle and T. Pock proposed a primal-dual algorithm to solve (1) that achieves $\mathcal{O}\left(\frac{1}{k}\right)$-convergence rate. This rate is guaranteed on a gap function and also requires both primal and dual domains to be bounded, which is unfortunately not applicable to (2). In addition, the guarantee of their methods relies on ergodic or weighted averaging sequences. Note that, in sparse and low-rank optimization and image processing, taking averaging sequence unfortunately destroys desired structures of approximate solutions. In addition, as also presented with numerical evidence, averaging sequences perform poorly in practice.

In [70], the authors proposed a homotopy algorithm called Homotopy Smoothing algorithm (HOPS) which also essentially relies on Nesterov's smoothing technique [49]. HOPS employs a similar strategy to ours in the sense of having a double loop structure. However, this method suffers from several drawbacks. First, it only applies to unconstrained problems as in (1), but not to (2) due to the unboundedness of the dual domain. Second, it requires knowing $\varepsilon_0 = P(x^0) - P(x^\star)$ to be able to set the initial smoothness parameter. Third, HOPS requires tuning the number of inner iterations and the rate at which the smoothness parameter is going to be reduced. The alternative of HOPS to alleviate this issue requires a bounded primal domain which further restricts the usage of their method.

For constrained problem (2), among different methods, augmented Lagrangian (ALM), alternating direction method of multipliers (ADMM), alternating minimization algorithms (AMA), and penalty methods are the most popular. Inexact augmented Lagrangian methods (iALM) [37,45,69] relies on a double loop structure similar to our method. However, termination rules for

Table 6 Summary of algorithms that require two proximal operators each iteration. Note that $z^k := \sum_{k=1}^{K} w^k x^k$ where K is the maximum number of iterations and w^k are the weights. For solving the unconstrained problem with ALM/ADMM methods, we split the problem. Practicality column refers to whether using the iterate in the convergence rate gives a fast practical performance or not.

Algorithm	g is Lipschitz	$g = \delta_{\{b\}}$	Type of rate	Set ϵ	Practicality
Nesterov [49]	$P(x^k) - P^\star \leq \mathcal{O}\left(\max\left(\epsilon, \frac{1}{\epsilon k^2}\right)\right)$	Not applicable	**Non-ergodic**	Yes	No
Chambolle-Pock [13]	$G(z^k) \leq \mathcal{O}\left(\frac{1}{k}\right)$	Convergence	Ergodic	**No**	No
Linearized ALM [68]	$P(z^k) - P^\star \leq \mathcal{O}\left(\frac{1}{k}\right)$	$\lvert f(z^k) - f^\star\rvert \leq \mathcal{O}\left(\frac{1}{k}\right)$ $\lVert Ax - b\rVert \leq \mathcal{O}\left(\frac{1}{k}\right)$	Ergodic	**No**	No
Inexact ALM [69]	$P(z^k) - P^\star \leq \mathcal{O}\left(\max\left(\epsilon_k, \beta_k\right)\right)$	$\lvert f(z^k) - f^\star\rvert \leq \mathcal{O}\left(\max\left(\epsilon_k + \beta_k\right)\right)$ $\lVert Ax - b\rVert \leq \mathcal{O}\left(\beta_k\right)$	**Non-ergodic**	Yes	No
Linearized ADMM [68]	$P(z_1^k, z_2^k) - P^\star = \mathcal{O}\left(\frac{1}{k}\right)$	$\lvert f_1(z_1^k) + f_2(z_2^k) - f_1^\star - f_2^\star\rvert \leq \mathcal{O}\left(\frac{1}{k}\right)$ $\lVert A_1 z_1^k + A_2 z_2^k - b\rVert \leq \mathcal{O}\left(\frac{1}{k}\right)$	Ergodic	**No**	No
ASGARD [60]	$P(x^k) - P^\star \leq \mathcal{O}\left(\frac{1}{k}\right)$	$\lvert f(x^k) - f^\star\rvert \leq \mathcal{O}\left(\frac{1}{k}\right)$ $\lVert Ax^k - b\rVert \leq \mathcal{O}\left(\frac{1}{k}\right)$	**Non-ergodic**	**No**	No
This paper (Algorithm 1)	$P(x^k) - P^\star \leq \mathcal{O}\left(\frac{1}{k}\right)$	$\lvert f(x^k) - f^\star\rvert \leq \mathcal{O}\left(\frac{1}{k}\right)$ $\lVert Ax^k - b\rVert \leq \mathcal{O}\left(\frac{1}{k}\right)$	**Non-ergodic**	**No**	**Yes**

these methods require the desired accuracy ε to be set a priori. In addition, in practice, it is not easy to check when the inner problem is solved to an ε_k-accuracy in the k-th iteration. Such an estimate is often derived from the worst-case complexity bound of the underlying solution method, and therefore, the corresponding algorithm is not efficient in practice.

While ADMM works really well and is widely used in practice, AMA is rarely used and requires additional conditions to converge. The best-known convergence rate of ADMM and its variants such as linearized ADMM and preconditioned ADMM is $\mathcal{O}\left(\frac{1}{k}\right)$ under standard assumptions [33,41,42,52, 68]. Moreover, this rate is given in an ergodic sense, and examples show that such a rate is optimal. See [10] for more information about the behavior of ADMM. In practice, however, the ergodic rate is rather pessimistic, which is much slower than the last iterate sequence (see Subsection 4.1 as an example). So far, we are not aware of any work showing an $\mathcal{O}\left(\frac{1}{k}\right)$-rate of the standard ADMM or its linearized and preconditioned ADMM in the last iterate. A recent work [39] combined preconditioned/linearized ADMM and Nesterov's accelerated schemes to achieve an $\mathcal{O}\left(\frac{1}{k}\right)$-non-ergodic convergence rate.

Penalty methods use a quadratic penalty term to move the constraints to the objective and solve the subproblems by changing the penalty parameter [36,43]. Similar to iALM, these methods also do not have clear implementable termination rules for the inner loop. In addition, they do not involve dual variables. Therefore, they are often less competitive with primal-dual methods. A recent work [59] proposed a new alternating quadratic penalty algorithm to solve (2) that has the same $\mathcal{O}\left(\frac{1}{k}\right)$-non-ergodic convergence rate as in this paper. Nevertheless, this method is completely different from this paper and does not have an update on the dual center.

