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

An adaptive superfast inexact proximal augmented Lagrangian method for smooth nonconvex composite optimization problems

Arnesh Sujanani · Renato D.C. Monteiro

Received: date / Accepted: date

Abstract This work presents an adaptive superfast proximal augmented Lagrangian (AS-PAL) method for solving linearly-constrained smooth nonconvex composite optimization problems. Each iteration of AS-PAL inexactly solves a possibly nonconvex proximal augmented Lagrangian (AL) subproblem obtained by an aggressive/adaptive choice of prox stepsize with the aim of substantially improving its computational performance followed by a full Lagrangian multiplier update. A major advantage of AS-PAL compared to other AL methods is that it requires no knowledge of parameters (e.g., size of constraint matrix, objective function curvatures, etc) associated with the optimization problem, due to its adaptive nature not only in choosing the prox stepsize but also in using a crucial adaptive accelerated composite gradient variant to solve the proximal AL subproblems. The speed and efficiency of AS-PAL is demonstrated through extensive computational experiments showing that it can solve many instances more than ten times faster than other state-of-the-art penalty and AL methods, particularly when high accuracy is required.

Keywords first-order accelerated method · augmented Lagrangian method · smooth weakly convex function · linearly-constrained nonconvex composite optimization · iteration complexity · adaptive method

Mathematics Subject Classification (2010) 90C26 · 90C30 · 65K10 · 90C60 · 47J22

1 Introduction

The main goal of this paper is to present the theoretical analysis and the excellent computational performance of an adaptive superfast proximal augmented Lagrangian method, referred to as AS-PAL, for solving the linearly-constrained smooth nonconvex composite optimization (SNCO) problem

$$\phi^* := \min\{\phi(z) := f(z) + h(z) : Az = b\}, \tag{1.1}$$

where $A : \Re^n \to \Re^l$ is a linear operator, $b \in \Re^l$, $h : \Re^n \to (-\infty, \infty]$ is a closed proper convex function which is M_h-Lipschitz continuous on its compact domain, and $f : \Re^n \to \Re$ is a real-valued differentiable nonconvex function which is m_f-weakly convex and whose gradient is L_f–Lipschitz continuous. AS-PAL is essentially an adaptive version of the IAIPAL method and the NL-IAIPAL method studied in [15, 16], but, in contrast to these methods, it does not require knowledge of the above parameters m_f, L_f, and M_h.

An iteration of AS-PAL has a similar pattern to the ones of the methods in [15, 16] and is also based on the augmented Lagrangian (AL) function $\mathcal{L}_c(z; p)$ defined as

$$\mathcal{L}_c(z; p) := \tilde{\mathcal{L}}_c(z; p) + h(z), \tag{1.2}$$

These authors were partially supported by AFORS Grant FA9550-22-1-0088.

Arnesh Sujanani · Renato D.C. Monteiro
School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332-0205
E-mail: asujanani6@gatech.edu & monteiro@isye.gatech.edu
Versions: v0 (July 24, 2022), v1 (October 3, 2022), v2 (May 13, 2023), v3 (August 31, 2023).

where $\tilde{\mathcal{L}}_c(\cdot;p)$ is the smooth part of (1.2) defined as

$$\tilde{\mathcal{L}}_c(z;p) := f(z) + \langle p, Az-b\rangle + \frac{c}{2}\|Az-b\|^2 \quad \forall z \in \Re^n. \tag{1.3}$$

More specifically, its rough description is as follows: given $(z_{k-1}, p_{k-1}) \in \mathcal{H} \times \Re^l$ and a pair of positive scalars (λ_k, c_k), it computes z_k as a suitable approximate solution of the possibly nonconvex proximal subproblem

$$\min_u \left\{ \lambda_k \mathcal{L}_{c_k}(u; p_{k-1}) + \frac{1}{2}\|u - z_{k-1}\|^2 \right\}, \tag{1.4}$$

and p_k according to the full Lagrange multiplier update

$$p_k = p_{k-1} + c_k(Az_k - b). \tag{1.5}$$

every $\lceil \alpha_k c_k \rceil$ iterations for some $\alpha_k > 0$. Hence, if $\alpha_k = c_k^{-1}$, p_k is updated every single iteration. Based on the fact that (1.4) is strongly convex whenever the prox stepsize λ_k is chosen in $(0, 1/m_f)$, the methods of [15,16] set $\lambda_k = 0.5/m_f$ for every k and solve each strongly-convex subproblem using an accelerated composite gradient (ACG) method (see for example [12,28,30]).

Our contributions: Since it is empirically observed that the larger λ_k is, the faster the procedure outlined above in (1.4)-(1.5) approaches a desired approximate solution of (1.1), AS-PAL adaptively chooses the prox stepsize λ_k to be a scalar which is usually much larger than $0.5/m_f$. As (1.4) may become nonconvex with such a choice of λ_k, a standard ACG method applied to (1.4) may fail to obtain a desirable approximate solution of (1.4). To remedy this situation, AS-PAL uses a new adaptive ACG method for solving (1.4) which accounts for the fact that (1.4) may be nonconvex and the Lipschitz constant of the gradient of the objective function of (1.4) may be unknown. Thus, in contrast to the methods of [15, 16], AS-PAL has the interesting feature of requiring no knowledge of the parameters m_f, L_f and M_h (see the first paragraph above) underlying (1.4) in view of its ability to adaptively generate the prox stepsize λ_k and the estimate of the Lipschitz constant of the gradient of the objective function of (1.4) within the adaptive ACG method. Moreover, as was shown for the method of [15], under the assumption that a Slater point exists, it is also shown that, for any given tolerance pair $(\hat{\rho}, \hat{\eta}) \in \Re^2_{++}$, AS-PAL finds a $(\hat{\rho}, \hat{\eta})$-approximate stationary solution of (1.1), i.e., a triple (z, p, w) satisfying

$$w \in \nabla f(z) + \partial h(z) + A^* p, \quad \|w\| \le \hat{\rho}, \quad \|Az - b\| \le \hat{\eta}, \tag{1.6}$$

in at most $\tilde{\mathcal{O}}(\hat{\eta}^{-1/2}\hat{\rho}^{-2} + \hat{\rho}^{-3})$ (resp., $\tilde{\mathcal{O}}(\hat{\eta}^{-1/2}\hat{\rho}^{-2} + \hat{\rho}^{-2.5})$) iterations if $\alpha_k = c_k^{-1}$ (resp., $\alpha_k = \alpha$ for some $\alpha > 0$) for every $k \ge 1$. Finally, a major advantage of AS-PAL is that it substantially improves the computational performance of the methods in [15,16], whose performance was already substantially better than other existing methods for solving (1.1). Our extensive computational results of section 4 show that AS-PAL can efficiently compute highly accurate solutions for all problems tested, while the other methods can fail to do so in many of these problems. AS-PAL can often find such solutions in just a few seconds or minutes while all the other methods may take several hours to do so.

Literature review. We only focus on relatively recent papers dealing with the iteration complexity of augmented Lagrangian (AL) type methods. In the convex setting, AL-based methods have been widely studied for example in [1, 2, 19, 20, 25, 26, 29, 32, 35].

We now discuss AL type methods in the nonconvex setting of (1.1). Such methods typically perform a Lagrange multiplier update of the form

$$p_k = (1-\theta)(p_{k-1} + \chi_k c_k(Az_k - b))$$

for $\theta \in [0, 1)$ and $\chi_k \in [0, 1]$ at every $k \ge 1$. Various proximal AL methods for solving both linearly and nonlinearly constrained SNCO problems have been studied in [6, 15, 16, 18, 27, 34, 37, 38, 39]. Unlike these methods, AS-PAL is the first universal proximal AL method in that it requires no knowledge of of parameters (e.g., size of constraint matrix, objective function curvatures, etc) associated with the optimization problem. We now highlight some important distinctions between the aforementioned methods. More specifically, [6,18,27] present proximal AL methods based on a perturbed augmented Lagrangian function and an under-relaxed multiplier update. Papers [15,16] present an accelerated proximal AL method based off the classical augmented Lagrangian function and a full multiplier update. The method in [34] is an AL-based method which reverses

the direction of the multiplier update. Papers [37, 38, 39, 40] study AL type variants based on the Moreau envelope. Finally, non-proximal AL methods for solving SNCO problems are studied in [21, 33].

We now discuss papers that are tangentially related to this work. Penalty methods for SNCO problems have been studied in [13, 14, 17, 24]. It is worth mentioning that AS-PAL extends the methods in [15, 16] by allowing for an adaptive prox stepsize, similar to the way the method of [14] extends the one in [13]. Finally, paper [9] studies a penalty-ADMM method that solves an equivalent reformulation of (1.1) while the paper [22] presents an inexact proximal point method applied to the function defined as $\phi(z)$ if z is feasible and $+\infty$ otherwise.

Before closing this literature review, we list the assumptions of some of the above methods in Table 1.1 and give a summary of these methods in Table 1.2, which compares these methods in terms of iteration complexities, necessary conditions, and ranges that parameters θ and χ_k can take.

$\mathcal{B}$	Either (i) the quantity $\sup_{x\in\operatorname{dom} h}\|\phi(x)\|$ is finite, (ii) $\operatorname{dom} h$ is bounded, and/or (iii) the feasible set is bounded.
$\mathcal{A}$	If the constraints have an affine component of the form $Ax = b$ then A has full row rank.
$\mathcal{F}$	There exists some $\nu > 0$ such that $\nu\|Ax_k - b\| \le \operatorname{dist}(0, A^*(Ax_k - b) + c_k^{-1}\partial h(x_k))$ for generated iterates $\{x_k\}_{k\ge1}$ and $\{c_k\}_{k\ge1}$.
$\mathcal{N}$	The function h restricted to its domain is r-Lipschitz continuous.
$\mathcal{SP}$	There exists $\bar{x} \in \operatorname{int}(\operatorname{dom} h)$ such that $Ax = b$.

Table 1.1 Abbreviations for common boundedness and regularity conditions. It has been shown (see [16]) that $\mathcal{N}$ is equivalent to requiring that, for every $x \in \operatorname{dom} h$, there exists $r > 0$ such that that $\partial h(x) \subseteq \mathcal{N}_{\operatorname{dom} h}(x) + \mathcal{B}_r$ where $\mathcal{B}_r = \{x : \|x\| \le r\}$.

Name	Best Complexity	λ_k	θ	χ_k	Conditions
QP-AIPP [13]	$\mathcal{O}(\varepsilon^{-3})$	$\Theta(m_f^{-1})$	-	-	None
R-QP-AIPP [14]	$\tilde{\mathcal{O}}(\varepsilon^{-3})$	$(0,\infty)$	-	-	$\mathcal{B}$
iPPP [23]	$\tilde{\mathcal{O}}(\varepsilon^{-5/2})$	$\mathcal{O}(m_f^{-1})$	-	-	$\mathcal{B},\mathcal{N},\mathcal{SP}$
iALM (2019) [33]	$\tilde{\mathcal{O}}(\varepsilon^{-3})$	-	0	$\mathcal{O}(c_k^{-1})$	$\mathcal{B},\mathcal{F}$
iALM (2020) [21]	$\tilde{\mathcal{O}}(\varepsilon^{-5/2})$	-	0	$\mathcal{O}(c_k^{-1})$	$\mathcal{B},\mathcal{F}$
PProx-PDA[1] [6]	$\mathcal{O}(\theta^{-2}\varepsilon^{-4})$	$\mathcal{O}(\theta L_f^{-1})$	$(0,1)$	1	$\mathcal{B}$, $\mathcal{A}$
θ-IPAAL[2] [27]	$\tilde{\mathcal{O}}(\theta^{-15/4}\varepsilon^{-5/2})$	$\Theta(\theta m_f^{-1})$	$(0,1)$	1	$\mathcal{N},\mathcal{SP}$
IAIPAL [16]	$\tilde{\mathcal{O}}(\varepsilon^{-5/2})$	$\Theta(m_f^{-1})$	0	$\{0,1\}$	$\mathcal{B},\mathcal{N},\mathcal{SP}$
AS-PAL	$\tilde{\mathcal{O}}(\varepsilon^{-5/2})$	$(0,\infty)$	0	$\{0,1\}$	$\mathcal{B},\mathcal{N},\mathcal{SP}$

Table 1.2 Comparison of penalty and AL-based methods with AS-PAL. For convenience, let $\varepsilon = \min\{\hat{\rho},\hat{\eta}\}$, and let $\tilde{\mathcal{O}}(\cdot)$ be the same as $\mathcal{O}(\cdot)$ with all logarithmic dependencies on ε removed.

Organization of the paper. The paper is laid out as follows. Subsection 1.1 presents basic definitions and notation used throughout the paper. Section 2 contains two subsections. The first describes the problem of interest and the assumptions made on it. The second formally states the AS-PAL method and its main complexity result. Section 3 is dedicated to proving the main complexity result. Section 4 presents extensive computational experiments which demonstrate the efficiency of AS-PAL. The Appendix contains two subsections. Appendix A presents the ADAP-FISTA algorithm which is used to solve possibly nonconvex unconstrained subproblems while Appendix B presents technical results which are used to prove that the sequence of the Lagrange multipliers generated by AS-PAL is bounded.

1.1 Basic Definitions and Notations

This subsection presents notation and basic definitions used in this paper.

[1] This method generates prox subproblems of the form $\arg\min_{x\in X}\{\lambda h(x) + c\|Ax - b\|^2/2 + \|x - x_0\|^2/2\}$ and the analysis of [6] makes the strong assumption that they can be solved exactly for any x_0, c, and λ.

[2] It is also shown that conditions $\mathcal{N}$ and $\mathcal{SP}$ can be removed to yield a complexity of $\tilde{\mathcal{O}}(\theta^{-7/2}\varepsilon^{-3})$.

Let $\Re_+$ and $\Re_{++}$ denote the set of nonnegative and positive real numbers, respectively. We denote by $\Re^n$ an n-dimensional inner product space with inner product and associated norm denoted by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$, respectively. We use $\Re^{l\times n}$ to denote the set of all $l\times n$ matrices and $\mathbb{S}^+_n$ to denote the set of positive semidefinite matrices in $\Re^{n\times n}$. The smallest positive singular value of a nonzero linear operator $Q:\Re^n\to\Re^l$ is denoted by ν^+_Q. For a given closed convex set $Z\subset\Re^n$, its boundary is denoted by ∂Z and the distance of a point $z\in\Re^n$ to Z is denoted by $\mathrm{dist}(z,Z)$. The indicator function of Z, denoted by δ_Z, is defined by $\delta_Z(z)=0$ if $z\in Z$, and $\delta_Z(z)=+\infty$ otherwise. For any $t>0$ and $b\geq 0$, we let $\log^+_b(t):=\max\{\log t, b\}$, and we define $\mathcal{O}_1(\cdot)=\mathcal{O}(1+\cdot)$.

The domain of a function $h:\Re^n\to(-\infty,\infty]$ is the set $\mathrm{dom}\, h:=\{x\in\Re^n: h(x)<+\infty\}$. Moreover, h is said to be proper if $\mathrm{dom}\, h\neq\emptyset$. The set of all lower semi-continuous proper convex functions defined in $\Re^n$ is denoted by $\overline{\mathrm{Conv}}\ \Re^n$. The ε-subdifferential of a proper function $h:\Re^n\to(-\infty,\infty]$ is defined by

$$\partial_\varepsilon h(z):=\{u\in\Re^n: h(z')\geq h(z)+\langle u,z'-z\rangle-\varepsilon,\quad\forall z'\in\Re^n\} \tag{1.7}$$

for every $z\in\Re^n$. The classical subdifferential, denoted by $\partial h(\cdot)$, corresponds to $\partial_0 h(\cdot)$. Recall that, for a given $\varepsilon\geq 0$, the ε-normal cone of a closed convex set C at $z\in C$, denoted by $N^\varepsilon_C(z)$, is

$$N^\varepsilon_C(z):=\{\xi\in\Re^n:\langle\xi,u-z\rangle\leq\varepsilon,\quad\forall u\in C\}.$$

The normal cone of a closed convex set C at $z\in C$ is denoted by $N_C(z)=N^0_C(z)$. If ψ is a real-valued function which is differentiable at $\bar z\in\Re^n$, then its affine approximation $\ell_\psi(\cdot,\bar z)$ at $\bar z$ is given by

$$\ell_\psi(z;\bar z):=\psi(\bar z)+\langle\nabla\psi(\bar z),z-\bar z\rangle\quad\forall z\in\Re^n. \tag{1.8}$$

2 The AS-PAL method

This section consists of two subsections. The first one precisely describes the problem of interest and its assumptions. The second one motivates and states the AS-PAL method and presents its main complexity result.

2.1 Problem of Interest

This subsection presents the main problem of interest and discusses the assumptions underlying it.

Consider problem (1.1) where $A:\Re^n\to\Re^l$, $b\in\Re^l$ and functions $f,h:\Re^n\to(-\infty,\infty]$ satisfy the following assumptions:

(C1) $h\in\overline{\mathrm{Conv}}(\Re^n)$ is M_h-Lipschitz continuous on $\mathcal{H}:=\mathrm{dom}\, h$ and the diameter

$$D_h:=\sup\{\|z-z'\|:z,z'\in\mathcal{H}\}$$

of $\mathcal{H}$ is finite;

(C2) A is a nonzero linear operator and there exists $\bar z\in\mathrm{int}\,(\mathcal{H})$ such that $A\bar z=b$;

(C3) f is nonconvex and differentiable on $\Re^n$, and there exists $L_f\geq m_f>0$ such that for all $z,z'\in\Re^n$,

$$\|\nabla f(z')-\nabla f(z)\|\leq L_f\|z'-z\|, \tag{2.1}$$

$$f(z')-\ell_f(z';z)\geq-\frac{m_f}{2}\|z'-z\|^2. \tag{2.2}$$

2.2 The AS-PAL method

This subsection motivates and states the AS-PAL method and presents its main complexity result.

