An Adaptive Proximal ADMM for Nonconvex Linearly Constrained Composite Programs [*]

Leandro Farias Maia[†] David H. Gutman[‡] Renato D.C. Monteiro[§]
Gilson N. Silva[¶]

July 12, 2024 (1st revision: June 30, 2025)

Abstract

This paper develops an adaptive proximal alternating direction method of multipliers (ADMM) for solving linearly constrained, composite optimization problems under the assumption that the smooth component of the objective is weakly convex, while the non-smooth component is convex and block-separable. The proposed method is adaptive to all problem parameters, including smoothness and weak convexity constants, and allows each of its block proximal subproblems to be inexactly solved. Each iteration of our adaptive proximal ADMM consists of two steps: the sequential solution of each block proximal subproblem; and adaptive tests to decide whether to perform a full Lagrange multiplier and/or penalty parameter update(s). Without any rank assumptions on the constraint matrices, it is shown that the adaptive proximal ADMM obtains an approximate first-order stationary point of the constrained problem in a number of iterations that matches the state-of-the-art complexity for the class of proximal ADMM's. The three proof-of-concept numerical experiments that conclude the paper suggest our adaptive proximal ADMM enjoys significant computational benefits.

Keywords: proximal ADMM, nonseparable, nonconvex composite optimization, iteration-complexity, augmented Lagrangian function

1 Introduction

This paper develops an adaptive proximal alternating direction method of multipliers, called A-ADMM, for solving the linearly constrained, smooth, weakly convex, composite optimization problem

$$\phi^* = \min_{y\in\mathbb{R}^n} \{\phi(y) := f(y) + h(y) : Ay = b\}, \tag{1}$$

where $A : \mathbb{R}^n \to \mathbb{R}^l$ is a linear operator, $b \in \mathbb{R}^l$ is a vector in the image of A, h is a proper closed convex function which is Lipschitz continuous on its compact domain and, for some positive integer B (the number of blocks) and positive integer vector $(n_1,\ldots,n_t)$ such that $n = \sum_{t=1}^B n_t$, has the blockwise representation $h(y) = \sum_{t=1}^B h_t(y_t)$ for every $y = (y_1,\ldots,y_B) \in \mathbb{R}^{n_1}\times\cdots\times\mathbb{R}^{n_B}$, and f is a real-valued weakly convex differentiable function on the domain of h whose gradient satisfies a blockwise Lipschitz condition. Thus, in

[*] **Funding**: The first and second authors are supported by the NSF grant #2410328. The third author has been partially supported by AFOSR Grants FA9550-22-1-0088 and FA9550-25-1-0131. The last author has been supported by CNPq Grants 401864/2022-7 and 306593/2022-0

[†] Department of Industrial and Systems Engineering, Texas A&M University, College Station, TX, 77843. farias-maia@gmail.com

[‡] Department of Industrial and Systems Engineering, Texas A&M University, College Station, TX, 77843. dhgutman@gmail.com

[§] School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332-0205. monteiro@isye.gatech.edu

[¶] Department of Mathematics, Federal University of Piauí, Teresina, PI, 64049-550. gilson.silva@ufpi.edu.br

terms of this blockwise representation, $f(y)$, $h(y)$, and Ay, can be written as

$$f(y) = f(y_1, \ldots, y_B), \quad Ay = \sum_{t=1}^{B} A_t y_t, \quad h(y) = \sum_{t=1}^{B} h_t(y_t), \tag{2}$$

where, for each $t \in \{1, \ldots, B\}$, h_t is a proper closed convex function with compact domain and $A_t : \mathbb{R}^{n_t} \to \mathbb{R}^l$ is a linear operator.

The goal in this paper is to find a (ρ, η)-stationary solution of (1)-(2), i.e., a quadruple $(\hat{x}, \hat{p}, \hat{v}, \hat{\varepsilon}) \in (\operatorname{dom} h) \times A(\mathbb{R}^n) \times \mathbb{R}^l \times \mathbb{R}_+$ satisfying

$$\hat{v} \in \nabla f(\hat{x}) + \partial_{\hat{\varepsilon}} h(\hat{x}) + A^* \hat{p}, \quad \sqrt{\|\hat{v}\|^2 + \hat{\varepsilon}} \leq \rho, \quad \|A\hat{x} - b\| \leq \eta, \tag{3}$$

where $(\rho, \eta) \in \mathbb{R}^2_{++}$ is a given tolerance pair.

A popular primal-dual algorithmic framework for solving problem (1) that takes advantage of its block structure (2) is the proximal ADMM, which is based on the augmented Lagrangian (AL) function,

$$\mathcal{L}_c(y; p) := \phi(y) + \langle p, Ay - b \rangle + \frac{c}{2} \|Ay - b\|^2, \tag{4}$$

where $c > 0$ is a penalty parameter. Given $(\tilde{y}^{k-1}, \tilde{q}^{k-1}, c_{k-1})$, the proximal ADMM finds the next triple $(\tilde{y}^k, \tilde{q}^k, c_k)$ as follows. Starting from $\tilde{y}^{k-1}$, it first performs ℓ_k iterations of a block inexact proximal point (IPP) method applied to $\mathcal{L}_{c_{k-1}}(\cdot\,; \tilde{q}^{k-1})$ to obtain $\tilde{y}_k$ where ℓ_k is a positive integer. Next, it performs a Lagrange multiplier update according to

$$\tilde{q}^k = (1 - \theta)\Big[\tilde{q}^{k-1} + \chi c_k \left(A\tilde{y}^k - b\right)\Big], \tag{5}$$

where $\theta \in [0, 1)$ is a dampening parameter and χ is a positive relaxation parameter, and chooses a scalar $c_k \geq c_{k-1}$ as the next penalty parameter.

We now formally describe how a proximal ADMM obtains $\tilde{y}^k$ from $\tilde{y}^{k-1}$. It sets $z^0 = \tilde{y}^{k-1}$, and for some positive integer ℓ_k, it performs a block IPP iteration from z^{j-1} to obtain z^j for every $j = 1, \ldots \ell_k$, and finally sets $\tilde{y}^k = z^{\ell_k}$. The block IPP iteration to obtain z^j from z^{j-1} consists of inexactly solving, sequentially from $t = 1$ to $t = B$, the t-th block proximal AL subproblem with prox stepsize λ_t

$$z_t^j \approx \operatorname{argmin}_{u_t \in \mathbb{R}^{n_t}} \left\{ \lambda_t \mathcal{L}_{c_{k-1}}(z_{<t}^j, u_t, z_{>t}^{j-1}; \tilde{q}^{k-1}) + \frac{1}{2}\|u_t - z_t^{j-1}\|^2 \right\}, \tag{6}$$

and finally setting $\tilde{y}^k = z^{\ell_k}$.

The recent publication [29] proposes a version of a proximal ADMM for solving (1)-(2) which assumes that $\ell_k = 1$, $\lambda_1 = \cdots = \lambda_B$, and $(\chi, \theta) \in (0, 1]^2$ satisfies

$$2\chi B(2 - \theta)(1 - \theta) \leq \theta^2, \tag{7}$$

and hence that $\theta = 0$ is not allowed in [29].

One of the main contributions of [29] is that its convergence guarantees do not require *the last block condition*, $\operatorname{Im}(A_B) \supseteq \{b\} \cup \operatorname{Im}(A_1) \cup \ldots \cup \operatorname{Im}(A_{B-1})$ and $h_B \equiv 0$, that hinders many instances of proximal ADMM, see [8, 18, 55, 59]. However, the main drawbacks of the proximal ADMM of [29] include: (i) the strong assumption (7) on (χ, θ); (ii) subproblem (6) must be solved exactly; (iii) the stepsize parameter λ is conservative and requires the knowledge of f's weak convexity parameter; (iv) it (conservatively) updates the Lagrange multiplier after each primal update cycle (i.e., $\ell_k = 1$); (v) its iteration-complexity has a high dependence on the number of blocks B, namely, $\mathcal{O}(B^8)$; (vi) its iteration-complexity bound depends linearly on θ^{-1}, and hence grows to infinity as θ approaches zero. Paper [29] also presents computational results comparing its proximal ADMM with a more practical variant where (θ, χ), instead of satisfying (7), is set to $(0, 1)$. Intriguingly, this $(\theta, \chi) = (0, 1)$ regime substantially outperforms the theoretical regime of (7) in the provided computational experiments. No convergence analysis for the $(\theta, \chi) = (0, 1)$ regime is forwarded in [29]. Thus, [29] leaves open the tantalizing question of whether the convergence of proximal ADMM with

$(\theta, \chi) = (0, 1)$ can be theoretically secured.

Contributions: This work partially addresses the convergence analysis issue raised above by studying a *completely parameter-free* proximal ADMM, with $(\theta, \chi) = (0, 1)$ and ℓ_k adaptively chosen, called A-ADMM. Rather than making the conservative determination that $\ell_k = 1$, the studied adaptive method ensures the dual updates are committed as frequently as possible. It is shown that A-ADMM finds a (ρ, η)-stationary solution in $\mathcal{O}(B \max\{\rho^{-3}, \eta^{-3}\})$ iterations. A-ADMM also exhibits the following additional features:

- Similar to the proximal ADMM of [29], its complexity is established without assuming that the *last block condition* holds.
- Compared to the $\mathcal{O}(B^8 \max\{\rho^{-3}, \eta^{-3}\})$ iteration-complexity of the proximal ADMM of [29], the one for A-ADMM vastly *improves the dependence on* B.
- A-ADMM uses an adaptive scheme that adaptively computes *variable block prox stepsizes*, instead of constant ones that require knowledge of weakly convex parameters for f as in the proximal ADMM of [29]. Specifically, while the method of [29] chooses $\lambda_1 = \ldots = \lambda_B \in (0, 1/(2\bar{m})]$ where $\bar{m}$ is a weakly convex parameter for $f(y)$ relative to the whole y, A-ADMM adaptively generates possibly distinct λ_t's that are larger than $1/(2m_t)$ (and hence $1/(2\bar{m})$) where m_t is the weakly convex parameter of f relative to its t-th block y_t. Thus, A-ADMM allows some of (or all) the subproblems (6) to be non-convex.
- A-ADMM is also adaptive to Lipschitz parameters.
- In contrast to the proximal ADMM in [29], A-ADMM allows the block proximal subproblems (6) to be either exactly or *inexactly* solved.

Related Works: ADMM methods with $B = 1$ are well-known to be equivalent to augmented Lagrangian methods. Several references have studied augmented Lagrangian and proximal augmented Lagrangian methods in the convex (see e.g., [1, 2, 34, 35, 36, 38, 45, 50, 57]) and nonconvex (see e.g. [5, 6, 19, 24, 28, 32, 33, 39, 54, 58, 59, 60]) settings. Moreover, ADMMs and proximal ADMMs in the convex setting have also been broadly studied in the literature (see e.g. [5, 7, 9, 10, 11, 12, 13, 15, 16, 17, 44, 51, 52]). So from now on, we just discuss proximal ADMM variants where f is nonconvex and $B > 1$.

A discussion of the existent literature on nonconvex proximal ADMM is best framed by dividing it into two different corpora: those papers that assume the last block condition and those that do not. Under the *last block condition*, the iteration-complexity established is $\mathcal{O}(\varepsilon^{-2})$, where $\varepsilon := \min\{\rho, \eta\}$. Specifically, [8, 18, 55, 56] introduce proximal ADMM approaches assuming $B = 2$, while [25, 26, 40, 41] present (possibly linearized) proximal ADMMs assuming $B \geq 2$. A first step towards removing the last block condition was made by [26] which proposes an ADMM-type method applied to a penalty reformulation of (1)-(2) that artificially satisfies the last block condition. This method possesses an $\mathcal{O}(\varepsilon^{-6})$ iteration-complexity bound.

On the other hand, development of ADMM-type methods directly applicable to (1)-(2) is considerably more challenging and only a few works addressing this topic have surfaced. In addition to [26], earlier contributions to this topic were obtained in [23, 54, 59]. More specifically, [23, 59] develop a novel small stepsize ADMM-type method without establishing its complexity. Finally, [54] considers an interesting but unorthodox negative stepsize for its Lagrange multiplier update, that sets it outside the ADMM paradigm, and thus justifies its qualified moniker, "scaled dual descent ADMM".

1.1 Structure of the Paper

In this subsection, we outline this article's structure. This section's lone remaining subsection, Subsection 1.2, briefly lays out the basic definitions and notation used throughout. Section 2 introduces a notion of an inexact solution of A-ADMM's foundational block proximal subproblem (6) along with efficient subroutines designed to find said solutions. Section 3 presents the static version of the main algorithm of this chapter, S-ADMM, and states the theorem governing its iteration-complexity (Theorem 3.2). Section 4 provides the detailed proof of the iteration-complexity theorem for S-ADMM and presents all supporting technical lemmas. Section 5 introduces the centerpiece algorithm of this work, a proximal ADMM method with constant stepsizes, namely A-ADMM. It also states the main theorem of this chapter (Theorem 5.2), which establishes the iteration-complexity of the method. Section 6 extends A-ADMM to an adaptive stepsizes version and briefly describes how to obtain a completely adaptive method. Section 7 presents proof-of-concept numerical experiments that display the superb efficiency of A-ADMM for three different problem classes. Section 8 gives some concluding remarks that suggest further research directions. Finally, Appendix A presents some technical results on convexity and linear algebra, while Appendix B describes an adaptive accelerated gradient method and its main properties.

1.2 Notation, Definitions, and Basic Facts

This subsection lists the elementary notation deployed throughout the paper. Let $\mathbb{R}$ denote the set of real numbers, and $\mathbb{R}_+$ and $\mathbb{R}_{++}$ denote the set of non-negative and positive real numbers, respectively. We assume that the n-dimensional Euclidean space, $\mathbb{R}^n$, is equipped with an inner product, $\langle \cdot\,, \cdot\rangle$.

The norm induced by $\langle \cdot\,, \cdot\rangle$ is denoted by $\|\cdot\|$. Let $\mathbb{R}^n_{++}$ and $\mathbb{R}^n_+$ denote the set of vectors in $\mathbb{R}^n$ with positive and non-negative entries, respectively. The smallest positive singular value of a nonzero linear operator $Q : \mathbb{R}^n \to \mathbb{R}^l$ is denoted ν^+_Q and its operator norm is $\|Q\| := \sup\{\|Q(w)\| : \|w\| = 1\}$. If S is a symmetric and positive definite matrix, the norm induced by S on $\mathbb{R}^n$, denoted by $\|\cdot\|_S$, is defined as $\|\cdot\|_S = \langle \cdot\,, S(\cdot)\rangle^{1/2}$. For $x = (x_1, \ldots, x_B) \in \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_B}$, we define the aggregated quantities

$$x_{<t} := (x_1, \ldots, x_{t-1}), \quad x_{>t} := (x_{t+1}, \ldots, x_B), \quad x_{\le t} := (x_{<t}, x_t), \quad x_{\ge t} := (x_t, x_{>t}). \tag{8}$$

For a given closed, convex set $Z \subset \mathbb{R}^n$, we let ∂Z designate its boundary. The distance of a point $z \in \mathbb{R}^n$ to Z, measured in terms of $\|\cdot\|$, is denoted $\mathrm{dist}(z, Z)$. Let $\log(\cdot)$ denote the logarithm base 2, and for any $s > 0$ and $b \ge 0$, let $\log^+_b(s) := \max\{\log s, b\}$.

For a given function $g : \mathbb{R}^n \to (-\infty, \infty]$, let $\mathrm{dom}\, g := \{x \in \mathbb{R}^n : g(x) < +\infty\}$ denotes the effective domain of g. We say that g is proper if $\mathrm{dom}\, g \neq \emptyset$. The set of all lower semi-continuous proper convex functions defined in $\mathbb{R}^n$ is denoted by $\overline{\mathrm{Conv}}\,(\mathbb{R}^n)$. For $\varepsilon \ge 0$, the ε-subdifferential of $g \in \overline{\mathrm{Conv}}\,(\mathbb{R}^n)$ at $z \in \mathrm{dom}\, g$ is

$$\partial_\varepsilon g(z) := \{w \in \mathbb{R}^n \ : \ g(\tilde z) \ge g(z) + \langle w\,, \tilde z - z\rangle - \varepsilon, \forall \tilde z \in \mathbb{R}^n\}.$$

When $\varepsilon = 0$, the ε-subdifferential recovers the classical subdifferential, $\partial g(\cdot) := \partial_0 g(\cdot)$. It is well-known (see [22, Prop. 1.3.1 of Ch. XI]) that for any $\beta > 0$ and $g \in \overline{\mathrm{Conv}}\,(\mathbb{R}^n)$,

$$\partial_\varepsilon(\beta g)(\cdot) = \beta \partial_{(\varepsilon/\beta)} g(\cdot). \tag{9}$$

Moreover, if $h_i \in \overline{\mathrm{Conv}}\,(\mathbb{R}^{n_i})$ for $i = 1, \ldots, B$ and $h(y) := \sum_{t=1}^B h_t(y_t)$ for any $y = (y_1, \ldots, y_B)$, then we have (see [22, Remark 3.1.5 of Ch. XI])

$$\partial_\varepsilon h(y) = \cup\{\partial_{\varepsilon_1} h_1(y_1) \times \ldots \times \partial_{\varepsilon_B} h_B(y_B) : \varepsilon_t \ge 0, \ \ \varepsilon_1 + \cdots + \varepsilon_B \le \varepsilon\}. \tag{10}$$

2 Assumptions and an Inexact Solution Concept

This section contains two subsections. The first one details a few mild technical assumptions imposed on the main problem (1)-(2). The second one introduces a notion of an inexact stationary point for the block proximal subproblem (6) along with an efficient method for finding such points.

2.1 Assumptions for Problem (1)-(2)

The main problem of interest in this paper is problem (1) with the block structure as in (2). It is assumed that vector $b \in \mathbb{R}^l$, linear operator $A : \mathbb{R}^n \to \mathbb{R}^l$, and functions $f : \mathbb{R}^n \to (-\infty, \infty]$ and $h_t : \mathbb{R}^{n_t} \to (-\infty, \infty]$ for $t = 1, \ldots, B$, satisfy the following conditions:

(A1) $h_t \in \overline{\mathrm{Conv}}\,(\mathbb{R}^{n_t})$ is prox friendly (i.e., its proximal operator is easily computable) and its domain $\mathcal{H}_t$ is compact;

(A2) there exists $M_h \ge 0$ such that $h(\cdot)$ as in (2) restricted to $\mathcal{H} := \mathcal{H}_1 \times \cdots \times \mathcal{H}_B$ is M_h-Lipschitz continuous;

(A3) for some $m = (m_1, \dots, m_B) \in \mathbb{R}^B_+$, function f is block m-weakly convex, i.e., for every $t \in \{1, \dots, B\}$,

$$f(x_{<t}, \cdot, x_{>t}) + \delta_{\mathcal{H}_t}(\cdot) + \frac{m_t}{2}\| \cdot \|^2 \text{ is convex for all } x \in \mathcal{H};$$

(A4) f is differentiable on $\mathcal{H}$ and, for every $t \in \{1, \dots, B-1\}$, there exists $L_{>t} \geq 0$ such that

$$\|\nabla_{x_t} f(x_{\leq t}, \tilde{x}_{>t}) - \nabla_{x_t} f(x_{\leq t}, x_{>t})\| \leq (L_{>t})\|\tilde{x}_{>t} - x_{>t}\| \quad \forall\, x, \tilde{x} \in \mathcal{H}; \tag{11}$$

(A5) A is nonzero and there exists $\bar{\mathrm{x}} \in \{x \in \mathcal{H} : Ax = b\} \neq \emptyset$ such that $\bar{d} := \mathrm{dist}(\bar{\mathrm{x}}, \partial\mathcal{H}) > 0$.

We now make some remarks about the above assumptions. First, since $\mathcal{H}$ is compact, it follows from (A2) that the scalars

$$D_h := \sup_{z \in \mathcal{H}} \|z - \bar{x}\|, \quad \nabla_f := \sup_{u \in \mathcal{H}} \|\nabla f(u)\|, \quad \underline{\phi} := \inf_{u \in \mathcal{H}} \phi(u), \quad \overline{\phi} := \sup_{u \in \mathcal{H}} \phi(u), \tag{12}$$

are bounded. Second, (A3) allows m_t to be zero for some or all $t \in \{1, \dots, B\}$. Finally, (A5) ensures that (1) has a Slater point.

2.2 An Inexact Solution Concept for (6)

This subsection introduces our notion (Definition 2.1) of an inexact solution of the block proximal AL subproblem (6). To cleanly frame this solution concept, observe that (6) can be cast in the form

$$\min\{\psi(z) := \psi^{(\mathrm{s})}(z) + \psi^{(\mathrm{n})}(z) : z \in \mathbb{R}^n\}, \tag{13}$$

where

$$\psi^{(\mathrm{s})}(\cdot) = \lambda_t \hat{\mathcal{L}}_c(y^i_{<t}, \cdot, y^{i-1}_{>t}; \tilde{q}^{k-1}) + \frac{1}{2}\| \cdot - y^{i-1}_t\|^2, \quad \psi^{(\mathrm{n})}(\cdot) = \lambda_t h_t(\cdot), \tag{14}$$

and $\hat{\mathcal{L}}_c(\cdot\,; \tilde{q}^{k-1})$ is the smooth part of (4), defined as

$$\hat{\mathcal{L}}_c(y; \tilde{q}^{k-1}) := f(y) + \left\langle \tilde{q}^{k-1}, Ay - b \right\rangle + \frac{c}{2}\left\|Ay - b\right\|^2. \tag{15}$$

Hence, to describe a notion of an inexact solution for (6), it is suffices to do so in the context of (13). Assume that:

(B1) $\psi^{(\mathrm{s})} : \mathbb{R}^n \to \mathbb{R}$ is a differentiable function;

(B2) $\psi^{(\mathrm{n})} \in \overline{\mathrm{Conv}}\,(\mathbb{R}^n)$.