Compared to our previous work [60], ASGARD, our new algorithm shares some similarities but also has several differences. First, it has inner and outer loops but the guarantee is on the overall iterations. Second, it works with any Bregman divergence induced by a general prox-function when solving (1), while ASGARD only works with the Bregman distances induced by a strongly convex and Lipschitz gradient prox-function. This excludes some important Bregman divergences such as the Kullback-Leibler (KL) divergence. Third, our algorithm allows us to use different norms while computing proximal operators, compared to ASGARD which works with only Euclidean norms. Fourth, it automatically restarts both the primal and dual variables as well as the parameters. It also has a rigorous convergence guarantee, while the practical restarting variant of ASGARD does not have convergence guarantee.

We developed a novel analysis for our double loop structured smoothing algorithm which allowed us to derive flexible rules for parameters in both unconstrained and constrained problems, in contrast to [70]. Our analysis gives insights on the heuristic restarting strategies in [60] as well as on the number of inner iterations in the algorithm. It also gives explicit number of iterations for the inner subproblems and does not require to predefine the horizon as opposed to iALM. Table 7 summarizes the key differences between different methods we have discussed in this paper.

Table 7 A comparison with previous work (β is a smoothness parameter defined in (8)).

ADMM/iALM	Penalty / HOPS / ASGARD	This work
Constant or adaptive β.	Analytically drive β to 0.	Analytically drive β to 0.
Update the dual center.	Do not move the dual center.	Update the dual center.
Theory is driven by the convergence in the dual.	Do not analyze the convergence of the dual.	Only analyze the stability of the primal-dual sequence.
Inner problems are solved inexactly.	Inner problems are solved inexactly.	Only ensure stability for the number of inner iterations and smoothness parameter.

Acknowledgements

Q.Tran-Dinh was supported in part by the National Science Foundation(NSF), Grant No. DMS-1619884, and Nafosted (no. 101.01-2017.315), Vietnam. A. Alacaoglu and V. Cevher were supported in part by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant Agreement No. 725594 - time-data and Swiss National Science Foundation (SNSF) under Grant Number 200021-178865/1.

6 Appendix: The proof of technical results

This appendix provides the missing proof of the results in the main text.

6.1 The proof of Example 3.1.

In this example, we have $b_{\mathcal{Y}}(y, \dot{y}) = \frac{1}{2}\|y - \dot{y}\|^2$. First, from the definition (21) of $g_\beta(Ax; \dot{y})$, by using the definition of $s_{\mathcal{K}}$, we write

$$\begin{aligned} g_\beta(Ax;\dot{y}) &= \min_{u\in\mathcal{K}}\max_{y\in\mathbb{R}^n}\left\{\langle Ax - b - u, y\rangle - \beta b_{\mathcal{Y}}(y,\dot{y})\right\} \\ &= \min_{u\in\mathcal{K}}\max_{y\in\mathbb{R}^n}\left\{\langle Ax - b - u, y\rangle - \tfrac{\beta}{2}\|y-\dot{y}\|^2\right\}. \end{aligned}$$

The optimality condition of the max problem on the right hand side of the previous inequality is $Ax-b-u-\beta(y-\dot{y}) = 0$, which implies $y = \dot{y} + \frac{1}{\beta}(Ax-b-u)$. In this case, $\langle Ax-b-u, y\rangle - \frac{\beta}{2}\|y-\dot{y}\|^2 = \frac{1}{2\beta}\|Ax-b-u\|^2 + \langle \dot{y}, Ax-b-u\rangle = \frac{1}{2\beta}\|Ax - b - u + \beta\dot{y}\|^2 - \frac{\beta}{2}\|\dot{y}\|^2$. Hence, we obtain

$$\begin{aligned} g_\beta(Ax;\dot{y}) &= \min_{u\in\mathcal{K}}\left\{\tfrac{1}{2\beta}\|u - (Ax - b + \beta\dot{y})\|^2\right\} - \tfrac{\beta}{2}\|\dot{y}\|^2 \\ &= \tfrac{1}{2\beta}\mathrm{dist}_{\mathcal{K}}\left(Ax - b + \beta\dot{y}\right)^2 - \tfrac{\beta}{2}\|\dot{y}\|^2, \end{aligned}$$

which is (22). In addition, this implies $u = \mathrm{proj}_{\mathcal{K}}\left(Ax - b + \beta\dot{y}\right)$. Hence, we obtain $y^*_\beta(Ax;\dot{y}) = \dot{y} + \frac{1}{\beta}(Ax-b-u) = \dot{y} + \frac{1}{\beta}\left(Ax - b - \mathrm{proj}_{\mathcal{K}}\left(Ax - b + \beta\dot{y}\right)\right)$, which is exactly (23).

If $\mathcal{K}$ is a cone, then using Moreau's decomposition [2, Theorem 6.30], we can show that

$$Ax - b + \beta\dot{y} - \mathrm{proj}_{\mathcal{K}}\left(Ax - b + \beta\dot{y}\right) = \mathrm{proj}_{\mathcal{K}^{\circ}}\left(Ax - b + \beta\dot{y}\right),$$

where K° is the polar set of $\mathcal{K}$. Since $\mathcal{K}$ is a cone, $\mathcal{K}^{\circ} = -\mathcal{K}^{*}$, where $\mathcal{K}^{*}$ is the dual cone of $\mathcal{K}$. Hence, we have $y^{*}_{\beta}(Ax;\dot{y}) = \mathrm{proj}_{-\mathcal{K}^{*}}\left(\dot{y} + \frac{1}{\beta}(Ax - b)\right)$. □

6.2 **The proof of Lemma 3.1: Optimality bounds**.

The proof of Lemma 3.1 is based on the following lemma.

