SIAM J. OPTIM.
Vol. 31, No. 1, pp. 217–243

AN AVERAGE CURVATURE ACCELERATED COMPOSITE GRADIENT METHOD FOR NONCONVEX SMOOTH COMPOSITE OPTIMIZATION PROBLEMS*

JIAMING LIANG† AND RENATO D. C. MONTEIRO†

Abstract. This paper presents an accelerated composite gradient (ACG) variant, referred to as the AC-ACG method, for solving nonconvex smooth composite minimization problems. As opposed to well-known ACG variants that are based on either a known Lipschitz gradient constant or a sequence of maximum observed curvatures, the current one is based on the average of all past observed curvatures. More specifically, AC-ACG uses a positive multiple of the average of all observed curvatures until the previous iteration as a way to estimate the "function curvature" at the current point and then two resolvent evaluations to compute the next iterate. In contrast to other variable Lipschitz estimation variants, e.g., the ones based on the maximum curvature, AC-ACG always accepts the aforementioned iterate regardless of how poor the Lipschitz estimation turns out to be. Finally, computational results are presented to illustrate the efficiency of AC-ACG on both randomly generated and real-world problem instances.

Key words. smooth nonconvex composite programming, average curvature, accelerated composite gradient methods, first-order methods, iteration-complexity, line search free methods

AMS subject classifications. 49M05, 49M37, 65K05, 65Y20, 68Q25, 90C26, 90C30

DOI. 10.1137/19M1294277

1. Introduction.

In this paper, we study an accelerated composite gradient–(ACG-) type algorithm for solving a nonconvex smooth composite optimization (SCO) problem

$$\phi_* := \min\{\phi(z) := f(z) + h(z) : z \in \mathbb{R}^n\}, \tag{1}$$

where f is a real-valued differentiable (possibly nonconvex) function with an M-Lipschitz continuous gradient on $\operatorname{dom} h$ and $h : \mathbb{R}^n \to (-\infty, \infty]$ is a proper lower semicontinuous convex function with a bounded domain.

A large class of algorithms for solving (1) sets the next iterate y_{k+1} as the unique optimal solution $y(\tilde{x}_k; M_k)$ of the linearized prox subproblem

$$y(\tilde{x}_k; M_k) := \operatorname{argmin}\left\{\ell_f(x; \tilde{x}_k) + h(x) + \frac{M_k}{2}\|x - \tilde{x}_k\|^2 : x \in \mathbb{R}^n\right\}, \tag{2}$$

where $\ell_f(x; \tilde{x}_k) := f(\tilde{x}_k) + \langle \nabla f(\tilde{x}_k), x - \tilde{x}_k\rangle$, the prox-center $\tilde{x}_k$ is chosen as either the current iterate y_k (as in unaccelerated algorithms) or a convex combination of y_k and another auxiliary iterate x_k (as in accelerated algorithms), and M_k is good upper curvature of f at $\tilde{x}_k$, i.e., $M_k > 0$, and satisfies

$$\mathcal{C}(y(\tilde{x}_k; M_k); \tilde{x}_k) \le M_k, \tag{3}$$

*Received by the editors October 21, 2019; accepted for publication (in revised form) October 25, 2020; published electronically January 13, 2021.
https://doi.org/10.1137/19M1294277
Funding: This work was partially supported by ONR grant N00014-18-1-2077. The first author was also partially supported by NSF grant CCF-1740776 through a joint fellowship from the Algorithms and Randomness Center (ARC) and the Transdisciplinary Research Institute for Advancing Data Science (TRIAD), the latter part of the TRIPODS program at NSF located at Georgia Tech (http://triad.gatech.edu).