Recall from the introduction that the AS-PAL method, whose goal is to find a $(\hat\rho,\hat\eta)$-approximate stationary solution as in (1.6), is an iterative method which, at its k-th step, computes a stepsize $\lambda_k>0$, an approximate solution z_k of (1.4), and the next multiplier p_k (see the second paragraph of the Introduction).

Before stating AS-PAL, we briefly motivate its steps below. First, as mentioned in the Introduction, AS-PAL allows the prox stepsize λ_k to be large in the sense that it does not have to be in the interval $(0,1/m_f)$ where strong convexity of the objective function of (1.4) is guaranteed. Second, the purpose of steps 1 and 2

of AS-PAL is to adaptively find the aforementioned triple (λ_k, z_k, p_k) using the accelerated gradient method, namely, ADAP-FISTA described in Appendix A, with tentative choices of λ_k. More specifically, if ADAP-FISTA with the current stepsize fails to generate a triple as above, the stepsize is halved and ADAP-FISTA is once again called until it succeeds. Third, AS-PAL checks for successful termination in its step 3 (i.e. it checks if the current iterate is a $(\hat{\rho}, \hat{\eta})$-approximate stationary solution). Finally, step 4 provides a test for when to increase the penalty parameter c_k.

We are now ready to provide a complete description of the AS-PAL method. A detailed discussion of its steps is given in the paragraphs following its description.

AS-PAL Method

Input: functions (f, h), scalars $\sigma \in (0, 1/2)$, $\chi \in (0, 1)$, and $\beta > 1$, an initial prox stepsize $\lambda_0 > 0$, an initial point $z_0 \in \mathcal{H}$, an initial penalty parameter $c_0 > 0$, a sequence of positive integers $\{d_l\}_{l \geq 1}$, and a tolerance pair $(\hat{\rho}, \hat{\eta}) \in \Re^2_{++}$.
Output: a triple (z, p, w) satisfying (1.6).

0. set $k = 1$, $l = 1$, $k_1 = 1$, and

$$p_0 = 0, \quad q_0 = 0, \quad \lambda = \lambda_0, \quad C_\sigma = \frac{2(1-\sigma)^2}{1-2\sigma}, \quad M_0^f = \lambda_0 c_0 \|A\|^2 + 1, \tag{2.3}$$

and define

$$\tilde{c}_l = 2^{l-1} c_0 \quad \forall l \geq 1; \tag{2.4}$$

1. set $c_k = \tilde{c}_l$ and choose $M_k^i \in [1, \overline{M}_k]$ where

$$\overline{M}_k := \max\{M_{k-1}^f, \lambda_0 c_k \|A\|^2 + 1\} \tag{2.5}$$

and call the ADAP-FISTA method described in Appendix A with inputs (χ, β, σ),

$$x_0 = z_{k-1}, \quad (\mu, L_0) = (1/2, M_k^i), \tag{2.6}$$

$$\psi_s = \lambda \tilde{\mathcal{L}}_{c_k}(\cdot, p_{k-1}) + \frac{1}{2}\|\cdot - z_{k-1}\|^2, \quad \psi_n = \lambda h; \tag{2.7}$$

2. if ADAP-FISTA fails or its output (z, u, L) (if it succeeds) does not satisfy the inequality

$$\lambda \mathcal{L}_{c_k}(z_{k-1}, p_{k-1}) - \left[\lambda \mathcal{L}_{c_k}(z, p_{k-1}) + \frac{1}{2}\|z - z_{k-1}\|^2\right] \geq \langle u, z_{k-1} - z\rangle, \tag{2.8}$$

then set $\lambda = \lambda/2$ and go to step 1; else, set $(\lambda_k, M_k^f) = (\lambda, L)$, $(z_k, u_k) = (z, u)$, and

$$w_k := \frac{u_k + z_{k-1} - z_k}{\lambda_k}, \tag{2.9}$$

$$q_k = p_{k-1} + c_k(Az_k - b), \quad p_k = \begin{cases} q_k, & k \equiv k_l \bmod d_l, \\ p_{k-1}, & \text{otherwise}, \end{cases} \tag{2.10}$$

and go to step 3;
3. if $\|w_k\| \leq \hat{\rho}$ and $\|Az_k - b\| \leq \hat{\eta}$, then stop with success and output $(z, q, w) = (z_k, q_k, w_k)$; else, go to step 4;
4. if $k \geq k_l + 1$ and

$$\Delta_k := \frac{1}{\sum_{i=k_l+1}^k \lambda_i}\left[\mathcal{L}_{c_k}(z_{k_l}, p_{k_l}) - \mathcal{L}_{c_k}(z_k, p_k)\right] \leq \max\left\{\frac{\sum_{i=k_l+1}^k \lambda_i \|w_i\|^2}{2C_\sigma \sum_{i=k_l+1}^k \lambda_i}, \frac{\hat{\rho}^2}{2C_\sigma}\right\}, \tag{2.11}$$

then set $k_{l+1} = k + 1$ and $l \leftarrow l + 1$;
5. set $k \leftarrow k + 1$, and go to step 1.

We now introduce some basic terminology, definitions and remarks about AS-PAL. First, AS-PAL makes two types of iterations, namely, the outer iterations indexed by k and the ACG iterations performed within each ADAP-FISTA call in step 1. Second, it follows from Proposition A.1(a) that the total number of resolvent evaluations[3] made by ADAP-FISTA is on the same order of magnitude as its total number of ACG iterations. Third, define the l-th cycle $\mathcal{C}_l$ as

$$\mathcal{C}_l = \{k_l, \ldots, k_{l+1} - 1\} \quad \forall l \geq 1, \tag{2.12}$$

where k_l are the indices computed in step 4 of AS-PAL. The goal of the test (2.11) is to decide when to start a new cycle. The test at the beginning of step 4 and the definition of $k_l + 1$ in step 4 ensure that $k_{l+1} - k_l \geq 2$ and hence that every cycle has at least two indices. The definition of $\tilde{c}_l$ in step 0 and the update rule for c_k in step 1 imply that

$$\mathcal{C}_l := \{k \geq 1 : c_k = \tilde{c}_l\} \quad \forall l \geq 1. \tag{2.13}$$

Fourth, it is shown in Proposition 3.1(b) below that the output λ_k and (z_k, u_k) in step 2 of AS-PAL satisfies

$$\|u_k\| \leq \sigma \|z_k - z_{k-1}\|, \quad u_k \in \lambda_k \left[\nabla_z \tilde{\mathcal{L}}_{c_k}(z_k; p_{k-1}) + \partial h(z_k)\right] + z_k - z_{k-1} \tag{2.14}$$

where σ is part of the input of AS-PAL. Since (2.14) with $\sigma = 0$ reduces to the optimality condition for (1.4), (z_k, u_k) can be viewed as an approximate stationary solution of (1.4) where the residual u_k is relaxed from being zero to a quantity that is now relatively bounded as in (2.14). Finally, it is shown in Proposition 3.1 below that the triple $(z, q, w) = (z_k, q_k, w_k)$ computed in step 2 satisfies the inclusion in (1.6) for every $k \geq 1$. As a consequence, if AS-PAL terminates in step 3, then the triple (z, q, w) output by this step is a $(\hat{\rho}, \hat{\eta})$-approximate solution of (1.1).

The goal of steps 1 and 2 is to obtain a prox stepsize $\lambda_k \leq \lambda_{k-1}$ and a pair (z_k, u_k) satisfying (2.8) and (2.14). This is done by successively calling ADAP-FISTA with inputs given by (2.6) and (2.7), where the first call is done with $\lambda = \lambda_{k-1}$ and subsequent ones with λ equal to the previous prox stepsize divided by two. Since the condition that $\lambda \leq 0.5/m_f$ implies that ψ_s in (2.7) is 1/2-strongly convex, it follows from Proposition A.2 that an ADAP-FISTA call with such λ (and input as in (2.6) and (2.7)) always terminates successfully with a pair $(z_k, u_k) = (z, u)$ satisfying both (2.8) and (2.14). Thus, the above adaptive procedure must eventually terminate. Even though some ADAP-FISTA calls may fail during the above procedure due to λ being potentially large, its major appeal is that it generates a sequence of prox stepsizes $\{\lambda_k\}$ which are much larger than the conservative sequence where $\lambda_k = 0.5/m_f$ for every $k \geq 1$. This feature seems to be a strong contributor to the excellent practical performance of AS-PAL as demonstrated by our computational results in Section 4.

AS-PAL as stated above is not fully determined as it does not specify how to choose M_k^i in step 1 and the sequence of positive integers $\{d_l\}_{l \geq 1}$ used in update (2.10). We now specify possible choices for these two quantities. Step 1 provides a wide range of values to choose M_k^i from, namely, $[1, \overline{M}_k]$ where $\overline{M}_k$ is as in (2.5). Our implementation of AS-PAL in section 4 sets M_k^i equal to M_{k-1}^f which, in view of (2.5), is clearly in the above interval. Two possible choices for the sequence $\{d_l\}_{l \geq 1}$ are described in Lemma 2.1 below and the complexities of AS-PAL with these choices of d_l are given in the paragraph following Theorem 2.2. The complexity established in this result holds though for an arbitrary sequence of positive integers $\{d_l\}$.

It is interesting to note that AS-PAL is a universal method in the sense that it does not require any knowledge of the parameters m_f, L_f, and M_h associated with (f, h). Moreover, if M_k^i is chosen as above, it does not even require the computation of $\|A\|$, and hence of the maximum eigenvalue of AA^*.

In the remaining part of this section, we state the main complexity result for AS-PAL, whose proof is the main focus of Section 3. Before stating the main result, we first introduce the following quantities:

$$\phi_* := \inf_{z \in \Re^n} \phi(z), \quad \bar{d} := \operatorname{dist}(\bar{z}, \partial\mathcal{H}), \quad \underline{\lambda} := \min\{\lambda_0, 1/(4m_f)\} \tag{2.15}$$

$$\nabla_f := \sup_{z \in \mathcal{H}} |\nabla f(z)|, \quad \kappa_p := \frac{2D_h(M_h + \nabla_f + \underline{\lambda}^{-1}(1+\sigma)D_h)}{\bar{d}\nu_A^+} \tag{2.16}$$

$$S := \sup_{z \in \mathcal{H}} |\phi(z)|, \quad \kappa_d := S + \frac{9\kappa_p^2}{2c_0} - \phi_*, \tag{2.17}$$

[3] A resolvent evaluation of h is an evaluation of $(I + \gamma\partial h)^{-1}(\cdot)$ for some $\gamma > 0$.

where λ_0, c_0, and σ are input parameters for AS-PAL, (m_f, L_f) are as in (C3), $\bar{z}$ is as in (C2), M_h is as in (C1), D_h is as in (C1), and ν_A^+ is as in Subsection 1.1. Note that assumptions (C1) and (C3) imply that S and ∇_f are finite.

Before stating the main result for AS-PAL, we introduce a quantity $\hat{c}(\hat{\rho}, \hat{\eta})$ which is used to express the complexity of AS-PAL. Indeed, define $\hat{c}(\hat{\rho}, \hat{\eta})$ as the smallest $\tilde{c}_l$ such that

$$\tilde{c}_l \geq \max\left\{\frac{2\kappa_p}{\hat{\eta}}, \frac{8C_\sigma\kappa_p^2}{d_l\underline{\lambda}\hat{\rho}^2}\right\}, \tag{2.18}$$

where $\tilde{c}_l$ is as in (2.4), and C_σ, $\underline{\lambda}$, and κ_p, are as in (2.3), (2.15), and (2.16), respectively.

The following simple result describes useful properties about $\hat{c}$ for two choices of $\{d_l\}$.

Lemma 2.1 *The following statements hold:*

(a) if $d_l = 1$ for all $l \geq 1$, then

$$\hat{c}(\hat{\rho}, \hat{\eta}) \leq \max\left\{\frac{4\kappa_p}{\hat{\eta}}, \frac{16C_\sigma\kappa_p^2}{\underline{\lambda}\hat{\rho}^2}, c_0\right\}; \tag{2.19}$$

(b) if, for some $\alpha > 0$, $d_l = \lceil \alpha\tilde{c}_l \rceil$ for all $l \geq 1$, then

$$\hat{c}(\hat{\rho}, \hat{\eta}) \leq \max\left\{\frac{4\kappa_p}{\hat{\eta}}, \frac{4\sqrt{2C_\sigma\kappa_p^2/(\underline{\lambda}\alpha)}}{\hat{\rho}}, c_0\right\}. \tag{2.20}$$

Proof For simplicity, denote $\hat{c}(\hat{\rho}, \eta)$ by $\hat{c}$.

(a) The proof of (a) is obvious.

(b) If $\hat{c} = \tilde{c}_1$, then $\hat{c} = c_1 = c_0$, and hence (2.20) holds. Otherwise, in view of the definition of $\hat{c}$, it follows that $\tilde{c}_l = \hat{c}/2$ violates (2.18), i.e.,

$$\hat{c} < 2\max\left\{\frac{2\kappa_p}{\hat{\eta}}, \frac{8C_\sigma\kappa_p^2}{\underline{\lambda}\hat{\rho}^2\,\lceil\alpha(\hat{c}/2)\rceil}\right\},$$

due to the assumption that $d_l = \lceil \alpha\tilde{c}_l \rceil$. Relation (2.20) now immediately from the above inequality. ∎

Thus, it follows from the above result that, in terms of the tolerances only, $\hat{c}(\hat{\rho}, \hat{\eta})$ is

$$\begin{aligned} &\mathcal{O}_1(\hat{\rho}^{-1} + \hat{\eta}^{-1}) \quad \text{if } d_l = \Theta(\tilde{c}_l), \\ &\mathcal{O}_1(\hat{\rho}^{-2} + \hat{\eta}^{-1}) \quad \text{if } d_l = \Theta(1). \end{aligned}$$

The following result describes the ACG iteration/resolvent evaluation complexity for AS-PAL.

Theorem 2.2 *Let a tolerance pair $(\hat{\rho}, \hat{\eta}) \in \Re^2_{++}$ be given and consider the quantity $\hat{c}(\hat{\rho}, \hat{\eta})$ defined just before Lemma 2.1. Moreover, assume that the initial prox stepsize λ_0 of AS-PAL satisfies*

$$\lambda_0 = \Omega(m_f^{-1}), \quad \log_0^+(m_f\lambda_0) \leq \mathcal{O}(1 + \kappa_d/(\underline{\lambda}\hat{\rho}^2)),$$

where m_f is as in (C3), $\underline{\lambda}$ is as in (2.15), and κ_d is as in (2.17). Then, AS-PAL outputs a $(\hat{\rho}, \hat{\eta})$-approximate stationary solution of (1.1) in

$$\mathcal{O}\left(\left[1 + \frac{m_f\kappa_d}{\hat{\rho}^2}\right]\sqrt{\mathcal{M}_{\lambda_0}(\hat{c})}\left[\log_1^+\left(\mathcal{M}_{\lambda_0}(\hat{c}) + \frac{\hat{c}}{c_0}\right)\right]^2\right) \tag{2.21}$$

ACG iterations/resolvent evaluations, where $\hat{c} := \hat{c}(\hat{\rho}, \hat{\eta})$ and

$$\mathcal{M}_\lambda(c) := 1 + \lambda(L_f + c\|A\|^2) \quad \forall c, \lambda \in \Re. \tag{2.22}$$

Before ending this subsection, we specialize bound (2.21) for the two choices of sequences $\{d_l\}$ as in Lemma 2.1. Indeed, Lemma 2.1 and the definition of $\mathcal{M}_\lambda(c)$ in (2.22) imply that bound (2.21) reduces to

$$\begin{aligned} &\tilde{\mathcal{O}}_1(\hat{\rho}^{-5/2} + \hat{\rho}^{-2}\hat{\eta}^{-1/2}) \quad \text{if } d_l = \Theta(\tilde{c}_l), \\ &\tilde{\mathcal{O}}_1(\hat{\rho}^{-3} + \hat{\rho}^{-2}\hat{\eta}^{-1/2}) \quad \text{if } d_l = \Theta(1). \end{aligned}$$

Even though setting $d_l = 1$ results in a slightly worse complexity, we have observed that performing a Lagrange multiplier update at every iteration results in more efficient computational performance.

3 Proof of Theorem 2.2

This section is dedicated to proving Theorem 2.2.

The first proposition below shows that, in every iteration of AS-PAL, the loop within steps 1 and 2 always stops and shows key properties of its output.

Proposition 3.1 *The following statements about AS-PAL hold for every $k \geq 1$:*

(a) the function ψ_s in (2.7) has $\mathcal{M}_{\lambda_0}(c_k)$-Lipschitz continuous gradient everywhere on $\Re^n$, and hence $\bar{L} = \mathcal{M}_{\lambda_0}(c_k)$ satisfies (A.2);

(b) the loop within steps 1 and 2 of its k-th iteration always ends and the output $(z_k, u_k, w_k, q_k, p_k, \lambda_k, M_k^f)$ obtained at the end of step 2 satisfies (2.14) and

$$\left(\frac{1-\sigma}{\sigma}\right)\|u_k\| \leq \|\lambda_k w_k\| \leq (1+\sigma)\|z_k - z_{k-1}\|; \tag{3.1}$$

$$\lambda_k \mathcal{L}_{c_k}(z_{k-1}, p_{k-1}) - \left[\lambda_k \mathcal{L}_{c_k}(z_k, p_{k-1}) + \frac{1}{2}\|z_k - z_{k-1}\|^2\right] \geq \langle u_k, z_{k-1} - z_k\rangle; \tag{3.2}$$

$$w_k \in \nabla f(z_k) + \partial h(z_k) + A^* q_k; \tag{3.3}$$

$$M_k^f \leq \max\{M_k^i, \omega \mathcal{M}_{\lambda_0}(c_k)\}; \tag{3.4}$$

$$\lambda_0 \geq \lambda_k \geq \underline{\lambda}, \tag{3.5}$$

where $\omega = 2\beta/(1-\chi)$, λ_0 is the initial prox stepsize, and $\underline{\lambda}$ is as in (2.15); moreover, every prox stepsize λ generated within the aforementioned loop is in $[\underline{\lambda}, \lambda_0]$.