Lemma 6.1. *Suppose that $b_{\mathcal{Y}}$ has an $L_{b_{\mathcal{Y}}}$-Lipschitz gradient such that $L_{b_{\mathcal{Y}}} \in (0, +\infty)$. Let $(x^{\star}, y^{\star})$ be a saddle point of the Lagrange function of* (1) *and*

$$S_{\beta}(\bar{x},\dot{y}) := \max_{y\in\mathbb{R}^n}\Big\{f(\bar{x}) + \langle y, Ax\rangle - g^{*}(y) - \beta b_{\mathcal{Y}}(y,\dot{y}) - f(x^{\star}) - g(Ax^{\star})\Big\}.$$

Let $\beta_b := \beta L_{b_{\mathcal{Y}}}$ and $\bar{y}^{*}_{\beta} := y^{*}_{\beta}(A\bar{x},\dot{y})$. Then, we have

$$\begin{cases} f(\bar{x}) + g\big(A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\big) - P^{\star} \geq -\beta\left\|y^{\star}\right\|_{\mathcal{Y}}\|\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\|_{\mathcal{Y},*} \\ f(\bar{x}) + g\big(A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\big) - P^{\star} \leq S_{\beta}(\bar{x},\dot{y}) + \beta\left\|\dot{y}\right\|_{\mathcal{Y}}\|\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\|_{\mathcal{Y},*} \\ \mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}, \mathrm{dom}(g)\right) \leq \beta\|\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\|_{\mathcal{Y},*} \\ \qquad\qquad\qquad \leq \beta_b\Big[\left\|y^{\star} - \dot{y}\right\|_{\mathcal{Y}} + \Big(\left\|y^{\star} - \dot{y}\right\|^2_{\mathcal{Y}} + \frac{2}{\beta_b}S_{\beta}(\bar{x},\dot{y})\Big)^{1/2}\Big] \end{cases}$$

Proof. First, from the KKT condition (7) we have $y^{\star} \in \partial g(Ax^{\star})$. Using this expression, we can show that

$$\begin{aligned} &f(\bar{x}) + g\big(A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\big) - f(x^{\star}) - g(Ax^{\star}) \\ &\quad\geq \langle -A^{\top}y^{\star}, \bar{x} - x^{\star}\rangle + \langle y^{\star}, A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y}) - Ax^{\star}\rangle \\ &\quad= -\beta\langle y^{\star}, \nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\rangle \\ &\quad\geq -\beta\|y^{\star}\|_{\mathcal{Y}}\|\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\|_{\mathcal{Y},*}, \end{aligned} \tag{33}$$

which proves the first estimate in Lemma 6.1.

Next, we consider optimality condition of

$$\bar{y}^{*}_{\beta} := \mathrm{argmin}_{y}\Big\{\langle A\bar{x}, y\rangle - g^{*}(y) - \beta b_{\mathcal{Y}}(y,\dot{y})\Big\}$$

as follows:

$$0 \in -A\bar{x} + \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y}) + \partial g^{*}(\bar{y}^{*}_{\beta}).$$

This implies that $\bar{y}^{*}_{\beta} \in \partial g(A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y}))$, and in particular we have $A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y}) \in \mathrm{dom}(g)$. Hence, we obtain

$$\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}, \mathrm{dom}(g)\right) \leq \beta\|\nabla_y b_{\mathcal{Y}}(\bar{y}^{*}_{\beta},\dot{y})\|_{\mathcal{Y},*}. \tag{34}$$

Now, using the equality of the Fenchel-Young inequality and the definition of $g_\beta(A\bar{x},\dot{y})$, one can show that

$$\begin{aligned}
f(\bar{x}) &+ g\big(A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\big) - f(x^\star) - g(Ax^\star)\\
&= S_\beta(\bar{x},\dot{y}) + g\big(A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\big) - g_\beta(A\bar{x},\dot{y})\\
&= S_\beta(\bar{x},\dot{y}) + \langle \bar{y}^*_\beta, A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\rangle - g^*(\bar{y}^*_\beta)\\
&\qquad - \langle A\bar{x},\bar{y}^*_\beta\rangle + g^*(\bar{y}^*_\beta) + \beta b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\\
&= S_\beta(\bar{x},\dot{y}) - \beta\langle \bar{y}^*_\beta, \nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\rangle + \beta b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\\
&\le S_\beta(\bar{x},\dot{y}) - \beta\langle \dot{y}, \nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\rangle - \frac{\beta}{2L_{b_{\mathcal{Y}}}}\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|^2_{\mathcal{Y},*}, \qquad (35)
\end{aligned}$$

where we used the following bound in the last inequality of (35):

$$0 = b_{\mathcal{Y}}(\dot{y},\dot{y}) \ge b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y}) + \langle \nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y}), \dot{y} - \bar{y}^*_\beta\rangle + \frac{1}{2L_{b_{\mathcal{Y}}}}\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|^2_{\mathcal{Y},*}.$$

The second inequality of Lemma 6.1 follows directly from (35) using the Cauchy-Schwartz inequality.

Finally, combining (35) and (33), we obtain

$$\begin{aligned}
0 &\le S_\beta(\bar{x},\dot{y}) + \beta\langle y^\star - \dot{y}, \nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\rangle - \frac{\beta}{2L_{b_{\mathcal{Y}}}}\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|^2_{\mathcal{Y},*}\\
&\le S_\beta(\bar{x},\dot{y}) + \beta\,\|y^\star - \dot{y}\|_{\mathcal{Y}}\,\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|_{\mathcal{Y},*} - \frac{\beta}{2L_{b_{\mathcal{Y}}}}\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|^2_{\mathcal{Y},*}.
\end{aligned}$$

Solving the quadratic inequation $0 \le S_\beta(\bar{x},\dot{y}) + \beta\|y^\star - \dot{y}\|_{\mathcal{Y}} t - \frac{\beta}{2L_{b_{\mathcal{Y}}}}t^2$ in $t := \|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|_{\mathcal{Y},*} \ge 0$, we deduce that

$$\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|_{\mathcal{Y},*} \le L_{b_{\mathcal{Y}}}\Big[\,\|y^\star - \dot{y}\|_{\mathcal{Y}} + \Big(\,\|y^\star - \dot{y}\|^2_{\mathcal{Y}} + \tfrac{2}{\beta L_{b_{\mathcal{Y}}}}S_\beta(\bar{x},\dot{y})\Big)^{1/2}\Big].$$

Substituting this estimate into (34), we obtain the last estimate of Lemma 6.1. □

Proof of Lemma 3.1. The third inequality of (17) directly follows from Lemma 6.1. Note that it is void if $L_{b_{\mathcal{Y}}} = +\infty$. The first inequality of (17) is proved with the same arguments as the first inequality of Lemma 6.1. For the second inequality, we have two cases:

1. If $L_{b_{\mathcal{Y}}} < +\infty$, then we can use Lemma 6.1 to get

$$\begin{aligned}
f(\bar{x}) + g(\bar{z}) - P^\star &\le f(\bar{x}) + g\big(A\bar{x} - \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\big) - P^\star\\
&\qquad + \widehat{M}_g\|\bar{z} - A\bar{x} + \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|_{\mathcal{Y},*}\\
&\le S_\beta(\bar{x},\dot{y}) + \beta(2\widehat{M}_g + \|\dot{y}\|_{\mathcal{Y}})\|\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta,\dot{y})\|_{\mathcal{Y},*},
\end{aligned}$$

where we used the triangle inequality and definition of projection to get

$$\begin{aligned}\|\bar{z} - A\bar{x} &+ \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta, \dot{y})\|_{\mathcal{Y},*} \\ &\leq \|\bar{z} - A\bar{x}\| + \|A\bar{x} - A\bar{x} + \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta, \dot{y})\|_{\mathcal{Y},*} \\ &\leq 2\|A\bar{x} - A\bar{x} + \beta\nabla_y b_{\mathcal{Y}}(\bar{y}^*_\beta, \dot{y})\|_{\mathcal{Y},*}.\end{aligned}$$

2. If $D_{\mathcal{Y}} < +\infty$, then g is Lipschitz continuous and $\mathrm{dom}(g) = \mathbb{R}^n$. Hence,

$$\begin{aligned}f(\bar{x}) + g(\bar{z}) - P^\star &= f(\bar{x}) + g(A\bar{x}) - P^\star \\ &= S_\beta(\bar{x}, \dot{y}) + g(A\bar{x}) - h_\beta(A\bar{x}, \dot{y}) \leq S_\beta(\bar{x}, \dot{y}) + \beta D_{\mathcal{Y}}.\end{aligned}$$

Combining both bounds we get the second inequality of (17). □

7 The proof of Theorem 3.1: Convergence of Algorithm 1

We present the full proof of Theorem 3.1 in this section.

One-stage inequality. With the same argument as in [61], we can prove the following estimate at the k-th iteration at the stage s of the outer loop, i.e., $K_s \leq k < K_{s+1} := K_s + m_s$, of Algorithm 1:

$$\begin{aligned}S_{\beta_s}(\bar{x}^{k+1}; \dot{y}^s) + \tfrac{\tau_k^2\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{k+1}) &\leq (1 - \tau_k) S_{\beta_s}(\bar{x}^k; \dot{y}^s) \\ &+ \tfrac{\tau_k^2\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^k),\end{aligned} \tag{36}$$

where $S_\beta(\bar{x}; \dot{y}) := P_\beta(\bar{x}; \dot{y}) - P(x^\star)$. Note that this estimate remains true if we use APG with Option 2. In order to use FISTA as an inner loop solver, instead of APG, we need a quadratic prox-function for the primal space. Except from this technical detail, the inequality remains the same.

Next, by strong convexity of $b_{\mathcal{Y}}(\cdot, \dot{y})$, the optimality condition of g_β-subproblem and convexity of $g^*(\cdot)$, we have

$$\begin{aligned}g_\beta(A\bar{x}; \dot{y}) &= \max_{y\in\mathbb{R}^n}\left\{\langle A\bar{x}, y\rangle - g^*(y) - \beta b_{\mathcal{Y}}(y, \dot{y})\right\} \\ &\geq \langle A\bar{x}, y^\star\rangle - g^*(y^\star) - \beta b_{\mathcal{Y}}(y^\star, \dot{y}) + \beta b_{\mathcal{Y}}(y^\star, y^*_\beta(A\bar{x}; \dot{y})).\end{aligned} \tag{37}$$

Now, from the optimality condition of (1), we have $-A^\top y^\star \in \partial f(x^\star)$. Using this inclusion and convexity of f, we can derive

$$f(\bar{x}) \geq f(x^\star) + \langle -A^\top y^\star, \bar{x} - x^\star\rangle. \tag{38}$$

Combining (37) and (38), we get

$$\begin{aligned}S_\beta(\bar{x}; \dot{y}) = P_\beta(\bar{x}; \dot{y}) - P(x^\star) &= f(\bar{x}) + g_\beta(A\bar{x}; \dot{y}) - (f(x^\star) + g(Ax^\star)) \\ &\geq -\beta b_{\mathcal{Y}}(y^\star, \dot{y}) + \beta b_{\mathcal{Y}}(y^\star, y^*_\beta(A\bar{x}; \dot{y})).\end{aligned} \tag{39}$$

From (36), for $K_s \leq k \leq K_s + m_s - 1$ we obtain

$$\tfrac{1}{\tau_k^2} S_{\beta_s}(\bar{x}^{k+1}; \dot{y}^s) + \tfrac{\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{k+1}) \leq \tfrac{1-\tau_k}{\tau_k^2} S_{\beta_s}(\bar{x}^k; \dot{y}^s) + \tfrac{\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^k). \tag{40}$$

By (39), we have $\beta b_{\mathcal{Y}}(y^\star, \dot{y}) + S_\beta(\bar{x}; \dot{y}) \geq 0$. Let us define $D_k^s := S_{\beta_s}(\bar{x}^k; \dot{y}^s) + \beta_s b_{\mathcal{Y}}(y^\star, \dot{y}^s)$. By adding $\frac{1}{\tau_k^2}\beta_s b_{\mathcal{Y}}(y^\star, \dot{y}^s)$ to both sides of (40) and using the definition of D_k^s, we obtain

$$\tfrac{1}{\tau_k^2} D_{k+1}^s + \tfrac{\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{k+1}) \leq \tfrac{(1-\tau_k)}{\tau_k^2} D_k^s + \tfrac{\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^k) + \tfrac{\beta_s}{\tau_k} b_{\mathcal{Y}}(y^\star, \dot{y}^s). \quad (41)$$

Let us choose $\tau_k = \frac{2}{k-K_s+2}$. Then, it is clear that $\tau_{K_s} = 1$. Moreover, $\frac{1-\tau_k}{\tau_k^2} = \frac{(k-K_s+2)(k-K_s)}{4} \leq \frac{(k-K_s+1)^2}{4} = \frac{1}{\tau_{k-1}^2}$. In this case, we can overestimate (41) as

$$\tfrac{1}{\tau_k^2} D_{k+1}^s + \tfrac{\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{k+1}) \leq \tfrac{1}{\tau_{k-1}^2} D_k^s + \tfrac{\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^k) + \tfrac{\beta_s}{\tau_k} b_{\mathcal{Y}}(y^\star, \dot{y}^s). \quad (42)$$