Definition 2.1 *For a given $z^0 \in \mathrm{dom}\,\psi^{(\mathrm{n})}$ and parameter $\sigma \in \mathbb{R}_+$, a triple $(\bar{z}, \bar{r}, \bar{\varepsilon}) \in \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}_+$ satisfying*

$$\bar{r} \in \nabla\psi^{(\mathrm{s})}(\bar{z}) + \partial_{\bar{\varepsilon}}\psi^{(\mathrm{n})}(\bar{z}) \quad \text{and} \quad \|\bar{r}\|^2 + 2\bar{\varepsilon} \leq \sigma\|z^0 - \bar{z}\|^2 \tag{16}$$

is called a $(\sigma; z^0)$-stationary solution of (13) with composite term $\psi^{(\mathrm{n})}$.

We now make some remarks about Definition 2.1. First, if $\sigma = 0$, then the inequality in (16) implies that $(\bar{r}, \bar{\varepsilon}) = (0, 0)$, and hence the inclusion in (16) implies that $\bar{z}$ is an exact stationary point of (13), i.e., it satisfies $0 \in \nabla\psi^{(\mathrm{s})}(\bar{z}) + \partial\psi^{(\mathrm{n})}(\bar{z})$. Thus, if the triple $(\bar{z}, \bar{r}, \bar{\varepsilon})$ is a $(\sigma; z^0)$-stationary solution of (13), then $\bar{z}$ can be viewed as an approximate stationary solution of (13) where the residual pair $(\bar{r}, \bar{\varepsilon})$ is bounded according to (16) (instead of being zero as in the exact case). Second, if $\bar{z}$ is an exact stationary point of (13) (e.g., $\bar{z}$ is an exact solution of (13)), then the triple $(\bar{z}, 0, 0)$ is a $(\sigma; z^0)$-stationary point of (13) for any $\sigma \in \mathbb{R}_+$.

In general, an exact solution or stationary point of (13) is not easy to compute. In such a case, Proposition 3.5 of [21] (see also Subsection 2.3 in [31] and Appendix A in [53]) establishes the iteration-complexity of a variant of the original Nesterov's accelerated gradient method [47] (see also [46] and [48, Chapter 2]) to find such an approximate solution under the assumptions that either $\psi^{(\mathrm{s})}$ is convex or is strongly convex.

Appendix B describes the ADAP-FISTA method of [53] and its main properties (see Proposition B.1 in Appendix B). The main reason for focusing on this method instead of the ones above are: i) in addition to being able to find an approximate solution as in Definition 2.1 when $\psi^{(s)}$ is strongly convex and its gradient is Lipschitz continuous, ADAP-FISTA is also applicable to instances where $\psi^{(s)}$ is weakly convex (and hence possibly nonconvex), and; ii) ADAP-FISTA provides a key and easy to check inequality whose validity at every iteration guarantees its successful termination. These two properties of ADAP-FISTA play an important role in the development of adaptive ADMMs with variable prox stepsizes for solving nonconvex instances of (1) (see Section 6). Finally, ADAP-FISTA shares similar features with other accelerated gradient methods (e.g., see [14, 20, 21, 27, 31, 43, 46, 48]) in that: it has similar complexity guarantees regardless of whether it succeeds or fails (e.g., see [27, 30, 49]); it successfully terminates when $\psi^{(s)}$ is μ-strongly convex; and it performs a line search for estimating a local Lipschitz constant for the gradient of $\psi^{(s)}$ (e.g., see [14]).

3 A Static Proximal ADMM

This section contains two subsections. The first one describes an important component of an ADMM method, namely, a subroutine called B-IPP for performing a block IPP iteration within ADMM as mentioned in the paragraph containing (4) and (5). Subsection 3.2 describes an ADMM with constant penalty parameter for solving (1)-(2), referred to as the static ADMM (S-ADMM for short). The qualifier “static” is attached because this variant keeps the penalty parameter constant. The main algorithm of this paper, A-ADMM, presented in Section 5, uses S-ADMM as an important component, i.e., it invokes S-ADMM with c doubled each time.

3.1 A block IPP black-box: B-IPP

The goal of this subsection is to state B-IPP, its main properties, and relevant remarks about it.

We start by describing B-IPP.

Subroutine B-IPP

Input: $(z, p, \lambda, c) \in \mathcal{H} \times A(\mathbb{R}^n) \times \mathbb{R}^B_{++} \times \mathbb{R}_{++}$
Output: $(z^+, v^+, \delta_+, \lambda^+) \in \mathcal{H} \times \mathbb{R}^l \times \mathbb{R}_{++} \times \mathbb{R}^B_{++}$

```
1: STEP 1: Block-IPP Iteration
2: for $t = 1, \ldots, B$ do
3:     set $\lambda_t^+ = \lambda_t$
4:     compute a $(1/8; z_t)$-stationary solution $(z_t^+, r_t^+, \varepsilon_t^+)$ of

       $$\min_{u \in \mathbb{R}^{n_t}} \left\{ \lambda_t^+ \hat{\mathcal{L}}_c(z_{<t}^+, u, z_{>t}; p) + \frac{1}{2}\|u - z_t\|^2 + \lambda_t^+ h_t(u) \right\} \tag{17}$$

       with composite term $\lambda_t^+ h_t(\cdot)$ (see Definition 2.1)
5: $z^+ \leftarrow (z_1^+, \ldots, z_B^+)$ and $\lambda^+ \leftarrow (\lambda_1^+, \ldots, \lambda_B^+)$

6: STEP 2: Computation of the residual pair $(v^+, \delta_+)$ for $(z^+, p)$
7: for $t = 1, \ldots, B$ do
8:     $v_t^+ \leftarrow \nabla_t f(z_{<t}^+, z_t^+, z_{>t}^+) - \nabla_t f(z_{<t}^+, z_t^+, z_{>t}) + \frac{r_t^+}{\lambda_t^+} + cA_t^* \sum_{s=t+1}^B A_s(z_s^+ - z_s) - \frac{1}{\lambda_t^+}(z_t^+ - z_t)$
9: $v^+ \leftarrow (v_1^+, \ldots, v_B^+)$ and $\delta_+ \leftarrow (\varepsilon_1^+/\lambda_1^+) + \ldots + (\varepsilon_B^+/\lambda_B^+)$
10: return $(z^+, v^+, \delta_+, \lambda^+)$
```

We now clarify some aspects of B-IPP. First, line 4 requires a subroutine to find an approximate solution of (17) as in Definition 2.1. A detailed discussion giving examples of such subroutine will be given in the second paragraph after Proposition 3.1. Second, Proposition 3.1 shows that the iterate z^+ and the residual pair (v^+, δ_+) computed in lines 5 and 9, respectively, satisfy the approximate stationary inclusion

$v^+ \in \nabla f(z^+) + \partial_{\delta_+} h(z^+) + \mathrm{Im}(A^*)$. Hence, upon termination of B-IPP, its output satisfies the first condition in (3) (though it may not necessarily fulfill the remaining two) and establishes an important bound on the residual pair (v^+, δ_+) in terms of a Lagrangian function variation that will be used later to determine a suitable potential function.

We now make some remarks about the prox stepsizes. First, B-IPP does not change the prox stepsize and hence, in principle, both λ and λ^+ could be removed from its input and output, respectively. ADMMs based on B-IPP will result in (constant stepsize) ADMM variants that keep their prox stepsize constant throughout. In Subsection 6, we will consider adaptive stepsize ADMM variants based on a adaptive version of B-IPP. The reason for including λ and λ^+ on the input and output of the constant stepsize B-IPP, and its corresponding ADMMs, at this early stage is to facilitate the descriptions of their adaptive counterparts, which will essentially involve a minimal but important change to line 4 of B-IPP.

The quantities

$$\zeta_1 := 100 \max\left\{1\,,\ \max_{1\le t\le B} m_t\right\} + 24L^2 + 1, \qquad \zeta_2 := 24(B-1)\|A\|_\dagger^2, \tag{18}$$

where

$$L := \sqrt{\sum_{t=1}^{B-1} (L_{>t})^2}, \qquad \|A\|_\dagger := \sqrt{\sum_{t=1}^{B} \|A_t\|^2}, \tag{19}$$

with scalars $L_{>t}$'s as in (11) and submatrices A_t's as in (2), are used in the following statement of the main result of this subsection.

Proposition 3.1 *Assume that $(z^+, v^+, \delta_+, \lambda^+) = $ B-IPP(z, p, λ, c) for some $(z, p, \lambda, c) \in \mathcal{H} \times A(\mathbb{R}^n) \times \mathbb{R}^B_{++} \times \mathbb{R}_{++}$. Then,*

$$v^+ \in \nabla f(z^+) + \partial_{\delta_+} h(z^+) + A^*[p + c(Az^+ - b)]. \tag{20}$$

Moreover, if the prox stepsize $\lambda \in \mathbb{R}^B_{++}$ input to B-IPP is chosen as

$$\lambda_t = \frac{1}{2\max\{m_t, 1\}} \qquad \forall t \in \{1, \ldots, B\}, \tag{21}$$

then:

(a) for any $t \in \{1, \ldots, B\}$, the smooth part of the objective function of the t-th block subproblem (17) is $(1/2)$-strongly convex;

(b) it holds that

$$\|v^+\|^2 + \delta_+ \le (\zeta_1 + c\zeta_2)\Big[\mathcal{L}_c(z; p) - \mathcal{L}_c(z^+; p)\Big], \tag{22}$$

where ζ_1 and ζ_2 are as in (18).

We postpone the proof of Proposition 3.1 to the end of Subsection 6.1 and, for now, only make some remarks about it.

First, the inclusion (20) shows that (v^+, δ_+) is a residual pair for the point z^+. Second, the inequality in (22) provides a bound on the magnitude of the residual pair (v^+, δ_+) in terms of a variation of the Lagrangian function which, in the analysis of the next section, will play the role of a potential function. Third, the prox stepsize selection in (21) is so as to guarantee that (22) holds and will play no further role in the analyses of ADMM's presented in the subsequent sections. Fourth, adaptive ways of choosing the prox stepsizes will be discussed on Subsection 6 which will yield ADMM's that do not require knowledge of the parameters m_t's. They are designed to guarantee that a slightly modified version of the inequality in (22) holds (i.e., with a different choice of constant ζ_1), and hence enable the arguments and proofs of the subsequent sections to follow through similarly for ADMM's based on an adaptive stepsize version of B-IPP.

We now comment on the possible ways to obtain a $(1/8; z_t)$-stationary solution of (17) as required in line 4 of B-IPP. As already observed in the paragraph following Definition 2.1, if an exact solution z_t^+ of (17) can be computed in closed form, then $(z_t^+, v_t^+, \varepsilon_t^+) = (z_t^+, 0, 0)$ is a $(1/8; z_t)$-stationary solution of (17). Another approach is to use ADAP-FISTA described in Appendix B. Specifically, assume that $\nabla_{x_t} f(x_1, \ldots, x_B)$ exists for every $x = (x_1, \ldots, x_B) \in \mathcal{H}$ and is $\tilde{L}_t$-Lipschitz continuous with respect to the t-th block x_t. Using this assumption and Proposition 3.1(a), it follows from statement (c) of Proposition B.1 in Appendix B with $\tilde{M} = 1 + \lambda_t(\tilde{L}_t + c\|A_t\|^2)$ that ADAP-FISTA with input $(\sigma, z^0, M_0, \mu_0) = (1/\sqrt{8}, z_t, \lambda_t c\|A_t\|^2, 1/2)$ obtains a $(1/8; z_t)$-stationary solution of (17). Moreover, since $M_0 \le \tilde{M}$, $\mu_0 = 1/2$, and $\lambda_t \le 1/2$ for every $t \in \{1, \ldots, B\}$, Proposition B.1(a) ensures that the number of iterations performed by ADAP-FISTA to obtain such a near-stationary solution is bounded (up to logarithmic terms) by $\mathcal{O}([\tilde{L}_t + c\|A_t\|^2]^{1/2})$.

Finally, as already observed before, B-IPP is a key component that is invoked once in every iteration of the ADMMs presented in subsequent sections. The complexity bounds for these ADMMs will be given in terms of ADMM iterations (and hence B-IPP calls) and will not take into account the complexities of implementing line 4. The main reason for doing so is the possible different ways of solving the block subproblems (e.g., in closed form, or using an ACG variant, or some other convex optimization solver). Nevertheless, the discussion in the previous paragraph provides ways of estimating the contribution of each block to the overall algorithmic effort.

3.2 The S-ADMM Method

This subsection describes S-ADMM, an ADMM with constant penalty parameter for solving (1)-(2).

We start by elaborating S-ADMM.

Algorithm 1 S-ADMM

Universal Input: $\rho > 0$, $\alpha \in [\rho^2, +\infty)$, $C \in [\rho, +\infty)$
Input: $(y^0, q^0, \lambda^0, c) \in \mathcal{H} \times A(\mathbb{R}^n) \times \mathbb{R}^B_{++} \times \mathbb{R}_{++}$
Output: $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}, \hat{\lambda})$

```
1:  $T_0 = 0$, $k = 0$
2:  for $i \leftarrow 1, 2, \ldots$ do
3:      $(y^i, v^i, \delta_i, \lambda^i) =$ B-IPP$(y^{i-1}, q^{i-1}, \lambda^{i-1}, c)$
4:      if $\|v^i\|^2 + \delta_i \le \rho^2$ then                          ▷ termination criteria
5:          $k \leftarrow k + 1$,  $j_k \leftarrow i$                          ▷ end of the final epoch
6:          $q^i = q^{i-1} + c(Ay^i - b)$
7:          return $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}, \hat{\lambda}) = (y^i, q^i, v^i, \delta_i, \lambda^i)$

8:      $T_i = \mathcal{L}_c(y^{i-1}; q^{i-1}) - \mathcal{L}_c(y^i; q^{i-1}) + T_{i-1}$
9:      if $\|v^i\|^2 + \delta_i \le C^2$ and $\frac{\rho^2}{\alpha(k+1)} \ge \frac{T_i}{i}$ then
10:         $k \leftarrow k + 1$,  $j_k \leftarrow i$                          ▷ end of epoch $\mathcal{I}_k$
11:         $q^i = q^{i-1} + c(Ay^i - b)$                        ▷ Lagrange multiplier update
12:     else
13:         $q^i = q^{i-1}$
```

We now make comments about S-ADMM. The iteration index i counts the number of iterations of S-ADMM, referred to as S-ADMM iterations throughout the paper. Index k counts the number of Lagrange multiplier updates performed by S-ADMM. The index j_k computed either in lines 5 or 10 of S-ADMM is the iteration index where the k-th Lagrange multiplier occurs. It is shown in Theorem 3.2(a) that the total number of iterations performed by S-ADMM is finite, and hence that the index j_k is well-defined. If the inequality in line 4 is satisfied, S-ADMM performs the last Lagrange multiplier update and stops in line 7. Otherwise, depending on the test in line 9, S-ADMM performs a Lagrange multiplier in line 11, or leaves it unchanged in line 13, and in both cases moves on to the next iteration.

We next define a few concepts that will be used in the discussion and analysis of S-ADMM. Define the k-th epoch $\mathcal{I}_k$ as the index set

$$\mathcal{I}_k := \{j_{k-1} + 1, \ldots, j_k\}, \tag{23}$$

with the convention that $j_0 = 0$, and let

$$(\tilde{y}^k, \tilde{q}^k, \tilde{\lambda}^k, \tilde{T}_k) := (y^{j_k}, q^{j_k}, \lambda^{j_k}, T_{j_k}) \quad \forall k \geq 0 \quad \text{and} \quad (\tilde{v}^k, \tilde{\delta}_k) := (v^{j_k}, \delta_{j_k}) \quad \forall k \geq 1. \tag{24}$$

We now make three additional remarks about the logic of S-ADMM regarding the prox stepsize and the Lagrange multiplier. First, since the prox stepsize λ^+ output by B-IPP is equal to the prox stepsize λ input to it, it follows from line 3 of S-ADMM that $\lambda^i = \lambda^{i-1}$, and hence that $\lambda^i = \lambda^0$ for every $i \geq 1$. Second, due to the definition of j_k, it follows that $q^i = q^{i-1}$ for every $i \in \{j_{k-1}+1, \ldots, j_k - 1\} = \mathcal{I}_k \setminus \{j_k\}$, which implies that

$$q^{i-1} = q^{j_{k-1}} = \tilde{q}^{k-1} \quad \forall i \in \mathcal{I}_k. \tag{25}$$

Moreover, (24) and (25) with $i = j_k$ imply that

$$\tilde{q}^k = q^{j_k} = q^{j_k - 1} + c(Ay^{j_k} - b) = \tilde{q}^{k-1} + c(A\tilde{y}^k - b) \quad \forall k \geq 1. \tag{26}$$

Noting that (A5) implies that $b \in \mathrm{Im}(A)$, and using the assumption that $\tilde{q}^0 = q^0 \in \mathrm{Im}(A)$, identity (26), and a simple induction argument, we conclude that $\tilde{q}^k \in A(\mathbb{R}^n)$ for every $k \geq 1$.

Before stating the main result of this subsection, we define the quantities

$$\begin{aligned} \Upsilon(C) &:= \frac{2D_h M_h + (2D_h + 1)(C + C^2 + \nabla_f)}{\bar{d}\nu_A^+}, \\ \Gamma(y^0, q^0; c) &:= \overline{\phi} - \underline{\phi} + c\|Ay^0 - b\|^2 + \left[\frac{4(\zeta_1 + c\zeta_2)}{\alpha} + 1\right] \frac{\|q^0\|^2 + \Upsilon^2(C)}{c}, \end{aligned} \tag{27}$$

where (y^0, q^0, λ^0, c) is the input of S-ADMM, (ζ_1, ζ_2) is as in (18), M_h and $\bar{d}$ are as in (A2) and (A5), respectively, (D_h, ∇_f) is as in (12), and ν_A^+ is the smallest positive singular value of the nonzero linear operator A.

The main iteration-complexity result for S-ADMM, whose proof is given in Section 4, is stated next.

Theorem 3.2 (S-ADMM Complexity) *Assume that $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}, \hat{\lambda}) =$ S-ADMM(y^0, q^0, λ^0, c) for some $(y^0, q^0, \lambda^0, c) \in \mathcal{H} \times A(\mathbb{R}^n) \times \mathbb{R}^B_{++} \times \mathbb{R}_{++}$ such that*

$$\lambda_t^0 = \frac{1}{2\max\{m_t, 1\}} \quad \forall t \in \{1, \ldots, B\}. \tag{28}$$

Then, for any tolerance pair $(\rho, \eta) \in \mathbb{R}^2_{++}$, the following statements hold for S-ADMM:

(a) its total number of iterations (and hence B-IPP calls) is bounded by

$$\left(\frac{\zeta_1 + c\zeta_2}{\rho^2}\right) \Gamma(y^0, q^0; c) + 1, \tag{29}$$

where (ζ_1, ζ_2) and $\Gamma(y^0, q^0; c)$ are as in (18) and (27), respectively;

(b) its output $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}, \hat{\lambda})$ satisfies $\hat{\lambda} = \lambda^0$,

$$\hat{v} \in \nabla f(\hat{y}) + \partial_{\hat{\varepsilon}} h(\hat{y}) + A^* \hat{q} \quad \textit{and} \quad \|\hat{v}\|^2 + \hat{\varepsilon} \leq \rho^2, \tag{30}$$

and the following bounds

$$c\|A\hat{y} - b\| \leq 2\max\{\|q^0\|, \Upsilon(C)\} \quad \textit{and} \quad \|\hat{q}\| \leq \max\{\|q^0\|, \Upsilon(C)\}; \tag{31}$$

(c) if $c \geq 2\max\{\|q^0\|, \Upsilon(C)\}/\eta$, then the output $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}, \hat{\lambda})$ of S-ADMM is a (ρ, η)-stationary solution of problem (1)-(2) according to (3).

We now make some remarks about Theorem 3.2. First, Theorem 3.2(b) implies that S-ADMM returns a quintuple $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}, \hat{\lambda})$ satisfying both conditions in (30), but not necessarily the feasibility condition $\|A\hat{y} - b\| \le \eta$. However, Theorem 3.2(c) guarantees that, if c is chosen large enough, i.e., $c = \Omega(\eta^{-1})$, then the feasibility also holds, and hence that $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta})$ is a (ρ, η)-stationary solution of (1)-(2).