Proof (a) Using inequality (2.1) and the definition of $\mathcal{L}_{c_k}(\cdot; p_{k-1})$ in (1.2) we easily see that its smooth part, namely, $\mathcal{L}_{c_k}(\cdot; p_{k-1}) - h(\cdot)$, has $(L_f + c_k\|A\|^2)$-Lipschitz continuous gradient everywhere on $\Re^n$. Hence, in view of the definitions of ψ_s in (2.7) and $M_\lambda(\cdot)$ in (2.22), and the fact that $\lambda \leq \lambda_0$, we conclude that (a) holds.

(b) We first claim that if the loop consisting of steps 1 and 2 of the k-iteration of AS-PAL stops, then (2.14), (3.1), (3.2), (3.3), and (3.4) hold. Indeed, assume that the loop consisting of steps 1 and 2 of the k-th iteration of AS-PAL stops. It then follows from the logic within step 1 and 2 of AS-PAL that the last ADAP-FISTA call within the loop stops successfully and (3.2) holds. Since (a) implies that $\bar{L} = \mathcal{M}_{\lambda_0}(c_k)$ satisfies relation (A.2), it follows Proposition A.1(b) with (ψ_s, ψ_n) as in (2.7), $x_0 = z_{k-1}$, and $L_0 = M_k^i$ that the triple $(z_k, u_k, M_k^f) = (y, u, L)$ satisfies the inequality in (2.14), relation (3.4), and the inclusion in (2.14)

$$u_k \in \lambda_k[\nabla f(z_k) + \partial h(z_k) + A^* q_k] + z_k - z_{k-1}.$$

Now, using the definition of w_k in (2.9), we easily see that the above inclusion is equivalent to (3.3) and that the inequality in (2.14) together with the triangle inequality for norms imply the two inequalities in (3.1).

We now claim that if step 1 is performed with a prox stepsize $\lambda \leq 1/(2m_f)$ in the k-th iteration, then for every $j > k$, we have that $\lambda_{j-1} = \lambda$ and the j-th iteration performs step 1 only once. To show the claim, assume that $\lambda \leq 1/(2m_f)$. Using this assumption, the definition of $\mathcal{L}_c$ in (1.2), and the assumption (2.2) that f is m_f-weakly convex, we see that the function ψ_s in (2.7) is strongly convex with modulus $1 - \lambda m_f \geq 1/2$. Since each ACG call is performed in step 1 of AS-PAL with $\mu = 1/2$, it follows immediately from Proposition A.2 with (ψ_s, ψ_n) as in (2.7) that ADAP-FISTA terminates successfully and outputs a pair (z, u) satisfying $u \in \partial(\psi_s + \psi_n)(z)$. This inclusion, the definition of (ψ_s, ψ_n), and the definition of subdifferential in (1.7), then imply that (2.8) holds. Hence, in view of the termination criteria of step 2 of AS-PAL, it follows that $\lambda_k = \lambda$. It is then easy to see, by the way λ is updated in step 2 of AS-PAL, that λ is not halved in the $(k+1)$-th iteration or any subsequent iteration, hence proving the claim.

It is now straightforward to see that the above two claims, the fact that the initial value of the prox stepsize is equal to λ_0, and the way λ_k is updated in AS-PAL, imply that the lemma holds. ∎

The main goal of the following result is to establish a bound on the number of ACG iterations performed by each ADAP-FISTA call in step 1 of AS-PAL.

Proposition 3.2 *The following statements about AS-PAL hold for any $k \geq 1$:*

(a) the quantity $\overline{M}_k$ *as in* (2.5) *satisfies*

$$\overline{M}_k \leq \omega \mathcal{M}_{\lambda_0}(c_k)$$

where $\omega = 2\beta/(1-\chi)$*;*

(b) every ACG call in step 1 of the k*-th iteration of AS-PAL performs*

$$\mathcal{O}_1\left(\sqrt{\mathcal{M}_{\lambda_0}(c_k)}\,\log_1^+ \mathcal{M}_{\lambda_0}(c_k)\right) \tag{3.6}$$

ACG iterations/resolvent evaluations.

Proof (a) The inequality with $k = 1$ is immediate due to the last equality in (2.3), the fact that $\omega > 1$, and the definition of $\overline{M}_1$ in (2.5). It now suffices to show that $\overline{M}_k \leq \max\{\overline{M}_{k-1}, \omega\mathcal{M}_{\lambda_0}(c_k)\}$ for every $k \geq 2$ since this claim, the facts that $c_k \leq c_{k+1}$ and $\overline{M}_1 \leq \omega\mathcal{M}_{\lambda_0}(c_1)$, and a simple induction argument, imply that the inequality holds for every $k \geq 2$. To show this claim, assume that $k \geq 2$. It follows from the definition of $\overline{M}_k$, relation (3.4), and the definition of $\mathcal{M}_\lambda(\cdot)$ in (2.22), that

$$\overline{M}_k = \max\{M_{k-1}^f, \lambda_0 c_k \|A\|^2 + 1\} \leq \max\{M_{k-1}^i, \omega\mathcal{M}_{\lambda_0}(c_k)\}.$$

The claim now follows from the fact that step 1 chooses M_{k-1}^i in the interval $[1, \overline{M}_{k-1}]$.

(b) First note that Proposition 3.1(a) implies that $\bar{L} = \mathcal{M}_{\lambda_0}(c_k)$ satisfies (A.2). Moreover, it follows from statement (a) and the fact that $M_k^i \leq \overline{M}_k$ that $M_k^i = \mathcal{O}(\mathcal{M}_{\lambda_0}(c_k))$. Since each ADAP-FISTA call in step 1 is made with $(\mu, L_0) = (1/2, M_k^i)$, it then follows from Proposition A.1(a) that (b) holds. ∎

The subsequent technical result characterizes the change in the augmented Lagrangian function between consecutive iterations of the AS-PAL method.

Lemma 3.3 *For every* $k \geq 1$*, we have:*

$$\mathcal{L}_{c_k}(z_k, p_k) - \mathcal{L}_{c_k}(z_k, p_{k-1}) = \frac{1}{c_k}\|p_k - p_{k-1}\|^2, \tag{3.7}$$

and

$$\frac{\lambda_k}{C_\sigma}\|w_k\|^2 \leq \mathcal{L}_{c_k}(z_{k-1}, p_{k-1}) - \mathcal{L}_{c_k}(z_k, p_k) + \frac{1}{c_k}\|p_k - p_{k-1}\|^2 \tag{3.8}$$

where C_σ *is as in* (2.3).

Proof Identity (3.7) follows immediately from the definition of the Lagrangian in (1.2) and the second relation in (2.10). Now, using relation (3.2), the first inequality in (3.1), and the definitions of C_σ and w_k in (2.3) and (2.9), respectively, we conclude that:

$$\begin{aligned}
&\lambda_k\mathcal{L}_{c_k}(z_{k-1}, p_{k-1}) - \lambda_k\mathcal{L}_{c_k}(z_k, p_{k-1}) \overset{(3.2)}{\geq} \frac{1}{2}\|z_k - z_{k-1}\|^2 + \langle u_k, z_{k-1} - z_k\rangle \\
&= \frac{1}{2}\|z_{k-1} - z_k + u_k\|^2 - \frac{1}{2}\|u_k\|^2 \overset{(2.9)}{=} \frac{1}{2}\|\lambda_k w_k\|^2 - \frac{1}{2}\|u_k\|^2 \\
&\overset{(3.1)}{\geq} \frac{1}{2}\|\lambda_k w_k\|^2 - \frac{\sigma^2}{2(1-\sigma)^2}\|\lambda_k w_k\|^2 = \frac{1-2\sigma}{2(1-\sigma)^2}\|\lambda_k w_k\|^2 \overset{(2.3)}{=} \frac{\|\lambda_k w_k\|^2}{C_\sigma}.
\end{aligned} \tag{3.9}$$

Inequality (3.8) now follows by dividing (3.9) by λ_k and combining the resulting inequality with (3.7). ∎

The result below, which establishes boundedness of the sequence of Lagrange multipliers, makes use of a technical result in the Appendix, namely Lemma B.3.

Proposition 3.4 *The sequences* $\{q_k\}$ *and* $\{p_k\}$ *generated by AS-PAL satisfy*

$$\|q_k\| \leq \kappa_p, \quad \|p_k\| \leq \kappa_p \tag{3.10}$$

where κ_p *is defined in* (2.16).

Proof Using the second inequality in (3.1), the triangle inequality, the second inequality in (3.5), and the definitions of D_h and ∇_f in (C1) and (2.16), respectively, we conclude that

$$\|w_k - \nabla f(z_k)\| \overset{(3.1)}{\leq} \frac{1}{\lambda_k}(1+\sigma)\|z_k - z_{k-1}\| + \nabla_f \overset{(3.5)}{\leq} \frac{D_h(1+\sigma)}{\underline{\lambda}} + \nabla_f. \tag{3.11}$$

Now, using the inclusion in (3.3), the relation in (3.11), Lemma B.3(b) with $(z,q,r) = (z_k, q_k, w_k - \nabla f(z_k))$ and $q^- = p_{k-1}$, and the definition of κ_p in (2.16), we conclude that for every $k \geq 1$:

$$\|q_k\| \overset{(B.4)}{\leq} \max\left\{\|p_{k-1}\|, \frac{2D_h(M_h + \|w_k - \nabla f(z_k)\|)}{\bar{d}\nu_A^+}\right\} \overset{(3.11)}{\leq} \max\left\{\|p_{k-1}\|, \kappa_p\right\}. \tag{3.12}$$

We now use an induction argument to show that (3.10) holds. Indeed, (3.10) holds for $k = 0$ since $p_0 = 0$ and $q_0 = 0$ in view of step 0 of AS-PAL. Suppose then that (3.10) holds for $k = k-1$. The relation in (3.12) together with the induction hypothesis then immediately imply that $\|q_k\| \leq \max\{\|p_{k-1}\|, \kappa_p\} = \kappa_p$. Since p_k is equal to either q_k or p_{k-1} in view of the second identity in (2.10), we also conclude that $\|p_k\| \leq \kappa_p$. ∎

Recall that the l-th cycle $\mathcal{C}_l$ of AS-PAL is defined in (2.13). The following result shows that the sequence $\{\|\|w_k\|\}_{k\in\mathcal{C}_l}$ is bounded and can be controlled by $\{\Delta_k\}_{k\in\mathcal{C}_l}$ plus a term which is $\mathcal{O}(1/d_l\tilde{c}_l)$.

Lemma 3.5 *Consider the sequences $\{(z_k, p_k, w_k)\}_{k\in\mathcal{C}_l}$ and $\{\Delta_k\}$ generated by AS-PAL. Then, for every $k \in \mathcal{C}_l$ such that $k \geq k_l + 1$, we have:*

$$\frac{\sum_{i=k_l+1}^k \lambda_i \|w_i\|^2}{\sum_{i=k_l+1}^k \lambda_i} \leq C_\sigma \left(\Delta_k + \frac{4\kappa_p^2}{\underline{\lambda} d_l \tilde{c}_l}\right) \tag{3.13}$$

where C_σ, $\underline{\lambda}$, and κ_p are as in (2.3), (2.15), and (2.16), respectively and k_l is the first index in $\mathcal{C}_l$.

Proof For $k \in \mathcal{C}_l$, define $I(k) := \{i : p_i \neq p_{i-1}, k_l + 1 \leq i \leq k\}$. It is easy to see from the update rule for p_k in (2.10) that $|I(k)| \leq (k - k_l)/d_l$. Moreover, the bound on p_k in (3.10) and the relation $(a+b)^2 \leq 2a^2 + 2b^2$ for $a, b \in \mathbb{R}$ imply that $\|p_j - p_{j-1}\|^2 \leq 2\|p_j\|^2 + 2\|p_{j-1}\|^2 \leq 4\kappa_p^2$. It then follows from the previous two bounds that for any $k \in \mathcal{C}_l$,

$$\sum_{i=k_l+1}^k \|p_i - p_{i-1}\|^2 = \sum_{i\in I(k)} \|p_i - p_{i-1}\|^2 \leq \frac{4(k-k_l)\kappa_p^2}{d_l}. \tag{3.14}$$

Hence, summing inequality (3.8) from $k_l + 1$ to k, relation (3.14), and the fact that $c_k = \tilde{c}_l$ for every $k \in \mathcal{C}_l$, imply that, for any $k \in \mathcal{C}_l$ such that $k \geq k_l + 1$, there holds

$$\begin{aligned}
\frac{1}{C_\sigma}\sum_{i=k_l+1}^k \lambda_i \|w_i\|^2 &\overset{(3.8)}{\leq} \sum_{i=k_l+1}^k \left[\mathcal{L}_{c_i}(z_{i-1}, p_{i-1}) - \mathcal{L}_{c_i}(z_i, p_i) + \frac{1}{c_i}\|p_i - p_{i-1}\|^2\right] \\
&\overset{j\in\mathcal{C}_l}{=} \sum_{i=k_l+1}^k \left[\mathcal{L}_{\tilde{c}_l}(z_{i-1}, p_{i-1}) - \mathcal{L}_{\tilde{c}_l}(z_i, p_i) + \frac{1}{\tilde{c}_l}\|p_i - p_{i-1}\|^2\right] \\
&= \mathcal{L}_{\tilde{c}_l}(z_{k_l}, p_{k_l}) - \mathcal{L}_{\tilde{c}_l}(z_k, p_k) + \frac{1}{\tilde{c}_l}\sum_{i=k_l+1}^k \|p_i - p_{i-1}\|^2 \\
&\overset{(3.14)}{\leq} \mathcal{L}_{\tilde{c}_l}(z_{k_l}, p_{k_l}) - \mathcal{L}_{\tilde{c}_l}(z_k, p_k) + \frac{4(k-k_l)\kappa_p^2}{d_l\tilde{c}_l} \\
&= \left(\sum_{i=k_l+1}^k \lambda_i\right)\Delta_k + \frac{4(k-k_l)\kappa_p^2}{d_l\tilde{c}_l},
\end{aligned}$$

where the last equality follows from the definition of Δ_k in (2.11). Now, using the above bound, and (3.5) we have:

$$\frac{\sum_{i=k_l+1}^k \lambda_i \|w_i\|^2}{\sum_{i=k_l+1}^k \lambda_i} \leq C_\sigma \left(\Delta_k + \frac{4(k-k_l)\kappa_p^2}{d_l\tilde{c}_l\sum_{i=k_l+1}^k \lambda_i}\right) \overset{(3.5)}{\leq} C_\sigma\left(\Delta_k + \frac{4\kappa_p^2}{\underline{\lambda} d_l\tilde{c}_l}\right),$$

which immediately implies the result. ∎

The next result establishes bounds on $\|Az_k - b\|$ and on the quantity Δ_k defined in (2.11).

Lemma 3.6 *Consider the sequence of iterates $\{(z_k, c_k, p_k)\}_{k\in\mathcal{C}_l}$ generated during the l-th cycle of AS-PAL and let Δ_k be as in (2.11). Then, for every $k \in \mathcal{C}_l$,*

(a) we have

$$\|Az_k - b\| \leq \frac{2\kappa_p}{\tilde{c}_l}; \tag{3.15}$$

(b) if additionally $k \geq k_l + 1$, then

$$\Delta_k \leq \frac{\kappa_d}{\sum_{i=k_l+1}^{k} \lambda_i}, \tag{3.16}$$

where κ_d is as in (2.17) and k_l denotes the first index in $\mathcal{C}_l$.

Proof (a) Let $k \in \mathcal{C}_l$. Using the update for q_k in (2.10), the triangle inequality, and the bounds on q_k and p_k in (3.10), we have

$$\|Az_k - b\| \overset{(2.10)}{=} \frac{\|q_k - p_{k-1}\|}{c_k} \overset{k\in\mathcal{C}_l}{\leq} \frac{\|q_k\| + \|p_{k-1}\|}{\tilde{c}_l} \overset{(3.10)}{\leq} \frac{2\kappa_p}{\tilde{c}_l},$$

which immediately proves (3.15).

(b) Recall from (2.13) that $\mathcal{C}_l := \{k : c_k = \tilde{c}_l := 2^{l-1}c_0\}$. Then, using the Cauchy-Schwarz inequality, the definition of the Lagrangian function in (1.2), the definition of S in (2.17), the bound on p_k in (3.10), relation (3.15), and the fact that $\tilde{c}_l \geq c_0$, we have

$$\mathcal{L}_{\tilde{c}_l}(z_{k_l}, p_{k_l}) \leq S + \|p_{k_l}\|\|Az_{k_l} - b\| + \frac{\tilde{c}_l}{2}\|Az_{k_l} - b\|^2 \overset{(3.15)}{\leq} S + \|p_{k_l}\|\left(\frac{2\kappa_p}{\tilde{c}_l}\right) + \frac{2\kappa_p^2}{\tilde{c}_l} \overset{(3.10)}{\leq} S + \frac{4\kappa_p^2}{c_0}. \tag{3.17}$$

Let $k \in \mathcal{C}_l$ be such that $k \geq k_l + 1$. Using the bound on p_k in (3.10), the fact that $\tilde{c}_l \geq c_0$, the definition of ϕ_* in (2.15), and completing the square, we have:

$$\mathcal{L}_{\tilde{c}_l}(z_k, p_k) - \phi_* \geq \mathcal{L}_{\tilde{c}_l}(z_k, p_k) - (f + h)(z_k) = \frac{1}{2}\left\|\frac{p_k}{\sqrt{\tilde{c}_l}} + \sqrt{\tilde{c}_l}(Az_k - b)\right\|^2 - \frac{\|p_k\|^2}{2\tilde{c}_l} \overset{(3.10)}{\geq} -\frac{\kappa_p^2}{2c_0}. \tag{3.18}$$

Hence, it follows from the definition of Δ_k in (2.11) and relations (3.17) and (3.18) that

$$\Delta_k = \frac{1}{\sum_{i=k_l+1}^{k} \lambda_i}\left(\mathcal{L}_{\tilde{c}_l}(z_{k_l}, p_{k_l}) - \mathcal{L}_{\tilde{c}_l}(z_k, p_k)\right) \leq \frac{1}{\sum_{i=k_l+1}^{k} \lambda_i}\left(S + \frac{9\kappa_p^2}{2c_0} - \phi_*\right).$$

Thus, (3.16) immediately follows from the definition of κ_d in (2.17). ∎

The following result establishes bounds on the number of ACG and outer iterations performed during an AS-PAL cycle and shows that AS-PAL outputs a $(\hat{\rho}, \hat{\eta})$-approximate stationary solution of (1.1) within a logarithmic number of cycles.