Taking a telescope from $k = K_s + 1$ to $k = K_{s+1} - 1 = K_s + m_s - 1$ of (42) and reuse (41) for $k = K_s$, we obtain

$$\begin{aligned} D_{K_{s+1}}^s &+ \tfrac{\tau_{K_{s+1}-1}^2\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{K_{s+1}}) \leq \tfrac{\tau_{K_{s+1}-1}^2(1-\tau_{K_s})}{\tau_{K_s}^2} D_{K_s}^s \\ &+ \tfrac{\tau_{K_{s+1}-1}^2\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{K_s}) + \beta_s \tau_{K_{s+1}-1}^2 b_{\mathcal{Y}}(y^\star, \dot{y}^s) \textstyle\sum_{j=K_s}^{K_s+m_s-1} \frac{1}{\tau_j} \\ &\overset{(i)}{\leq} \tfrac{\tau_{K_{s+1}-1}^2\|A\|^2}{\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{K_s}) + \beta_s \tau_{K_{s+1}-1}^2 b_{\mathcal{Y}}(y^\star, \dot{y}^s) \textstyle\sum_{j=K_s}^{K_s+m_s-1} \frac{1}{\tau_j}, \end{aligned}$$

where (i) holds since $\tau_{K_s} = 1$. Since $\tau_k = \frac{2}{k-K_s+2}$, we have $\tau_{K_{s+1}-1} = \frac{2}{m_s+1}$ and $\sum_{j=K_s}^{K_s+m_s-1} \frac{1}{\tau_j} = \frac{m_s(m_s+3)}{4}$. Using this relation, the last estimate leads to

$$D_{K_{s+1}}^s + \tfrac{4\|A\|^2}{(m_s+1)^2\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{K_{s+1}}) \leq \tfrac{4\|A\|^2}{(m_s+1)^2\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{K_s}) + \tfrac{\beta_s m_s(m_s+3)}{(m_s+1)^2} b_{\mathcal{Y}}(y^\star, \dot{y}^s).$$

Since $D_{K_{s+1}}^s = S_{\beta_s}(\bar{x}^{K_{s+1}}; \dot{y}^s) + \beta_s b_{\mathcal{Y}}(y^\star, \dot{y}^s) \geq \beta_s b_{\mathcal{Y}}(y^\star, y_{\beta_s}^*(A\bar{x}^{K_{s+1}}; \dot{y}^s)) = \beta_s b_{\mathcal{Y}}(y^\star, \dot{y}^{s+1})$, the last estimate leads to

$$\begin{aligned} \beta_s b_{\mathcal{Y}}(y^\star, \dot{y}^{s+1}) + \tfrac{4\|A\|^2}{(m_s+1)^2\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{K_{s+1}}) &\leq \tfrac{4\|A\|^2}{(m_s+1)^2\beta_s} d_{\mathcal{X}}(x^\star, \hat{x}^{K_s}) \\ &+ \tfrac{\beta_s(m_s+3)}{(m_s+1)^2} b_{\mathcal{Y}}(y^\star, \dot{y}^s). \end{aligned} \quad (43)$$

The s-stage inequality. Using the update rule $m_{s+1} \leftarrow \lfloor \omega(m_s+1) + 1 \rfloor - 1$ at Step 16 of Algorithm 1, we have

$$\omega(m_s+1) \leq m_{s+1} + 1 \leq \omega(m_s+1) + 1. \quad (44)$$

Hence, recursively applying this inequality, one can get

$$m_0\omega^s \leq (m_0+1)\omega^s - 1 \leq m_s \leq \big(m_0 + \tfrac{\omega}{\omega-1}\big)\omega^s - \tfrac{\omega}{\omega-1} \leq \kappa_0\omega^s, \quad (45)$$

where $\kappa_0 := m_0 + \frac{\omega}{\omega-1} > 0$.

From $\beta_{s+1} := \frac{\beta_s(m_{s+1}+1)}{\omega\sqrt{m_{s+1}(m_{s+1}+3)}}$ at Step 17 of Algorithm 1, we get $\beta_{s+1} \leq \frac{\beta_s}{\omega} \leq \frac{\beta_0}{\omega^{s+1}}$. Next, we need to lower bound β_s. Clearly, for $m_s \geq 1$, we have

$$\tfrac{m_{s+1}+1}{\sqrt{m_{s+1}(m_{s+1}+3)}} \geq 1 - \tfrac{1}{m_{s+1}} \geq 0.$$

In this case, we can estimate $\beta_{s+1} = \frac{\beta_s(m_{s+1}+1)}{\omega\sqrt{m_{s+1}(m_{s+1}+3)}} \geq \frac{\beta_s}{\omega}\left(1 - \frac{1}{m_{s+1}}\right) = \frac{\beta_s}{\omega} - \frac{\beta_s}{m_{s+1}\omega}$. Substituting (45) on m_{s+1} and β_s into this inequality, we obtain

$$\beta_{s+1} \geq \tfrac{\beta_s}{\omega} - \tfrac{c_0}{\omega^{2s+1}}, \quad \text{where} \quad c_0 := \tfrac{\beta_0}{\omega m_0}.$$

This condition leads to $\omega\beta_{s+1} + \frac{c_0}{\omega^{2s}} \geq \beta_s$. By induction, we can show that $\omega^s\beta_s + c_0\sum_{j=0}^{s-1}\frac{1}{\omega^j} \geq \beta_0$, which leads to

$$\beta_s \geq \tfrac{1}{\omega^s}\left(\beta_0 - \tfrac{c_0\omega(\omega^s-1)}{(\omega-1)\omega^s}\right) \geq \beta_0\left(1 - \tfrac{1}{m_0(\omega-1)}\right)\tfrac{1}{\omega^s}. \tag{46}$$

Here, we use the fact that

$$\rho_0 := \beta_0 - \frac{c_0\omega(\omega^s-1)}{(\omega-1)\omega^s} \geq \beta_0 - \frac{c_0\omega}{\omega-1} = \beta_0\left(1 - \frac{1}{m_0(\omega-1)}\right) > 0$$

since $m_0 > \frac{1}{\omega-1}$. This condition gives us a lower bound on β_s.