Second, assuming for simplicity that $q^0 = 0$, it follows from (27) and (29) that the overall complexity of S-ADMM is

$$\mathcal{O}\left(\frac{1+c}{\rho^2}\left(1 + c\|Ay^0 - b\|^2\right)\right).$$

If the initial point y^0 satisfies $c\|Ay^0 - b\|^2 = \mathcal{O}(1)$, then the bound further reduces to $\mathcal{O}((1+c)\rho^{-2})$. Moreover, under the assumption made in Theorem 3.2(c), i.e., that $c = \Theta(\eta^{-1})$, then the above two complexity estimates reduces to $\mathcal{O}(\eta^{-2}\rho^{-2})$ if y^0 is arbitrary and to $\mathcal{O}(\eta^{-1}\rho^{-2})$ if y^0 satisfies $c\|Ay^0 - b\|^2 = \mathcal{O}(1)$.

Finally, it is worth discussing the dependence of the complexity bound (29) in terms of number of blocks B only. It follows from the definition of ζ_2 in (18) that $\zeta_2 = \Theta(B)$. This implies that $\Gamma(y^0, q^0; c) = \mathcal{O}(1 + B/\alpha)$ due to (27), and hence that the complexity bound (29) is $\mathcal{O}(B(1 + B\alpha^{-1})$. Thus, if α is chosen to be $\alpha = \Omega(B)$ then the dependence of (29) in terms of B only is $\mathcal{O}(B)$.

In Section 5, we present an ADMM variant, namely A-ADMM, which gradually increases the penalty parameter and achieves the complexity bound $\mathcal{O}(\eta^{-1}\rho^{-2})$ of the previous paragraph regardless of the choice of the initial point y^0. Specifically, A-ADMM repeatedly invokes S-ADMM using a warm-start strategy, i.e., if c is the penalty parameter used in the previous S-ADMM call and $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}, \hat{\lambda})$ denotes its output, then the current S-ADMM call uses $(\hat{y}, \hat{q}, \hat{\lambda}, 2c)$ as input, and hence with a doubled penalty parameter.

4 The Proof of S-ADMM's Complexity Theorem (Theorem 3.2)

This section gives the proof of Theorem 3.2.

Its first result shows that every iterate $(y^i, v^i, \delta_i, \lambda^i)$ of S-ADMM satisfies the stationary inclusion $v^i \in \nabla f(y^i) + \partial_{\delta_i} h(y^i) + \mathrm{Im}(A^*)$ and derives a bound on the residual error (v^i, δ_i).

Lemma 4.1 *The following statements about S-ADMM hold:*

(a) for every iteration index $i \ge 1$,

$$v^i \in \nabla f(y^i) + \partial_{\delta_i} h(y^i) + A^*[q^{i-1} + c(Ay^i - b)]; \tag{32}$$

(b) if the initial prox stepsize λ^0 is chosen as in (28), then for every iteration index $i \ge 1$,

$$\frac{1}{\zeta_1 + c\zeta_2}(\|v^i\|^2 + \delta_i) \le \mathcal{L}_c(y^{i-1}; q^{i-1}) - \mathcal{L}_c(y^i; q^{i-1}) = T_i - T_{i-1}. \tag{33}$$

Proof: In view of line 3 of S-ADMM, we have that $(y^i, v^i, \delta_i, \lambda^i) = \text{B-IPP}(y^{i-1}, q^{i-1}, \lambda^{i-1}, c)$. Moreover, if λ^0 is chosen according to (28) then we have that $\lambda_t^i = \lambda_t^0 = 1/\left(2\max\{m_t, 1\}\right)$ for every $i \ge 1$ and $t \in \{1, \ldots, B\}$ since every iteration of S-ADMM does not change the prox stepsize (see its line 3). The result now follows from these two observations, Proposition 3.1 with $(z, p, \lambda, c) = (y^{i-1}, q^{i-1}, \lambda^{i-1}, c)$ and $(z^+, v^+, \delta_+, \lambda^+) = (y^i, v^i, \delta_i, \lambda^i)$, and line 8 of S-ADMM. ■

We now make some remarks about Lemma 4.1. First, (33) implies that: $\{T_i\}$ is nondecreasing; and, if $T_i = T_{i-1}$, then $(v^i, \delta_i) = (\mathbf{0}, 0)$, which together with (32), implies that the algorithm stops in line 7 with an exact stationary point for problem (1)-(2). In view of this remark, it is natural to view $\{T_i\}$ as a potential sequence. Second, if $\{T_i\}$ is bounded, (33) immediately implies that the quantity $\|v^i\|^2 + \delta_i$ converges to zero, and hence that y^i eventually becomes a near stationary point for problem (1)-(2), again in view of (32). A major effort of our analysis will be to show that $\{T_i\}$ is bounded.

With the above goal in mind, the following result gives an expression for T_i that plays an important role in our analysis.

Lemma 4.2 *If i is an iteration index generated by* S-ADMM *such that $i \in \mathcal{I}_k$, then*

$$T_i = \left[\mathcal{L}_c(\tilde{y}^0;\tilde{q}^0) - \mathcal{L}_c(y^i;\tilde{q}^{k-1})\right] + \frac{1}{c}\sum_{\ell=1}^{k-1}\|\tilde{q}^\ell - \tilde{q}^{\ell-1}\|^2. \tag{34}$$

Proof: We first note that

$$\begin{aligned} T_i - T_1 &= \sum_{j=2}^{i}(T_j - T_{j-1}) = \sum_{j=1}^{i-1}(T_{j+1}-T_j) = \sum_{j=1}^{i-1}\left[\mathcal{L}_c(y^j;q^j) - \mathcal{L}_c(y^{j+1};q^j)\right] \\ &= \sum_{j=1}^{i-1}\left[\mathcal{L}_c(y^j;q^j) - \mathcal{L}_c(y^j;q^{j-1})\right] + \sum_{j=1}^{i-1}\left[\mathcal{L}_c(y^j;q^{j-1}) - \mathcal{L}_c(y^{j+1};q^j)\right]. \end{aligned} \tag{35}$$

Moreover, using the definition of T_i with $i=1$ (see line 8 of S-ADMM), the fact that $q^{i-1} = \tilde{q}^{k-1}$ due to (24) and simple algebra, we have

$$\begin{aligned} &T_1 + \sum_{j=1}^{i-1}\left[\mathcal{L}_c(y^j;q^{j-1}) - \mathcal{L}_c(y^{j+1};q^j)\right] = T_1 + \mathcal{L}_c(y^1;q^0) - \mathcal{L}_c(y^i;q^{i-1}) \\ &= [\mathcal{L}_c(y^0;q^0) - \mathcal{L}_c(y^1;q^0)] + [\mathcal{L}_c(y^1;q^0) - \mathcal{L}_c(y^i;\tilde{q}^{k-1})] = \mathcal{L}_c(y^0;q^0) - \mathcal{L}_c(y^i;\tilde{q}^{k-1}). \end{aligned} \tag{36}$$

Using the definition of the Lagrangian function (see definition in (4)), relations (26) and (25), we conclude that for any $\ell \le k$,

$$\mathcal{L}_c(y^j;q^j) - \mathcal{L}_c(y^j;q^{j-1}) \overset{(4)}{=} \left\langle Ay^j - b\,, q^j - q^{j-1}\right\rangle \overset{(26),(25)}{=} \begin{cases} 0 & \text{, if } j \in \mathcal{I}_\ell \setminus \{j_\ell\}; \\ \dfrac{\|\tilde{q}^\ell - \tilde{q}^{\ell-1}\|^2}{c} & \text{, if } j = j_\ell, \end{cases}$$

and hence that

$$\sum_{j=1}^{i-1}\left[\mathcal{L}_c(y^j;q^j) - \mathcal{L}_c(y^j;q^{j-1})\right] = \frac{1}{c}\sum_{\ell=1}^{k-1}\|\tilde{q}^\ell - \tilde{q}^{\ell-1}\|^2.$$

Identity (34) now follows by combining the above identity with the ones in (35) and (36). ■

The next technical result will be used to establish an upper bound on the first term of the right hand side of (34).

Lemma 4.3 *For any given $c>0$ and pairs $(u,p) \in \mathcal{H}\times\mathbb{R}^l$ and $(\tilde{u},\tilde{p}) \in \mathcal{H}\times\mathbb{R}^l$, we have*

$$\mathcal{L}_c(u;p) - \mathcal{L}_c(\tilde{u};\tilde{p}) \le \overline{\phi} - \underline{\phi} + c\|Au-b\|^2 + \frac{1}{2c}\max\{\|p\|,\|\tilde{p}\|\}^2 \tag{37}$$

where $(\overline{\phi},\underline{\phi})$ is as in (12).

Proof: Using the definitions of $\mathcal{L}_c(\,\cdot\,;\,\cdot\,)$ and $\underline{\phi}$ as in (4) and (12), respectively, we have

$$\begin{aligned} \mathcal{L}_c(\tilde{u};\tilde{p}) - \underline{\phi} &\overset{(12)}{\ge} \mathcal{L}_c(\tilde{u};\tilde{p}) - (f+h)(\tilde{u}) \\ &\overset{(4)}{=} \langle \tilde{p}, A\tilde{u}-b\rangle + \frac{c}{2}\|A\tilde{u}-b\|^2 = \frac{1}{2}\left\|\frac{\tilde{p}}{\sqrt{c}} + \sqrt{c}(A\tilde{u}-b)\right\|^2 - \frac{\|\tilde{p}\|^2}{2c} \ge -\frac{\|\tilde{p}\|^2}{2c}. \end{aligned}$$

On the other hand, using the definitions of $\mathcal{L}_c(\,\cdot\,;\,\cdot\,)$ and $\bar{\phi}$ as in (4) and (12), respectively, and the Cauchy-Schwarz inequality, we have

$$\begin{aligned} \mathcal{L}_c(u;p) - \overline{\phi} &\overset{(12)}{\le} \mathcal{L}_c(u;p) - (f+h)(u) \overset{(4)}{=} \langle p, Au-b\rangle + \frac{c\|Au-b\|^2}{2} \\ &\le \left(\frac{\|p\|^2}{2c} + \frac{c\|Au-b\|^2}{2}\right) + \frac{c\|Au-b\|^2}{2} = \frac{\|p\|^2}{2c} + c\|Au-b\|^2. \end{aligned}$$

Combining the above two relations, we then conclude that (37) holds. ■

The following result shows that the sequence $\{T_i\}$ generated by S-ADMM is bounded.

Proposition 4.4 *The following statements about* S-ADMM *hold:*

(a) its total number E *of epochs is bounded by* $\lceil(\zeta_1 + c\zeta_2)/\alpha\rceil$ *where* ζ_1 *and* ζ_2 *are as in* (18)*;*

(b) for every iteration index i*, we have* $T_i \le \Lambda_{\mathrm{E}}(y^0; c)$*;*

(c) the number of iterations performed by S-ADMM *is bounded by*

$$1 + \left(\frac{\zeta_1 + c\zeta_2}{\rho^2}\right)\Lambda_{\mathrm{E}}(y^0; c), \tag{38}$$

where

$$\Lambda_{\mathrm{E}}(y^0; c) := \overline{\phi} - \underline{\phi} + c\|A\tilde{y}^0 - b\|^2 + \frac{Q_{\mathrm{E}}^2}{2c} + \frac{(\zeta_1 + c\zeta_2)F_{\mathrm{E}}^2}{c\alpha}, \tag{39}$$

and

$$\begin{aligned} Q_{\mathrm{E}} &:= \max\left\{\|\tilde{q}^k\| : k \in \{0, \ldots, \mathrm{E}-1\}\right\}, \\ F_{\mathrm{E}} &:= \begin{cases} 0 & , \textit{ if } \mathrm{E} = 1 \\ \max\left\{\|\tilde{q}^k - \tilde{q}^{k-1}\| : k \in \{1, \ldots, \mathrm{E}-1\}\right\} & , \textit{ if } \mathrm{E} \ne 1. \end{cases} \end{aligned} \tag{40}$$

Proof: (a) Assume for the sake of contradiction that S-ADMM generates an epoch $\mathcal{I}_K$ such that $K > \lceil(\zeta_1 + c\zeta_2)/\alpha\rceil$, and hence $K \ge 2$. Using the fact that j_{K-1} is the last index of $\mathcal{I}_{K-1}$ and noting the epoch termination criteria in line 9 of S-ADMM, we then conclude that $\tilde{T}_{K-1}/j_{K-1} \le \rho^2/K$. Also, since S-ADMM did not terminate during epochs $\mathcal{I}_1, \ldots, \mathcal{I}_{K-1}$, it follows from its termination criterion in line 4 that $\|v^i\|^2 + \delta_i > \rho^2$ for every iteration $i \le j_{K-1}$. These two previous observations, (33) with $i \in \{1, \ldots, j_{K-1}\}$, the facts that $T_0 = 0$ by definition and $T_{j_{K-1}} = \tilde{T}_{K-1}$ due to (24), imply that

$$\rho^2 < \frac{1}{j_{K-1}}\sum_{i=1}^{j_{K-1}}(\|v^i\|^2 + \delta_i) \overset{(33)}{\le} \frac{\zeta_1 + c\zeta_2}{j_{K-1}}\sum_{i=1}^{j_{K-1}}(T_i - T_{i-1}) = \frac{(\zeta_1 + c\zeta_2)\tilde{T}_{K-1}}{j_{K-1}} \le \frac{(\zeta_1 + c\zeta_2)}{\alpha K}\rho^2.$$

Since this inequality and the assumption (for the contradiction) that $K > \lceil(\zeta_1 + c\zeta_2)/\alpha\rceil$ yield an immediate contradiction, the conclusion of the statement follows.

(b) Since $\{T_i\}$ is nondecreasing, it suffices to show that $T_i \le \Lambda_{\mathrm{E}}(y^0; c)$ holds for any $i \in \mathcal{I}_{\mathrm{E}}$, where E is the total number of epochs of S-ADMM (see statement (a)). It follows from the definition of Q_{E}, and Lemma 4.3 with $(u, p) = (\tilde{y}^0; \tilde{q}^0)$ and $(\tilde{u}, \tilde{p}) = (y^i; \tilde{q}^{\mathrm{E}-1})$, that

$$\mathcal{L}_c(\tilde{y}^0; \tilde{q}^0) - \mathcal{L}_c(y^i; \tilde{q}^{\mathrm{E}-1}) \le \overline{\phi} - \underline{\phi} + c\|A\tilde{y}^0 - b\|^2 + \frac{1}{2c}Q_{\mathrm{E}}^2.$$

Now, using the definition of F_{E} as in (40), we have that

$$\frac{1}{c}\sum_{j=1}^{\mathrm{E}-1}\|\tilde{q}^j - \tilde{q}^{j-1}\|^2 \le \frac{(\mathrm{E}-1)}{c}F_{\mathrm{E}}^2 \le \frac{(\zeta_1 + c\zeta_2)}{c\alpha}F_{\mathrm{E}}^2,$$

where the last inequality follows from the fact that $\mathrm{E} - 1 \le (\zeta_1 + c\zeta_2)/\alpha$ due to statement (a). The inequality $T_i \le \Lambda_{\mathrm{E}}(y^0; c)$ now follows from the two inequalities above and identity (34) with $k = \mathrm{E}$.

(c) Assume by contradiction that there exists an iteration index i generated by S-ADMM such that

$$i > \left(\frac{\zeta_1 + c\zeta_2}{\rho^2}\right)\Lambda_{\mathrm{E}}(y^0; c) + 1. \tag{41}$$

Since S-ADMM does not stop at any iteration index smaller than i, the stopping criterion in line 4 is violated at these iterations, i.e., $\|v^j\|^2 + \delta_j > \rho^2$ for every $j \le i - 1$. Hence, it follows from (33), the previous inequality, the fact that $T_0 = 0$ due to line 1 of S-ADMM, and statement (b) that

$$\frac{(i-1)\rho^2}{\zeta_1 + c\zeta_2} < \frac{1}{\zeta_1 + c\zeta_2}\sum_{j=1}^{i-1}(\|v^j\|^2 + \delta_j) \le \sum_{j=1}^{i-1}(T_j - T_{j-1}) = T_{i-1} - T_0 \le T_i \le \Lambda_{\mathrm{E}}(y^0, c),$$

which contradicts (41). Thus, statement (c) holds. ■

We now make some remarks about Lemma 4.4. First, Lemma 4.4(a) shows that the number of epochs depends linearly on c. Second, Lemma 4.4(c) shows that the total number of iterations of S-ADMM is bounded but the derived bound is given in terms of the quantities Q_{E} and F_{E} in (40), both of which depend on the magnitude of the sequence of generated Lagrange multipliers $\{\tilde{q}_k : k = 1, \ldots, E\}$. Hence, the bound in (38) is algorithm-dependent in that it depends on the sequence $\{\tilde{q}^k\}$ generated by S-ADMM.

In what follows, we derive a bound on the total number of iterations performed by S-ADMM that depends only on the instance of (1)-(2) under consideration. With this goal in mind, the following result provides a uniform bound on the sequence of Lagrange multipliers generated by S-ADMM that depends only on the instance of (1)-(2).

Lemma 4.5 *The following statements about* S-ADMM *hold:*

(a) it holds that

$$\|\tilde{q}^k\| \leq \max\{\|q^0\|, \Upsilon(C)\}, \quad \forall\ k \in \{1, \ldots, \mathrm{E}\}; \tag{42}$$

(b) if i is an iteration index such that $\|v^i\|^2 + \delta_i \leq C^2$, then

$$c\|Ay^i - b\| \leq 2\max\{\|q^0\|, \Upsilon(C)\};$$

(c) it holds that

$$\|\tilde{q}^k - \tilde{q}^{k-1}\| \leq 2\max\{\|q^0\|, \Upsilon(C)\} \quad \forall\ k \in \{1, \ldots, \mathrm{E}\}. \tag{43}$$

Proof: (a) We first define the index set

$$\mathcal{I}_k(C) := \{i \in \mathcal{I}_k : \|v^i\|^2 + \delta_i \leq C^2\}, \tag{44}$$

where $C > 0$ is part of the input for S-ADMM and (v^i, δ_i) is as in line 9 of B-IPP. We now claim that the vector pair $(\tilde{q}^{k-1}, y^i)$ satisfies

$$\|\tilde{q}^{k-1} + cA(y^i - b)\| \leq \max\{\|\tilde{q}^{k-1}\|, \Upsilon(C)\}, \quad \forall i \in \mathcal{I}_k(C). \tag{45}$$

To show the claim, let $i \in \mathcal{I}_k(C)$ be given. To simplify notation, define

$$\underline{q}^i := \tilde{q}^{k-1} + c(Ay^i - b) \quad \text{and} \quad r^i := v^i - \nabla f(y^i)$$

and note that the triangle inequality for norms, and the definitions of ∇_f in (12) and $\mathcal{I}_k(C)$ in (44), imply that

$$\delta_i + \|r^i\| \leq C^2 + \|v^i\| + \|\nabla f(y^i)\| \leq C^2 + (C + \nabla_f). \tag{46}$$

The fact that $(y^i, v^i, \delta_i, \lambda^i) = $ B-IPP$(y^{i-1}, q^{i-1}, \lambda^{i-1}, c)$ (see line 3 of S-ADMM), the definitions of $\underline{q}^i$ and r^i, Lemma 4.1(a), and identity (25), imply that $r^i \in \partial_{\delta_i} h(y^i) + A^*\underline{q}^i$. Hence, the pair $(q^-, \varrho) = (q^{i-1}, c)$ and the quadruple $(z, q, r, \delta) = (y^i, \underline{q}^i, r^i, \delta_i)$ satisfy the conditions in (65) of Lemma A.3. Thus , the conclusion of the same lemma, the definitions of $\Upsilon(\cdot)$ and $\upsilon(\cdot)$ in (27) and (67), respectively, inequality (46), and the fact that υ is non-decreasing, imply that

$$\|\underline{q}^i\| \overset{(66)}{\leq} \max\left\{\|q^{i-1}\|, \upsilon(\|r^i\| + \delta_i)\right\} \overset{(46)}{\leq} \max\left\{\|q^{i-1}\|, \upsilon(C + C^2 + \nabla_f)\right\} \overset{(25)}{=} \max\{\|\tilde{q}^{k-1}\|, \Upsilon(C)\},$$

where the equality follows from (25). We have thus proved that the claim holds.

We now show that (42) holds. Using (45) with $i = j_k$, the facts that $\underline{q}^{j_k} = \tilde{q}^k = \tilde{q}^{k-1} + c(A\tilde{y}^k - b)$ due to (26), and $\tilde{y}^k = y^{j_k}$ due to (24), and the triangle inequality for norms, we have that

$$\|\tilde{q}^k\| = \|\tilde{q}^{k-1} + cA(\tilde{y}^k - b)\| \leq \max\{\|\tilde{q}^{k-1}\|, \Upsilon(C)\}.$$

Inequality (42) now follows by recursively using the last inequality and the fact that $\tilde{q}^0 = q^0$ due to (24).

(b) Using that $c(Ay^i - b) = \underline{q}^i - \tilde{q}^{k-1}$ and (45) we have

$$c\|Ay^i - b\| \leq \|\underline{q}^i\| + \|\tilde{q}^{k-1}\| \overset{(45)}{\leq} \max\{\|\tilde{q}^{k-1}\|, \Upsilon(C)\} + \|\tilde{q}^{k-1}\| \overset{(42)}{\leq} 2\max\{\|\tilde{q}^0\|, \Upsilon(C)\}$$

where the last inequality above follows from (42) with $k = k - 1$ and the fact that $\tilde{q}^0 = q^0$ due to (24).