Proposition 3.7 *The following statements about AS-PAL hold:*

(a) every cycle performs at most

$$\left\lceil 2 + \frac{2C_\sigma\kappa_d}{\underline{\lambda}\hat{\rho}^2} \right\rceil \tag{3.19}$$

outer iterations, where $\underline{\lambda}$, κ_d, and C_σ are as in (2.15), (2.17), and (2.3) respectively; moreover, if λ_0 is such that $\lambda_0 = \Omega(m_f^{-1})$ and $\log_0^+(m_f\lambda_0) \leq \mathcal{O}(1 + \kappa_d/(\underline{\lambda}\hat{\rho}^2))$, then the number of ACG calls within an arbitrary cycle is $\mathcal{O}(1 + m_f\kappa_d/\hat{\rho}^2)$;

(b) any cycle l generated by AS-PAL has the property that its penalty parameter $\tilde{c}_l$ satisfies $\tilde{c}_l \leq \hat{c}(\hat{\rho}, \hat{\eta})$ where $\hat{c}(\hat{\rho}, \hat{\eta})$ is as in (2.18); as a consequence, the number of cycles of AS-PAL is bounded by

$$\log_1^+\left(\frac{2\hat{c}(\hat{\rho}, \hat{\eta})}{c_0}\right) \tag{3.20}$$

where c_0 is the initial penalty parameter for AS-PAL.

Proof (a) Fix a cycle l and let k_l denote the first index in $\mathcal{C}_l$ (see (2.12)). If some $k \in \mathcal{C}_l$ is such that

$$k > k_l + \frac{2C_\sigma \kappa_d}{\underline{\lambda}\hat{\rho}^2} \tag{3.21}$$

then

$$\Delta_k \overset{(3.16)}{\leq} \frac{\kappa_d}{\sum_{i=k_l+1}^{k} \lambda_i} \overset{(3.5)}{\leq} \frac{\kappa_d}{\underline{\lambda}(k-k_l)} \overset{(3.21)}{\leq} \frac{\hat{\rho}^2}{2C_\sigma}, \tag{3.22}$$

which clearly implies that Δ_k satisfies inequality (2.11) and hence that the l-th cycle ends at or before the k-th iteration. Hence, the first part of (a) follows immediately from this conclusion. To prove the second part, first note that the number of times λ is divided by 2 in step 2 of AS-PAL is at most $\lceil \log_0^+(\lambda_0/\underline{\lambda})/\log 2 \rceil$, in view of the last conclusion of Proposition 3.1. This observation, the conclusion of the first part, the two conditions imposed on λ_0, and the definition of $\underline{\lambda}$ in (2.15), then imply that the number of ACG calls within an arbitrary cycle is $\mathcal{O}(1 + m_f\kappa_d/\hat{\rho}^2)$.

(b) Assume by contradiction that AS-PAL generates a $\tilde{c}_l$ such that $\tilde{c}_l > \hat{c}(\hat{\rho}, \hat{\eta})$. This assumption and the definition of $\hat{c}(\hat{\rho}, \hat{\eta})$ in (2.18) then imply that $l > 1$ and $\tilde{c}_{l-1} \geq \hat{c}(\hat{\rho}, \hat{\eta})$. It follows from this observation together with Lemma 3.6(a) with $l = l - 1$ that for every $k \in \mathcal{C}_{l-1}$,

$$\|Az_k - b\| \overset{(3.15)}{\leq} \frac{2\kappa_p}{\tilde{c}_{l-1}} \overset{(2.18)}{<} \eta. \tag{3.23}$$

This implies that $\min_{i \in \mathcal{C}_{l-1}} \|w_i\| > \hat{\rho}$ in view of the fact that AS-PAL does not stop succcesfully in step 3 of its $(l-1)$-th cycle. Letting k_{l-1} denote the first index of the $(l-1)$-th cycle, this conclusion together with Lemma 3.5 with $l = l - 1$ then imply that

$$\hat{\rho}^2 < \frac{\sum_{i=k_{l-1}+1}^{k} \lambda_i \|w_i\|^2}{\sum_{i=k_{l-1}+1}^{k} \lambda_i} \leq C_\sigma \left(\Delta_k + \frac{4\kappa_p^2}{\underline{\lambda} d_{l-1} \tilde{c}_{l-1}} \right) \leq C_\sigma \Delta_k + \frac{\hat{\rho}^2}{2},$$

where the third inequality follows from the fact that $\tilde{c}_{l-1}$ satisfies (2.18). Using this last conclusion, we can easily see that (2.11) is violated for every $k \in \mathcal{C}_{l-1}$ such that $k \geq k_{l-1} + 1$, a conclusion that contradicts the fact that the $(l-1)$-th cycle terminated in step 4 of AS-PAL. ∎

We are now ready to prove Theorem 2.2.

Proof (of Theorem 2.2) First, note that the assumptions that $\lambda_0 = \Omega(m_f^{-1})$, $\log_0^+(m_f\lambda_0) \leq \mathcal{O}(1 + \kappa_d/(\underline{\lambda}\hat{\rho}^2))$, the definition of $\underline{\lambda}$ in (2.15), and the second conclusion of Proposition 3.7(a) imply that every cycle of AS-PAL performs $\mathcal{O}(1 + m_f\kappa_d/\hat{\rho}^2)$ ACG calls. Second, Proposition 3.7(b) implies that $\tilde{c}_l \leq \hat{c}$ and hence that $\mathcal{M}_{\lambda_0}(\tilde{c}_l) \leq \mathcal{M}_{\lambda_0}(\hat{c})$ in view of the definition of $\mathcal{M}_\lambda(\cdot)$ in Theorem 2.2. The result then immediately follows from the above observations, Proposition 3.2(b) with $c_k = \tilde{c}_l$, and the bound (3.20) on the number of cycles performed by AS-PAL. ∎

4 Numerical Experiments

This section showcases the numerical performance of AS-PAL against five other benchmark algorithms for solving five classes of linearly-constrained SNCO problems. It contains three subsections. The first presents the numerical results of the algorithms on three different linearly-constrained SNCO matrix problems. The second presents the numerical results of the algorithms on two different linearly-constrained SNCO vector problems. The last subsection contains comments about the numerical results.

We have implemented two different more aggressive variants of AS-PAL, which we refer to as ASL and ASL-2. The details of both variants are described as follows. First, both variants differ from AS-PAL in that they allow the prox stepsize to be doubled in step 5 of any iteration if it has not been halved in step 2 and the number of iterations performed by its ACG call in step 1 has not exceeded a pre-specified number. Second, since the prox stepsize is allowed to increase in these variants, the initial prox stepsize is taken to be relatively small. Third, our implementation chooses the following values for the input parameters of ASL and ASL-2:

$$\sigma = 0.1, \quad \mu = 1/4, \quad \chi = 0.001, \quad \beta = 1.25, \quad p_0 = 0.$$

Fourth, as mentioned in Section 2.2, ASL and ASL-2 set M_k^i equal to M_{k-1}^f. Finally, the key difference between ASL and ASL-2 is that ASL performs a Lagrange multiplier update at each of its outer iterations while ASL-2 performs a Lagrange multiplier update every few iterations. More specifically, for every cycle $l \geq 1$, ASL (resp., ASL-2) chooses the scalar d_l in (2.10) as $d_l = 1$ (resp., $d_l = \tilde{c}_l$).

The two variants of our method are bench-marked against the following five algorithms. Specifically, ASL and ASL-2 are bench-marked against the iALM method of [21], two variants of the S-prox-ALM of [38, 39] (nicknamed SPA1 and SPA2), the dampened augmented Lagrangian method of [18] (nicknamed ADL), the inexact proximal augmented Lagrangian method of [16] (nicknamed IPL), and the relaxed quadratic penalty method of [14] (nicknamed RQP). We next describe the implementation details of each of these five algorithms. The implementation of iALM chooses the parameters σ, β_0, w_0, $\mathbf{y}^0$, and γ_k as

$$\sigma = 5, \quad \beta_0 = 1, \quad w_0 = 1, \quad \mathbf{y}^0 = 0, \quad \gamma_k = \frac{(\log 2)\,\|Ax^1\|}{(k+1)\,[\log(k+2)]^2},$$

for every $k \geq 1$. Furthermore, the implementation of iALM uses the ACG subroutine called APG. The starting point for the k^{th} APG call is the prox center for the k^{th} prox subproblem. The implementations of SPA1 and SPA2 choose the parameters α, p, c, β, y_0, and z_0 as

$$\alpha = \frac{\Gamma}{4}, \quad p = 2(L_f + \Gamma\|A\|^2), \quad c = \frac{1}{2(L_f + \Gamma\|A\|^2)}, \quad \beta = 0.5, \quad y_0 = 0, \quad z_0 = x_0,$$

where $\Gamma = 1$ in SPA1 and $\Gamma = 10$ in SPA2. ADL sets its initial prox stepsize $\lambda_0 = 10$, its initial penalty parameter $c_0 = 1$, and its parameters $(\sigma, \chi, \theta) = (0.3, 1/6, 1/2)$. The implementation of IPL sets $\sigma = 0.3$, initial penalty parameter $c_0 = 1$, and constant prox stepsize $\lambda = 1/(2m_f)$. RQP uses the AIPPv2 variant in [14] with initial prox stepsize $\lambda_0 = 1/m_f$, $\sigma = 0.3$, and parameters $(\theta, \tau) = (4, 10\,[\lambda L_f + 1])$. Finally, note that IPL and RQP solve each prox subproblem using the ACG variant in [28] with an adaptive line search for the ACG variant's stepsize parameter as described in [10].

We describe the type of solution each of the methods aims to find. That is, given a linear operator A, functions f and h satisfying assumptions described in Subsection 2.1, an initial point $z_0 \in \mathcal{H}$, and tolerance pair $(\hat{\rho}, \hat{\eta}) \in \Re^2_{++}$, each of the methods aims to find a triple (z, p, w) satisfying:

$$w \in \nabla f(z) + \partial h(z) + A^* p, \quad \frac{\|w\|}{1 + \|\nabla f(z_0)\|} \leq \hat{\rho}, \quad \frac{\|Az - b\|}{1 + \|Az_0 - b\|} \leq \hat{\eta}, \tag{4.1}$$

where $\|\cdot\|$ signifies the Euclidean norm when solving vector problems and the Frobenius norm when solving matrix problems.

The tables below report the runtimes and the total number of ACG iterations needed to find a triple satisfying (4.1). The bold numbers in the tables of this section indicate the algorithm that performed the best for that particular metric (i.e. runtime or ACG iterations). It will be seen in the following subsections that the adaptive methods ASL, ASL-2, and RQP are the most consistent ones among all the methods considered. More specifically, within the specified time limit for each problem class, ASL and ASL-2 converged in all instances considered in our experiments while RQP converged in approximately 88% of them. To compare ASL and RQP on a particular problem class more closely, we also report in each table caption the following average time ratio (ATR) between these two methods defined as

$$ATR = \frac{1}{N}\sum_{i=1}^{N} a_i/r_i, \tag{4.2}$$

where N is the number of class instances that both methods were able to solve and a_i and r_i are the runtimes of ASL and RQP for instance i, respectively.

Finally, we note that all experiments were performed in MATLAB 2020a and run on a Macbook Pro with 8-core Intel Core i9 processor and 32 GB of memory. All codes for these experiments are also available online[4].

[4] See https://github.com/asujanani6/AS-PAL.

4.1 Linearly-Constrained SNCO Matrix Problems

This subsection compares ASL and ASL-2 against the methods iALM, IPL, ADL, and RQP on three different linearly-constrained SNCO matrix problems. It is divided into three sub-subsections with each one dealing with a specific linearly-constrained SNCO problem.

We now make some remarks about the experiments of this subsection. The two variants SPA1 and SPA2 of [38, 39] are not included in our benchmark of this subsection since they are only guaranteed to converge when h is the indicator function of a polyhedron, a condition which does not apply to any of the problems considered in this subsection. Also, since § 4.1.3 considers a slight generalization of the linearly-constrained SNCO problem (1.1) where the linear constraints $Ax = b$ are replaced with the more general set linear constraints $Ax \in S$, we have slightly modified the codes for ASL and ASL-2 to handle these more general constraints and then compared them with RQP since it is the only other method whose code can currently solve this extended formulation.

4.1.1 Sparse PCA

This § 4.1.1 considers the sparse principal components analysis problem studied in [5]. That is, given integer k, positive scalar pair $(\vartheta, b) \in \Re^2_{++}$, and matrix $\Sigma \in S^n_+$, consider the following sparse principal component analysis (PCA) problem:

$$\begin{aligned} \min_{\Pi,\Phi} \ & \langle \Sigma, \Pi \rangle_F + \sum_{i,j=1}^n q_\vartheta(\Phi_{ij}) + \vartheta \sum_{i,j=1}^n |\Phi_{ij}| \\ \text{s.t. } & \Pi - \Phi = 0, \quad (\Pi, \Phi) \in \mathcal{F}^k \times \Re^{n\times n} \end{aligned}$$

where $\mathcal{F}^k = \{M \in S^n_+ : 0 \preceq M \preceq I, \operatorname{tr} M = k\}$ denotes the k–Fantope and q_ϑ is the minimax concave penalty (MCP) function given by

$$q_\vartheta(t) := \begin{cases} -t^2/(2b), & \text{if } |t| \le b\vartheta, \\ b\vartheta^2/2 - \vartheta|t|, & \text{if } |t| > b\vartheta, \end{cases} \quad \forall t \in \Re.$$

Parameters	**Iteration Count/Runtime (seconds)**					
(k, m_f, L_f)	iALM	IPL	ADL	RQP	ASL	ASL-2
(5,125,125)	*/*	1438/14.84	6674/29.34	377974/2037.71	**376/0.86**	817/1.78
(10,125,125)	*/*	1559/8.33	*/*	44194/211.82	485/1.07	**437/1.06**
(20,125,125)	*/*	1400/7.02	28040/158.96	14229/72.78	**489/1.10**	506/1.11
(5,200,200)	*/*	6555/43.98	*/*	282559/1478.51	516/1.15	**366/0.88**
(10,200,200)	*/*	7470/47.53	*/*	12335/85.69	513/1.11	**447/0.99**
(20,200,200)	*/*	20132/118.99	*/*	61016/319.77	**421/0.96**	519/1.15
(5,250,250)	*/*	10391//44.25	5766/25.68	350333/1300.26	**523/1.10**	535/1.15
(10,250,250)	211991/574.91	32566/175.81	*/*	143796/943.87	**645/1.38**	915/1.94
(20,250,250)	236490/628.55	199353/985.49	*/*	*/*	**486/1.08**	815/1.90
$(5,10^3,10^3)$	*/*	567358/2873.66	12717/52.65	*/*	**918/1.91**	995/2.13
$(10,10^3,10^3)$	*/*	*/*	*/*	*/*	**759/1.70**	1234/2.81
$(20,10^3,10^3)$	*/*	*/*	52557/236.03	*/*	**1626/3.41**	2252/4.71

Table 4.1 Iteration counts and runtimes (in seconds) for the Sparse PCA problem in § 4.1.1. The tolerances are set to 10^{-5}. Entries marked with * did not converge in the time limit of 3600 seconds. The ATR metric is 0.0050.

For our experiments in this subsection, we choose $\vartheta = 100$ and allow b to vary. Observe that the curvature parameters are $m_f = L_f = 1/b$. We also generate the matrix Σ according to an eigenvalue decomposition $\Sigma = P\Lambda P^T$, based on a parameter pair (s, k), where k is as in the problem description and s is a positive integer. Specifically, we choose $\Lambda = (100, 1, ..., 1)$, the first column of P to be a sparse vector whose first s entries are $1/\sqrt{s}$, and the other entries of P to be sampled randomly from the standard Gaussian distribution. For our experiments, we fix $s = 5$ and allow k to vary. Also, for every problem instance, the initial starting point is chosen as $(\Pi_0, \Phi_0) = (D_k, 0)$ where D_k is a diagonal matrix whose first k entries are 1 and whose remaining entries are 0.

Parameters	**Outer Iterations/Average Inner Iterations**				
(k, m_f, L_f)	IPL	ADL	RQP	ASL	ASL-2
(5,125,125)	52/27.65	58/115.07	960/393.72	26/14.46	30/27.23
(10,125,125)	59/26.42	*/*	943/46.87	27/17.96	28/15.61
(20,125,125)	48/29.17	39/718.97	447/31.83	28/17.46	30/16.87
(5,200,200)	84/78.04	*/*	86983/3.25	26/19.85	28/13.07
(10,200,200)	170/43.94	*/*	671/18.38	27/19	30/14.9
(20,200,200)	167/120.55	*/*	291/209.68	29/14.52	31/16.74
(5,250,250)	248/41.90	67/86.06	110293/3.18	27/19.37	28/19.11
(10,250,250)	1544/21.09	*/*	365/393.96	27/23.89	28/32.68
(20,250,250)	10842/18.39	*/*	*/*	29/16.76	31/26.29
$(5,10^3,10^3)$	34515/16.44	234/54.35	*/*	29/31.66	31/32.10
$(10,10^3,10^3)$	*/*	*/*	*/*	29/26.17	32/38.56
$(20,10^3,10^3)$	*/*	40/1313.93	*/*	30/54.20	32/70.38

Table 4.2 Number of outer iterations and average number of inner iterations per outer iteration for the Sparse PCA problem in § 4.1.1.