†School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0205 USA (jiaming.liang@gatech.edu, monteiro@isye.gatech.edu).

where

$$\mathcal{C}(y;\tilde{x}) := \frac{2\,[f(y) - \ell_f(y;\tilde{x})]}{\|y-\tilde{x}\|^2}. \tag{4}$$

Regardless of the choice of $\tilde{x}_k$, it is well-known that the smaller the sequence $\{M_k\}$ is, the faster the convergence rate of the method becomes. Hence, it is desirable to choose $M_k = \bar{M}_k$ where $\bar{M}_k$, referred to as the local curvature of f at $\tilde{x}_k$, is the smallest value of M_k satisfying (3). However, since finding $\bar{M}_k$ is generally time-consuming, alternative strategies that upper estimate $\bar{M}_k$ are used. A common one is a backtracking procedure that initially sets M_k to be the maximum of all the observed curvatures $\mathcal{C}_1, \ldots, \mathcal{C}_{k-1}$ where $\mathcal{C}_i := \mathcal{C}(y_{i+1}; \tilde{x}_i)$ for every $i \geq 1$. It then checks whether M_k is a good curvature of f at $\tilde{x}_k$. If so, it sets $y_{k+1} = y(\tilde{x}_k; M_k)$; otherwise, it updates $M_k \leftarrow \eta M_k$ for some parameter $\eta > 1$, and then repeats this same step again. Such an approach has been used extensively in the literature dealing with composite gradient methods both in the context of convex and nonconvex SCO (N-SCO) problems (see, for example, [3, 7, 16, 22]) and can be efficient particularly for those SCO instances where a sharp upper bound M on the smallest Lipschitz constant $\bar{M}$ of ∇f on $\operatorname{dom} h$ is not available.

This paper investigates an ACG variant for solving the N-SCO problem where M_k is computed as a positive multiple of the average of all observed curvatures up to the previous iteration. As opposed to ACG variants based on the scheme outlined above as well as other ACG variants, AC-ACG always computes a new step regardless of whether M_k overestimates or underestimates $\mathcal{C}_k$. More specifically, if M_k overestimates $\mathcal{C}_k$, then a composite step as in (2) is taken; otherwise, y_{k+1} is set to be a convex combination of y_k and an auxiliary iterate x_{k+1}, which is obtained by a resolvent evaluation of h. It is worth noting that both of these steps are used in previous ACG variants but only one of them is used at a time. The main result of the paper establishes a convergence rate for AC-ACG. More specifically, it states that k iterations of the AC-ACG method generate a pair (y, v) satisfying $v \in \nabla f(y) + \partial h(y)$ and $\|v\|^2 = \mathcal{O}(M_k/k)$ where M_k is as in the beginning of this paragraph. Since M_k is usually much smaller than $\bar{M}$ or even $\bar{M}_k$, this convergence rate bound explains the efficiency of AC-ACG to solve both randomly generated and real-world problem instances of (1) used in our numerical experiments. Finally, it is shown that AC-ACG also has similar iteration-complexity as previous ACG variants (e.g., [6, 10, 15, 16]).

Related works. The first complexity analysis of an ACG algorithm for solving (1) under the assumption that f is a nonconvex differentiable function whose gradient is Lipschitz continuous and that h is a simple lower semicontinuous convex function is established in the novel work [6]. Inspired by [6], many papers have proposed other ACG variants for solving (1) under the aforementioned assumptions (see, e.g., [5, 7, 16]) or even under the relaxed assumption that h is nonconvex (see, e.g., [13, 14, 26]). It is worth mentioning that (i) in contrast to [6, 16], the other works deal with hybrid-type accelerated methods that resort to unaccelerated composite gradient steps whenever a certain descent property is not satisfied, and (ii) in contrast to the methods of [7, 13, 16] that choose M_k adaptively in a manner similar to that described in the second paragraph in section 1, the methods in [5, 6, 14, 26] work with a constant sequence $\{M_k\}$, namely, $M_k = M$ for some $M > \bar{M}$. Section 3 provides a more detailed overview of ACG variants for solving both convex and nonconvex SCO problems which includes most of the ones just mentioned.