(c) Statement (c) follows from (26), the triangle inequality for norms, statement (b) with $i = j_k$ and the fact that $\tilde{y}^k = y^{j_k}$ due to (24). ■

Proof of Theorem 3.2: (a) It follows from Proposition 4.4(c) that the total number of iterations generated by S-ADMM is bounded by the expression in (38). Now, recalling that E is the last epoch generated by S-ADMM, using (40), (42) and (43) we see that $Q_{\rm E} \le \max\{\|q^0\|, \Upsilon(C)\}$ and $F_{\rm E} \le 2\max\{\|q^0\|, \Upsilon(C)\}$, which implies that $\Lambda_{\rm E}(y^0; c) \le \Gamma(y^0, q^0; c)$, where $\Lambda_{\rm E}(y^0; c)$ and $\Gamma(y^0, q^0; c)$ are as in (39) and (27), respectively. The conclusion now follows from the two previous observations.

(b) We first prove that the inclusion in (30) holds. It follows from Lemma 4.1(a) with $i = j_{\rm E}$ and (24) with $k = {\rm E}$ that

$$\tilde{v}^{\rm E} \in \nabla f(\tilde{y}^{\rm E}) + \partial_{\tilde{\delta}_{\rm E}} h(\tilde{y}^{\rm E}) + A^*[q^{j_{\rm E}-1} + c(A\tilde{y}^{\rm E} - b)].$$

Using (25) with $i = j_{\rm E}$, (26) with $k = {\rm E}$, and the fact that $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}) = (\tilde{y}^{\rm E}, \tilde{q}^{\rm E}, \tilde{v}^{\rm E}, \tilde{\delta}_{\rm E})$, we conclude that the inclusion in (30) holds. The inequality in (30) follows from the fact that S-ADMM terminates in line 4 with the condition $\|\hat{v}\|^2 + \hat{\delta} = \|v^{j_{\rm E}}\|^2 + \delta_{j_{\rm E}} \le \rho^2$ satisfied. The first inequality in (31) follows from Lemma 4.5(b) with $i = j_{\rm E}$ and the fact that $\tilde{y}^{\rm E} = y^{j_{\rm E}}$ due to (24). Finally, the second inequality in (31) follows from Lemma 4.5(a) and the fact that $(y^{j_{\rm E}}, q^{j_{\rm E}}) = (\tilde{y}^{\rm E}, \tilde{q}^{\rm E})$ due to (24).

(c) Using the assumption that $c \ge 2\max\{\|q^0\|, \Upsilon(C)\}/\eta$, statement (b) guarantees that S-ADMM outputs $\hat{y} = y^{j_k}$ satisfying $\|A\hat{y} - b\| \le [2\max\{\|q^0\|, \Upsilon(C)\}]/c \le \eta$. Hence, the conclusion that $(\hat{y}, \hat{q}, \hat{v}, \hat{\delta}) = (\tilde{y}^k, \tilde{q}^k, \tilde{v}^k, \tilde{\delta}_k)$ satisfies (3) follows from the previous inequality, the inclusion in (30), and the last inequality in (31). ■

5 The A-ADMM Method

This section describes A-ADMM, the main algorithm of this paper, and its iteration-complexity. In contrast to S-ADMM which keeps the penalty parameter constant, A-ADMM adaptively changes the penalty parameter. The version of A-ADMM presented in this section keeps the prox stepsize constant throughout since it performs multiple calls to the S-ADMM which, as already observed, also has this same attribute. An adaptive variant of A-ADMM with variable prox stepsizes is presented in Section 6.

A-ADMM is formally stated next.

Algorithm 2 A-ADMM

Universal Input: tolerance pair $(\rho, \eta) \in \mathbb{R}^2_{++}$, $\alpha \in [\rho^2, +\infty)$, and $C \in [\rho, +\infty)$
Input: $x^0 \in \mathcal{H}$ and $\gamma^0 = (\gamma^0_1, \ldots, \gamma^0_B) \in \mathbb{R}^B_{++}$
Output: $(\hat{x}, \hat{p}, \hat{v}, \hat{\varepsilon}, \hat{c})$

```
1: $p^0 = (p^0_1, \ldots, p^0_B) \leftarrow (0, \ldots, 0)$ and $c_0 \leftarrow 1/[1 + \|Ax^0 - b\|]$
2: for $\ell \leftarrow 1, 2, \ldots$ do
3:     $(x^\ell, p^\ell, v^\ell, \varepsilon_\ell, \gamma^\ell) = $ S-ADMM$(x^{\ell-1}, p^{\ell-1}, \gamma^{\ell-1}, c_{\ell-1})$
4:     $c_\ell = 2c_{\ell-1}$
5:     if $\|Ax^\ell - b\| \le \eta$ then
6:         $(\hat{x}, \hat{p}, \hat{v}, \hat{\varepsilon}, \hat{c}) = (x^\ell, p^\ell, v^\ell, \varepsilon_\ell, c_\ell)$
7:         return $(\hat{x}, \hat{p}, \hat{v}, \hat{\varepsilon}, \hat{c})$
```

We now make some remarks about A-ADMM. First, even though an initial penalty parameter c_0 is prescribed in line 1 for the sake of analysis simplification, A-ADMM can be equally shown to converge for other choices of c_0. Second, it uses a "warm-start" strategy for calling S-ADMM, i.e., after the first call to S-ADMM, the input of any S-ADMM call is the output of the previous S-ADMM call. Third, Lemma 5.1 below and Theorem 3.2(b) imply that each S-ADMM call in line 3 of A-ADMM generates a quintuple $(x^\ell, p^\ell, v^\ell, \varepsilon_\ell, \gamma^\ell)$ satisfying the first two conditions in (3), but not necessarily the last one, i.e., the feasibility condition which is tested in line 5. To ensure that this condition is also attained, A-ADMM doubles the penalty parameter c (see its line 4) every iteration. Since the first inequality in (31) ensures that $\|Ax^\ell - b\| = \mathcal{O}(1/c_\ell)$, this penalty update scheme guarantees that the test in line 5 will eventually be satisfied, and A-ADMM will terminate with a (ρ,η)-stationary solution of (1)-(2).

Before describing the main result, we define the following constant, and which appear in the total iteration-complexity,

$$\bar\Gamma(x^0;C) := \overline{\phi} - \underline{\phi} + \frac{8\zeta_2\Upsilon^2(C)}{\alpha} + 2\Upsilon^2(C)\left(\frac{4\zeta_1}{\alpha}+9\right)(1+\|Ax^0-b\|), \tag{47}$$

where (ζ_1,ζ_2) is as in (18) and $\Upsilon(C)$ is as in (27).

Recalling that every A-ADMM iteration makes a S-ADMM call, the following result translates the properties of S-ADMM established in Theorem 3.2 to the context of A-ADMM.

Lemma 5.1 *Let ℓ be an iteration index of A-ADMM. Then, the following statements hold:*

(a) the sequences $\{(x^k,p^k,v^k,\varepsilon_k,\gamma^k)\}_{k=1}^{\ell}$ and $\{c_k\}_{k=1}^{\ell}$ satisfy

$$v^k \in \nabla f(x^k) + \partial_{\varepsilon_k} h(x^k) + A^*p^k \quad \textit{and} \quad \max_{1\le k\le \ell}\|v^k\|^2 + \varepsilon_k \le \rho^2, \tag{48}$$

the identity $\gamma^k = \gamma^0$, and the following bounds

$$\max_{1\le k\le\ell}\|p^k\| \le \Upsilon(C) \quad \textit{and} \quad \max_{1\le k\le\ell} c_k\|Ax^k-b\| \le 4\Upsilon(C); \tag{49}$$

(b) the number of iterations performed by the S-ADMM call within the ℓ-th iteration of A-ADMM (see line 3 of A-ADMM) is bounded by

$$\left(\frac{\zeta_1 + c_{\ell-1}\zeta_2}{\rho^2}\right)\bar\Gamma(x^0;C)+1, \tag{50}$$

where $\bar\Gamma(x^0;C)$ is as in (47);

(c) if $c_\ell \ge 4\Upsilon(C)/\eta$ then $(x^\ell,p^\ell,v^\ell,\varepsilon_\ell,\gamma^\ell)$ is a (ρ,η)-stationary solution of problem (1)-(2).

Proof: (a) Using Theorem 3.2(b) with $(y^0,q^0,\lambda^0,c) = (x^{k-1},p^{k-1},\gamma^{k-1},c_{k-1})$ and noting line 3 of A-ADMM, we conclude that for any $k\in\{1,\ldots,\ell\}$, the quintuple $(x^k,p^k,v^k,\varepsilon_k,\gamma^k)$ satisfies (48) and the conditions

$$\gamma^k = \gamma^{k-1}, \quad \|p^k\| \le \max\{\|p^{k-1}\|,\Upsilon(C)\}, \quad c_{k-1}\|Ax^k-b\| \le 2\max\{\|p^{k-1}\|,\Upsilon(C)\}. \tag{51}$$

A simple induction argument applied to both the identity and the first inequality in (51), with the fact that $p^0=0$, show that $\gamma^k=\gamma^0$ and that the first inequality in (49) holds. The second inequality in (51), the assumption that $p^0=0$, the fact that $c_k = 2c_{k-1}$ for every $k\in\{1,\ldots,\ell\}$, and the first inequality in (49), imply that the second inequality in (49) also holds.

(b) Theorem 3.2(a) with $(y^0,q^0,\lambda^0,c) = (x^{\ell-1},p^{\ell-1},\gamma^{k-1},c_{\ell-1})$ imply that the total number of iterations performed by the S-ADMM call within the ℓ-th iteration of A-ADMM is bounded by

$$\left(\frac{\zeta_1 + c_{\ell-1}\zeta_2}{\rho^2}\right)\Gamma(x^{\ell-1},p^{\ell-1};c_{\ell-1})+1,$$

where $\Gamma(\cdot,\cdot;\cdot)$ is as in (27). Thus, to show (50), it suffices to show that $\Gamma(x^{\ell-1},p^{\ell-1};c_{\ell-1}) \le \bar\Gamma(x_0;C)$.

Before showing the above inequality, we first show that $c_{\ell-1}\|Ax^{\ell-1}-b\|^2 \le 16\Upsilon^2(C)/c_0$ for every index ℓ. Indeed, this observation trivially holds for $\ell=1$ due to the fact that $c_0 = 1/(1+\|Ax^0-b\|)\le 1$ (see line 1 of A-ADMM) and the assumption that $\Upsilon(C)\ge 1$. Moreover, the second inequality in (49) and the fact that $c_{\ell-1}\ge c_0$ show that the inequality also holds for $\ell>1$, and thus it holds for any $\ell\ge 1$.

Using the last conclusion, the definition of $\Gamma(x^{\ell-1}, p^{\ell-1}; c_{\ell-1})$, the fact that $c_{\ell-1} \geq c_0$, and the first inequality in (49), we have

$$\begin{aligned}\Gamma(x^{\ell-1}, p^{\ell-1}; c_{\ell-1}) &\leq \overline{\phi} - \underline{\phi} + \frac{16\Upsilon^2(C)}{c_0} + \left[\frac{4\zeta_1}{\alpha c_0} + \frac{1}{c_0} + \frac{4\zeta_2}{\alpha}\right](\|p^{\ell-1}\|^2 + \Upsilon^2(C)) \\ &\overset{(49)}{\leq} \overline{\phi} - \underline{\phi} + \frac{16\Upsilon^2(C)}{c_0} + \left[\frac{4\zeta_1}{\alpha c_0} + \frac{1}{c_0} + \frac{4\zeta_2}{\alpha}\right](2\Upsilon^2(C)) \\ &= \overline{\phi} - \underline{\phi} + \frac{8\zeta_2\Upsilon^2(C)}{\alpha} + \frac{2\Upsilon^2(C)}{c_0}\left(\frac{4\zeta_1}{\alpha} + 9\right) = \bar{\Gamma}(x_0; C),\end{aligned}$$

where the last identity follows from $c_0 = 1/(1 + \|Ax^0 - b\|)$ and the definition of $\bar{\Gamma}(x_0; C)$ in (47).

(c) Assume that $c_\ell \geq 4\Upsilon(C)/\eta$. This assumption, the first inequality in (49), and the fact that $c_\ell = 2c_{\ell-1}$, immediately imply that $c_{\ell-1} \geq 2\max\{\|p^{\ell-1}\|, \Upsilon(C)\}/\eta$. The statement now follows from the previous observation, line 3 of A-ADMM, and Theorem 3.2(c) with $(y^0, q^0, \lambda^0, c) = (x^{\ell-1}, p^{\ell-1}, \gamma^{\ell-1}, c_{\ell-1})$. ■

The next result describes the iteration-complexity of A-ADMM in terms of total ADMM iterations (and hence B-IPP calls within S-ADMM).

Theorem 5.2 (A-ADMM Complexity) *The following statements about A-ADMM hold:*

(a) it obtains a (ρ, η)-stationary solution of (1)-(2) in no more than $\log[1 + 4\Upsilon(C)/(c_0\eta)] + 1$ calls to S-ADMM;

(b) its total number of S-ADMM iterations (and hence B-IPP calls within S-ADMM) is bounded by

$$\frac{8\zeta_2\bar{\Gamma}(x^0; C)\Upsilon(C)}{\rho^2\eta} + \frac{\zeta_2 c_0\bar{\Gamma}(x^0; C)}{\rho^2} + \left[1 + \frac{\zeta_1\bar{\Gamma}(x^0; C)}{\rho^2}\right]\log\left(2 + \frac{8\Upsilon(C)}{c_0\eta}\right) \tag{52}$$

where (ζ_1, ζ_2), $\Upsilon(C)$ and $\bar{\Gamma}(x^0; C)$ are as in (18), (27) and (47), respectively, and c_0 is defined in line 1 of A-ADMM.

Proof: (a) Assume for the sake of contradiction that A-ADMM generates an iteration index $\hat{\ell}$ such that $\hat{\ell} > 1 + \log[1 + 4\Upsilon(C)/(c_0\eta)] > 1$, and hence

$$c_{\hat{\ell}-1} = c_0 2^{\hat{\ell}-1} > c_0\left(1 + \frac{4\Upsilon(C)}{c_0\eta}\right) > \frac{4\Upsilon(C)}{\eta}.$$

Using Lemma 5.1(c) with $\ell = \hat{\ell} - 1 \geq 1$, we conclude that the quintuple $(x^{\hat{\ell}-1}, p^{\hat{\ell}-1}, v^{\hat{\ell}-1}, \varepsilon_{\hat{\ell}-1}, \gamma^{\hat{\ell}-1})$ is a (ρ, η) stationary solution of problem (1)-(2), and hence satisfies $\|Ax^{\hat{\ell}-1} - b\| \leq \eta$. In view of line 5 of A-ADMM, this implies that A-ADMM stops at the $(\hat{\ell} - 1)$-th iteration, which hence contradicts the fact that $\hat{\ell}$ is an iteration index. We have thus proved that (a) holds.

(b) Let $\tilde{\ell}$ denote the total number of S-ADMM calls and observe that $\tilde{\ell} \leq 1 + \log[1 + 4\Upsilon(C)/(c_0\eta)]$ due to (a). Now, using Lemma 5.1(b) and the previous observation, we have that the overall number of iterations performed by S-ADMM is bounded by

$$\begin{aligned}\sum_{\ell=1}^{\tilde{\ell}}\left[\left(\frac{\zeta_1 + c_{\ell-1}\zeta_2}{\rho^2}\right)\bar{\Gamma}(x^0; C) + 1\right] &= \left[1 + \frac{\zeta_1\bar{\Gamma}(x^0; C)}{\rho^2}\right]\tilde{\ell} + \frac{\zeta_2\bar{\Gamma}(x^0; C)}{\rho^2}\sum_{\ell=1}^{\tilde{\ell}} c_{\ell-1} \\ &\leq \left[1 + \frac{\zeta_1\bar{\Gamma}(x^0; C)}{\rho^2}\right]\tilde{\ell} + \frac{c_0\zeta_2\bar{\Gamma}(x^0; C)}{\rho^2}\left(2^{\tilde{\ell}} - 1\right) \\ &\leq \left[1 + \frac{\zeta_1\bar{\Gamma}(x^0; C)}{\rho^2}\right]\tilde{\ell} + \frac{c_0\zeta_2\bar{\Gamma}(x^0; C)}{\rho^2}\left(1 + \frac{8\Upsilon(C)}{c_0\eta}\right).\end{aligned}$$

The result now follows by using that $\tilde{\ell} \leq 1 + \log[1 + 4\Upsilon(C)/(c_0\eta)]$. ■

We now make some comments about Theorem 5.2. First, it follows from Theorem 5.2(a) that the final penalty parameter generated by A-ADMM is $\mathcal{O}(\eta^{-1})$. Second, it follows from Theorem 5.2(a) that A-ADMM ends with a (ρ,η)-stationary solution of (1)-(2) by calling S-ADMM (and hence doubling the penalty parameter) no more than $\mathcal{O}(\log(\eta^{-1}))$ times. Third, under the mild assumption that $\|Ax^0 - b\| = \mathcal{O}(1)$, Theorem 5.2(b) and the fact that $\zeta_2 = 0$ when $B = 1$ (see (18)), imply that the complexity of A-ADMM, in terms of the tolerances only, is:

- $\tilde{\mathcal{O}}(\rho^{-2}\eta^{-1})$ if $B > 1$, and thus $\tilde{\mathcal{O}}(\epsilon^{-3})$;
- $\tilde{\mathcal{O}}(\rho^{-2})$ if $B = 1$, and thus $\tilde{\mathcal{O}}(\epsilon^{-2})$,

where $\epsilon := \min\{\rho, \eta\}$. On the other hand, S-ADMM only achieves the above complexities with (a generally non-computable) $c = \Theta(\eta^{-1})$ and with the condition that $c\|Ax^0 - b\|^2 = \mathcal{O}(1)$ (see the first paragraph following Theorem 3.2), or equivalently, $\|Ax^0 - b\| = \mathcal{O}(\eta^{1/2})$, and hence the initial point x^0 being nearly feasible. Finally, the above complexity for $B = 1$ is similar to those derived for some AL methods (e.g., see [33, Theorem 2.3(b)], [53, Proposition 3.7(a)] and [59, 60]) in terms of tolerance dependencies.

6 S-ADMM and A-ADMM variants with adaptive prox stepsizes

This section outlines S-ADMM and A-ADMM variants with adaptive prox stepsizes, which requires no knowledge of the weakly convexity parameters m_t's. This section contains two subsections. Subsection 6.1 formally describes a variable prox stepsize version of B-IPP, referred to as AB-IPP. Subsection 6.2 presents adaptive prox stepsize versions of S-ADMM and A-ADMM based on AB-IPP (instead of B-IPP) and argues that their iteration-complexities are similar to their corresponding constant stepsizes versions.

6.1 AB-IPP: A variable prox stepsize version of B-IPP

This subsection formally describes AB-IPP and states the main result of this section which describes the main properties of AB-IPP. This result can be viewed as a generalization of Proposition 3.1 to the AB-IPP context. The subsection ends with a proof of Proposition 3.1, which follows as consequence of the main result and the fact (established in this subsection too) that B-IPP is a special case of AB-IPP.

Using the fact that S-ADMM and A-ADMM redundantly included prox stepsizes in their input and output, the description of their adaptive prox stepsize versions now requires minimal changes to the presentation of the previous subsections. Specifically, instead of calling B-IPP, the adaptive prox stepsize version of S-ADMM now calls the subroutine AB-IPP stated below. Moreover, the adaptive prox stepsize version of A-ADMM is the same as Algorithm 2 but with the understanding that line 3 now calls the adaptive prox stepsize version of S-ADMM.