We now describe the specific parameters that ASL, ASL-2, and RQP choose for this class of problems. ASL, ASL-2, and RQP choose the initial penalty parameter, $c_0 = 1$. Both ASL and ASL-2 allow the prox stepsize to be doubled at the end of an iteration if the number of iterations by its ACG call does not exceed 4. Finally, ASL and ASL-2 also take M_0^1 defined in step 1 of AS-PAL to be 1 and the initial prox stepsize to be $20/m_f$.

The numerical results are presented in Table 4.1. Table 4.1 compares ASL and ASL-2 with four of the benchmark algorithms namely, iALM, IPL, ADL, and RQP. Iteration counts and runtimes for all instances are presented. The tolerances are set as $\hat{\rho} = \hat{\eta} = 10^{-5}$ and a time limit of 3600 seconds, or 1 hour, is imposed. Entries marked with * did not converge in the time limit.

This § 4.1.1 is unique in that it more closely investigates the behavior of ASL and ASL-2 and the four bench-marked algorithms on all Sparse PCA instances tested.

Table 4.2 presents the number of outer iterations and the average number of inner iterations per outer iteration performed by each of the methods for the same Sparse PCA instances considered in Table 4.1. As demonstrated by the table, both ASL and ASL-2 perform the least number of outer iterations and average inner iterations among all the methods considered.

To illustrate the robustness of ASL with respect to different choices of input parameters, we compare it with six different sets of input parameters, of which the first one (namely, ASL-I0) is the one used to generate the results for ASL and ASL-2 in Table 4.1. The six choices of input parameters are described in Table 4.3 and the corresponding iteration counts and runtimes performed by ASL are given in Table 4.4.

	Input Parameter Comparison					
Parameters	ASL-I0	ASL-I1	ASL-I2	ASL-I3	ASL-I4	ASL-I5
σ	0.1	0.3	0.3	0.4	0.45	0.35
μ	0.25	0.5	0.125	0.167	0.1	0.125
χ	0.001	0.01	0.01	0.0001	0.0001	0.00001
β	1.25	1.5	1.75	2	1.2	2
λ_0	$20/m_f$	$0.05/m_f$	$2/m_f$	$0.1/m_f$	$0.5/m_f$	$10/m_f$

Table 4.3 Six different choices of input parameters.

From Table 4.4, we can see that the average CPU time of ASL over the 12 instances were 1.40s, 2.12s, 2.01s, 2.48s, 2.39s, and 1.87s, showing that the performance of ASL is robust under different choices of input parameters.

Parameters	Iteration Count/Runtime (seconds)					
(k, m_f, L_f)	ASL-I0	ASL-I1	ASL-I2	ASL-I3	ASL-I4	ASL-I5
(5,125,125)	376/0.86	607/1.61	**336/0.80**	641/1.56	475/1.07	359/0.81
(10,125,125)	485/1.07	538/1.27	**345/0.83**	719/1.66	586/1.27	424/0.98
(20,125,125)	489/1.10	509/1.18	**380/0.89**	589/1.36	485/1.10	485/1.10
(5,200,200)	516/1.15	815/1.83	546/1.22	750/1.76	701/1.55	**446/1.03**
(10,200,200)	513/1.11	1113/2.40	626/1.41	816/1.88	875/1.89	**480/1.07**
(20,200,200)	421/0.96	665/1.50	428/1.01	776/1.71	597/1.38	**379/0.90**
(5,250,250)	**523/1.10**	737/1.77	591/1.33	886/2.12	823/1.87	570/1.21
(10,250,250)	**645/1.38**	838/1.89	690/1.57	983/2.15	1016/2.21	646/1.42
(20,250,250)	**486/1.08**	539/1.28	501/1.17	661/1.61	568/1.29	643/1.45
(5,1000,1000)	**918/1.91**	1738/3.62	1567/3.25	1790/3.88	1629/3.39	1075/2.27
(10,1000,1000)	**759/1.70**	2024/4.32	2420/4.97	2184/4.60	2821/5.73	1952/4.12
(20,1000,1000)	1626/3.41	**1305/2.82**	2790/5.70	2674/5.52	2898/5.94	2936/6.11

Table 4.4 Performance of ASL for solving the Sparse PCA problem of § 4.1.1 for six different choices of input parameters. The tolerances are set to 10^{-5}.

4.1.2 Nonconvex QSDP

Given a pair of dimensions $(\ell, n) \in \mathbb{N}^2$, a scalar pair $(\tau_1, \tau_2) \in \Re^2_{++}$, linear operators $\mathcal{A} : \mathbb{S}^n_+ \mapsto \Re^\ell$, $\mathcal{B} : \mathbb{S}^n_+ \mapsto \Re^n$, and $\mathcal{C} : \mathbb{S}^n_+ \mapsto \Re^\ell$ defined by

$$[\mathcal{A}(Z)]_i = \langle A_i, Z\rangle\,, \quad [\mathcal{B}(Z)]_j = \langle B_j, Z\rangle\,, \quad [\mathcal{C}(Z)]_i = \langle C_i, Z\rangle\,,$$

for matrices $\{A_i\}_{i=1}^\ell, \{B_j\}_{j=1}^n, \{C_i\}_{i=1}^\ell \subseteq \Re^{n\times n}$, positive diagonal matrix $D \in \Re^{n\times n}$, and a vector pair $(b, d) \in \Re^\ell \times \Re^\ell$, this § 4.1.2 considers the following nonconvex quadratic semidefinite programming (QSDP) problem:

$$\begin{aligned} \min_Z \ & \left[f(Z) := -\frac{\tau_1}{2}\|D\mathcal{B}(Z)\|^2 + \frac{\tau_2}{2}\|\mathcal{C}(Z) - d\|^2\right] \\ \text{s.t. } & \mathcal{A}(Z) = b, \quad Z \in P^n, \end{aligned}$$

where $P^n = \{Z \in \mathbb{S}^n_+ : \text{trace}\,(Z) = 1\}$.

For our experiments in § 4.1.2, we choose dimensions $(l, n) = (30, 100)$. The matrices A_i, B_j, and C_i are generated so that only 5% of their entries are nonzero. The entries of A_i, B_j, C_i, and d (resp. D) are generated by sampling from the uniform distribution $\mathcal{U}[0, 1]$ (resp. $\mathcal{U}[1, 1000]$). We generate the vector b as $b = \mathcal{A}(E/n)$, where E is the diagonal matrix in $\Re^{n\times n}$ with all ones on the diagonal. The initial starting point z_0 is generated as a random matrix in $\mathbb{S}^+_n$. The specific procedure for generating it is described in [18]. Finally, we choose $(\tau_1, \tau_2) \in \Re^2_{++}$ so that $L_f = \lambda_{\max}(\nabla^2 f)$ and $m_f = -\lambda_{\min}(\nabla^2 f)$ are the various values given in the tables of this subsection.

We now describe the specific parameters that ASL, ASL-2, RQP, and iALM choose for this class of problems. ASL, ASL-2, and RQP choose the initial penalty parameter, $c_0 = 1$. Both ASL and ASL-2 allow the prox stepsize to be doubled at the end of an iteration if the number of iterations by its ACG call does not exceed 75. ASL and ASL-2 also take M_0^1 defined in step 1 of AS-PAL to be 100 and the initial prox stepsize to be $1/(20m_f)$. Finally, the auxillary parameters of iALM are given by:

$$B_i = \|A_i\|_F, \quad L_i = 0, \quad \rho_i = 0 \quad \forall i \geq 1.$$

The numerical results are presented in two tables, Table 4.5 and Table 4.6. The first table, Table 4.5, compares ASL and ASL-2 with three of the benchmark algorithms namely, iALM, IPL, and RQP. For ASL-2, not only are iteration counts and runtimes presented, but also the percentage of its outer iterations where a full Lagrange multiplier (LM) update is performed is reported. The tolerances are set as $\hat{\rho} = \hat{\eta} = 10^{-5}$ and a time limit of 10800 seconds, or 3 hours, is imposed. Table 4.6 presents the same exact instances as Table 4.5 but now with tolerances set as $\hat{\rho} = \hat{\eta} = 10^{-6}$ and a time limit of 14400 seconds, or 4 hours. Entries marked with * did not converge in the time limit. Table 4.6 only compares ASL and ASL-2 with iALM and RQP since these were the only algorithms to converge for every instance with tolerances set at 10^{-5}.

As seen from Table 4.5 and Table 4.6, ASL was the best performing method converging the fastest in 85% and 90% of the instances tested for tolerances 10^{-5} and 10^{-6}, respectively. On average, ASL also took

Parameters		Iteration Count/Runtime (seconds)					LM Update %
m_f	L_f	iALM	IPL	RQP	ASL	ASL-2	ASL-2
10^0	10^1	230272/1772.75	27345/322.82	9887/91.62	**9647/85.38**	**9647**/92.05	100%
10^0	10^2	91421/516.42	4575/53.13	7085/73.65	**1498/13.28**	**1498**/13.55	100%
10^0	10^3	113405/587.24	1403/19.54	9486/112.42	960/8.32	**937/8.29**	58.33%
10^0	10^4	393953/1794.35	3140/31.54	10019/91.49	**1824/15.98**	2217/18.89	55%
10^0	10^5	1938432/9473.18	16282/166.02	15719/145.12	**9883/85.50**	12064/103.56	55.56%
10^1	10^2	347506/1556.07	*/*	15971/140.53	**4417/38.31**	20692/176.30	14.29%
10^1	10^3	177264/750.97	*/*	10945/96.93	2151/18.98	**1807/16.10**	55.56%
10^1	10^4	129617/2008.28	1296/15.58	9838/93.88	**1273/11.14**	2403/20.42	55.56%
10^1	10^5	287924/1305.24	3410/35.51	8040/75.28	**2262/19.91**	2479/21.58	53.84%
10^1	10^6	1473676/7865.52	15855/164.07	12696/120.96	**10305/92.97**	11204/95.53	53.13%
10^2	10^4	182388/844.88	*/*	10803/99.14	**1261/11.55**	1644/13.94	60%
10^2	10^6	450561/2612.55	4503/59.18	10804/128.30	**2990/25.63**	3088/27.20	54.55%
10^2	10^7	1034041/4612.17	20235/207.29	14622/137.38	**11893/104.67**	12982/109.49	52.63%
10^3	10^4	552738/2435.47	*/*	18530/172.45	**1368/12.42**	1827/15.44	55%
10^3	10^5	220303/937.05	*/*	14929/138.71	**3543/31.34**	4144/36.30	65.71%
10^3	10^7	371617/1791.77	5969/56.92	11230/96.31	**5121/44.50**	5260/44.59	64.81%
10^3	10^8	1634409/7250.46	23075/245.84	**13465/133.86**	17371/149.43	18878/200.56	61.40%
10^4	10^5	450523/1908.97	54984/529.99	18981/168.37	**4756/42.61**	5408/48.00	74.47%
10^4	10^6	248709/1055.40	*/*	15876/143.12	**6293/54.94**	6974/58.95	76.19%
10^4	10^8	491118/2230.07	7959/83.00	13184/125.98	**7187/63.00**	7669/65.02	71.23%

Table 4.5 Iteration counts and runtimes (in seconds) for the Nonconvex QSDP problem in § 4.1.2. For ASL-2, the percentage of its outer iterations where a full Lagrange multiplier (LM) update is performed is also reported. The tolerances are set to 10^{-5}. Entries marked with * did not converge in the time limit of 10800 seconds. The ATR metric is 0.3831.

Parameters		Iteration Count/Runtime (Seconds)				LM Update %
m_f	L_f	iALM	RQP	ASL	ASL-2	ASL-2
10^0	10^1	555086/2257.85	35311/333.79	**15699/138.34**	**15699**/141.45	100%
10^0	10^2	268608/1091.32	27247/237.73	**2130/18.24**	**2130**/19.00	100%
10^0	10^3	355922/1497.22	26981/243.00	2073/17.59	**1937/16.70**	58.82%
10^0	10^4	1317510/5523.22	60908/563.85	**2453/21.21**	3099/27.25	51.17%
10^0	10^5	*/*	70646/699.03	**10479/91.90**	12895/119.70	53.57%
10^1	10^2	1297322/5529.71	68257/676.14	**7114/61.71**	34580/324.34	13.74%
10^1	10^3	526262/2254.62	41340/381.30	24596/212.61	**22721/201.09**	52.78%
10^1	10^4	370204/1565.84	35879/322.52	**2098/18.06**	2793/23.64	55%
10^1	10^5	998029/4212.47	42708/387.52	**3848/32.55**	4507/38.33	52.94%
10^1	10^6	*/*	36575/325.00	**10710/90.22**	12351/105.07	52.94%
10^2	10^4	689898/2954.75	39912/377.93	**1847/15.83**	2187/18.54	56.52%
10^2	10^6	1345701/5725.54	49506/448.12	**3658/31.57**	4143/35.74	53.85%
10^2	10^7	*/*	43571/399.37	**12300/111.21**	14147/119.15	52.50%
10^3	10^4	1714445/7243.63	64949/594.67	**1611/13.94**	2514/22.43	54.55%
10^3	10^5	596094/2740.11	40706/363.31	**3769/32.95**	4789/40.52	64.86%
10^3	10^7	1625487/6691.03	57454/511.35	**7867/68.70**	26978/229.32	61.33%
10^3	10^8	*/*	45759/399.79	**18245/163.00**	19577/168.74	62.07%
10^4	10^5	1376159/6145.45	*/*	**5030/43.15**	6207/54.05	73.47%
10^4	10^6	995529/4392.18	51540/489.43	**6552/56.81**	7333/62.09	75%
10^4	10^8	1309587/5634.84	72323/659.91	**8096/71.30**	8939/78.08	69.23%

Table 4.6 Iteration counts and runtimes (in seconds) for the Nonconvex QSDP problem in § 4.1.2. For ASL-2, the percentage of its outer iterations where a full Lagrange multiplier (LM) update is performed is also reported. The tolerances are set to 10^{-6}. Entries marked with * did not converge in the time limit of 14400 seconds. The ATR metric is 0.1616.

only roughly 0.3831 and 0.1616 amount of time that RQP took to converge for tolerances 10^{-5} and 10^{-6}, respectively. ASL-2 was the second best-performing method converging the fastest in 20% of instances tested.

4.1.3 Bounded Matrix Completion (BMC)

This § 4.1.3 considers the bounded matrix completion problem studied in [36]. That is, given a dimension pair $(p,q) \in \mathbb{N}^2$, positive scalar triple $(\upsilon, \tau_m, \theta) \in \Re^3_{++}$, scalar pair $(u,l) \in \Re^2$, matrix $Q \in \Re^{p\times q}$, and indices Ω, consider the following bounded matrix completion (BMC) problem:

$$\min_X \frac{1}{2}\|P_\Omega(X-Q)\|^2 + \tau_m \sum_{i=1}^{\min\{p,q\}} [\kappa(\sigma_i(X)) - \kappa_0\sigma_i(X)] + \tau_m\kappa_0\|X\|_*$$
$$\text{s.t. } l \le X_{ij} \le u \quad \forall (i,j) \in \{1,...,p\}\times\{1,...,q\},$$

where $\|\cdot\|_*$ denotes the nuclear norm, the function P_Ω is the linear operator that zeros out any entry not in Ω, the function $\sigma_i(X)$ denotes the i^{th} largest singular value of X, and

$$\kappa_0 := \frac{\upsilon}{\theta}, \quad \kappa(t) := \upsilon \log\left(1 + \frac{|t|}{\theta}\right) \quad \forall t \in \Re.$$

Parameters				Iteration Count			Runtime (seconds)		
θ	τ_m	m_f	L_f	RQP	ASL	ASL-2	RQP	ASL	ASL-2
1/2	0.5	2	2	130	**79**	1904	209.07	**139.52**	3166.11
1/2	1	4	4	128	**119**	1821	207.21	**189.75**	3041.02
1/2	2	8	8	1233	**457**	1734	2075.16	**931.91**	2452.90
1/3	0.5	4.5	4.5	384	**51**	1984	1229.60	**61.22**	2445.73
1/3	1	9	9	513	**76**	1913	1360.69	**101.04**	2400.80
1/3	2	18	18	*	**494**	1858	*	**1001.82**	2264.55
1/4	0.5	8	8	601	**66**	1734	928.01	**85.28**	2180.79
1/4	1	16	16	680	**90**	1698	1077.89	**147.76**	2247.67
1/5	0.5	12.5	12.5	488	**193**	454	1653.75	**313.51**	622.90
1/5	1	25	25	859	**227**	418	1494.94	**475.09**	567.75
1/6	0.5	18	18	838	**96**	1858	1359.45	**137.72**	2472.10
1/6	1	36	36	617	**221**	1815	962.52	**358.25**	2610.81
1/7	0.5	24.5	24.5	770	**142**	397	1232.90	**195.66**	560.27
1/7	1	49	49	789	**355**	365	1213.75	**462.08**	483.12

Table 4.7 Iteration counts and runtimes (in seconds) for the BMC problem in § 4.1.3. The tolerances are set to 10^{-3}. Entries marked with * did not converge in the time limit of 7200 seconds. The ATR metric is 0.3004.