Now, from the update rule of β_{s+1} again, we have $\frac{\omega^2\beta_{s+1}^2 m_{s+1}(m_{s+1}+3)}{(m_{s+1}+1)^2} = \beta_s^2$ and $\frac{\omega^2}{(m_{s+1}+1)^2} \leq \frac{1}{(m_s+1)^2}$ as in (44). Plugging these estimates into (43), we obtain

$$\begin{aligned}\tfrac{4\|A\|^2}{(m_{s+1}+1)^2}d_{\mathcal{X}}(x^\star, \hat{x}^{K_{s+1}}) + \tfrac{\beta_{s+1}^2 m_{s+1}(m_{s+1}+3)}{(m_{s+1}+1)^2}b_{\mathcal{Y}}(y^\star, \dot{y}^{s+1}) \leq \\ \tfrac{1}{\omega^2}\Big[\tfrac{4\|A\|^2}{(m_s+1)^2}d_{\mathcal{X}}(x^\star, \hat{x}^{K_s}) + \tfrac{\beta_s^2 m_s(m_s+3)}{(m_s+1)^2}b_{\mathcal{Y}}(y^\star, \dot{y}^s)\Big].\end{aligned} \tag{47}$$

By induction and using that $\hat{x}^0 = \bar{x}^0$, we obtain

$$\begin{aligned}\tfrac{4\|A\|^2}{(m_s+1)^2}d_{\mathcal{X}}(x^\star, \hat{x}^{K_s}) + \tfrac{\beta_s^2 m_s(m_s+3)}{(m_s+1)^2}b_{\mathcal{Y}}(y^\star, \dot{y}^s) \leq \tfrac{1}{\omega^{2s}}\Big[\tfrac{4\|A\|^2}{(m_0+1)^2}d_{\mathcal{X}}(x^\star, \bar{x}^0) \\ + \tfrac{\beta_0^2 m_0(m_0+3)}{(m_0+1)^2}b_{\mathcal{Y}}(y^\star, \dot{y}^0)\Big].\end{aligned} \tag{48}$$

Since $m_s(m_s+3) \geq m_s - 1$, combining (48) and (43), we obtain

$$\begin{aligned}S_{\beta_s}(\bar{x}^{K_{s+1}}; \dot{y}^s) &\leq \tfrac{1}{\beta_s\omega^{2s}}\left[\tfrac{4\|A\|^2}{(m_0+1)^2}d_{\mathcal{X}}(x^\star, \bar{x}^0) + \tfrac{\beta_0^2 m_0(m_0+3)}{(m_0+1)^2}b_{\mathcal{Y}}(y^\star, \dot{y}^0)\right] \\ &\leq \tfrac{R_0^2}{\beta_s\omega^{2s}},\end{aligned} \tag{49}$$

where $R_0 := \left[\frac{4\|A\|^2}{(m_0+1)^2}d_{\mathcal{X}}(x^\star, \bar{x}^0) + \frac{\beta_0^2 m_0(m_0+3)}{(m_0+1)^2}b_{\mathcal{Y}}(y^\star, \dot{y}^0)\right]^{1/2}$.

Using $m_s \leq \kappa_0\omega^s$ in (45) to estimate the total number of iterations K_{s+1} of Algorithm 1 as

$$K_{s+1} = \sum_{i=0}^{s} m_i \leq \kappa_0\sum_{i=0}^{s}\omega^i = \kappa_0\left(\tfrac{\omega^{s+1}-1}{\omega-1}\right).$$

This condition leads to $\omega^s \geq \frac{(\omega-1)K_{s+1}+\kappa_0}{\omega\kappa_0}$. We use this to bound ω^s via the number of iterations K_{s+1}, and denote $\rho_0 := \beta_0\big(1 - \frac{1}{m_0(\omega-1)}\big) > 0$.

Using (45) and (46) of β_s and m_s into (49), we obtain

$$S_{\beta_s}(\bar{x}^{K_{s+1}}; \dot{y}^s) \leq \tfrac{R_0^2}{\rho_0 \omega^s} \leq \tfrac{\omega\kappa_0 R_0^2}{\rho_0[(\omega-1)K_{s+1}+\kappa_0]}. \tag{50}$$

Our next step is using (48) to bound $\|\dot{y}^s - y^\star\|_{\mathcal{Y}}$. Clearly, $\frac{\beta_s^2 m_s(m_s+3)}{(m_s+1)^2} = \frac{\beta_{s-1}^2}{\omega^2} \geq \frac{\rho_0^2}{\omega^{2s}}$ by (46). Using (48), and strong convexity of $b_{\mathcal{Y}}$ with respect to the given norm, we can show that

$$\tfrac{1}{2}\|\dot{y}^s - y^\star\|_{\mathcal{Y}}^2 \leq b_{\mathcal{Y}}(y^\star, \dot{y}^s) \leq \tfrac{R_0^2}{\rho_0^2}. \tag{51}$$

The first estimate of (18)**.** Let us denote $\bar{z}^{K_s}$ the projection of $A\bar{x}^{K_s}$ onto $\mathrm{dom}(g)$. Using Lemma 3.1, and we write

$$f(\bar{x}^{K_s}) + g(\bar{z}^{K_s}) - P^\star \geq -\|y^\star\|_{\mathcal{Y}}\ \mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s}, \mathrm{dom}(g)\right),$$

which is the first estimate of (18).

The third estimate of (18)**.** We define $\beta_{b,s} := \beta_s L_{b_{\mathcal{Y}}}$. Using Lemma 3.1, we have

$$\begin{aligned}\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_{s+1}}, \mathrm{dom}(g)\right) &\leq \beta_s \|\nabla_y b_{\mathcal{Y}}(y^*_\beta(A\bar{x}^{K_{s+1}}, \dot{y}^s), \dot{y}^s)\|_{\mathcal{Y},*} \\ &\leq \beta_{b,s}\Big[\|y^\star - \dot{y}^s\|_{\mathcal{Y}} + \Big(\|y^\star - \dot{y}^s\|_{\mathcal{Y}}^2 + \tfrac{2}{\beta_{b,s}} S_{\beta_s}(\bar{x}^{K_{s+1}}, \dot{y}^s)\Big)^{1/2}\Big].\end{aligned}$$

We note that, by using (46) and (49), we can bound $\frac{2S_{\beta_s}(\bar{x}^{K_{s+1}}; \dot{y}^s)}{\beta_s} \leq \frac{2R_0^2}{\rho_0^2}$. Using this upper bound and (51) we obtain

$$\begin{aligned}\mathrm{dist}_{\mathcal{Y},*}\left(A\bar{x}^{K_s}, \mathrm{dom}(g)\right) &\leq \beta_{b,s-1}\Big(\tfrac{\sqrt{2}R_0}{\rho_0} + \big[(2 + \tfrac{2}{L_{b_{\mathcal{Y}}}})\tfrac{R_0^2}{\rho_0^2}\big]^{1/2}\Big) \\ &\leq \tfrac{(\sqrt{2}+2)\beta_0 L_{b_{\mathcal{Y}}}\omega\kappa_0 R_0}{\rho_0[(\omega-1)K_s+\kappa_0]},\end{aligned}$$

which is the third bound of (18).