Subroutine AB-IPP

```
Input: $(z, p, \lambda, c) \in \mathcal{H} \times A(\mathbb{R}^n) \times \mathbb{R}^B_{++} \times \mathbb{R}_{++}$
Output: $(z^+, v^+, \delta_+, \lambda^+) \in \mathcal{H} \times \mathbb{R}^l \times \mathbb{R}_{++} \times \mathbb{R}^B_{++}$
1: STEP 1: Block-IPP Iteration
2: for $t = 1, \ldots, B$ do
3:     set $\lambda_t^+ = \lambda_t$
4:     compute a $(1/8; z_t)$-stationary solution $(z_t^+, r_t^+, \varepsilon_t^+)$ of (17) with composite term $\lambda_t^+ h_t(\cdot)$
5:     if $z_t^+$ does NOT satisfy

       $\mathcal{L}_c(z^+_{<t}, z_t, z_{>t}; p) - \mathcal{L}_c(z^+_{<t}, z_t^+, z_{>t}; p) \geq \frac{1}{8\lambda_t^+}\|z_t^+ - z_t\|^2 + \frac{c}{4}\|A_t(z_t^+ - z_t)\|^2$    (53)

       then
6:         $\lambda_t^+ \leftarrow \lambda_t^+/2$ and go to line 4.
7: $z^+ \leftarrow (z_1^+, \ldots, z_B^+)$ and $\lambda^+ \leftarrow (\lambda_1^+, \ldots, \lambda_B^+)$
8: STEP 2: Proceed exactly as in STEP 2 of B-IPP to obtain $(v^+, \delta_+)$
```

We now clarify some aspects of AB-IPP. First, in contrast to B-IPP which outputs λ^+ satisfying $\lambda^+ = \lambda$, AB-IPP has to perform a search for λ_t^+ in the loop consisting of lines 4 to 6, referred to as the t-th AB-IPP loop in our discussion below. Specifically, starting with λ_t^+ set to λ_t, each iteration of the t-th AB-IPP loop halves λ_t^+ and the loop terminates when a prox stepsize λ_t^+ satisfying (53) is generated. Second, it follows from Definition 2.1 that line 4 of AB-IPP yields a triple $(z_t^+, r_t^+, \varepsilon_t^+)$ such that

$$\begin{gathered} r_t^+ \in \nabla \left[\lambda_t^+ \hat{\mathcal{L}}_c(z_{<t}^+, \cdot, z_{>t}; p) + \frac{1}{2}\| \cdot - z_t\|^2 \right](z_t^+) + \partial_{\varepsilon_t^+}(\lambda_t^+ h_t)(z_t^+), \\ \|r_t^+\|^2 + 2\varepsilon_t^+ \le \frac{1}{8}\|z_t^+ - z_t\|^2, \end{gathered} \tag{54}$$

where $\hat{\mathcal{L}}_c(z_{<t}^+, \cdot, z_{>t}; p)$ is defined in (15). Third, the main motivation to enforce condition (53) is that it allows us to show (see Proposition 6.3(b) below) an inequality similar to the one in (33) which, as already observed in the second remark of the paragraph immediately following Proposition 3.1, plays a fundamental role in the analysis of S-ADMM given in Sections 3 and 4.

The next result shows that if the input λ for AB-IPP is as in (21), then B-IPP can be viewed as a special case of AB-IPP.

Lemma 6.1 *Let $t \in \{1, \ldots, B\}$ be given and assume that the prox stepsize λ_t^+ at a certain iteration of the t-th AB-IPP loop satisfies $\lambda_t^+ \in (0, 1/(2m_t)]$. Then, the following statements hold for this t-th loop iteration:*

(a) the smooth part of the objective function of the t-th block subproblem (17) is $(1/2)$-strongly convex;

(b) the triple $(z_t^+, r_t^+, \varepsilon_t^+)$ obtained at the end of this iteration is a $(1/8, z_t)$-stationary solution of (17) which satisfies inequality (53); thus the t-th loop ends at this iteration.

As a consequence, the following statements hold:

(c) if $\lambda_t \in (0, 1/(2m_t)]$, then the t-th AB-IPP loop performs only one iteration and outputs $\lambda_t^+ = \lambda_t$;

(d) if the input λ for AB-IPP is as in (21), then AB-IPP is identical to B-IPP; hence, for every $t \in \{1, \ldots, B\}$, only one loop iteration is performed and the prox stepsize λ^+ output by AB-IPP is equal to λ.

Proof: (a) The assumption that $\lambda_t^+ \in (0, 1/(2m_t)]$ implies that the matrix $B_t := (1 - \lambda_t^+ m_t)I + \lambda_t^+ c A_t^* A_t$ is clearly positive definite, and hence defines the norm $\| \cdot \|_{B_t}$ whose square is

$$\| \cdot \|_{B_t}^2 := \langle \, \cdot \, , \, B_t(\cdot) \, \rangle \ge \lambda_t^+ c \|A_t(\cdot)\|^2 + \frac{1}{2}\| \cdot \|^2. \tag{55}$$

Now, let $\Psi_t(\cdot)$ denote the smooth part of the objective function of the t-th block subproblem (17), i.e.,

$$\Psi_t(\cdot) := \lambda_t^+ \hat{\mathcal{L}}_c(z_{<t}^+, \cdot, z_{>t}; p) + \frac{1}{2}\| \cdot - z_t\|^2 \tag{56}$$

where $\hat{\mathcal{L}}_c(\cdot\,; p)$ is as in (15). Moreover, using assumption (A3), the above definitions of B_t and the norm $\| \cdot \|_{B_t}$, the definitions of $\hat{\mathcal{L}}_c(\cdot\,; \cdot)$ and $\Psi_t(\cdot)$ in (15) and (56), respectively, we easily see that the function $\Psi_t(\cdot) - \frac{1}{2}\| \cdot \|_{B_t}^2$ is convex, and hence $\Psi_t(\cdot)$ is $(1/2)$-strongly convex due to the inequality in (55).

(b) The conclusion that the triple $(z_t^+, r_t^+, \varepsilon_t^+)$ obtained at the end of this iteration is a $(1/8, z_t)$-stationary solution of (17) follows from line 4 of AB-IPP. We now show that (53) holds. Due to (54) and (56), the quadruple $(z_t^+, r_t^+, \varepsilon_t^+, \lambda_t^+)$ satisfies

$$r_t^+ \overset{(54)}{\in} \nabla \Psi_t(z_t^+) + \partial_{\varepsilon_t^+}(\lambda_t^+ h_t)(z_t^+) \subset \partial_{\varepsilon_t^+}\left[\Psi_t + \lambda_t^+ h_t\right](z_t^+)$$

where the set inclusion is due to [22, Thm. 3.1.1 of Ch. XI], and the fact that $\Psi_t(\cdot)$ and $\lambda_t^+ h_t(\cdot)$ are convex functions. Since $\Psi_t(\cdot) - \frac{1}{2}\| \cdot \|_{B_t}^2$ is convex, it follows from Lemma A.4 with $\psi = \Psi_t + \lambda_t^+ h_t$,

$(\xi, \tau, Q) = (1, 1, B_t)$, $(u, y, v) = (z_t, z_t^+, r_t^+)$, and $\eta = \varepsilon_t^+$, that

$$
\begin{aligned}
&\lambda_t^+ \mathcal{L}_c(z_{<t}^+, z_t, z_{>t}; p) - \left[\lambda_t^+ \mathcal{L}_c(z_{<t}^+, z_t^+, z_{>t}; p) + \frac{1}{2}\|z_t^+ - z_t\|^2\right] \\
&= \left[\Psi_t(z_{<t}^+, z_t, z_{>t}; p) + \lambda_t^+ h_t(z_t)\right] - \left[\Psi_t(z_{<t}^+, z_t^+, z_{>t}; p) + \lambda_t^+ h_t(z_t^+)\right] \\
&\geq \frac{1}{4}\|z_t^+ - z_t\|_{B_t}^2 - 2\varepsilon_t^+ + \langle r_t^+, z_t - z_t^+\rangle \overset{(55)}{\geq} \frac{\lambda_t^+ c}{4}\|A_t(z_t^+ - z_t)\|^2 - 2\varepsilon_t^+ + \langle r_t^+, z_t - z_t^+\rangle,
\end{aligned}
$$

where the first equality follows from (4), (15), and (56), the first inequality is due to Lemma A.4, and the last inequality is due to (55). Using the previous inequality, the inequality $ab \leq (a^2 + b^2)/2$ with $(a, b) = (\sqrt{2}\|r_t^+\|, (1/\sqrt{2})\|z_t^+ - z_t\|)$, and the condition on the error (r_t^+, ε_t^+) in (54), we conclude that

$$
\begin{aligned}
&\mathcal{L}_c(z_{<t}^+, z_t, z_{>t}; p) - \mathcal{L}_c(z_{<t}^+, z_t^+, , z_{>t}; p) \\
&\qquad \geq \frac{1}{2\lambda_t^+}\|z_t^+ - z_t\|^2 + \frac{c}{4}\|A_t(z_t^+ - z_t)\|^2 - \frac{1}{\lambda_t^+}\left(\|\sqrt{2}r_t^+\| \left\|\frac{1}{\sqrt{2}}(z_t^+ - z_t)\right\| + 2\varepsilon_t^+\right) \\
&\qquad \geq \frac{1}{2\lambda_t^+}\|z_t^+ - z_t\|^2 + \frac{c}{4}\|A_t(z_t^+ - z_t)\|^2 - \frac{1}{\lambda_t^+}\left(\|r_t^+\|^2 + \frac{1}{4}\|z_t^+ - z_t\|^2 + 2\varepsilon_t^+\right) \\
&\qquad \overset{(54)}{\geq} \frac{1}{2\lambda_t^+}\|z_t^+ - z_t\|^2 + \frac{c}{4}\|A_t(z_t^+ - z_t)\|^2 - \frac{1}{\lambda_t^+}\left(\frac{1}{4} + \frac{1}{8}\right)\|z_t^+ - z_t\|^2 \\
&\qquad = \frac{1}{8\lambda_t^+}\|z_t^+ - z_t\|^2 + \frac{c}{4}\|A_t(z_t^+ - z_t)\|^2,
\end{aligned}
$$

and hence that (53) holds. By the logic of AB-IPP, it then follows that the t-th AB-IPP loop terminates at the current loop iteration with λ_t^+ being the final t-th stepsize output by AB-IPP. We have thus proved that (b) holds.

(c) The assumption that $\lambda_t \in (0, 1/(2m_t)]$ implies that the t-th AB-IPP loop starts with a prox stepsize $\lambda_t^+ \in (0, 1/(2m_t)]$. Hence, (c) follows immediately from statement (b).

(d) We first observe that if the input λ for AB-IPP is as in (21) then it satisfies the assumption of (c). Hence, statement (d) follows immediately from (c). ■

It follows from Lemma 6.1 that, if function f restricted to its t-th block variable is convex, i.e., $m_t = 0$, then it follows from Lemma 6.1(c) that the t-th AB-IPP loop terminates in one iteration with $\lambda_t^+ = \lambda_t$. Hence, AB-IPP does not update λ_t when $m_t = 0$.

The next result shows that any of the AB-IPP loops must terminate.

Lemma 6.2 *For every $t \in \{1, \ldots, B\}$, the t-th AB-IPP loop terminates in at most $1 + \lceil \log(1 + 4m_t\lambda_t) \rceil$ iterations with a $(1/8, z_t)$-stationary solution $(z_t^+, r_t^+, \varepsilon_t^+)$ of (17) satisfying condition (53) and the inequality $\lambda_t^+ \geq \min\{\lambda_t, 1/(4m_t)\}$.*

Proof: Assume for the sake of contradiction that there exists an iteration j for the t-th AB-IPP loop such that $j > 1 + \lceil \log(1 + 4m_t\lambda_t) \rceil$, and hence $j \geq 2$. In view of line 6 of AB-IPP, the stepsize λ_t^+ at the beginning of the $(j-1)$-th loop iteration satisfies

$$
\lambda_t^+ = \frac{\lambda_t}{2^{j-2}} = \frac{2\lambda_t}{2^{j-1}} \leq \frac{2\lambda_t}{4\lambda_t m_t} = \frac{1}{2m_t},
$$

where the inequality follows from the fact that $j - 1 \geq \log(1 + 4m_t\lambda_t)$. In view of the previous inequality, Lemma 6.1(b) implies that inequality (53) holds, and hence that the t-th AB-IPP loop ends at the $(j-1)$-th iteration, a conclusion that contradicts the assumption that j is an iteration of this loop. The conclusion that $\lambda_t^+ \geq \min\{\lambda_t, 1/(4m_t)\}$ follows from Lemma 6.1(b) and the fact that λ_t^+ is halved at every loop iteration for which (53) does not hold. ■

The following result, which is a more general version of Proposition 3.1, describes the main properties of the quadruple $(z^+, v^+, \delta_+, \lambda^+)$ output by AB-IPP.

Proposition 6.3 *Assume that $(z^+, v^+, \delta_+, \lambda^+) = \text{AB-IPP}(z, p, \lambda, c)$ for some $(z, p, \lambda, c) \in \mathcal{H} \times A(\mathbb{R}^n) \times \mathbb{R}_{++}^B \times \mathbb{R}_{++}$. Then the following statements hold:*

(a) there holds

$$\Delta\mathcal{L}_c := \mathcal{L}_c(z;p) - \mathcal{L}_c(z^+;p) \ge \frac{1}{8}\sum_{t=1}^{B} \frac{\|z_t^+ - z_t\|^2}{\lambda_t^+} + \frac{c}{4}\sum_{t=1}^{B} \|A_t(z_t^+ - z_t)\|^2; \tag{57}$$

(b) the quadruple $(z^+, v^+, \delta_+, \lambda^+)$ satisfies

$$\begin{aligned} v^+ &\in \nabla f(z^+) + \partial_{\delta_+} h(z^+) + A^*[p + c(Az^+ - b)],\\ \|v^+\|^2 + \delta_+ &\le \Big[1 + 50\left(\lambda^+_{\min}\right)^{-1} + 48L^2\lambda^+_{\max} + c\zeta_2\Big]\Big[\mathcal{L}_c(z;p) - \mathcal{L}_c(z^+;p)\Big], \end{aligned} \tag{58}$$

where ζ_2 is as in (18), L is as in (19), and

$$\lambda^+_{\min} := \min_{1\le t\le B}\{\lambda_t^+\}, \quad \lambda^+_{\max} := \max_{1\le t\le B}\{\lambda_t^+\}. \tag{59}$$

Proof: (a) We first observe that Lemma 6.2 implies that the t-th AB-IPP loop terminates with a triple $(z_t^+, r_t^+, \varepsilon_t^+)$ satisfying (53). Hence, summing (53) from $t = 1$ to $t = B$, we conclude that (57), and hence statement (a), holds.
(b) We first prove the inclusion in (58). To simplify notation, let $p^+ = p + c(Az^+ - b)$. Using this definition, relation (9) with $(\varepsilon, \beta) = (\varepsilon_t^+, \lambda_t^+)$, we easily see that (54) implies that

$$\begin{aligned} \frac{r_t^+}{\lambda_t^+} &\overset{(54)}{\in} \nabla_{z_t^+} f(z^+_{<t}, z_t^+, z_{>t}) + A_t^*\left[p + c[A(z^+_{<t}, z_t^+, z_{>t}) - b]\right] + \frac{1}{\lambda_t^+}(z_t - z_t^+) + \partial_{(\varepsilon_t^+/\lambda_t^+)} h_t(z_t^+)\\ &= \nabla_{z_t^+} f(z^+_{<t}, z_t^+, z_{>t}) + A_t^*\left(p^+ - c\sum_{s=t+1}^{B} A_s(z_s^+ - z_s)\right) + \frac{1}{\lambda_t^+}(z_t - z_t^+) + \partial_{(\varepsilon_t^+/\lambda_t^+)} h_t(z_t^+), \end{aligned}$$

for every $t \in \{1, \ldots, B\}$. Rearranging the above inclusion and using the definition of v_t^+ (see line 9 of B-IPP), we see that for every $t \in \{1, \ldots, B\}$,

$$v_t^+ \in \nabla_{z_t^+} f(z^+) + \partial_{(\varepsilon_t^+/\lambda_t^+)} h_t(z_t^+) + A_t^* p^+.$$

Now using (10) with $(\varepsilon, \varepsilon_t) = (\delta_+, \varepsilon_t^+/\lambda_t^+)$ for every $t \in \{1, \ldots, B\}$, and $\delta_+ = (\varepsilon_1^+/\lambda_1^+) + \ldots + (\varepsilon_B^+/\lambda_B^+)$ (see line 9 of B-IPP), we have that

$$\partial_{\delta_+} h(z^+) \supset \partial_{(\varepsilon_1^+/\lambda_1^+)} h_1(z_1^+) \times \ldots \times \partial_{(\varepsilon_B^+/\lambda_B^+)} h_B(z_B^+),$$

and we conclude that the inclusion in (58) holds.

We now prove the inequality in (58). Using (54), (57), and that $1/\lambda_t^+ \le (\lambda^+_{\min})^{-1}$ due to (59), we have

$$\begin{aligned} \sum_{t=1}^{B}\left(2\frac{\|r_t^+\|^2}{(\lambda_t^+)^2} + \frac{\varepsilon_t^+}{\lambda_t^+}\right) &\overset{(59)}{\le} [2(\lambda^+_{\min})^{-1} + 1]\sum_{t=1}^{B}\left(\frac{\|r_t^+\|^2 + \varepsilon_t^+}{\lambda_t^+}\right)\\ &\overset{(54)}{\le} [2(\lambda^+_{\min})^{-1} + 1]\sum_{t=1}^{B}\left(\frac{\|z_t^+ - z_t\|^2}{8\lambda_t^+}\right) \overset{(57)}{\le} \left[2\left(\lambda^+_{\min}\right)^{-1} + 1\right]\Delta\mathcal{L}_c. \end{aligned}$$

Defining $D_t := \|v_t^+ - r_t^+/\lambda_t^+\|^2$, using the previous inequality, the definition of δ_+ (see line 9 of B-IPP), and that $\|a + b\|^2 \le 2\|a\|^2 + 2\|b\|^2$, for any $a, b \in \mathbb{R}^n$, we have

$$\|v^+\|^2 + \delta_+ = \sum_{t=1}^{B}\left(\|v_t^+\|^2 + \frac{\varepsilon_t^+}{\lambda_t^+}\right) \le \sum_{t=1}^{B}\left(2D_t + 2\frac{\|r_t^+\|^2}{(\lambda_t^+)^2} + \frac{\varepsilon_t^+}{\lambda_t^+}\right) \le 2\sum_{t=1}^{B} D_t + \left[2(\lambda^+_{\min})^{-1} + 1\right]\Delta\mathcal{L}_c. \tag{60}$$

We will now bound $\sum_{t=1}^{B} D_t$. Using (57) and that $\lambda_s^+ \le \lambda^+_{\max}$ due to (59), we have

$$\|z^+_{>t} - z_{>t}\|^2 = \sum_{s=t+1}^{B}\|z_s^+ - z_s\|^2 \overset{(59)}{\le} \left(\lambda^+_{\max}\sum_{s=t+1}^{B}\frac{\|z_s^+ - z_s\|^2}{\lambda_s^+}\right) \overset{(57)}{\le} 8\lambda^+_{\max}\Delta\mathcal{L}_c. \tag{61}$$

Moreover, it follows from the definitions of D_t given above and v_t (see line 9 of B-IPP), the Cauchy-Schwarz inequality, and assumption (A4), that

$$\begin{aligned} D_t = \left\| v_t^+ - \frac{r_t^+}{\lambda_t^+} \right\|^2 &= \left\| \nabla_{z_t^+} f(z_{\le t}^+, z_{>t}^+) - \nabla_{z_t^+} f(z_{\le t}^+, z_{>t}) + A_t \left(c \sum_{s=t+1}^{B} A_s(z_s^+ - z_s) \right) - \frac{(z_s^+ - z_s)}{\lambda_t^+} \right\|^2 \\ &\le 3 \left(\|\nabla_{z_t^+} f(z_{\le t}^+, z_{>t}^+) - \nabla_{z_t^+} f(z_{\le t}^+, z_{>t})\|^2 + \left(c\|A_t\| \sum_{s=t+1}^{B} \|A_s(z_s^+ - z_s)\| \right)^2 + \frac{\|z_s^+ - z_s\|^2}{(\lambda_t^+)^2} \right) \\ &\overset{(11)}{\le} 3 \left((L_{>t})^2 \|z_{>t}^+ - z_{>t}\|^2 + c^2 \|A_t\|^2 (B-t) \sum_{s=t+1}^{B} \|A_s(z_s^+ - z_s)\|^2 + (\lambda_{\min}^+)^{-1} \frac{\|z_s^+ - z_s\|^2}{\lambda_t^+} \right) \\ &\overset{(57),(61)}{\le} 3 \left\{ \left[8\lambda_{\max}^+ (L_{>t})^2 + 4c\|A_t\|^2 (B-1) \right] \Delta\mathcal{L}_c + \left(\lambda_{\min}^+\right)^{-1} \frac{\|z_s^+ - z_s\|^2}{\lambda_t^+} \right\}. \end{aligned}$$

Summing up the previous inequality from $t = 1$ to $t = B$, using the definitions of L and $\|A\|_\dagger$ as in (19), and using inequality (57), we have

$$\begin{aligned} \sum_{t=1}^{B} D_t &\overset{(19)}{\le} \left[12c\|A\|_\dagger^2 (B-1) + 24\lambda_{\max}^+ L^2 \right] \Delta\mathcal{L}_c + 3 \left(\lambda_{\min}^+\right)^{-1} \sum_{t=1}^{B} \frac{\|z_t^+ - z_t\|^2}{\lambda_t^+} \\ &\overset{(57)}{\le} 12 \left[c\|A\|_\dagger^2 (B-1) + 2\lambda_{\max}^+ L^2 + 2\left(\lambda_{\min}^+\right)^{-1} \right] \Delta\mathcal{L}_c. \end{aligned}$$

Inequality (58) now follows by combining (60) with the previous inequality, and using the definition of ζ_2 as in (18). ■

We now use Lemma 6.1 and Proposition 6.3 to show that Proposition 3.1 holds.