Other approaches toward solving (1) use an inexact proximal point scheme where each prox subproblem is constructed to be (possibly strongly) convex and hence

efficiently solvable by a convex ACG variant. Papers [4, 10, 23] propose a descent unaccelerated inexact proximal-type method, which works with a larger prox stepsize and hence has a better outer iteration-complexity than the approaches in the previous paragraph. Paper [15] presents an accelerated inexact proximal point method that performs an accelerated step with a large prox stepsize in every outer iteration and requires a prox subproblem to be approximately solved by an ACG variant in the same way as in the algorithms presented in [4, 10].

Definitions and notations. The set of real numbers is denoted by $\mathbb{R}$. The set of nonnegative real numbers and the set of positive real numbers are denoted by $\mathbb{R}_+$ and $\mathbb{R}_{++}$, respectively. Let $\mathbb{R}^n$ denote the standard n-dimensional Euclidean space with inner product and norm denoted by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$, respectively. The Frobenius inner product and Frobenius norm in $\mathbb{R}^{m\times n}$ are denoted by $\langle\cdot,\cdot\rangle_F$ and $\|\cdot\|_F$, respectively. The set of real $n\times n$ symmetric matrices is denoted by $\mathcal{S}^n$, and we define $\mathcal{S}^n_+$ to be the subset of $\mathcal{S}^n$ consisting of the positive semidefinite matrices. The indicator function I_S of a set $S\subset\mathbb{R}^n$ is defined as $I_S(z)=0$ for every $z\in S$, and $I_S(z)=\infty$, otherwise. The cardinality of a finite set $\mathcal{A}$ is denoted by $|\mathcal{A}|$. Let $\mathcal{O}_1(\cdot)$ denote $\mathcal{O}(\cdot+1)$ where $\mathcal{O}$ is the big O notation.

Let $\psi:\mathbb{R}^n\to(-\infty,+\infty]$ be given. The effective domain of ψ is denoted by $\operatorname{dom}\psi:=\{x\in\mathbb{R}^n:\psi(x)<\infty\}$ and ψ is proper if $\operatorname{dom}\psi\neq\emptyset$. Moreover, a proper function $\psi:\mathbb{R}^n\to(-\infty,+\infty]$ is said to be μ-strongly convex for some $\mu\geq 0$ if

$$\psi(\alpha z+(1-\alpha)u)\leq\alpha\psi(z)+(1-\alpha)\psi(u)-\frac{\alpha(1-\alpha)\mu}{2}\|z-u\|^2$$

for every $z,u\in\operatorname{dom}\psi$ and $\alpha\in[0,1]$. If ψ is differentiable at $\bar z\in\mathbb{R}^n$, then its affine approximation $\ell_\psi(\cdot;\bar z)$ at $\bar z$ is defined as

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

The subdifferential of ψ at $z\in\mathbb{R}^n$ is denoted by $\partial\psi(z)$. The set of all proper lower semicontinuous convex functions $\psi:\mathbb{R}^n\to(-\infty,+\infty]$ is denoted by $\overline{\operatorname{Conv}}(\mathbb{R}^n)$.

Organization of the paper. Section 2 describes the N-SCO problem and the assumptions made on it. It also presents the AC-ACG method for solving the N-SCO problem and describes the main result of the paper, which establishes a convergence rate bound for AC-ACG in terms of the average of observed curvatures. Section 3 contains three subsections. The first subsection reviews three ACG variants for solving convex SCO (C-SCO) problems. The second (resp., third) one reviews pure (resp., hybrid) ACG variants for solving N-SCO problems. Section 4 provides the proof of the main result stated in section 2. Section 5 presents computational results illustrating the efficiency of the AC-ACG method. Section 6 presents some concluding remarks. Finally, the appendix contains a technical result.

2. The AC-ACG method for solving the N-SCO problem.

This section presents the main algorithm studied in this paper, namely, an ACG method based on a sequence of average curvatures, and derives a convergence rate for it expressed in terms of this sequence. More specifically, it describes the N-SCO problem and the assumptions made on it, presents the AC-ACG method, and states the main result of the paper, i.e., the convergence rate of the AC-ACG method.