The second estimate of (18)**.** Using Lemma 3.1, we have

$$\begin{aligned}f(\bar{x}^{K_{s+1}}) + g(\bar{z}^{K_{s+1}}) - P^\star &\leq S_{\beta_s}(\bar{x}^{K_{s+1}}, \dot{y}^s) \\ &\quad + \beta_s \min\Big\{D_{\mathcal{Y}}, (2\widehat{M}_g + \|\dot{y}^s\|_{\mathcal{Y}})\|\nabla_y b_{\mathcal{Y}}(\dot{y}^{s+1}, \dot{y}^s)\|_{\mathcal{Y},*}\Big\} \\ &\leq \tfrac{\omega\kappa_0 R_0^2}{\rho_0[(\omega-1)K_{s+1}+\kappa_0]} \\ &\quad + \min\Big\{D_{\mathcal{Y}}, \tfrac{(\sqrt{2}+2)L_{b_{\mathcal{Y}}}R_0}{\rho_0}\big[2\widehat{M}_g + \|y^\star\|_{\mathcal{Y}} + \tfrac{\sqrt{2}R_0}{\rho_0}\big]\Big\}\tfrac{\beta_0\omega\kappa_0}{(\omega-1)K_{s+1}+\kappa_0},\end{aligned}$$

which implies the second estimate of (18) by substituting $s+1$ by s. □

References

1. S. Arora, M. Khodak, N. Saunshi, and K. Vodrahalli. A Compressed Sensing View of Unsupervised Text Embeddings, Bag-of-n-Grams, and LSTMs. In *International Conference on Learning Representations*, 2018.
2. H. H. Bauschke and P. Combettes. *Convex analysis and monotone operators theory in Hilbert spaces.* Springer-Verlag, 2nd edition, 2017.
3. A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009.
4. A. Beck and M. Teboulle. Smoothing and first order methods: A unified framework. *SIAM J. Optim.*, 22(2):557–580, 2012.
5. Dimitri P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods.* Athena Scientific, 1996.
6. Allan Borodin, Ran El-Yaniv, and Vincent Gogan. Can we learn to beat the best stock. *J. Artif. Intell. Res.(JAIR)*, 21:579–594, 2004.
7. Jonathan M Borwein, Jon D Vanderwerff, et al. *Convex functions: constructions, characterizations and counterexamples*, volume 109. Cambridge University Press Cambridge, 2010.
8. Radu Ioan Boţ and Christopher Hendrich. A variable smoothing algorithm for solving convex optimization problems. *TOP*, 23(1):124–150, 2012.
9. R.I. Bot and C. Hendrich. A double smoothing technique for solving unconstrained nondifferentiable convex optimization problems. *Compt. Optim. Appl.*, 54(2):239–262, 2013.
10. S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.
11. J. Brodie, I. Daubechies, C. De Mol, D. Giannone, and I. Loris. Sparse and stable markowitz portfolios. *Proceedings of the National Academy of Sciences*, 106(30):12267–12272, 2009.
12. E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.
13. A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vis.*, 40(1):120–145, 2011.
14. A. Chambolle and T. Pock. On the ergodic convergence rates of a first-order primal–dual algorithm. *Math. Program.*, 159(1-2):253–287, 2016.
15. C.-C. Chang and C.-J. Lin. LIBSVM: A library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.
16. G. Chen and M. Teboulle. Convergence analysis of a proximal-like minimization algorithm using Bregman functions. *SIAM J. Optim.*, 3(3):538–543, 1993.
17. S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.
18. P. Combettes and J.-C. Pesquet. Signal recovery by proximal forward-backward splitting. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer-Verlag, 2011.
19. L. Condat. A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms. *J. Optim. Theory Appl.*, 158:460–479, 2013.
20. D. Davis. Convergence rate analysis of the forward-Douglas-Rachford splitting scheme. *SIAM J. Optim.*, 25(3):1760–1786, 2015.
21. D. Davis and W. Yin. Faster convergence rates of relaxed Peaceman-Rachford and ADMM under regularity assumptions. *Math. Oper. Res.*, 2014.
22. O. Devolder, F. Glineur, and Y. Nesterov. Double smoothing technique for large-scale linearly constrained convex optimization. *SIAM J. Optim.*, 22(2):702–727, 2012.
23. D.L. Donoho. Compressed sensing. *IEEE Trans. Inf. Theory*, 25(4):1289–1306, 2006.
24. J. Eckstein. Nonlinear proximal point algorithms using Bregman functions, with applications to convex programming. *Math. Oper. Res.*, 18(1):202–226, 1993.