Proof of Proposition 3.1: Assume that the input λ for AB-IPP is as in (21). Lemma 6.1(d) then implies that B-IPP is identical to a version of AB-IPP where only one loop iteration is performed and the prox stepsize λ^+ output by it is equal to λ. Hence, all of the results about AB-IPP also apply to B-IPP. In view of this observation, the inclusion (20) follows immediately from Proposition 6.3(b), and statement (a) follows from Lemma 6.1(a) and the fact that $\lambda_t^+ = \lambda_t \in (0, 1/(2m_t)]$ for all $t \in \{1, \ldots, B\}$, where the inclusion is due to (21). Noting that the definitions of $\lambda_{\min}^+$ and $\lambda_{\max}^+$ in (59), the fact that $\lambda^+ = \lambda$, and relation (21), imply that

$$\left(\lambda_{\min}^+\right)^{-1} = \frac{1}{\min_{1\le t\le B} \lambda_t} = \max_{1\le t\le B} \frac{1}{\lambda_t} \overset{(21)}{=} 2\max\left\{1, \max_{1\le t\le B} m_t\right\}, \qquad \lambda_{\max}^+ = \max_{1\le t\le B} \lambda_t \overset{(21)}{\le} \frac{1}{2}.$$

we then conclude that statement (b) of Proposition 3.1 follows from Proposition 6.3(b) and the definition of ζ_2 in (18). ■

6.2 Adaptive prox stepsize ADMMs

This subsection argues that similar complexity results obtained for S-ADMM can also be derived for its adaptive stepsize analog which, instead of invoking B-IPP in its line 3, calls its adaptive counterpart AB-IPP presented in Subsection 6.1. We refer to this modified S-ADMM as the adaptive prox stepsize S-ADMM.

We start with some remarks about the stepsizes $\{\lambda_t^k\}$ generated by the adaptive prox stepsize S-ADMM. First, a very simple induction applied to Lemma 6.2 shows that the stepsize vector $\lambda^k = (\lambda_1^k, \ldots, \lambda_B^k)$ output by the k-th AB-IPP call within the adaptive prox stepsize S-ADMM satisfies

$$\min_{1\le t\le B} \lambda_t^k \ge \min_{1\le t\le B} \left\{ \lambda_t^0, \frac{1}{4m_t} \right\} =: \underline{\lambda}, \qquad \max_{1\le t\le B} \lambda_t^k \le \max_{1\le t\le B} \lambda_t^0.$$

Moreover, it follows from Proposition 6.3(b) with $(z, p, \lambda, c) = (y^{i-1}, q^{i-1}, \lambda^{i-1}, c)$, the fact that $(y^i, v^i, \delta_i, \lambda^i) =$ AB-IPP$(y^{i-1}, q^{i-1}, \lambda^{i-1}, c)$, and the above two bounds, that the inclusion (32) holds and

$$\|v^i\|^2 + \delta_i \le \left[1 + 50\underline{\lambda}^{-1} + 48L^2 \left(\max_{1\le t\le B} \lambda_t^0 \right) + c\zeta_2 \right] \left[\mathcal{L}_c(y^{i-1}; q^{i-1}) - \mathcal{L}_c(y^i; q^{i-1}) \right]. \qquad (62)$$

Using the observation that all the complexity results for S-ADMM were derived using (33), one can similarly obtain complexity results for the adaptive prox stepsize S-ADMM (and hence for the corresponding adaptive prox stepsize version of A-ADMM) using (62) and a similar reasoning. For example, the complexity of the adaptive prox stepsize S-ADMM is

$$1 + \frac{1}{\rho^2}\left[1 + 50\underline{\lambda}^{-1} + 48L^2\left(\max_{1\le t\le B}\lambda_t^0\right) + c\zeta_2\right]\Gamma(y^0, q^0; c), \tag{63}$$

which is the analog of (29). Similarly, the complexity of the adaptive prox stepsize A-ADMM is

$$\frac{\zeta_2\bar{\Gamma}(x^0;C)}{\rho^2}\left[\frac{8\Upsilon(C)}{\eta} + c_0\right] + \left[1 + \frac{\left(50\underline{\lambda}^{-1} + 48L^2\left(\max_{1\le t\le B}\lambda_t^0\right)\right)\bar{\Gamma}(x^0;C)}{\rho^2}\right]\log\left(2 + \frac{8\Upsilon(C)}{c_0\eta}\right),$$

which is the analog of (52).

Implementation of the t-th AB-IPP loop : Throughout our presentation in this section, we have assumed that a $(1/8, z_t)$-stationary solution of the t-th block subproblem (17) can be obtained in line 4 of AB-IPP. Such an assumption is reasonable if an exact solution z_t^+ of (17) can be computed in closed form since then $(z_t^+, v_t^+, \varepsilon_t^+) = (z_t^+, 0, 0)$ is a $(1/8; z_t)$-stationary solution of (17).

We now discuss the issues of finding a $(1/8, z_t)$-stationary solution of (17) using the ADAP-FISTA described in Appendix B with input $(\mu_0, M_0) = (1/2, \lambda_t c\|A_t\|^2)$, and hence with the same input as in the discussion on the second last paragraph of Subsection 3.1. Recall that in that paragraph, as well as in here, we assume that $\nabla_{x_t} f(x_1, \ldots, x_B)$ is $\tilde{L}_t$-Lipschitz continuous with respect to x_t. If $\lambda_t^+ > 1/(2m_t)$ in line 4, then ADAP-FISTA may not be able to find the required near-stationary solution. This is due to the fact that the smooth part of the objective function of (17) with the above input is not necessarily (1/2)-strongly convex (see Lemma 6.1(c)), which can cause failure of ADAP-FISTA (see Proposition B.1(c) with $\mu_0 = 1/2$). Nevertheless, regardless of whether ADAP-FISTA succeeds or fails, a similar reasoning as in the second last paragraph of Subsection 3.1 shows that it terminates in $\mathcal{O}([\lambda_t^+(\tilde{L}_t + c\|A_t\|^2)]^{1/2})$ iterations. Moreover, failure of ADAP-FISTA signals that the current prox stepsize λ_t^+ is too large. In such a situation, λ_t^+ should be halved regardless of whether (53) is satisfied or not. An argument similar to the one used in the proof of Lemma 6.2 shows that this slightly modified version of AB-IPP terminates in at most $1 + \lceil \log(1 + 4m_t\lambda_t) \rceil$ loop iterations.

7 Numerical Experiments

This section showcases the numerical performance of A-ADMM on three linearly and box constrained, nonconvex (weakly convex) programming problems. Subsections 7.1 and 7.3 focus on a quadratic problem, while Subsection 7.2 focuses on the distributed *Cauchy loss* function [37]. Subsections 7.1 and 7.2 employ fewer blocks, each with a wide dimensional range, whereas Subsection 7.3 uses a large number of one-dimensional blocks. These three proof-of-concept experiments indicate that A-ADMM may not only substantially outperform the relevant benchmarking methods in practice, but also be relatively robust to the relationship between block counts and sizes.

To provide an adequate benchmark for A-ADMM, we compare six algorithmic variants in the tables presented in the following subsections. All variants follow the same core A-ADMM framework, differing only in the use of adaptive or constant prox stepsizes and in the presence or absence of Lagrange multiplier updates:

- A-ADMM-Adapt / A-ADMM-Const: the original method with adaptive or constant prox stepsizes.

- Penalty-Adapt / Penalty-Const: penalty-only variants with no multiplier updates, obtained by removing line 6 and lines 8–13 from S-ADMM.

- v-ADMM-Adapt / v-ADMM-Const: vanilla ADMM variants with multiplier updates at each iteration, implemented by removing lines 8–13 from S-ADMM and adding the equation $q^i = q^{i-1} + c(Ay^i - b)$ at the end of each S-ADMM iteration.

For each of these variants, the total number of iterations ("Iter"), total runtime ("Time"), and the objective value at the solution ("$f+h$") are included in the tables presented in the following subsections. The tables also contain a column labeled "Mults" for the variants A-ADMM-Adapt and A-ADMM-Const, indicating the total number of Lagrange multiplier updates performed by the method. This column is omitted for PENALTY and v-ADMM, since the number of multipliers they perform is naturally clear.

Notably, we attempted to implement the algorithm proposed in [29] using different parameter choices (θ,χ) that theoretically should ensure convergence; see (5) and (7). However, since none of these choices managed to find the desired point within the iteration limit, we omitted them from our benchmarks.

In all three subsections, we assume that the blocks for the generated instances of (1)-(2) have the same size $\bar{n}$, i.e.,

$$\bar{n} = n_1 = \ldots = n_t,$$

and hence that $n = \bar{n}B$. So, the sizes of instances are determined by a triple $(B, \bar{n}, l)$ where l is the number of rows of the constraint matrix A. Instances with $\bar{n} < l$ are usually harder to solve since they further "deviate" from *the last block condition* described in the Introduction (see paragraph following (7)).

For all three problems, the non-smooth component is the indicator function:

$$h_i(x_i) \;=\; \begin{cases} 0 & \text{if } \|x_i\|_\infty \le \omega, \\ +\infty & \text{otherwise,} \end{cases} \qquad i \in \{1,\ldots,B\}, \tag{64}$$

for some $\omega \in \mathbb{R}_{++}$. The parameters $(\omega, B, \bar{n}, l)$ are specified at the beginning of each table. Each matrix $A_i \in \mathbb{R}^{l\times\bar{n}}$, corresponding to the linear constraint in each problem, is filled with i.i.d. standard-normal entries. To define b, we sample $x^b \in [-\omega/2, \omega/2]^{\bar{n}B}$ uniformly at random and set $b = \sum_{i=1}^B A_i x_i^b$. The initial iteration x^0 is drawn uniformly from $[-\omega/4,\ \omega/4]^{\bar{n}B}$. The penalty parameter and the Lagrange multiplier are chosen in accordance with line 1 of A-ADMM, i.e., $c_0 = 1/(1+\|Ax^0-b\|)$ and $p^0 = \mathbf{0}$. Moreover, we fix the input C as $C = 10^3\rho(1+\|\nabla f(x^0)\|)$. Finally, for consistency, all of the variants with adaptive stepsizes were initialized using the same value, setting $\lambda_i^0 = 100$ for each block.

For each of the problems in Subsections 7.1 and 7.2, we solve the corresponding subproblem (17) using the ADAP-FISTA routine described in Appendix B, with $(M_0, \beta, \mu, \chi) = (1, 1.2, 0.5, 0.001)$. The routine terminates once it produces a $(1/8, z_t)$-stationary point. If this criterion is not met, the current stepsize λ_t is halved, and ADAP-FISTA is restarted with the reduced stepsize. Since the subproblems in Subsection 7.3 are all one-dimensional, they are solved exactly without invoking ADAP-FISTA.

All algorithms executed were run for a maximum of $500,000$ iterations. Any algorithm reaching this limit required at least 10 milliseconds to complete. A method is considered to outperform another when it achieves both a lower iteration count and a shorter total runtime.

To ensure timely execution, each algorithm was terminated upon reaching the above iteration limit or upon finding an approximate stationary triple (x^+, p^+, v^+) satisfying the relative error criterion

$$v^+ \in \nabla f(x^+) + \partial h(x^+) + A^* p^+, \qquad \frac{\|v^+\|}{1+\|\nabla f(x^0)\|} \le \rho, \qquad \frac{\|Ax^+ - b\|}{1+\|Ax^0-b\|} \le \eta,$$

with $\rho = \eta = 10^{-5}$.

All experiments were implemented and executed in MATLAB 2024b and run on a macOS machine with an Apple M3 Max chip (14 Cores), and 96 GB of memory. For the sake of brevity, our benchmark only considers randomly generated dense instances. Its main goal is to demonstrate that the ADMMs presented in this work are promising.

7.1 Nonconvex Distributed Quadratic Programming Problem

This subsection studies the performance of A-ADMM-Adapt and A-ADMM-Const against its variants PENALTY-Adapt, PENALTY-Const, v-ADMM-Adapt, and v-ADMM-Const for finding stationary points of a box-constrained, nonconvex block distributed quadratic programming problem with B blocks (DQP).

For a given pair $(l, \bar{n}) \in \mathbb{N}^2_{++}$ with $l < \bar{n}B$, the B-block DQP is formulated as

$$\min_{x=(x_1,\ldots,x_B)\in\mathbb{R}^{\bar{n}B}} \left\{ f(x) = \sum_{i=1}^B \left[\frac{1}{2}x_i^T P_i x_i + \langle x_i, r_i\rangle\right] \;:\; \|x\|_\infty \le \omega \text{ and } \sum_{i=1}^B A_i x_i = b \right\},$$

where $\omega > 0$, $b \in \mathbb{R}^l$, $P_i \in \mathbb{R}^{\bar{n}\times\bar{n}}$ is a symmetric indefinite matrix, $x_i \in \mathbb{R}^{\bar{n}}$, $r_i \in \mathbb{R}^{\bar{n}}$ and $A_i \in \mathbb{R}^{l\times\bar{n}}$, for all $i \in \{1,\ldots,B\}$. It is easy to see that the above DQP instance is a special case of the general formulation (1)-(2).

We now outline the experimental setup used for the DQP problem. First, an orthonormal matrix Q_i is generated using the standard normal distribution. Then, a diagonal matrix D_i is constructed such that one-third of its diagonal entries are set to zero, while the remaining entries are drawn uniformly from the interval $[-10, 10]$, ensuring that at least one of them is negative. Next, the matrix P_i is defined as $P_i = Q_i^\top D_i Q_i$. It is straightforward to verify that if m_i denotes the smallest eigenvalue of D_i, then $m_i < 0$, and the function $f(x_{<i}, \cdot, x_{>i})$ is $|m_i|$-weakly convex. Hence, all variants with constant prox stepsizes set $\lambda_i^0 = 1/(2\max\{1, |m_i|\})$ for $i \in \{1,\ldots,B\}$. Finally, each vector r_i is generated independently, with entries drawn from the standard normal distribution.

The results of the experiments are summarized in Table 1. This table does not include results for the V-ADMM since both its constant and adaptive prox stepsize versions did not converge for any of the instances tested (and hence even for those with only two blocks).

Instance		A-ADMM-Adapt			A-ADMM-Const			Penalty-Adapt			Penalty-Const		
ω	$(B,\bar{n},l)$	Iters / Mult	Time	$f+h$	Iters / Mult	Time	$f+h$	Iters	Time	$f+h$	Iters	Time	$f+h$
100	(2,10,10)	**304** / 15	**0.136**	-9.431e+04	3161 / 14	0.748	-8.019e+04	885	0.219	-9.431e+04	2579	1.153	-1.101e+05
	(2,20,10)	**556** / 16	**0.107**	-8.051e+05	36870 / 18	42.213	-7.289e+05	394706	67.558	-8.006e+05	396830	872.954	-8.006e+05
	(2,20,20)	**2799** / 22	**0.573**	-2.567e+05	6262 / 17	4.280	-2.511e+05	*	*	*	*	*	*
	(2,50,25)	**2694** / 17	**1.089**	-1.238e+06	13298 / 25	46.919	-1.316e+06	*	*	*	*	*	*
	(5,10,10)	**3813** / 30	**0.863**	-8.151e+05	7071 / 30	3.580	-8.151e+05	433420	135.294	-8.151e+05	435685	834.517	-8.151e+05
	(5,20,10)	**3605** / 33	**0.777**	-1.581e+06	4049 / 16	2.437	-1.584e+06	4300	0.947	-1.468e+06	367877	1014.252	-1.501e+06
	(5,20,20)	**2499** / 43	**0.707**	-1.723e+06	6348 / 88	6.828	-1.697e+06	11440	3.156	-1.695e+06	41060	188.430	-1.676e+06
	(5,50,25)	**4982** / 24	**2.935**	-4.639e+06	9913 / 51	33.287	-4.643e+06	*	*	*	*	*	*
	(10,10,10)	**701** / 17	**0.153**	-1.354e+06	6328 / 27	2.084	-1.489e+06	*	*	*	331182	693.230	-1.491e+06
	(10,20,10)	4038 / 15	**1.285**	-4.033e+06	**3715** / 20	2.192	-4.118e+06	154722	89.783	-4.012e+06	232231	927.544	-4.118e+06
	(10,20,20)	**4875** / 18	**2.307**	-3.390e+06	10964 / 18	16.865	-3.390e+06	357554	205.330	-3.379e+06	*	*	*
	(10,50,25)	**1561** / 21	**1.160**	-8.426e+06	25596 / 16	182.247	-8.455e+06	*	*	*	*	*	*
1000	(2,10,10)	**6664** / 29	**0.741**	-3.146e+07	7287 / 26	3.708	-3.146e+07	9220	1.267	-2.958e+07	9947	6.759	-2.958e+07
	(2,20,10)	1585 / 17	0.176	-6.014e+07	4810 / 17	1.619	-6.014e+07	**132**	**0.015**	-6.200e+07	2533	0.342	-6.200e+07
	(2,20,20)	3315 / 18	**0.948**	-2.762e+07	**3174** / 18	1.917	-2.663e+07	358154	358.486	-2.762e+07	4231	7.252	-2.663e+07
	(2,50,25)	**5811** / 21	**1.939**	-1.620e+08	10339 / 27	28.421	-1.590e+08	471301	309.598	-1.600e+08	476761	8972.114	-1.600e+08
	(5,10,10)	**1760** / 20	**0.372**	-5.514e+07	6579 / 19	1.955	-5.514e+07	*	*	*	*	*	*
	(5,20,10)	**1934** / 20	**0.526**	-1.705e+08	8409 / 20	4.278	-1.705e+08	*	*	*	*	*	*
	(5,20,20)	**2057** / 36	**0.564**	-1.297e+08	7943 / 24	9.057	-1.274e+08	*	*	*	*	*	*
	(5,50,25)	**6968** / 20	**4.736**	-5.507e+08	10632 / 27	44.251	-5.641e+08	8251	9.658	-5.647e+08	30110	424.027	-5.625e+08
	(10,10,10)	**2455** / 19	**0.639**	-1.105e+08	4372 / 16	1.142	-1.164e+08	*	*	*	*	*	*
	(10,20,10)	**324** / 14	**0.120**	-2.134e+08	3523 / 18	2.502	-2.045e+08	82486	67.381	-2.134e+08	88201	604.193	-2.045e+08
	(10,20,20)	**2285** / 18	**1.089**	-3.403e+08	11324 / 25	20.784	-3.398e+08	338197	260.461	-3.396e+08	*	*	*
	(10,50,25)	**11369** / 86	**10.996**	-7.901e+08	12063 / 16	36.658	-8.010e+08	196168	367.267	-7.934e+08	228062	8159.721	-8.022e+08
100	(2,20,25)	**1056** / 21	**0.343**	-2.096e+05	18458 / 94	24.491	-1.810e+05	*	*	*	*	*	*
	(2,50,75)	86875 / 35	**30.592**	-6.286e+05	**46026** / 18	307.517	-6.286e+05	*	*	*	*	*	*
	(5,10,15)	**1241** / 15	**0.254**	-4.748e+05	5259 / 19	2.091	-4.747e+05	*	*	*	*	*	*
	(5,50,75)	**27946** / 20	**19.807**	-4.313e+06	34810 / 26	303.214	-4.247e+06	*	*	*	*	*	*
	(10,5,10)	**967** / 15	**0.233**	-3.886e+05	4198 / 21	1.406	-4.077e+05	415410	147.596	-3.850e+05	98031	119.803	-4.077e+05
	(10,50,75)	45452 / 35	**60.585**	-6.830e+06	**25929** / 21	326.712	-7.016e+06	*	*	*	*	*	*
1000	(2,20,25)	**30449** /58	**11.234**	-2.876e+07	15587 / 22	20.564	-2.876e+07	42878	15.526	-3.182e+07	44988	173.040	-3.182e+07
	(5,10,15)	**7555** /19	**1.523**	-4.982e+07	9367 / 20	6.557	-4.981e+07	*	*	*	*	*	*
	(10,5,10)	**798** /22	**0.206**	-8.085e+07	2985 / 33	1.424	-7.333e+07	*	*	*	*	*	*

Bolded values equal to the best algorithm according to iteration count or time. Column "Time" is measured in seconds.

** indicates the algorithm failed to find a stationary point meeting the tolerances by the 500,000th iteration.*

Table 1: Performance of A-ADMM and Penalty variants for the DQP problem.

We now make some remarks about the above numerical results. We begin by comparing the performance of the two A-ADMM variants. Both variants successfully converged on all instances, but A-ADMM-Adapt outperformed A-ADMM-Const on about 93% of the instances tested. We now compare A-ADMM-Adapt against Penalty-Adapt. While A-ADMM-Adapt converged successfully in all test instances, Penalty-Adapt converged in only about 63% of them. Moreover, A-ADMM-Adapt outperformed Penalty-Adapt on about 99% of the test instances.

In summary, the above results shows that A-ADMM-Adapt is better than its constant stepsize counterpart A-ADMM-Const, and A-ADMM-Adapt is better and more stable than Penalty-Adapt.

7.2 Distributed *Cauchy loss* function

This subsection studies the performance of A-ADMM-Adapt and A-ADMM-Const against its variants Penalty-Adapt, Penalty-Const, v-ADMM-Adapt, and v-ADMM-Const for finding stationary points of a box-constrained, nonconvex block distributed *Cauchy loss* function problem with B blocks.