The problem of interest in this paper is the N-SCO problem (1), where the following conditions are assumed to hold:

(A1) $h \in \overline{\text{Conv}}\,(\mathbb{R}^n)$;

(A2) f is a nonconvex differentiable function on $\operatorname{dom} h$ and there exist scalars $m \geq 0$, $M \geq 0$ such that for every $u, u' \in \operatorname{dom} h$,

$$-\frac{m}{2}\|u-u'\|^2 \leq f(u) - \ell_f(u;u'), \qquad \|\nabla f(u) - \nabla f(u')\| \leq M\|u-u'\|; \tag{5}$$

(A3) the diameter $D := \sup\{\|u-u'\| : u, u' \in \operatorname{dom} h\}$ is bounded.

Throughout the paper, we let $\bar{m}$ (resp., $\bar{M}$) denote the smallest scalar $m \geq 0$ (resp., $M \geq 0$) satisfying the first (resp., second) inequality in (5).

We now make some remarks about the above assumptions. First, the set of optimal solutions X_* is nonempty and compact in view of (A1)–(A3). Second, the second inequality in (5) implies

$$-\frac{M}{2}\|u-u'\|^2 \leq f(u) - \ell_f(u;u') \leq \frac{M}{2}\|u-u'\|^2 \quad \forall u, u' \in \operatorname{dom} h. \tag{6}$$

Third, the last remark together with the fact that f is nonconvex on $\operatorname{dom} h$ due to assumption (A2) implies that $0 < \bar{m} \leq \bar{M}$. Fourth, assumption (A3) is used in the proofs of Lemmas 4.1(b) and 4.3(b).

A necessary condition for $\hat{y}$ to be a local minimum of (1) is that $0 \in \nabla f(\hat{y}) + \partial h(\hat{y})$, i.e., $\hat{y}$ is a stationary point of (1). More generally, given a tolerance $\hat{\rho} > 0$, a pair $(\hat{y}, \hat{v})$ is called a $\hat{\rho}$-approximate stationary pair of (1) if

$$\hat{v} \in \nabla f(\hat{y}) + \partial h(\hat{y}), \quad \|\hat{v}\| \leq \hat{\rho}. \tag{7}$$

We are ready to state the AC-ACG method, which stops when a $\hat{\rho}$-approximate stationary pair of (1) is computed. AC-ACG requires as input a scalar $M \geq \bar{M}$ where $\bar{M}$ is defined in the paragraph following (A3).

Average curvature–accelerated composite gradient (AC-ACG)

---

0. Let a parameter $\gamma \in (0,1)$, a scalar $M \geq \bar{M}$, a tolerance $\hat{\rho} > 0$, and an initial point $y_0 \in \operatorname{dom} h$ be given and set $A_0 = 0$, $x_0 = y_0$, $M_0 = \gamma M$, $k = 0$, and

$$\alpha = \frac{0.9}{8}\left(1 + \frac{1}{0.9\gamma}\right)^{-1}; \tag{8}$$

1. compute

$$a_k = \frac{1 + \sqrt{1 + 4M_k A_k}}{2M_k}, \quad A_{k+1} = A_k + a_k, \quad \tilde{x}_k = \frac{A_k y_k + a_k x_k}{A_{k+1}}; \tag{9}$$

2. set $y^g_{k+1} = y(\tilde{x}_k; M_k)$ where $y(\cdot;\cdot)$ is as in (2) and compute

$$x_{k+1} = \operatorname*{argmin}_{u \in \mathbb{R}^n} \left\{ a_k\left[\ell_f(u;\tilde{x}_k) + h(u)\right] + \frac{1}{2}\|u - x_k\|^2 \right\}, \tag{10}$$

$$v_{k+1} = M_k(\tilde{x}_k - y^g_{k+1}) + \nabla f(y^g_{k+1}) - \nabla f(\tilde{x}_k); \tag{11}$$