25. Jonathan Eckstein and Benar Fux Svaiter. A family of projective splitting methods for the sum of two maximal monotone operators. *Math. Program.*, 111(1-2):173–199, 2008.
26. E. Elhamifar and R. Vidal. Sparse subspace clustering. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2009*, pages 2790–2797. IEEE, 2009.
27. E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.
28. J. E. Esser. *Primal-dual algorithm for convex models and applications to image restoration, registration and nonlocal inpainting.* PhD Thesis, University of California, Los Angeles, Los Angeles, USA, 2010.
29. O. Fercoq and Z. Qu. Restarting accelerated gradient methods with a rough strong convexity estimate. *Preprint: arXiv:1609.07358*, pages 1–23, 2016.
30. X. Gao and S.-Z. Zhang. First-order algorithms for convex optimization with nonseparable objective and coupled constraints. *Journal of the Operations Research Society of China*, 5(2):131–159, 2017.
31. A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.
32. P. Giselsson and S. Boyd. Monotonicity and Restart in Fast Gradient Methods. In *IEEE Conference on Decision and Control*, pages 5058–5063, Los Angeles, USA, December 2014. CDC.
33. B.S. He and X.M. Yuan. On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. *SIAM J. Numer. Anal.*, 50:700–709, 2012.
34. M. R. Hestenes. Multiplier and gradient methods. *J. Optim. Theory Appl.*, 4:303–320, 1969. 10.1007/BF00927673.
35. K. C. Kiwiel. Proximal minimization methods with generalized Bregman functions. *SIAM J. Control Optim.*, 35(4):1142–1168, 1997.
36. G. Lan and R.D.C. Monteiro. Iteration complexity of first-order penalty methods for convex programming. *Math. Program.*, 138(1):115–139, 2013.
37. G. Lan and R.D.C. Monteiro. Iteration-complexity of first-order augmented Lagrangian methods for convex programming. *Math. Program.*, 155(1-2):511–547, 2016.
38. Guanghui Lan, Zhaosong Lu, and Renato DC Monteiro. Primal-dual first-order methods with iteration-complexity for cone programming. *Mathematical Programming*, 126(1):1–29, 2011.
39. H. Li and Z. Lin. Accelerated Alternating Direction Method of Multipliers: an Optimal $\mathcal{O}(1/k)$ Nonergodic Analysis. *Journal of Scientific Computing*, pages 1–29, 2016.
40. A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150. Association for Computational Linguistics, 2011.
41. R.D.C. Monteiro and B.F. Svaiter. Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers. *SIAM J. Optim.*, 23(1):475–507, 2013.
42. R.D.C. Monteiro and B.F. Svaiter. Iteration-complexity of block-decomposition algorithms and the alternating minimization augmented Lagrangian method. *SIAM J. Optim.*, 23(1):475–507, 2013.
43. I. Necoara, A. Patrascu, and F. Glineur. Complexity of first-order inexact Lagrangian and penalty methods for conic convex programming. *Optim. Methods Soft.*, 34(2):305–335, 2019.
44. I. Necoara and J.A.K. Suykens. Applications of a smoothing technique to decomposition in convex optimization. *IEEE Trans. Automatic control*, 53(11):2674–2679, 2008.
45. V. Nedelcu, I. Necoara, and Q. Tran-Dinh. Computational Complexity of Inexact Gradient Augmented Lagrangian Methods: Application to Constrained MPC. *SIAM J. Optim. Control*, 52(5):3109–3134, 2014.
46. A. Nemirovskii and D. Yudin. *Problem Complexity and Method Efficiency in Optimization.* Wiley Interscience, 1983.
47. Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87 of *Applied Optimization.* Kluwer Academic Publishers, 2004.

48. Y. Nesterov. Excessive gap technique in nonsmooth convex minimization. *SIAM J. Optim.*, 16(1):235–249, 2005.
49. Y. Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, 2005.
50. V. Q. Nguyen, O. Fercoq, and V. Cevher. Smoothing technique for nonsmooth composite minimization with linear operator. *ArXiv preprint (arXiv:1706.05837)*, 2017.
51. B. O'Donoghue and E. Candes. Adaptive Restart for Accelerated Gradient Schemes. *Found. Comput. Math.*, 15:715–732, 2015.
52. Y. Ouyang, Y. Chen, G. Lan, and E. JR. Pasiliao. An accelerated linearized alternating direction method of multiplier. *SIAM J. Imaging Sci.*, 8(1):644–681, 2015.
53. N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013.
54. J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proc. of 2014 conference on Empirical methods in Natural language processing (EMNLP)*, pages 1532–1543, 2014.
55. F. Pourkamali-Anaraki and S. Becker. Efficient Solvers for Sparse Subspace Clustering. *arXiv preprint arXiv:1804.06291*, 2018.
56. R. T. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathemathics of Operations Research*, 1:97–116, 1976.
57. R. Shefi and M. Teboulle. Rate of Convergence Analysis of Decomposition Methods Based on the Proximal Method of Multipliers for Convex Minimization. *SIAM J. Optim.*, 24(1):269–297, 2014.
58. W. Su, S. Boyd, and E. Candes. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2510–2518, 2014.
59. Q. Tran-Dinh. Proximal Alternating Penalty Algorithms for Constrained Convex Optimization. *Computational Optimization and Applications, Online First*, 72(1):1–43, 2019.
60. Q. Tran-Dinh, O. Fercoq, and V. Cevher. A smooth primal-dual optimization framework for nonsmooth composite convex minimization. *SIAM J. Optim.*, 28(1):96–134, 2018.
61. P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *Submitted to SIAM J. Optim*, 2008.
62. J. von Neumann. *Functional Operators: The Geometry of Orthogonal Spaces.*, volume AM-22. Princeton University Press, 2016.
63. C. B. Vu. A splitting algorithm for dual monotone inclusions involving co-coercive operators. *Advances in Computational Mathematics*, 38(3):667–681, 2013.
64. H. Wang, G. Li, and G. Jiang. Robust regression shrinkage and consistent variable selection through the LAD-Lasso. *J. Business & Economic Statistics*, 25(3):347–355, 2007.
65. L. White, R. Togneri, W. Liu, and M. Bennamoun. Generating bags of words from the sums of their word embeddings. In *17th Int. Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, 2016.
66. B. E. Woodworth and N. Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in neural information processing systems (NIPS)*, pages 3639–3647, 2016.
67. S. J. Wright. Optimization Algorithms for Data Analysis. *IAS/Park City Mathematics Series*, pages 1–49, 2017.
68. Y. Xu. Accelerated first-order primal-dual proximal methods for linearly constrained composite convex programming. *SIAM J. Optim.*, 27(3):1459–1484, 2017.
69. Y. Xu. Iteration complexity of inexact augmented Lagrangian methods for constrained convex programming. *arXiv preprint arXiv:1711.05812*, 2017.
70. Yi Xu, Yan Yan, Qihang Lin, and Tianbao Yang. Homotopy smoothing for non-smooth problems with lower complexity than $o(1/epsilon)$. In *Advances in Neural Information Processing Systems*, pages 1208–1216, 2016.
71. J. Zhu, S. Rosset, R. Tibshirani, and T. J. Hastie. 1-norm support vector machines. In *Advances in neural information processing systems (NIPS)*, pages 49–56, 2004.