For a given pair $(l, \bar{n}) \in \mathbb{N}^2_{++}$, with $l < \bar{n}B$, the B-block *Cauchy loss* function is formulated as

$$\min_{x=(x_1,\ldots,x_B)\in\mathbb{R}^{\bar{n}B}} \left\{ f(x) = \sum_{i=1}^{B} \frac{\alpha_i^2}{2} \log\left[1 + \left(\frac{y_i - \langle x_i, z_i\rangle}{\alpha_i}\right)^2\right] \;:\; \|x\|_\infty \le \omega \text{ and } \sum_{i=1}^{B} A_i x_i = b \right\},$$

where $\omega > 0$, $b \in \mathbb{R}^l$, $\alpha_i > 0$, $y_i \in \mathbb{R}$, $(z_i, x_i) \in \mathbb{R}^{\bar{n}} \times \mathbb{R}^{\bar{n}}$, and $A_i \in \mathbb{R}^{l\times\bar{n}}$, for all $i \in \{1, \ldots, B\}$. It is not difficult to check that the distributed *Cauchy loss* problem fits within the template defined by (1)-(2).

We now outline the experimental setup used in the above problem. For each $i \in \{1, \ldots, B\}$, the scalar $y_i \in \mathbb{R}$ and the vector $z_i \in \mathbb{R}^{\bar{n}}$ are generated with entries drawn from the standard normal distribution, and the parameter α_i is sampled uniformly at random from the interval $[50, 100]$.

The results of this experiment are summarized in Tables 2 and 3. In both tables, the row labeled "$f+h$" is omitted, as its values consistently ranged from 10^{-12} to 10^{-9} in every instance. Table 2 reports performance on the cases $l \le \bar{n}$, while Table 3 focuses on the more challenging problems with $l > \bar{n}$, reporting only the three best-performing variants: A-ADMM-Adapt, Penalty-Adapt, and v-ADMM-Adapt.

Instance		A-ADMM-Adapt		A-ADMM-Const		Penalty-Adapt		Penalty-Const		v-ADMM-Adapt		v-ADMM-Const	
ω	$(B, \bar{n}, l)$	Iters/Mults	Time	Iters/Mults	Time	Iters	Time	Iters	Time	Iters	Time	Iters	Time
100	(2,10,5)	27 / 4	0.099	11215 / 4	0.517	**26**	**0.032**	11326	0.329	49	0.035	34610	1.392
	(2,10,10)	20 / 3	0.071	18400 / 1	0.732	**19**	**0.034**	18782	0.570	34	0.044	*	*
	(2,15,15)	1597 / 1	94.309	15687 / 1	0.576	1629	94.427	15808	**0.561**	**457**	6.183	36037	1.283
	(2,50,25)	**582** / 1	3.079	30297 / 4	1.413	593	3.079	30597	1.329	1280	**0.952**	*	*
	(2,50,50)	4208 / 2	384.667	185561 / 4	10.494	4282	387.620	188135	**9.882**	**3885**	33.250	*	*
	(5,10,10)	**19** / 3	0.034	5035 / 1	0.390	**19**	**0.032**	5131	0.371	34	10.718	11129	0.890
	(5,15,15)	**24** / 4	0.059	11408 / 1	1.015	**24**	**0.057**	11637	0.954	39	2.001	10422	0.915
	(5,20,10)	8 / 2	0.032	5820 / 1	0.510	**7**	**0.027**	5885	0.479	24	0.061	47553	4.383
	(5,50,25)	**10** / 1	**0.086**	18168 / 2	2.143	11	0.088	18454	2.076	25	0.130	*	*
	(10,10,10)	28 / 3	0.139	4138 / 1	0.679	**27**	**0.103**	4204	0.645	50	13.583	8093	1.418
	(10,15,15)	**15** / 3	0.117	4415 / 4	0.867	**15**	**0.114**	4454	0.822	19	0.127	8442	1.670
	(10,50,50)	18 / 4	0.294	16784 / 1	5.406	**17**	**0.285**	17059	5.256	54	0.486	47590	15.826
1000	(2,10,5)	45 / 4	0.039	74582 / 1	2.374	**44**	**0.036**	75829	2.194	356	0.334	*	*
	(2,10,10)	**7380** / 1	12.966	260746/1	8.657	7514	12.324	264975	**8.058**	44190	134.696	*	*
	(2,15,15)	**3435**/1	62.765	279478/1	10.380	3480	61.982	284659	**9.386**	5647	16.456	*	*
	(2,20,10)	**4711** / 1	0.622	95017 / 1	3.432	4742	**0.610**	95582	3.134	8202	1.088	*	*
	(2,50,50)	**1019** / 1	**2.140**	*	*	1032	2.197	*	*	332326	892.005	*	*
	(5,10,5)	**53** / 1	0.062	27448 / 1	2.083	54	**0.060**	27763	1.969	491	11.362	*	*
	(5,10,10)	**546** / 1	**1.119**	73937 / 1	5.638	557	1.126	71911	5.460	4713	76.192	*	*
	(5,10,15)	**179** / 1	**1.358**	114652 / 4	10.313	186	1.426	115423	9.816	1864	8.114	*	*
	(5,20,10)	63 / 4	0.125	73809 / 1	6.361	**62**	**0.119**	74937	6.036	438	0.835	*	*
	(5,50,50)	**302** / 1	**1.068**	318714 / 1	47.092	305	1.082	323073	46.081	16397	56.792	*	*
	(10,20,10)	**105** / 1	**0.236**	32974 / 1	6.114	107	0.238	33204	5.808	888	1.385	*	*
	(10,50,25)	**42** / 1	**0.453**	116753 / 1	30.268	43	0.455	118212	30.392	733	1.998	*	*
10000	(2,10,5)	991 / 1	0.575	291318 / 1	9.480	**987**	**0.560**	293390	8.802	17610	18.995	*	*
	(2,50,25)	314505 / 1	54.209	*	*	**307661**	**52.388**	*	*	*	*	*	*
	(5,10,5)	801 / 1	1.802	*	*	**799**	**1.794**	*	*	52001	47.409	*	*
	(5,20,10)	415 / 1	0.870	*	*	**414**	**0.866**	*	*	80115	26.152	*	*
	(10,20,10)	**578** / 1	**1.323**	461401 / 1	85.592	588	1.343	466743	82.175	39708	38.514	*	*
	(10,50,25)	1228 / 1	4.773	*	*	**1225**	**4.736**	*	*	157577	319.330	*	*

Bolded values equal to the best algorithm according to iteration count or time. Column "Time" is measured in seconds.

** indicates the algorithm failed to find a stationary point meeting the tolerances by the 500,000th iteration.*

Table 2: Performance of all variants for the *Cauchy loss* problem on instances with $\bar{n} \ge l$.

Instance		A-ADMM-Adapt		Penalty-Adapt		v-ADMM-Adapt	
ω	$(B,\bar{n},l)$	Iters/Mults	Time	Iters	Time	Iters	Time
100	(2,10,15)	**110** / 4	0.272	118	**0.119**	196	6.605
	(2,15,20)	**504** / 5	**12.214**	515	12.512	1096	54.083
	(2,20,25)	**1480** / 6	1246.694	1496	**1231.654**	*	*
	(5,10,15)	27 / 4	0.060	**26**	**0.055**	48	0.441
	(5,15,20)	**42** / 4	0.546	43	0.545	77	**0.501**
	(5,20,25)	17 / 3	0.060	**16**	**0.056**	32	0.112
	(10,10,15)	**105** / 1	**142.722**	111	144.726	641	350.003
	(10,15,20)	**62** / 1	10.162	64	10.182	100	**1.676**
	(10,20,25)	**23** / 1	**0.132**	25	0.143	44	0.311
1000	(2, 10,15)	**368** / 6	3.594	388	**3.569**	63070	221.984
	(2, 15,20)	**26992** / 1	406.725	27571	405.881	88764	**206.515**
	(5, 10,15)	**4656** / 4	640.031	4724	654.362	12499	**424.552**
	(5, 15,20)	**567** / 4	54.419	570	**51.992**	13246	294.429
	(10,10,15)	**58** / 1	**0.257**	60	0.273	216	1.854
	(10, 15,20)	**246** / 1	**0.636**	251	0.656	1030	10.360
10000	(2,10,15)	3163 / 8	4.339	**2771**	**3.823**	32093	26.532
	(5,10,15)	24063 / 1	161.108	**23989**	**158.182**	125086	498.810
	(10,10,15)	**2603** / 1	22.021	2649	**21.950**	59229	150.299

Bolded values equal to the best algorithm according to iteration count or time. Column "Time" is measured in seconds.

** indicates the algorithm failed to find a stationary point meeting the tolerances by the 500,000th iteration.*

Table 3: Performance of the adaptive variants for the *Cauchy loss* problem on instances with $\bar{n} < l$.

We now present some comments about the numerical results. From these tables, we first observe that the adaptive variants outperform their constant prox stepsize counterparts. Moreover, Both A-ADMM-Adapt and Penalty-Adapt converged on all 48 instances, while v-ADMM-Adapt converged on 46. In summary, the tables above show that for the *Cauchy loss* problem, A-ADMM-Adapt and Penalty-Adapt exhibit a similar behavior.

7.3 Nonconvex QP with Box Constraints

This subsection studies the performance of A-ADMM-Adapt and A-ADMM-Const against its variant Penalty-Adapt for solving a general nonconvex quadratic problem with box constraints (QP-BC). The QP-BC problem is formulated as

$$\min_{x=(x_1,\ldots,x_B)\in\mathbb{R}^{\bar{n}B}} \left\{ f(x) = \frac{1}{2}\langle x, Px\rangle + \langle r, x\rangle \ : \ \|x\|_\infty \le \omega \text{ and } Ax = b \right\},$$

where $P \in \mathbb{R}^{B\times B}$ is a symmetric indefinite matrix, $A \in \mathbb{R}^{l\times B}$, $(r,b) \in \mathbb{R}^B \times \mathbb{R}^l$, and $\omega > 0$. In this subsection, we view QP-BC as an extreme special case of (1)-(2) where each variable forms a block, and hence $B = n$ and $\bar{n} = 1$. In this case, the variable blocks correspond to individual coordinates. Consequently, each column of the matrix A defines an A_t matrix for $t \in \{1,\ldots,B\}$.

We now describe how we orchestrated our QP-BC experiments. First, an orthonormal matrix Q is generated using the standard normal distribution. Then, a diagonal matrix D_i is constructed such that one-third of its diagonal entries are set to zero, while the remaining entries are drawn uniformly from the interval $[-10, 10]$, ensuring that at least one of them is negative. Next, the matrix P is defined as $P = Q^\top DQ$. The vector r is generated using the standard normal distribution.

The results of this experiment are summarized in Table 4.

We now make some remarks about the above numerical results. We begin by comparing the performance of the two ADMM variants. Both variants successfully converged on all instances, but A-ADMM-Adapt outperformed A-ADMM-Const on about 63% of the instances tested.

We now compare A-ADMM-Adapt against Penalty-Adapt. While A-ADMM -Adapt converged successfully in all test instances, Penalty-Adapt converged in only about 63% of them. Moreover, A-ADMM-Adapt outperformed Penalty-Adapt on about 99% of the test instances.

Instance		A-ADMM-Adapt				A-ADMM-Const				Penalty-Adapt		
ω	(B,l)	Iters	Time	Mults	$f+h$	Iters	Time	Mults	$f+h$	Iters	Time	$f+h$
1	(50,20)	5122	2.482	18	-7.280e+01	**2935**	**1.422**	24	-7.309e+01	46373	22.298	-7.309e+01
	(50,40)	**24942**	**15.872**	34	-4.231e+01	25363	16.145	34	-4.231e+01	27842	17.646	-4.263e+01
	(100,10)	1903	5.386	37	-1.418e+02	**641**	**1.407**	17	-1.510e+02	3072	9.903	-1.431e+02
	(100,25)	**2739**	**8.347**	19	-2.070e+02	3034	8.676	19	-2.070e+02	*	*	*
	(100,50)	31213	136.013	101	-1.593e+02	**9113**	**39.417**	12	-1.799e+02	54522	227.096	-1.522e+02
	(100,75)	**69340**	377.528	20	-5.108e+01	69743	**373.452**	20	-5.108e+01	*	*	*
10	(50,20)	**2674**	**1.310**	12	-6.293e+03	20546	10.007	20	-6.528e+03	5067	2.455	-6.379e+03
	(50,40)	**27770**	**17.865**	13	-5.477e+02	176863	113.403	23	-6.229e+02	46646	29.852	-4.488e+02
	(100,10)	**446**	**1.110**	17	-1.504e+04	1240	2.799	24	-1.519e+04	15573	34.136	-1.463e+04
	(100,25)	6149	17.607	15	-4.293e+03	**5735**	**16.410**	10	-4.326e+03	*	*	*
	(100,50)	**9170**	**36.270**	21	-6.599e+03	20277	80.017	16	-6.495e+03	*	*	*
	(100,75)	**42857**	**217.284**	41	-5.032e+03	123079	624.127	15	-3.907e+03	*	*	*
100	(50,20)	**2530**	**1.236**	18	-8.944e+05	2567	1.253	17	-9.056e+05	6167	2.977	-8.944e+05
	(50,40)	**14982**	**9.710**	32	-3.017e+05	24800	16.117	26	-3.546e+05	34156	21.889	-3.580e+05
	(100,10)	**2148**	**4.723**	17	-8.557e+05	4787	10.527	18	-7.527e+05	277834	608.748	-7.892e+05
	(100,25)	**3474**	**9.905**	22	-9.911e+05	17908	51.079	48	-9.818e+05	81339	231.611	-9.911e+05
	(100,50)	29850	117.949	45	-8.377e+05	**24711**	**97.683**	23	-9.188e+05	*	*	*
	(100,75)	**19436**	**98.258**	22	-6.714e+05	52056	262.916	94	-6.770e+05	163000	823.867	-4.715e+05
1000	(50,20)	**3143**	**1.665**	19	-2.403e+07	14760	7.152	36	-2.194e+07	*	*	*
	(50,40)	85861	54.568	73	-1.556e+07	**85465**	**54.416**	75	-1.556e+07	*	*	*
	(100,10)	2757	6.260	22	-7.742e+07	**1413**	**3.096**	17	-7.958e+07	358497	784.694	-7.742e+07
	(100,25)	**4987**	**14.265**	45	-2.093e+08	12927	36.879	23	-2.088e+08	*	*	*
	(100,50)	28236	111.511	42	-3.153e+07	41103	162.337	20	-3.135e+07	**18548**	**73.107**	-3.266e+07
	(100,75)	**44632**	**226.221**	22	-2.793e+07	98181	497.447	27	-2.793e+07	*	*	*

Bolded values equal to the best algorithm according to iteration count or time. Column "Time" is measured in seconds.

** indicates the algorithm failed to find a stationary point meeting the tolerances by the 500,000th iteration.*

Table 4: Performance of A-ADMM variants for the nonconvex QP-BC

In summary, the above results confirm the findings of the previous subsections, i.e., A-ADMM-Adapt is better than its constant stepsize counterpart A-ADMM-Adapt, and A-ADMM-Adapt is more stable than Penalty-Adapt.

8 Concluding Remarks

We start by making some remarks about the analysis of this paper. Even though we have only considered proximal ADMMs, our analysis also applies to proximal penalty methods. By taking $C = \rho$ in Algorithm 1, the resulting method will only perform a single Lagrange multiplier (in line 6 of S-ADMM) at the end of each epoch. However, it can be easily seen from our convergence analysis that this last multiplier is not essential and can be removed, which yields a proximal penalty method (and hence, which never perform Lagrange multiplier updates).

We now discuss some possible extensions of our analysis in this paper. First, it would be interesting to develop proximal ADMM variants with more efficient Lagrange multiplier update rules than the one developed in this paper. Second, it would be interesting to develop proximal ADMM variants for composite optimization problems with block constraints given by $\sum_{t=1}^{B} g_t(x_t) \le 0$ where the components of $g_t : \mathbb{R}^{n_t} \to \mathbb{R}^l$ are convex for each $t = 1, \ldots, B$. Finally, this paper assumes that $\operatorname{dom} h$ is bounded (see assumption (A1)) and h restricted to its domain is Lipschitz continuous (see assumption (A2)). It would be interesting to extend its analysis to the case where these assumptions are removed.

A Technical Results for Proof of Lagrange Multipliers

This appendix provides some technical results to show that under certain conditions the sequence of Lagrange multipliers generated by S-ADMM is bounded.

The next two results, used to prove Lemma A.3, can be found in [18, Lemma B.3] and [33, Lemma 3.10], respectively.

Lemma A.1 *Let $A : \mathbb{R}^n \to \mathbb{R}^l$ be a nonzero linear operator. Then,*

$$\nu_A^+ \|u\| \leq \|A^* u\|, \quad \forall u \in A(\mathbb{R}^n).$$

Lemma A.2 *Let h be a function as in (A5). Then, for every $\delta \geq 0$, $z \in \mathcal{H}$, and $\xi \in \partial_\delta h(z)$, we have*

$$\|\xi\| \text{dist}(u, \partial\mathcal{H}) \leq [\text{dist}(u, \partial\mathcal{H}) + \|z - u\|]\, M_h + \langle \xi \, , z - u \rangle + \delta \quad \forall u \in \mathcal{H}$$

where $\partial\mathcal{H}$ denotes the boundary of $\mathcal{H}$.

The following result, whose statement is in terms of the δ-subdifferential instead of the classical subdifferential, is a slight generalization of [53, Lemma B.3]. For the sake of completeness, we also include its proof.

Lemma A.3 *Assume that $b \in \mathbb{R}^l$, linear operator $A : \mathbb{R}^n \to \mathbb{R}^l$, and function $h(\cdot)$, satisfy assumptions (A2) and (A5). If $(q^-, \varrho) \in A(\mathbb{R}^n) \times (0, \infty)$ and $(z, q, r, \delta) \in \text{dom}\, h \times A(\mathbb{R}^n) \times \mathbb{R}^n \times \mathbb{R}_+$ satisfy*

$$r \in \partial_\delta h(z) + A^* q \quad \textit{and} \quad q = q^- + \varrho(Az - b), \tag{65}$$

then we have

$$\|q\| \leq \max\left\{\|q^-\|, \upsilon\left(\|r\| + \delta\right)\right\} \tag{66}$$

where

$$\upsilon(t) := \frac{2D_h M_h + (2D_h + 1)t}{\bar{d}\nu_A^+} \quad \forall t \in \mathbb{R}_+, \tag{67}$$

M_h, $\bar{d} > 0$, and D_h, are as in (A5), (A6), and (12), respectively, ν_A^+ denotes the smallest positive singular value of A.

Proof: We first claim that

$$\bar{d}\nu_A^+ \|q\| \leq 2D_h\left(M_h + \|r\|\right) - \langle q \, , Az - b \rangle + \delta \tag{68}$$

holds. The assumption on (z, q, r, δ) implies that $r - A^* q \in \partial_\delta h(z)$. Hence, using the Cauchy-Schwarz inequality, the definitions of $\bar{d}$ and $\bar{x}$ in (A6), and Lemma A.2 with $\xi = r - A^* q$, and $u = \bar{x}$, we have:

$$\bar{d}\|r - A^* q\| - \left[\bar{d} + \|z - \bar{x}\|\right] M_h \overset{(A.2)}{\leq} \langle r - A^* q \, , z - \bar{x} \rangle + \delta \leq \|r\|\|z - \bar{x}\| - \langle q \, , Az - b \rangle + \delta. \tag{69}$$

Now, using the above inequality, the triangle inequality, the definition of D_h in (12), and the facts that $\bar{d} \leq D_h$ and $\|z - \bar{x}\| \leq D_h$, we conclude that:

$$\bar{d}\|A^* q\| + \langle q \, , Az - b \rangle \overset{(69)}{\leq} \left[\bar{d} + \|z - \bar{x}\|\right] M_h + \|r\|\left(D_h + \bar{d}\right) + \delta \leq 2D_h\left(M_h + \|r\|\right) + \delta. \tag{70}$$

Noting the assumption that $q \in A(\mathbb{R}^n)$, inequality (68) now follows from the above inequality and Lemma A.1.

We now prove (66). Relation (65) implies that $\langle q \, , Az - b \rangle = \|q\|^2/\varrho - \langle q^- \, , q \rangle / \varrho$, and hence that

$$\bar{d}\nu_A^+ \|q\| + \frac{\|q\|^2}{\varrho} \leq 2D_h(M_h + \|r\|) + \frac{\langle q^- \, , q \rangle}{\varrho} + \delta \leq 2D_h(M_h + \|r\|) + \frac{\|q\|}{\varrho}\|q^-\| + \delta, \tag{71}$$

where the last inequality is due to the Cauchy-Schwarz inequality. Now, letting W denote the right hand side of (66) and using (71), we conclude that

$$\left(\bar{d}\nu_A^+ + \frac{\|q\|}{\varrho}\right)\|q\| \overset{(71)}{\leq} \left(\frac{2D_h(M_h + \|r\|) + \delta}{W} + \frac{\|q\|}{\varrho}\right) W \leq \left(\bar{d}\nu_A^+ + \frac{\|q\|}{\varrho}\right) W, \tag{72}$$

and hence that (66) holds. ■

We conclude this section with a technical result of convexity which is used in the proof of Lemma 6.1. Its proof can be found in [42, Lemma A1] but for the sake of completeness a more detailed proof is given here.