We first describe the parameters considered for the above problem and some of its properties. First, the matrix Q is the user-movie ratings data matrix of the MovieLens 100K dataset[5]. Second, υ is chosen to be 0.5 and τ_m and θ are allowed to vary. Third, the curvature parameters are $m_f = 2\upsilon\tau_m/\theta^2$ and $L_f = \max\{1, m_f\}$. Fourth, the bounds are set to $(l,u) = (0,5)$ and the initial starting point is chosen as $X_0 = 0$. Finally, the above optimization problem can be written in the form:

$$\min_X f(X) + h(X)$$
$$\text{s.t. } \mathcal{A}(X) \in S,$$

where

$$f(X) = \frac{1}{2}\|P_\Omega(X-Q)\|^2 + \tau_m \sum_{i=1}^{\min\{p,q\}} [\kappa(\sigma_i(X)) - \kappa_0\sigma_i(X)], \quad h(X) = \tau_m\kappa_0\|X\|_*,$$
$$\mathcal{A}(X) = X, \quad S = \left\{Z \in \Re^{p\times q} : l \le Z_{ij} \le u,\ (i,j) \in \{1,...,p\}\times\{1,...,q\}\right\}.$$

[5] See the MovieLens 100K dataset containing 610 users and 9724 movies which can be found in https://grouplens.org/datasets/movielens/.

To deal with the more generalized constraints $\mathcal{A}(X) \in S$, both ASL and ASL-2 consider the following augmented Lagrangian function and Lagrange multiplier update:

$$\mathcal{L}_c(z,p) := f(z) + h(z) - \frac{\|p\|^2}{2c} + \frac{c}{2}\left\|\left(Az + \frac{p}{c}\right) - \Pi_S\left(Az + \frac{p}{c}\right)\right\|^2;$$
$$p_k := p_{k-1} + c_k\left[Az_k - \Pi_S\left(Az_k + \frac{p_{k-1}}{c_k}\right)\right],$$

where Π_S denotes the projection onto the set S.

We now describe the specific parameters that ASL, ASL-2, and RQP choose for this class of problems. ASL, ASL-2, and RQP choose the initial penalty parameter, $c_0 = 500$. Both ASL and ASL-2 allow the prox stepsize to be doubled at the end of an iteration if the number of iterations by its ACG call does not exceed 4. Finally, ASL and ASL-2 also take M_0^1 defined in step 1 of AS-PAL to be 1 and the initial prox stepsize to be $10/(m_f)$.

The numerical results are presented in Table 4.7. Table 4.7 compares ASL and ASL-2 with RQP. The tolerances are set as $\hat{\rho} = \hat{\eta} = 10^{-3}$ and a time limit of 7200 seconds, or 2 hours, is imposed. Entries marked with * did not converge in the time limit.

As seen from Table 4.7, ASL was the best performing method converging the fastest in 100% of the instances tested. On average, ASL also only took roughly 0.3004 amount of time that RQP took to converge.

4.2 Linearly-Constrained SNCO Vector Problems

This subsection compares ASL and ASL-2 against iALM, IPL, RQP, SPA1, and SPA2, on two different linearly-constrained SNCO vector problems.

4.2.1 Nonconvex QP

Given a pair of dimensions $(\ell, n) \in \mathbb{N}^2$, a scalar pair $(\tau_1, \tau_2) \in \Re^2_{++}$, matrices $A, C \in \Re^{\ell \times n}$ and $B \in \Re^{n \times n}$, positive diagonal matrix $D \in \Re^{n \times n}$, and a vector pair $(b, d) \in \Re^\ell \times \Re^\ell$, this § 4.2.1 considers the problem

$$\begin{array}{ll} \min\limits_z & \left[f(z) := -\dfrac{\tau_1}{2}\|DBz\|^2 + \dfrac{\tau_2}{2}\|Cz - d\|^2\right] \\ \text{s.t.} & Az = b, \quad z \in \Delta^n, \end{array}$$

where $\Delta^n := \{x \in \Re^n_+ : \sum_{i=1}^n x_i = 1\}$. For our experiments in § 4.2.1, we choose dimensions $(l, n) = (20, 1000)$ and generate the matrices A, B, and C to be fully dense. The entries of A, B, C, and d (resp. D) are generated by sampling from the uniform distribution $\mathcal{U}[0, 1]$ (resp. $\mathcal{U}[1, 1000]$). We generate the vector b as $b = A(\boldsymbol{e}/n)$ where $\boldsymbol{e}$ denotes the vector of all ones. The initial starting point z_0 is generated as $z^*/\sum_{i=1}^n z_i^*$, where the entries of z^* are sampled from the $\mathcal{U}[0, 1]$ distribution. Finally, we choose $(\tau_1, \tau_2) \in \Re^2_{++}$ so that $L_f = \lambda_{\max}(\nabla^2 f)$ and $m_f = -\lambda_{\min}(\nabla^2 f)$ are the various values given in the tables of this subsection.

We now describe the specific parameters that ASL, ASL-2, RQP, and iALM choose for this class of problems. ASL, ASL-2, and RQP choose the initial penalty parameter, $c_0 = 1$. Both ASL and ASL-2 allow the prox stepsize to be doubled at the end of an iteration if the number of iterations by its ACG call does not exceed 75. ASL and ASL-2 also take M_0^1 defined in step 1 of AS-PAL to be 100 and the initial prox stepsize to be $20/m_f$. Finally, the auxillary parameters of iALM are given by:

$$B_i = \|a_i\|, \quad L_i = 0, \quad \rho_i = 0 \quad \forall i \geq 1,$$

where a_i is the i^{th} row of A.

The numerical results are presented in two tables, Table 4.8 and Table 4.9. The first table, Table 4.8, compares ASL and ASL-2 with five of the benchmark algorithms namely, iALM, IPL, RQP, SPA1, and SPA2. The tolerances are set as $\hat{\rho} = \hat{\eta} = 10^{-4}$ and a time limit of 10800 seconds, or 3 hours, is imposed. Table 4.9 presents the same exact instances as Table 4.8 but now with tolerances set as $\hat{\rho} = \hat{\eta} = 10^{-6}$ and a time limit of 21600 seconds, or 6 hours. Table 4.9 only compares ASL and ASL-2 with iALM and RQP since these were the only algorithms to converge for every instance with tolerances set at 10^{-4}. Entries marked with * did not converge in the time limit.

Parameters	**Iteration Count/Runtime (seconds)**						
(m_f,L_f)	iALM	IPL	RQP	ASL	ASL-2	SPA1	SPA2
$(10^0,10^1)$	176005/4410.98	11202/373.54	**3905/111.11**	4762/152.64	4762/146.47	*/*	*/*
$(10^0,10^2)$	109988/1878.44	5382/155.70	6065/198.38	**1884/64.50**	**1884**/73.04	*/*	*/*
$(10^0,10^3)$	57869/1607.23	1210/55.25	11216/384.88	**645/18.75**	655/18.95	*/*	*/*
$(10^1,10^1)$	236655/4785.86	3958/144.45	3171/88.10	**1236/39.36**	**1236**/47.19	*/*	*/*
$(10^1,10^2)$	195714/3582.34	2319/84.14	6701/217.43	**1051/34.07**	**1051**/37.53	*/*	*/*
$(10^1,10^3)$	98865/2073.41	1171/41.98	7583/234.67	644/18.16	**608/17.58**	*/*	*/*
$(10^1,10^4)$	87595/3272.03	6506/280.97	15637/403.79	**924/26.29**	1000/29.04	*/*	*/*
$(10^2,10^3)$	366178/6637.79	*/*	7647/207.66	**778/23.34**	908/25.60	92872/3290.91	*/*
$(10^2,10^4)$	248673/4329.35	*/*	10421/283.96	**1375/43.01**	2145/68.75	120882/5363.80	257973/10644.96
$(10^2,10^5)$	130351/2310.50	19887/561.16	16250/447.53	2410/72.80	**2398/72.63**	205483/9317.19	213369/7548.79
$(10^3,10^3)$	363915/8111.85	*/*	4589/136.56	2001/63.97	**1997/59.91**	*/*	*/*
$(10^3,10^4)$	344723/6949.95	*/*	6023/596.66	4055/147.54	**2827/88.94**	*/*	158622/6136.37
$(10^3,10^5)$	291006/5714.73	16455/495.64	10067/279.27	**3007/110.53**	3157/122.54	*/*	286333/10761.87
$(10^3,10^6)$	141115/2527.15	21586/610.60	15991/423.22	**2208/86.19**	2921/104.64	269687/9718.92	175752/6267.26

Table 4.8 Iteration counts and runtimes (in seconds) for the Nonconvex QP problem in § 4.2.1. The tolerances are set to 10^{-4}. Entries marked with * did not converge in the time limit of 10800 seconds. The ATR metric is 0.3025.

Parameters	**Iteration Count/Runtime (seconds)**			
(m_f,L_f)	iALM	RQP	ASL	ASL-2
$(10^0,10^1)$	591803/9779.56	23935/599.52	**8276/323.31**	**8276**/360.02
$(10^0,10^2)$	698270/11336.43	62409/1579.43	**2474/87.09**	**2474**/100.31
$(10^0,10^3)$	551623/9146.99	84314/2232.40	**959/27.51**	1128/31.49
$(10^1,10^1)$	*/*	25312/703.17	**1628**/63.51	**1628/62.29**
$(10^1,10^2)$	*/*	53161/3386.99	**1793/54.43**	**1793**/58.80
$(10^1,10^3)$	*/*	54172/1438.63	**927/26.36**	1474/46.06
$(10^1,10^4)$	*/*	108376/3482.75	**1477/46.23**	1614/53.75
$(10^2,10^3)$	*/*	92292/2475.48	**1251/40.43**	1269/44.66
$(10^2,10^4)$	*/*	78775/2116.42	**1992/67.83**	2934/96.44
$(10^2,10^5)$	*/*	137886/3875.34	**3940/134.06**	4725/165.68
$(10^3,10^3)$	*/*	47491/1280.58	**2238/73.86**	2805/92.48
$(10^3,10^4)$	*/*	49708/596.66	6035/222.00	**4698/190.47**
$(10^3,10^5)$	*/*	52883/1481.81	**3863/161.51**	6110/216.05
$(10^3,10^6)$	*/*	108743/4083.58	**3396/139.33**	5083/158.99

Table 4.9 Iteration counts and runtimes (in seconds) for the Nonconvex QP problem in § 4.2.1. The tolerances are set to 10^{-6}. Entries marked with * did not converge in the time limit of 21600 seconds. The ATR metric is 0.1000.

As seen from Table 4.8 and Table 4.9, ASL was the best performing method converging the fastest in 64.3% and 92.9% of the instances tested for tolerances 10^{-4} and 10^{-6}, respectively. ASL-2 was the second-best performing method converging the fastest in 50% and 35.7% of the instances tested for tolerances 10^{-4} and 10^{-6}, respectively.

4.2.2 Nonconvex QP with Box Constraints

Given a pair of dimensions $(\ell, n) \in \mathbb{N}^2$, a scalar triple $(r, \tau_1, \tau_2) \in \Re^3_{++}$, matrices $A, C \in \Re^{\ell\times n}$ and $B \in \Re^{n\times n}$, positive diagonal matrix $D \in \Re^{n\times n}$, and a vector pair $(b, d) \in \Re^\ell \times \Re^\ell$, this § 4.2.2 considers the problem

$$\begin{aligned} \min_z \ & \left[f(z) := -\frac{\tau_1}{2}\|DBz\|^2 + \frac{\tau_2}{2}\|Cz - d\|^2 \right] \\ \text{s.t. } & Az = b, \\ & -r \le z_i \le r, \quad i \in \{1, ..., n\}. \end{aligned}$$

Parameters	**Iteration Count/Runtime (seconds)**						
(r,m_f,L_f)	iALM	IPL	RQP	ASL	ASL-2	SPA1	SPA2
$(5,10^0,10^1)$	203310/226.93	11274/17.93	49512/92.43	**7247/1.69**	9308/2.21	205576/335.91	1943184/2879.25
$(10,10^0,10^1)$	221433/334.57	9170/14.29	70736/132.59	**7043/1.67**	7178/1.80	128567/240.07	1176352/2139.45
$(20,10^0,10^1)$	192970/307.75	8363/14.59	58980/144.06	**5469/1.30**	5546/1.31	154035/374.16	1403641/2295.86
$(1,10^1,10^2)$	465159/858.38	*/*	326336/1156.69	**4509/1.08**	5455/1.32	133522/213.68	303003/524.57
$(2,10^1,10^2)$	862136/1141.23	*/*	399982/814.19	**8453/2.01**	10058/2.38	64280/107.55	447451/693.07
$(5,10^1,10^2)$	1857919/2476.33	*/*	174005/394.47	**8320/2.06**	11574/2.69	106715/238.12	488965/879.75
$(1,10^1,10^3)$	351468/510.03	*/*	47007/81.74	**8438/1.98**	9277/2.25	47583/65.33	123195/166.48
$(2,10^1,10^3)$	368578/481.14	*/*	69875/129.77	**6200/1.49**	7621/1.77	96971/123.39	161433/198.84
$(5,10^1,10^3)$	280346/329.16	*/*	116988/232.13	**5218/1.23**	6980/1.64	272448/361.41	161327/216.67
$(1,10^2,10^3)$	727587/908.05	*/*	104411/205.03	**4200/1.04**	4726/1.15	*/*	112604/154.60
$(2,10^2,10^3)$	964734/1225.22	21472/44.19	130903/253.02	**6432/1.56**	8113/1.99	*/*	53266/74.85
$(5,10^2,10^3)$	705884/890.93	11709/25.93	117945/226.21	**5137/1.26**	6124/1.45	*/*	47237/65.84
$(1,10^2,10^4)$	576627/864.79	255622/575.34	100193/200.17	**7796/1.85**	9085/2.16	155586/232.28	183307/274.50
$(2,10^2,10^4)$	1028921/1477.99	29123/57.82	165257/314.62	**7048/1.68**	15508/3.57	158192/256.01	196930/308.35
$(5,10^3,10^3)$	*/*	142961/253.35	225865/439.62	**26333/5.92**	45890/10.22	*/*	*/*
$(10,10^3,10^3)$	2474551/3522.28	*/*	168397/330.62	**14213/3.27**	80951/18.00	*/*	*/*
$(1,10^3,10^4)$	435881/667.19	71369/154.75	*/*	4724/1.16	**4633/1.03**	*/*	*/*
$(2,10^3,10^4)$	476462/584.73	23931/39.52	64649/100.27	**8971/2.12**	11009/2.59	*/*	*/*
$(5,10^3,10^4)$	521072/649.28	9829/17.02	*/*	**5943/1.45**	21743/5.01	*/*	*/*
$(1,10^3,10^5)$	*/*	347105/696.61	*/*	**8952/2.12**	31893/7.14	*/*	142702/231.41
$(2,10^3,10^5)$	1436029/2222.25	*/*	*/*	**9013/2.13**	11148/2.58	*/*	163317/397.06
$(5,10^3,10^5)$	*/*	106935/276.73	*/*	**11629/2.73**	12851/2.96	*/*	145047/192.72

Table 4.10 Iteration counts and runtimes (in seconds) for the Nonconvex QP problem with box constraints in § 4.2.2. The tolerances are set to 10^{-5}. Entries marked with * did not converge in the time limit of 3600 seconds. The ATR metric is 0.0097.

For our experiments in § 4.2.2, we choose dimensions $(l,n) = (20,100)$ and generate the matrices A, B, and C to be fully dense. The entries of A, B, C, and d (resp. D) are generated by sampling from the uniform distribution $\mathcal{U}[0,1]$ (resp. $\mathcal{U}[1,1000]$). We generate the vector b as $b = A(u)$ where u is a random vector in $\mathcal{U}[-r,r]^n$. The initial starting point z_0 is generated as a random vector in $\mathcal{U}[-r,r]^n$. We vary r across the different instances. Finally, we choose $(\tau_1,\tau_2) \in \Re^2_{++}$ so that $L_f = \lambda_{\max}(\nabla^2 f)$ and $m_f = -\lambda_{\min}(\nabla^2 f)$ are the various values given in the tables of this subsection.

We now describe the specific parameters that ASL, ASL-2, and RQP choose for this class of problems. ASL, ASL-2, and RQP choose the initial penalty parameter, $c_0 = 1$. Both ASL and ASL-2 allow the prox stepsize to be doubled at the end of an iteration if the number of iterations by its ACG call does not exceed 75. Finally, ASL and ASL-2 also take M_0^1 defined in step 1 of AS-PAL to be 100 and the initial prox stepsize to be $20/m_f$.

The numerical results are presented in Table 4.10. Table 4.10 compares ASL and ASL-2 with five of the benchmark algorithms namely, iALM, IPL, RQP, SPA1, and SPA2. The tolerances are set as $\hat{\rho} = \hat{\eta} = 10^{-5}$ and a time limit of 3600 seconds, or 1 hour, is imposed. Entries marked with * did not converge in the time limit.

As seen from Table 4.10, ASL was the best performing method converging the fastest in 95.5% of the instances tested. On average, ASL also only took roughly 0.0097 amount of time that RQP took to converge.

4.3 Comments about the numerical results

Overall, the adaptive methods ASL, ASL-2, and RQP were the most reliable and consistent, converging in almost every instance. ASL was clearly the most efficient, often converging much faster than RQP particularly when the required accuracy was high. As demonstrated by the results in Tables 4.5 and 4.6 and the ones in Tables 4.8 and 4.9, the ATR metric improves (decreases) as the required accuracy increases. Finally, ASL worked extremely fast on the problem classes of § 4.1.1 and § 4.2.2 as demonstrated by the results in Tables 4.1 and 4.10, respectively.

Acknowledgements The authors were partially supported by AFORS Grant FA9550-22-1-0088.

Code and Data Availability The code and data used for the experiments in this paper are publicly available in the AS-PAL GitHub repository [6].

Declarations

Conflict of Interest The authors declare they have no conflict of interest.

A ADAP-FISTA algorithm

A.1 ADAP-FISTA method

This subsection presents an adaptive ACG variant, called ADAP-FISTA, which is an important tool in the development of the AS-PAL method. We first introduce the assumptions on the problem it solves. ADAP-FISTA considers the following problem

$$\min\{\psi(x) := \psi_s(x) + \psi_n(x) : x \in \Re^n\} \tag{A.1}$$

where ψ_s and ψ_n are assumed to satisfy the following assumptions:

(I) $\psi_n : \Re^n \to \Re \cup \{+\infty\}$ is a possibly nonsmooth convex function;
(II) $\psi_s : \Re^n \to \Re$ is a differentiable function and there exists $\bar{L} \geq 0$ such that

$$\|\nabla\psi_s(z') - \nabla\psi_s(z)\| \leq \bar{L}\|z' - z\| \quad \forall z, z' \in \Re^n. \tag{A.2}$$

We now describe the type of approximate solution that ADAP-FISTA aims to find.