Lemma A.4 *Assume that $\xi > 0$, $\psi \in \overline{\operatorname{Conv}}\,(\mathbb{R}^n)$ and positive definite real-valued $n \times n$-matrix Q are such that $\psi - (\xi/2)\|\cdot\|_Q^2$ is convex and let $(y, v, \eta) \in \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}_+$ be such that $v \in \partial_\eta \psi(y)$. Then, for any $\tau > 0$,*

$$\psi(u) \geq \psi(y) + \langle v\,, u - y\rangle - (1+\tau^{-1})\eta + \frac{(1+\tau)^{-1}\xi}{2}\|u - y\|_Q^2 \quad \forall u \in \mathbb{R}^n. \tag{73}$$

Proof: Let $\psi_v := \psi - \langle v\,, \cdot\rangle$. The assumptions imply that ψ_v has a unique global minimum $\bar{y}$ and that

$$\psi_v(u) \geq \psi_v(\bar{y}) + \frac{\xi}{2}\|u - \bar{y}\|_Q^2 \geq \psi_v(y) - \eta + \frac{\xi}{2}\|u - \bar{y}\|_Q^2 \tag{74}$$

for every $u \in \mathbb{R}^n$. The above inequalities with $u = y$ imply that $(\xi/2)\|\bar{y} - y\|_Q^2 \leq \eta$. On the other hand, for any $\tilde{u}, u' \in \mathbb{R}^n$ and $\tau > 0$, it holds

$$\begin{aligned}\|\tilde{u} + u'\|^2 &= \|\tilde{u}\|^2 + \|u'\|^2 + 2\left\langle \frac{1}{\sqrt{\tau}}\tilde{u}\,, \sqrt{\tau}u'\right\rangle \leq \|\tilde{u}\|^2 + \|u'\|^2 + \frac{1}{\tau}\|\tilde{u}\|^2 + \tau\|u'\|^2\\ &= (1+\tau)\|u'\|^2 + (1+\tau^{-1})\|\tilde{u}\|^2\end{aligned}$$

which implies in

$$(1+\tau)^{-1}\|\tilde{u} + u'\|^2 \leq \|u'\|^2 + (1+\tau)^{-1}(1+\tau^{-1})\|\tilde{u}\|^2 = \|u'\|^2 + \tau^{-1}\|\tilde{u}\|^2.$$

Hence, adding and subtracting the term $(\tau^{-1}\xi/2)\|\bar{y} - y\|_Q^2$ in the right hand side of (74) and using the previous inequality with $\tilde{u} = u - \bar{y}$ and $u' = \bar{y} - y$, we obtain that

$$\begin{aligned}\psi_v(u) &\geq \psi_v(y) - \eta - \frac{\tau^{-1}\xi}{2}\|\bar{y} - y\|_Q^2 + \frac{\xi}{2}\left(\tau^{-1}\|y - \bar{y}\|_Q^2 + \|u - \bar{y}\|_Q^2\right)\\ &\geq \psi_v(y) - (1+\tau^{-1})\eta + \frac{(1+\tau)^{-1}\xi}{2}\|u - y\|_Q^2\end{aligned}$$

for every $u \in \mathbb{R}^n$. Hence, (73) follows from the above conclusion and the definition of ψ_v. ■

B ADAP-FISTA algorithm

This appendix section presents an adaptive variant of ACG, called ADAP-FISTA, for solving (13) under the assumption that (B1), (B2), and $\nabla\psi^{(\mathrm{s})}(\cdot)$ is $\tilde{M}$-Lipschitz continuous, i.e.,

$$\|\nabla\psi^{(\mathrm{s})}(z') - \nabla\psi^{(\mathrm{s})}(z)\| \leq \tilde{M}\|z' - z\| \quad \forall z, z' \in \mathbb{R}^n. \tag{75}$$

We would like to emphasize that the notations introduced in this appendix, related to the ADAP-FISTA, are local to this section and should not be confused with those used in previous sections. These choices are made to remain consistent with the original presentation of the algorithm in [53, Appendix A], and they do not carry the same interpretation as in the rest of the paper.

Before formally stating ADAP-FISTA, we give some comments. ADAP-FISTA requires as input an arbitrary positive estimate M_0 for the unknown parameter $\tilde{M}$. Moreover, ADAP-FISTA is a variant of SFISTA [3, 4, 48], which in turn is an ACG variant that solves instances of (13) with $\psi^{(\mathrm{s})}$ strongly convex and that requires the availability of a strong convex parameter for $\psi^{(\mathrm{s})}$. Since ADAP-FISTA is an enhanced version of SFISTA, it also uses as input a good guess μ_0 for what is believed to be a strong convex parameter of $\psi^{(\mathrm{s})}$ (even though such parameter may not exist as $\psi^{(\mathrm{s})}$ is not assumed to be strongly convex). In other words, ADAP-FISTA is used under the belief that $\psi^{(\mathrm{s})}$ is μ_0-strongly convex. If a key test inequality within ADAP-FISTA fails to be satisfied then it stops without finding a $(\sqrt{\sigma}; x_0)$-stationary solution of (13), but reaches the important conclusion that $\psi^{(\mathrm{s})}$ is not μ_0-strongly convex.

We are now ready to present the ADAP-FISTA algorithm below.

ADAP-FISTA Method

Input: $(x_0, M_0, \mu_0, \sigma) \in \mathrm{dom}\,\psi^{(\mathrm{n})} \times \mathbb{R}_{++} \times \mathbb{R}_{++} \times \mathbb{R}_{++}$ such that $M_0 > \mu_0$.

0. Let $\chi \in (0,1)$ and $\beta > 1$ be given, and set $y_0 = x_0$, $A_0 = 0$, $\tau_0 = 1$, and $j = 0$.

1. Set $M_{j+1} = M_j$.

2. Compute

$$a_j = \frac{\tau_j + \sqrt{\tau_j^2 + 4\tau_j A_j (M_{j+1} - \mu_0)}}{2(M_{j+1} - \mu_0)}, \quad \tilde{x}_j = \frac{A_j y_j + a_j x_j}{A_j + a_j},$$

$$y_{j+1} := \underset{v \in \mathrm{dom}\,\psi^{(\mathrm{n})}}{\mathrm{argmin}} \left\{ q_j(v; \tilde{x}_j, M_{j+1}) := \psi^{(\mathrm{s})}(\tilde{x}_j) + \langle \nabla \psi^{(\mathrm{s})}(\tilde{x}_j), v - \tilde{x}_j \rangle + \psi^{(\mathrm{n})}(v) + \frac{M_{j+1}}{2} \|v - \tilde{x}_j\|^2 \right\}. \quad (76)$$

If the inequality

$$\psi^{(\mathrm{s})}(\tilde{x}_j) + \langle \nabla \psi^{(\mathrm{s})}(\tilde{x}_j), y_{j+1} - \tilde{x}_j \rangle + \frac{(1-\chi) M_{j+1}}{2} \|y_{j+1} - \tilde{x}_j\|^2 \geq \psi^{(\mathrm{s})}(y_{j+1}) \quad (77)$$

holds, then go to step 3; else, set $M_{j+1} \leftarrow \beta M_{j+1}$ and repeat step 2.

3. Compute

$$\begin{aligned} A_{j+1} &= A_j + a_j, \quad \tau_{j+1} = \tau_j + a_j \mu_0, \\ s_{j+1} &= (M_{j+1} - \mu_0)(\tilde{x}_j - y_{j+1}), \\ x_{j+1} &= \frac{1}{\tau_{j+1}} \left[\mu_0 a_j y_{j+1} + \tau_j x_j - a_j s_{j+1} \right]. \end{aligned}$$

4. If the inequality

$$\|y_{j+1} - x_0\|^2 \geq \chi A_{j+1} M_{j+1} \|y_{j+1} - \tilde{x}_j\|^2, \quad (78)$$

holds, then go to step 5; otherwise, stop with **failure**.

5. Compute

$$u_{j+1} = \nabla \psi^{(\mathrm{s})}(y_{j+1}) - \nabla \psi^{(\mathrm{s})}(\tilde{x}_j) + M_{j+1}(\tilde{x}_j - y_{j+1}).$$

If the inequality

$$\|u_{j+1}\| \leq \sqrt{\sigma} \|y_{j+1} - x_0\| \quad (79)$$

holds, then stop with **success** and output $(y, u) := (y_{j+1}, u_{j+1})$; otherwise, $j \leftarrow j + 1$ and go to step 1.

We now make some remarks about ADAP-FISTA. First, steps 2 and 3 of ADAP-FISTA appear in the usual SFISTA for solving strongly convex version of (13), either with a static Lipschitz constant (i.e., $M_{j+1} = \tilde{M}$ for all $j \geq 0$), or with adaptive line search for M_{j+1} (e.g., as in step 2 of ADAP-FISTA). Second, the pair (y_{j+1}, u_{j+1}) always satisfies the inclusion in (16) (see [53, Lemma A.3]); hence, if ADAP-FISTA stops successfully in step 5, then the triple $(y_{j+1}, u_{j+1}, 0)$ is a (σ, x_0)-stationary solution of (13), due to (79). Finally, if condition (78) in step 4 is never violated, then ADAP-FISTA must stop successfully in step 5 (see Proposition B.1 below).

The following result describes the main properties of ADAP-FISTA.

Proposition B.1 *Assume that (B1) and (B2) hold and that $\nabla \psi^{(s)}(\cdot)$ is $\tilde{M}$-Lipschitz continuous. Then, the following statements about the* ADAP-FISTA *method with arbitrary input $(x_0, M_0, \mu_0, \sigma) \in \mathrm{dom}\,\psi^{(n)} \times \mathbb{R}_{++} \times \mathbb{R}_{++} \times \mathbb{R}_{++}$ hold:*

(a) it always stops (with either success or failure) in at most

$$\mathcal{O}\left(\sqrt{\frac{\tilde{M}+M_0}{\mu_0}}\log_1^+(\sigma^{-1/2}\tilde{M})\right) \tag{80}$$

iterations/resolvent evaluations;

(b) if it stops successfully with output (y,u), then the triple $(y,u,0)$ is a $(\sqrt{\sigma};x_0)$-stationary solution of (13) *(see Definition 2.1);*

(c) if $\psi^{(\mathrm{s})}(\cdot)$ is μ_0-strongly convex, then ADAP-FISTA *always terminates successfully, and therefore with a $(\sqrt{\sigma};x_0)$-stationary solution of* (13)*.*

We now make some remarks about Proposition B.1. First, if ADAP-FISTA fails then it follows from Proposition B.1(c) that $\psi^{(\mathrm{s})}$ is not $\tilde{\mu}$-strongly convex. Hence, failure of the method sends the message that $\psi^{(\mathrm{s})}$ is not "desirable", i.e., is far from being $\tilde{\mu}$-strongly convex. Second, if ADAP-FISTA successfully terminates (which can happen even if $\psi^{(\mathrm{s})}$ is not $\tilde{\mu}$-strongly convex), then Proposition B.1(b) guarantees that it finds the desired stationary solution. Third, if $\sigma^{-1/2}=\mathcal{O}(1)$ and $M_0=\mathcal{O}(\tilde{M})$, then (80) reduces to $\mathcal{O}((\tilde{M}/\mu_0)^{1/2})$.

References

[1] N. Aybat and G. Iyengar. A first-order smoothed penalty method for compressed sensing. *SIAM J. Optim.*, 21(1):287–313, 2011.

[2] N. Aybat and G. Iyengar. A first-order augmented Lagrangian method for compressed sensing. *SIAM J. Optim.*, 22(2):429–459, 2012.

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[4] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE transactions on image processing*, 18(11):2419–2434, 2009.

[5] D. P. Bertsekas. *Nonlinear programming.* Taylor & Francis, 3ed edition, 2016.

[6] E. Birgin, G. Haeser, and J. M. Martínez. Safeguarded augmented Lagrangian algorithms with scaled stopping criterion for the subproblems. *Computational Optimization and Applications*, 91:491–509, 2025.

[7] S. Boyd, N. Parikh, and E. Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers.* Now Publishers Inc, 2011.

[8] M. T. Chao, Y. Zhang, and J. B. Jian. An inertial proximal alternating direction method of multipliers for nonconvex optimization. *International Journal of Computer Mathematics*, 98(6):1199–1217, 2021.

[9] J. Eckstein and D. P. Bertsekas. On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1):293–318, 1992.

[10] J. Eckstein and M. C. Ferris. Operator-splitting methods for monotone affine variational inequalities, with a parallel application to optimal control. *INFORMS Journal on Computing*, 10(2):218–235, 1998.

[11] J. Eckstein and M. Fukushima. Some reformulations and applications of the alternating direction method of multipliers. In *Large scale optimization*, pages 115–134. Springer, 1994.

[12] J. Eckstein and B. F. Svaiter. A family of projective splitting methods for the sum of two maximal monotone operators. *Mathematical Programming*, 111(1):173–199, 2008.

[13] J. Eckstein and B. F. Svaiter. General projective splitting methods for sums of maximal monotone operators. *SIAM Journal on Control and Optimization*, 48(2):787–811, 2009.

[14] M. I. Florea and S. A. Vorobyov. An accelerated composite gradient method for large-scale composite objective problems. *IEEE Transactions on Signal Processing*, 67(2):444–459, 2018.

[15] D. Gabay. Applications of the method of multipliers to variational inequalities. In *Studies in mathematics and its applications*, volume 15, pages 299–331. Elsevier, 1983.

[16] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & mathematics with applications*, 2(1):17–40, 1976.

[17] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, 9(R2):41–76, 1975.

[18] M. L. N. Goncalves, J. G. Melo, and R. D. C. Monteiro. Convergence rate bounds for a proximal ADMM with over-relaxation stepsize parameter for solving nonconvex linearly constrained problems. *Pacific Journal of Optimization*, 15:379–398, 2019.

[19] D. Hajinezhad and M. Hong. Perturbed proximal primal–dual algorithm for nonconvex nonsmooth optimization. *Math. Program.*, 176:207–245, 2019.

[20] Y. He and R. Monteiro. An accelerated HPE-type algorithm for a class of composite convex-concave saddle-point problems. *SIAM J. Optim.*, 26(1):29–56, 2016.

[21] Y. He and R. D. C. Monteiro. Accelerating block-decomposition first-order methods for solving composite saddle-point and two-player Nash equilibrium problems. *SIAM J. Optim.*, 25:2182–2211, 2015.

[22] J. B. Hiriart-Urruty and C. Lemarechal. *Convex Analysis and Minimization Algorithms II. Advanced Theory and Bundle Methods*. Springer, Berlin, 1993.

[23] M. Hong, Z.-Q. Luo, and M. Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016.

[24] A. Izmailov, M. Solodov, and E. Uskov. Global convergence of augmented Lagrangian methods applied to optimization problems with degenerate constraints, including problems with complementarity constraints. *SIAM Journal on Optimization*, 22:1579–1606, 2012.

[25] Z. Jia, J. Huang, and Z. Wu. An incremental aggregated proximal ADMM for linearly constrained nonconvex optimization with application to sparse logistic regression problems. *Journal of Computational and Applied Mathematics*, 390:113384, 2021.

[26] B. Jiang, T. Lin, S. Ma, and S. Zhang. Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis. *Computational Optimization and Applications*, 72(1):115–157, 2019.

[27] W. Kong. Complexity-optimal and parameter-free first-order methods for finding stationary points of composite optimization problems. *SIAM Journal on Optimization*, 34(3):3005–3032, 2024.

[28] W. Kong and R. D. C. Monteiro. An accelerated inexact dampened augmented Lagrangian method for linearly constrained nonconvex composite optimization problems. *Comput. Optim. Appl.*, 2023.

[29] W. Kong and R. D. C. Monteiro. Global complexity bound of a proximal ADMM for linearly constrained nonseparable nonconvex composite programming. *SIAM Journal on Optimization*, 34(1):201–224, 2024.

[30] W. Kong, J. G. Melo, and R. D. Monteiro. An efficient adaptive accelerated inexact proximal point method for solving linearly constrained nonconvex composite problems. *Computational Optimization and Applications*, 76:305–346, 2020.

[31] W. Kong, J. G. Melo, and R. Monteiro. FISTA and Extensions - Review and New Insights. *Optimization Online*, 2021.

[32] W. Kong, J. G. Melo, and R. D. Monteiro. Iteration complexity of a proximal augmented Lagrangian method for solving nonconvex composite optimization problems with nonlinear convex constraints. *Mathematics of Operations Research*, 48(2):1066–1094, 2023.

[33] W. Kong, J. G. Melo, and R. D. C. Monteiro. Iteration complexity of an inner accelerated inexact proximal augmented Lagrangian method based on the classical Lagrangian function. *SIAM Journal on Optimization*, 33(1):181–210, 2023.

[34] G. Lan and R. D. C. Monteiro. Iteration-complexity of first-order penalty methods for convex programming. *Math. Program.*, 138(1):115–139, 2013.

[35] G. Lan and R. D. C. Monteiro. Iteration-complexity of first-order augmented Lagrangian methods for convex programming. *Math. Program.*, 155(1):511–547, 2016.

[36] Y. Liu, X. Liu, and S. Ma. On the nonergodic convergence rate of an inexact augmented Lagrangian framework for composite convex programming. *Math. Oper. Res.*, 44(2):632–650, 2019.

[37] P.-L. Loh. Statistical consistency and asymptotic normality for high-dimensional robust \$M\$-estimators. *The Annals of Statistics*, 45(2):866 – 896, 2017. doi: 10.1214/16-AOS1471. URL https://doi.org/10.1214/16-AOS1471.

[38] Z. Lu and Z. Zhou. Iteration-complexity of first-order augmented Lagrangian methods for convex conic programming. *SIAM journal on optimization*, 33(2):1159–1190, 2023.

[39] J. Melo, R. D. C. Monteiro, and H. Wang. Iteration-complexity of an inexact proximal accelerated augmented Lagrangian method for solving linearly constrained smooth nonconvex composite optimization problems. *Available on arXiv:2006.08048*, 2020.

[40] J. G. Melo and R. D. C. Monteiro. Iteration-complexity of a linearized proximal multiblock ADMM class for linearly constrained nonconvex optimization problems. *Optimization Online preprint*, 2017.

[41] J. G. Melo and R. D. C. Monteiro. Iteration-complexity of a Jacobi-type non-Euclidean ADMM for multi-block linearly constrained nonconvex programs. *Available on arXiv:1705.07229*, 2017.

[42] J. G. Melo, R. D. Monteiro, and H. Wang. A proximal augmented Lagrangian method for linearly constrained nonconvex composite optimization problems. *Journal of Optimization Theory and Applications*, 202(1):388–420, 2024.

[43] R. Monteiro, Ortiz, and B. F. Svaiter. An adaptive accelerated first-order method for convex optimization. *Comput. Optim. Appl.*, 64:31–73, 2016.

[44] R. D. C. Monteiro and B. F. Svaiter. Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers. *SIAM Journal on Optimization*, 23(1):475–507, 2013.

[45] I. Necoara, A. Patrascu, and F. Glineur. Complexity of first-order inexact Lagrangian and penalty methods for conic convex programming. *Optimization Methods and Software*, 34(2):305–335, 2019.

[46] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical programming*, 140 (1):125–161, 2013.

[47] Y. E. Nesterov. A method of solving a convex programming problem with convergence rate $O\bigl(\frac{1}{k^2}\bigr)$. *Dokl. Akad. Nauk SSSR*, 269(3):543–547, 1983.

[48] Y. E. Nesterov. *Introductory lectures on convex optimization : a basic course.* Kluwer Academic Publ., 2004.

[49] C. Paquette, H. Lin, D. Drusvyatskiy, J. Mairal, and Z. Harchaoui. Catalyst for gradient-based nonconvex optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 613–622. PMLR, 2018.

[50] A. Patrascu, I. Necoara, and Q. Tran-Dinh. Adaptive inexact fast augmented Lagrangian methods for constrained convex optimization. *Optim. Lett.*, 11(3):609–626, 2017.

[51] R. T. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of operations research*, 1(2):97–116, 1976.

[52] A. Ruszczyński. An augmented Lagrangian decomposition method for block diagonal linear programming problems. *Operations Research Letters*, 8(5):287–294, 1989.

[53] A. Sujanani and R. D. C. Monteiro. An adaptive superfast inexact proximal augmented Lagrangian method for smooth nonconvex composite optimization problems. *Journal of Scientific Computing*, 97 (2):34, 2023.

[54] K. Sun and X. A. Sun. Dual descent augmented Lagrangian method and alternating direction method of multipliers. *SIAM Journal on Optimization*, 34(2):1679–1707, 2024.

[55] A. Themelis and P. Patrinos. Douglas–Rachford splitting and ADMM for nonconvex optimization: Tight convergence results. *SIAM Journal on Optimization*, 30(1):149–181, 2020.

[56] Y. Wang, W. Yin, and J. Zeng. Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, 78(1):29–63, 2019.

[57] Y. Xu. Iteration complexity of inexact augmented Lagrangian methods for constrained convex programming. *Mathematical Programming*, 185:199–244, 2021.

[58] J. Zeng, W. Yin, and D. Zhou. Moreau envelope augmented Lagrangian method for nonconvex optimization with linear constraints. *J. Scientific Comp.*, 91(61), 2022.

[59] J. Zhang and Z.-Q. Luo. A proximal alternating direction method of multiplier for linearly constrained nonconvex minimization. *SIAM Journal on Optimization*, 30(3):2272–2302, 2020.

[60] J. Zhang and Z.-Q. Luo. A global dual error bound and its application to the analysis of linearly constrained nonconvex optimization. *SIAM Journal on Optimization*, 32(3):2319–2346, 2022.