Problem A: Given ψ satisfying the above assumptions, a point $x_0 \in \operatorname{dom}\psi_n$, a parameter $\sigma \in (0, \infty)$, the problem is to find a pair $(y, u) \in \operatorname{dom}\psi_n \times \Re^n$ such that

$$\|u\| \leq \sigma\|y - x_0\|, \quad u \in \nabla\psi_s(y) + \partial\psi_n(y). \tag{A.3}$$

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

ADAP-FISTA Method

0. Let initial point $x_0 \in \operatorname{dom}\psi_n$ and scalars $\mu > 0$, $L_0 > \mu$, $\chi \in (0, 1)$, $\beta > 1$, and $\sigma > 0$ be given, and set $y_0 = x_0$, $A_0 = 0$, $\tau_0 = 1$, and $j = 0$;
1. Set $L_{j+1} = L_j$;
2. Compute

$$a_j = \frac{\tau_j + \sqrt{\tau_j^2 + 4\tau_j A_j(L_{j+1} - \mu)}}{2(L_{j+1} - \mu)}, \quad \widetilde{x}_j = \frac{A_j y_j + a_j x_j}{A_j + a_j}, \tag{A.4}$$

$$y_{j+1} := \operatorname*{arg\,min}_{u \in \operatorname{dom}\psi_n} \left\{ q_j(u; \widetilde{x}_j, L_{j+1}) := \ell_{\psi_s}(u; \tilde{x}_j) + \psi_n(u) + \frac{L_{j+1}}{2}\|u - \widetilde{x}_j\|^2 \right\}, \tag{A.5}$$

If the inequality

$$\ell_{\psi_s}(y_{j+1}; \tilde{x}_j) + \frac{(1-\chi)L_{j+1}}{4}\|y_{j+1} - \tilde{x}_j\|^2 \geq \psi_s(y_{j+1}) \tag{A.6}$$

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

$$A_{j+1} = A_j + a_j, \quad \tau_{j+1} = \tau_j + a_j\mu, \tag{A.7}$$

$$s_{j+1} = (L_{j+1} - \mu)(\tilde{x}_j - y_{j+1}), \tag{A.8}$$

$$x_{j+1} = \frac{1}{\tau_{j+1}}\left[\mu a_j y_{j+1} + \tau_j x_j - a_j s_{j+1}\right]; \tag{A.9}$$

[6] See https://github.com/asujanani6/AS-PAL.

4. If the inequality

$$\|y_{j+1} - x_0\|^2 \geq \chi A_{j+1} L_{j+1} \|y_{j+1} - \tilde{x}_j\|^2, \tag{A.10}$$

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

5. Compute

$$u_{j+1} = \nabla\psi_s(y_{j+1}) - \nabla\psi_s(\tilde{x}_j) + L_{j+1}(\tilde{x}_j - y_{j+1}). \tag{A.11}$$

If the inequality

$$\|u_{j+1}\| \leq \sigma \|y_{j+1} - x_0\| \tag{A.12}$$

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

We now make some remarks about ADAP-FISTA. First, usual FISTA methods for solving the strongly convex version of (A.1) consist of repeatedly invoking only steps 2 and 3 of ADAP-FISTA either with a static Lipschitz constant (of the gradient), namely, $L_{j+1} = L$ for all $j \geq 0$ for some $L \geq \bar{L}$, or by adaptively searching for a suitable Lipschitz L_{j+1} (as in step 2 of ADAP-FISTA) satisfying a condition similar to (A.6). Second, the pair (y_{j+1}, u_{j+1}) always satisfies the inclusion in (A.3) (see Lemma A.3 below) so if ADAP-FISTA stops successfully in step 5, or equivalently (A.12) holds, the pair solves Problem A above. Finally, if condition (A.10) in step 4 is never violated, ADAP-FISTA must stop successfully in step 5 (see Proposition A.1 below).

We now discuss how ADAP-FISTA compares with existing ACG variants for solving (A.1) under the assumption that ψ_s is μ-strongly convex. Under this assumption, FISTA variants have been studied, for example, in [3, 11, 12, 28, 30], while other ACG variants have been studied, for example, in [7, 8, 31]. A crucial difference between ADAP-FISTA and these variants is that: i) ADAP-FISTA stops based on a different relative criterion, namely, (A.12) (see Problem A above) and attempts to approximately solve (A.1) in this sense even when ψ_s is not μ-strongly convex, and ii) ADAP-FISTA provides a key and easy to check inequality whose validity at every iteration guarantees its successful termination. On the other hand, ADAP-FISTA shares similar features with these other methods in that: i) it has a reasonable iteration complexity guarantee regardless of whether it succeeds or fails, and ii) it successfully terminates when ψ_s is μ-strongly convex (see Propositions A.1-A.2 below). Moreover, like the method in [3], ADAP-FISTA adaptively searches for a suitable Lipschitz estimate L_{j+1} that is used in (A.5).

We now present the main convergence results of ADAP-FISTA, which is invoked by AS-PAL for solving the sequence of subproblems (1.4). The first result, namely Proposition A.1 below, gives an iteration complexity bound regardless if ADAP-FISTA terminates with success or failure and shows that if ADAP-FISTA successfully stops, then it obtains a stationary solution of (A.1) with respect to a relative error criterion. The second result, namely Proposition A.2 below, shows that ADAP-FISTA always stops successfully whenever ψ_s is μ-strongly convex.

Proposition A.1 *The following statements about ADAP-FISTA hold:*

(a) if $L_0 = \mathcal{O}(\bar{L})$, it always stops (with either success or failure) in at most

$$\mathcal{O}_1\left(\sqrt{\frac{\bar{L}}{\mu}}\log_1^+(\bar{L})\right)$$

iterations/resolvent evaluations;

(b) if it stops successfully, it terminates with a triple $(y, u, L) \in \operatorname{dom}\psi_n \times \Re^n$ satisfying

$$u \in \nabla\psi_s(y) + \partial\psi_n(y), \quad \|u\| \leq \sigma\|y - x_0\|, \quad L \leq \max\{L_0, \omega\bar{L}\}. \tag{A.13}$$

Proposition A.2 *If ψ_s is μ-convex, then ADAP-FISTA always terminates with success and its output (y, u, L), in addition to satisfying (A.13) also satisfies the inclusion $u \in \partial(\psi_s + \psi_n)(y)$.*

The rest of this section is broken up into two subsections which are dedicated to proving Proposition A.1 and Proposition A.2, respectively.

A.2 Proof of Proposition A.1

This subsection is dedicated to proving Proposition A.1. The first lemma below presents key definitions and inequalities used in the convergence analysis of ADAP-FISTA.

Lemma A.3 *Define*

$$\omega = 2\beta/(1-\chi), \quad \zeta := \bar{L} + \max\{L_0, \omega\bar{L}\}. \tag{A.14}$$

Then, the following statements hold:

(a) $\{L_j\}$ is nondecreasing;
(b) for every $j \geq 0$, we have

$$\tau_j = 1 + A_j\mu, \quad \frac{\tau_j A_{j+1}}{a_j^2} = L_{j+1} - \mu; \tag{A.15}$$

$$L_0 \leq L_j \leq \max\{L_0, \omega\bar{L}\}; \tag{A.16}$$

$$u_{j+1} \in \nabla\psi_s(y_{j+1}) + \partial\psi_n(y_{j+1}), \quad \|u_{j+1}\| \leq \zeta\|y_{j+1} - \tilde{x}_j\|. \tag{A.17}$$

Proof (a) It is clear from the update rule in the beginning of Step 1 that $\{L_j\}$ is nondecreasing.

(b) The first equality in (A.15) follows directly from both of the relations in (A.7). The second equality in (A.15) follows immediately from the definition of a_j in (A.4) and the first relation in (A.7).

We prove (A.16) by induction. It clearly holds for $j = 0$. Suppose now (A.16) holds for $j \geq 0$ and let us show that it holds for $j+1$. Note that if $L_{j+1} = L_j$, then relation (A.16) immediately holds. Assume then that $L_{j+1} > L_j$. It then follows from the way L_{j+1} is chosen in step 1 that (A.6) is not satisfied with L_{j+1}/β. This fact together with the inequality (A.2) at the points $(y_{j+1}, \tilde{x}_j)$ imply that

$$\ell_{\psi_s}(y_{j+1}; \tilde{x}_j) + \frac{(1-\chi)L_{j+1}}{4\beta}\|y_{j+1} - \tilde{x}_j\|^2 < \psi_s(y_{j+1}) \overset{(A.2)}{\leq} \ell_{\psi_s}(y_{j+1}; \tilde{x}_j) + \frac{\bar{L}}{2}\|y_{j+1} - \tilde{x}_j\|^2. \tag{A.18}$$

The relation in (A.16) then immediately follows from the definition of ω in (A.14).

Now, by the definition of u_{j+1} in (A.11), triangle inequality, (A.2), the bound (A.16) on L_{j+1}, and the definition of ζ we have

$$\frac{\|u_{j+1}\|}{\|y_{j+1} - \widetilde{x}_j\|} \overset{(A.11)}{\leq} \frac{\|\nabla\psi_s(y_{j+1}) - \nabla\psi_s(\widetilde{x}_j)\|}{\|y_{j+1} - \widetilde{x}_j\|} + L_{j+1} \overset{(A.2)}{\leq} \bar{L} + L_{j+1} \overset{(A.16)}{\leq} \zeta$$

which immediately implies the inequality in (A.17). It follows from (A.5) and its associated optimality condition that $0 \in \nabla\psi_s(\widetilde{x}_j) + \partial\psi_n(y_{j+1}) - L_{j+1}(\widetilde{x}_j - y_{j+1})$, which in view of the definition of u_{j+1} in (A.11) implies the inclusion in (A.17). ∎

The result below gives some estimates on the sequence $\{A_j\}$, which will be important for the convergence analysis of the method.

Lemma A.4 *Define*

$$Q := 2\sqrt{\frac{\max\{L_0, \omega\bar{L}\}}{\mu}} \tag{A.19}$$

where ω is as in (A.14). Then, for every $j \geq 1$, we have

$$A_j L_j \geq \max\left\{\frac{j^2}{4}, \left(1 + Q^{-1}\right)^{2(j-1)}\right\}. \tag{A.20}$$

Proof Let integer $j \geq 1$ be given. Define $\xi_j = 1/(L_j - \mu)$. Using the first equality in (A.7) and the definition of a_j in (A.4), we have that for every $i \leq j$,

$$A_i \overset{(A.7)}{=} A_{i-1} + a_{i-1} \overset{(A.4)}{\geq} A_{i-1} + \left(\frac{\tau_{i-1}\xi_i}{2} + \sqrt{\tau_{i-1}\xi_i A_{i-1}}\right) \geq \left(\sqrt{A_{i-1}} + \frac{1}{2}\sqrt{\tau_{i-1}\xi_i}\right)^2.$$

Passing the above inequality to its square root and using Lemma A.3(a) and the fact that (A.15) implies that $\tau_{i-1} \geq \max\{1, \mu A_{i-1}\}$, we then conclude that for every $i \leq j$,

$$\sqrt{A_i} - \sqrt{A_{i-1}} \geq \frac{1}{2}\sqrt{\xi_i} \geq \frac{1}{2}\sqrt{\xi_j} \tag{A.21}$$

$$\sqrt{\frac{A_i}{A_{i-1}}} \geq 1 + \frac{1}{2}\sqrt{\mu\xi_i} \geq 1 + \frac{1}{2}\sqrt{\mu\xi_j} \geq 1 + Q^{-1} \tag{A.22}$$

where the last inequality in (A.22) follows from the definition of ξ_j, the relation in (A.16), and the definition of Q in (A.19). Adding the inequality in (A.21) from $i = 1$ to $i = j$ and using the fact that $A_0 = 0$, we conclude that $\sqrt{A_j} \geq j\sqrt{\xi_j}/2$ and hence that the first bound in (A.20) holds in view of the fact that $\xi_j \geq 1/L_j$. Now, multiplying the inequality in (A.22) from $i = 2$ to $i = j$ and using Lemma A.3(a) and the fact that $A_1 = \xi_1$, we conclude that $\sqrt{A_j} \geq \sqrt{\xi_1}(1 + Q^{-1})^{j-1} \geq \sqrt{\xi_j}(1 + Q^{-1})^{j-1}$, and hence that the second bound in (A.20) holds in view of the fact that $\xi_j \geq 1/L_j$. ∎

Proposition A.5 *Let ζ and Q be as in (A.14) and (A.19), respectively. ADAP-FISTA always stops (with either success or failure) and does so by performing at most*

$$\left\lceil (1+Q)\log_1^+\left(\frac{\zeta^2}{\chi\sigma^2}\right) + 1 \right\rceil + \left\lceil \frac{2\log_0^+(\bar{L}/((1-\chi)L_0))}{\log\beta} \right\rceil \tag{A.23}$$

iterations/resolvent evaluations.

Proof Let l denote the first quantity in (A.23). Using this definition and the inequality $\log(1+\alpha) \geq \alpha/(1+\alpha)$ for any $\alpha > -1$, it is easy to verify that

$$\left(1 + Q^{-1}\right)^{2(l-1)} \geq \frac{\zeta^2}{\chi\sigma^2}. \tag{A.24}$$

We claim that ADAP-FISTA terminates with success or failure in at most l iterations. Indeed, it suffices to show that if ADAP-FISTA has not stopped with failure up to (and including) the l-th iteration, then it must stop successfully at the l-th iteration. So, assume that ADAP-FISTA has not stopped with failure up to the l-th iteration. In view of step 4 of ADAP-FISTA, it follows that (A.10) holds with $j = l - 1$.

This observation together with the inequality in (A.17) with $j = l-1$, (A.20) with $j = l$, and (A.24), then imply that

$$\|y_l - x_0\|^2 \overset{(A.10)}{\geq} \chi A_l L_l \|y_l - \tilde{x}_{l-1}\|^2 \overset{(A.17)}{\geq} \frac{\chi}{\zeta^2} A_l L_l \|u_l\|^2 \overset{(A.20)}{\geq} \frac{\chi}{\zeta^2}\left(1 + Q^{-1}\right)^{2(l-1)} \|u_l\|^2 \overset{(A.24)}{\geq} \frac{1}{\sigma^2}\|u_l\|^2, \tag{A.25}$$

and hence that (A.12) is satisfied. In view of Step 5 of ADAP-FISTA, the method must successfully stop at the end of the l-th iteration. We have thus shown that the above claim holds. Moreover, in view of (A.16), it follows that the second term in (A.23) is a bound on the total number of times L_j is multiplied by β and step 2 is repeated. Since exactly one resolvent evaluation occurs every time step 2 is executed, the desired conclusion follows. ∎

We are now ready to give the proof of Proposition A.1.

Proof (of Proposition A.1) (a) The result immediately follows from Proposition A.5 and the assumption that $L_0 = \mathcal{O}(\bar{L})$.

(b) This is immediate from the termination criterion (A.12) in step 5 of ADAP-FISTA, the inclusion in (A.17), and relation (A.16).

∎

A.3 Proof of Proposition A.2

This subsection is dedicated to proving Proposition A.2. Thus, for the remainder of this subsection, assume that ψ_s is μ-strongly convex. The first lemma below presents important properties of the iterates generated by ADAP-FISTA.

Lemma A.6 *For every $j \geq 0$ and $x \in \Re^n$, define*

$$\gamma_j(x) := \ell_{\psi_s}(y_{j+1}, \tilde{x}_j) + \psi_n(y_{j+1}) + \langle s_{j+1}, x - y_{j+1}\rangle + \frac{\mu}{2}\|y_{j+1} - \tilde{x}_j\|^2 + \frac{\mu}{2}\|x - y_{j+1}\|^2, \tag{A.26}$$

where $\psi := \psi_s + \psi_n$ and s_{j+1} are as in (A.1) and (A.8), respectively. Then, for every $j \geq 0$, we have:

$$y_{j+1} = \arg\min_x \left\{\gamma_j(x) + \frac{L_{j+1} - \mu}{2}\|x - \tilde{x}_j\|^2\right\}; \tag{A.27}$$

$$x_{j+1} = \arg\min_{x \in \Re^n} \left\{a_j\gamma_j(x) + \tau_j\|x - x_j\|^2/2\right\}. \tag{A.28}$$

Proof Since $\nabla\gamma_j(y_{j+1}) = s_{j+1}$, it follows from (A.8) that y_{j+1} satisfies the optimality condition for (A.27), and thus the relation in (A.27) follows. Furthermore, we have that:

$$\begin{aligned} a_j\nabla\gamma_j(x_{j+1}) + \tau_j(x_{j+1} - x_j) &= a_j s_{j+1} + a_j\mu(x_{j+1} - y_{j+1}) + \tau_j(x_{j+1} - x_j) \\ &\overset{(A.7)}{=} a_j s_{j+1} - \mu a_j y_{j+1} - \tau_j x_j + \tau_{j+1}x_{j+1} \overset{(A.9)}{=} 0 \end{aligned}$$

and thus (A.28) follows. ∎

Before stating the next lemma, recall that if a closed function $\Psi : \Re^n \to \Re \cup \{+\infty\}$ is ν-convex with modulus $\nu > 0$, then it has an unique global minimum z^* and

$$\Psi(z^*) + \frac{\nu}{2}\|\cdot - z^*\|^2 \leq \Psi(\cdot). \tag{A.29}$$

Lemma A.7 *For every $j \geq 0$ and $x \in \Re^n$, we have*

$$\begin{aligned} A_j\gamma_j(y_j) + a_j\gamma_j(x) + \frac{\tau_j}{2}\|x_j - x\|^2 - \frac{\tau_{j+1}}{2}\|x_{j+1} - x\|^2 \\ \geq A_{j+1}\psi(y_{j+1}) + \frac{\chi A_{j+1}L_{j+1}}{2}\|y_{j+1} - \tilde{x}_j\|^2. \end{aligned} \tag{A.30}$$

Proof Using (A.28), the second identity in (A.7), and the fact that $\Psi_j := a_j\gamma_j(\cdot) + \tau_j\|\cdot - x_j\|^2/2$ is $(\tau_j + \mu a_j)$-convex, it follows from (A.29) with $\Psi = \Psi_j$ and $\nu = \tau_{j+1}$ that

$$a_j\gamma_j(x) + \frac{\tau_j}{2}\|x - x_j\|^2 - \frac{\tau_{j+1}}{2}\|x - x_{j+1}\|^2 \geq a_j\gamma_j(x_{j+1}) + \frac{\tau_j}{2}\|x_{j+1} - x_j\|^2 \quad \forall x \in \Re^n.$$

Using the convexity of γ_j, the definitions of A_{j+1} and $\widetilde{x}_j$ in (A.7) and (A.4), respectively, and the second equality in (A.15), we have

$$\begin{aligned} A_j\gamma_j(y_j) + a_j\gamma_j(x_{j+1}) + \frac{\tau_j}{2}\|x_{j+1} - x_j\|^2 \\ &\geq A_{j+1}\gamma_j\left(\frac{A_j y_j + a_j x_{j+1}}{A_{j+1}}\right) + \frac{\tau_j A_{j+1}^2}{2a_j^2}\left\|\frac{A_j y_j + a_j x_{j+1}}{A_{j+1}} - \frac{A_j y_j + a_j x_j}{A_{j+1}}\right\|^2 \\ &\overset{(A.4)}{\geq} A_{j+1}\min_x\left[\gamma_j(x) + \frac{\tau_j A_{j+1}}{2a_j^2}\|x - \widetilde{x}_j\|^2\right] \\ &\overset{(A.15)}{=} A_{j+1}\min_x\left\{\gamma_j(x) + \frac{L_{j+1} - \mu}{2}\|x - \widetilde{x}_j\|^2\right\} \\ &\overset{(A.27)}{=} A_{j+1}\left[\gamma_j(y_{j+1}) + \frac{L_{j+1} - \mu}{2}\|y_{j+1} - \widetilde{x}_j\|^2\right] \\ &\overset{(A.26)}{=} A_{j+1}\left[\ell_{\psi_s}(y_{j+1}; \tilde{x}_j) + \psi_n(y_{j+1}) + \frac{L_{j+1}}{2}\|y_{j+1} - \widetilde{x}_j\|^2\right] \\ &\overset{(A.6)}{\geq} A_{j+1}\left[\psi(y_{j+1}) + \frac{\chi L_{j+1}}{2}\|y_{j+1} - \widetilde{x}_j\|^2\right]. \end{aligned}$$

The conclusion of the lemma now follows by combining the above two relations. ∎

Lemma A.8 *For every $j \geq 0$, we have $\gamma_j \leq \psi$.*

Proof Define:

$$\tilde{\gamma}_j(x) := \ell_{\psi_s}(x;\tilde{x}_j) + \psi_n(x) + \frac{\mu}{2}\|x-\tilde{x}_j\|^2. \tag{A.31}$$

It follows immediately from the fact that ψ_s is μ-convex that $\tilde{\gamma}_j \leq \psi$. Furthermore, immediately from the definition of y_{j+1} in (A.5), we can write:

$$y_{j+1} = \arg\min_x \left\{ \tilde{\gamma}_j(x) + \frac{L_{j+1}-\mu}{2}\|x-\tilde{x}_j\|^2 \right\}. \tag{A.32}$$

Now, clearly from (A.32) and the definition of s_{j+1} in (A.8), we see that $s_{j+1} \in \partial\tilde{\gamma}_j(y_{j+1})$. Furthermore, since $\tilde{\gamma}_j$ is μ-convex, it follows from the subgradient rule for the sum of convex functions that the above inclusion is equivalent to $s_{j+1} \in \partial\left(\tilde{\gamma}_j(\cdot) - \frac{\mu}{2}\|\cdot - y_{j+1}\|^2\right)(y_{j+1})$. Hence, the subgradient inequality and the fact that $\tilde{\gamma}_j(x) \leq \psi(x)$ imply that for all $x \in \Re^n$:

$$\psi(x) \geq \tilde{\gamma}_j(x) \geq \tilde{\gamma}_j(y_{j+1}) + \langle s_{j+1}, x - y_{j+1}\rangle + \frac{\mu}{2}\|x-y_{j+1}\|^2 = \gamma_j(x)$$

and thus the statement of the lemma follows. ∎

Lemma A.9 *For every $j \geq 0$ and $x \in \operatorname{dom}\psi_n$, we have*

$$\eta_j(x) - \eta_{j+1}(x) \geq \frac{\chi A_{j+1}L_{j+1}}{2}\|y_{j+1}-\widetilde{x}_j\|^2$$

where

$$\eta_j(x) := A_j[\psi(y_j) - \psi(x)] + \frac{\tau_j}{2}\|x-x_j\|^2.$$

Proof Subtracting $A_{j+1}\psi(x)$ from both sides of the inequality in (A.30) and using Lemma A.8 we have

$$\begin{aligned} A_j\psi(y_j) + a_j\psi(x) - A_{j+1}\psi(x) + \frac{\tau_j}{2}\|x_j-x\|^2 - \frac{\tau_{j+1}}{2}\|x_{j+1}-x\|^2 \\ \geq A_{j+1}\psi(y_{j+1}) - A_{j+1}\psi(x) + \frac{\chi A_{j+1}L_{j+1}}{2}\|y_{j+1}-\widetilde{x}_j\|^2. \end{aligned}$$

The result now follows from the first equality in (A.7) and the definition of $\eta_j(x)$. ∎

We now state a result that will be important for deriving complexity bounds for ADAP-FISTA.

Lemma A.10 *For every $j \geq 0$ and $x \in \operatorname{dom}\psi_n$, we have*

$$A_j[\psi(y_j) - \psi(x)] + \frac{\tau_j}{2}\|x-x_j\|^2 \leq \frac{1}{2}\|x-x_0\|^2 - \frac{\chi}{2}\sum_{i=0}^{j-1} A_{i+1}L_{i+1}\|y_{i+1}-\widetilde{x}_i\|^2. \tag{A.33}$$

Proof Summing the inequality of Lemma A.9 from $j=0$ to $j=j-1$, using the facts that $A_0=0$ and $\tau_0=1$, and using the definition of $\eta_j(\cdot)$ in Lemma A.9 gives us the inequality of the lemma. ∎

We are now ready to give the proof of Proposition A.2.

Proof (of Proposition A.2) Since ψ_s is μ-convex, Lemma A.10 holds. Thus, using (A.33) with $x = y_j$, it follows that for all $j \geq 0$:

$$\|y_j - x_0\|^2 \overset{(A.33)}{\geq} \chi\sum_{i=1}^{j} A_iL_i\|y_i - \tilde{x}_{i-1}\|^2 \geq \chi A_jL_j\|y_j-\tilde{x}_{j-1}\|^2. \tag{A.34}$$

Hence, for all $j \geq 0$, relation (A.10) in step 4 of ADAP-FISTA is always satisfied and thus ADAP-FISTA never fails. In view of this observation and Proposition A.1, it follows that if ψ_s is μ-convex then ADAP-FISTA always terminates successfully with a (y,u,L) satisfying relation (A.13) in a finite number of iterations. The inclusion $u \in (\psi_s+\psi_n)(y)$ then follows immediately from the inclusion in (A.13) and the subgradient rule for the sum of convex functions. ∎

B Technical Results for Proof of Lagrange Multipliers

The following basic result is used in Lemma B.3. Its proof can be found, for instance, in [4, Lemma A.4]. Recall that ν_A^+ denotes the smallest positive singular value of a nonzero linear operator A .

Lemma B.1 *Let $A : \Re^n \to \Re^l$ be a nonzero linear operator. Then,*

$$\nu_A^+ \|u\| \leq \|A^* u\|, \quad \forall u \in A(\Re^n).$$

The following technical result, whose proof can be found in Lemma 3.10 of [16], plays an important role in the proof of Lemma B.3 below.

Lemma B.2 *Let h be a function as in (C1). Then, for every $\delta \geq 0$, $z \in \mathcal{H}$, and $\xi \in \partial_\delta h(z)$, we have*

$$\|\xi\| \mathrm{dist}(u, \partial\mathcal{H}) \leq [\mathrm{dist}(u, \partial\mathcal{H}) + \|z - u\|] M_h + \langle \xi, z - u \rangle + \delta \quad \forall u \in \mathcal{H} \tag{B.1}$$

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

Lemma B.3 *Assume that h is a function as in condition (C1) and $A : \Re^n \to \Re^l$ is a linear operator satisfying condition (C2). Assume also that the triple $(z, q, r) \in \Re^n \times A(\Re^n) \times \Re^n$ satisfy $r \in \partial h(z) + A^* q$. Then:*

(a) there holds

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

(b) if, in addition,

$$q = q^- + \chi(Az - b) \tag{B.3}$$

for some $q^- \in \Re^l$ and $\chi > 0$, then we have

$$\|q\| \leq \max\left\{ \|q^-\|, \frac{2D_h(M_h + \|r\|)}{\bar{d}\nu_A^+} \right\}. \tag{B.4}$$

Proof (a) The assumption on (z, q, r) implies that $r - A^* q \in \partial h(z)$. Hence, using the Cauchy-Schwarz inequality, the definitions of $\bar{d}$ and $\bar{z}$ in (2.15) and (C2), respectively, and Lemma B.2 with $\xi = r - A^* q$, $u = \bar{z}$, and $\delta = 0$, we have:

$$\bar{d}\|r - A^* q\| - \left[\bar{d} + \|z - \bar{z}\|\right] M_h \overset{(B.1)}{\leq} \langle r - A^* q, z - \bar{z} \rangle \leq \|r\|\|z - \bar{z}\| - \langle q, Az - b \rangle. \tag{B.5}$$

Now, using the above inequality, the triangle inequality, the definition of D_h in (C1), and the facts that $\bar{d} \leq D_h$ and $\|z - \bar{z}\| \leq D_h$, we conclude that:

$$\bar{d}\|A^* q\| + \langle q, Az - b \rangle \overset{(B.5)}{\leq} \left[\bar{d} + \|z - \bar{z}\|\right] M_h + \|r\| \left(D_h + \bar{d}\right) \leq 2D_h \left(M_h + \|r\|\right). \tag{B.6}$$

Noting the assumption that $q \in A(\Re^n)$, inequality (B.2) now follows from the above inequality and Lemma B.1.

(b) Relation (B.3) implies that $\langle q, Az - b \rangle = \|q\|^2/\chi - \langle q^-, q \rangle/\chi$, and hence that

$$\bar{d}\nu_A^+ \|q\| + \frac{\|q\|^2}{\chi} \leq 2D_h(M_h + \|r\|) + \frac{\langle q^-, q \rangle}{\chi} \leq 2D_h(M_h + \|r\|) + \frac{\|q\|}{\chi}\|q^-\|, \tag{B.7}$$

where the last inequality is due to the Cauchy-Schwarz inequality. Now, letting K denote the right hand side of (B.4) and using (B.7), we conclude that

$$\left(\bar{d}\nu_A^+ + \frac{\|q\|}{\chi}\right)\|q\| \overset{(B.7)}{\leq} \left(\frac{2D_h(M_h + \|r\|)}{K} + \frac{\|q\|}{\chi}\right) K \leq \left(\bar{d}\nu_A^+ + \frac{\|q\|}{\chi}\right) K, \tag{B.8}$$

and hence that (B.4) holds. ∎

References

1. N.S. Aybat and G. Iyengar. A first-order smoothed penalty method for compressed sensing. *SIAM J. Optim.*, 21(1):287–313, 2011.
2. N.S. Aybat and G. Iyengar. A first-order augmented Lagrangian method for compressed sensing. *SIAM J. Optim.*, 22(2):429–459, 2012.
3. 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.
4. 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. *Pac. J. Optim.*, 15(3):379–398, 2019.
5. Q. Gu, Z. Wang, and H. Liu. Sparse pca with oracle property. In *Advances in Neural Information Processing Systems 27*, pages 1529–1537. Curran Associates, Inc., 2014.
6. D. Hajinezhad and M. Hong. Perturbed proximal primal-dual algorithm for nonconvex nonsmooth optimization. *Math. Program.*, 176:207–245, 2019.
7. 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(4):2182–2211, 2015.
8. Y. He and R.D.C. Monteiro. An accelerated HPE-type algorithm for a class of composite convex-concave saddle-point problems. *SIAM J. Optim.*, 26(1):29–56, 2016.
9. B. Jiang, T. Lin, S. Ma, and S. Zhang. Structured nonconvex and nonsmooth optimization algorithms and iteration complexity analysis. *Comput. Optim. Appl.*, 72(3):115–157, 2019.
10. W. Kong. Accelerated inexact first-order methods for solving nonconvex composite optimization problems. *arXiv:2104.09685*, April 2021.
11. W. Kong. Complexity-optimal and curvature-free first-order methods for finding stationary points of composite optimization problems. *Available on arXiv:2205.13055*, 2022.
12. W. Kong, J. G. Melo, and R.D.C. Monteiro. FISTA and Extensions - Review and New Insights. *Optimization Online*, 2021.
13. W. Kong, J.G. Melo, and R.D.C. Monteiro. Complexity of a quadratic penalty accelerated inexact proximal point method for solving linearly constrained nonconvex composite programs. *SIAM J. Optim.*, 29(4):2566–2593, 2019.
14. W. Kong, J.G. Melo, and R.D.C. Monteiro. An efficient adaptive accelerated inexact proximal point method for solving linearly constrained nonconvex composite problems. *Comput. Optim. Appl.*, 76(2):305–346, 2019.
15. W. Kong, J.G. Melo, and R.D.C. Monteiro. Iteration-complexity of a proximal augmented Lagrangian method for solving nonconvex composite optimization problems with nonlinear convex constraints. *Mathematics of Operations Research*, 2023.
16. 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.
17. W. Kong and R.D.C. Monteiro. An accelerated inexact proximal point method for solving nonconvex-concave min-max problems. *SIAM Journal on Optimization*, 31(4):2558–2585, 2021.
18. 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.
19. G. Lan and R.D.C. Monteiro. Iteration-complexity of first-order penalty methods for convex programming. *Math. Program.*, 138(1):115–139, Apr 2013.
20. G. Lan and R.D.C. Monteiro. Iteration-complexity of first-order augmented Lagrangian methods for convex programming. *Math. Program.*, 155(1):511–547, Jan 2016.
21. Z. Li, P.-Y. Chen, S. Liu, S. Lu, and Y. Xu. Rate-improved inexact augmented Lagrangian method for constrained nonconvex optimization. *Available on arXiv:2007.01284*, 2020.
22. Z. Li and Y. Xu. Augmented Lagrangian based first-order methods for convex and nonconvex programs: nonergodic convergence and iteration complexity. *arXiv e-prints*, pages arXiv–2003, 2020.
23. Q. Lin, R. Ma, and Y. Xu. Inexact proximal-point penalty methods for non-convex optimization with non-convex constraints. *Available on Arxiv:1908.11518*, 2019.
24. Q. Lin, R. Ma, and Y Xu. Inexact proximal-point penalty methods for constrained non-convex optimization. *Available on arXiv:1908.11518*, 2020.
25. Y.F. 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.
26. Z. Lu and Z. Zhou. Iteration-complexity of first-order augmented Lagrangian methods for convex conic programming. *Available on arXiv:1803.09941*, 2018.
27. J.G. 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.
28. R.D.C. Monteiro, C. Ortiz, and B.F. Svaiter. An adaptive accelerated first-order method for convex optimization. *Comput. Optim. Appl.*, 64:31–73, 2016.
29. I. Necoara, A. Patrascu, and F. Glineur. Complexity of first-order inexact Lagrangian and penalty methods for conic convex programming. *Optim. Methods Softw.*, pages 1–31, 2017.
30. Y. E. Nesterov. *Introductory lectures on convex optimization : a basic course*. Kluwer Academic Publ., 2004.
31. Y.E. Nesterov. Gradient methods for minimizing composite functions. *Math. Program.*, pages 1–37, 2012.
32. 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.
33. M. Sahin, A. Eftekhari, A. Alacaoglu, F. Latorre, and V Cevher. An inexact augmented Lagrangian framework for nonconvex optimization with nonlinear constraints. *Available on arXiv:1906.11357*, 2019.
34. K. Sun and A. Sun. Dual Descent ALM and ADMM. *Available on arXiv:2109.13214*, 2022.
35. Y. Xu. Iteration complexity of inexact augmented Lagrangian methods for constrained convex programming. *Math. Program.*, 2019.
36. Q. Yao and J.T. Kwok. Efficient learning with a family of nonconvex regularizers by redistributing nonconvexity. *J. Mach. Learn. Res.*, 18:179–1, 2017.

37. J. Zeng, W. Yin, and D. Zhou. Moreau Envelope Augmented Lagrangian method for Nonconvex Optimization with Linear Constraints. *J. Scientific Comp.*, 91(61), April 2022.
38. J. Zhang and Z.-Q. Luo. A global dual error bound and its application to the analysis of linearly constrained nonconvex optimization. *Available on arXiv:2006.16440*, 2020.
39. J. Zhang and Z.-Q. Luo. A proximal alternating direction method of multiplier for linearly constrained nonconvex minimization. *SIAM J. Optim.*, 30(3):2272–2302, 2020.
40. J. Zhang, W. Pu, and Z. Luo. On the Iteration Complexity of Smoothed Proximal ALM for Nonconvex Optimization Problem with Convex Constraints. *Available on arXiv:2207.06304*, 2022.