3. if $\|v_{k+1}\| \leq \hat{\rho}$ then output $(\hat{y}, \hat{v}) = (y^g_{k+1}, v_{k+1})$ and **stop**; otherwise, compute

$$C_k = \max\left\{ \mathcal{C}(y^g_{k+1}; \tilde{x}_k), \frac{\|\nabla f(y^g_{k+1}) - \nabla f(\tilde{x}_k)\|}{\|y^g_{k+1} - \tilde{x}_k\|} \right\}, \tag{12}$$

$$C_k^{avg} = \frac{1}{k+1}\sum_{j=0}^{k} C_j, \tag{13}$$

$$M_{k+1} = \max\left\{ \frac{1}{\alpha} C_k^{avg}, \gamma M \right\}, \tag{14}$$

where $\mathcal{C}(\cdot;\cdot)$ is as in (4);

4. set

$$y_{k+1} = \begin{cases} y_{k+1}^b := \frac{A_k y_k + a_k x_{k+1}}{A_{k+1}} & \text{if } C_k > 0.9M_k; \\ y_{k+1}^g & \text{otherwise} \end{cases} \tag{15}$$

and $k \leftarrow k+1$, and go to step 1.

We add a few observations about the AC-ACG method. First, the first two identities in (9) imply that

$$A_{k+1} = M_k a_k^2. \tag{16}$$

Second, the AC-ACG method evaluates two gradients of f and exactly two resolvents of h (i.e., an evaluation of $(I + \lambda \partial h)^{-1}(\cdot)$ for some $\lambda > 0$) per iteration, namely, one in (2) and the other one in (10). Third, Theorem 2.1 below guarantees that AC-ACG always terminates and outputs a $\hat{\rho}$-approximate solution $(\hat{y}, \hat{v})$ (see step 3). Fourth, C_k is the most recent observed curvature, C_k^{avg} is the average of all observed curvatures obtained so far, and M_{k+1} is a modified average curvature that will be used in the next iteration to compute y_{k+2}^g. Fifth, the observed curvature C_k used here is different from the one mentioned in the introduction (see (3)) and it is more suitable for our theoretical analysis. Sixth, every iteration starts with a triple (A_k, x_k, y_k) and obtains the next one $(A_{k+1}, x_{k+1}, y_{k+1})$ as in (9), (10), and (2). The iterate y_{k+1} is chosen to be either $y_{k+1}^g = y(\tilde{x}_k; M_k)$ obtained in (2) or the convex combination y_{k+1}^b defined in (15) depending on whether the current curvature C_k is smaller than or equal to a multiple (e.g., 0.9) of the modified average curvature M_k or not, respectively. Seventh, in the iterations for which $C_k \leq 0.9M_k$ (called the good ones), M_k is clearly a good upper curvature of f at $\tilde{x}_k$ in view of the definitions of C_k and y_{k+1}^g in (12) and step 2 of AC-ACG, respectively, and the definition of a good curvature in (3). Thus, assuming that the frequency of good iterations is relatively high, it is reasonable to expect that the smaller the sequence $\{M_k\}$ is, the faster the convergence rate of AC-ACG will be (see the discussion after (3) in the introduction). Eighth, it follows as a consequence of the results of section 4 that the number of good iterations is relatively large (see Lemma 4.5) and that the overall effect of the bad ones are nicely under control (see Lemma 4.4). Moreover, Theorem 2.1 below states that the convergence rate of AC-ACG is directly proportional to $\sqrt{M_k}$ in that $\min\{\|v_i\| : i \leq k\} = \mathcal{O}(\sqrt{M_k}/\sqrt{k})$.

We now discuss the likelihood of M_{k+1}, or equivalently, $\gamma_{k+1} := M_{k+1}/M$, being small. First observe that (14) implies that $\gamma_{k+1} \geq \gamma$. Hence, let us examine the situation in which $\gamma_{k+1} = \gamma$, i.e., γ_{k+1} reaches its lowest possible value for a fixed $\gamma \in (0,1)$. Clearly, it follows from (14) that $\gamma_{k+1} = \gamma$ if and only if

$$\frac{C_k^{avg}}{M} \leq \alpha\gamma. \tag{17}$$

Moreover, in view of (8) and the fact that $\gamma < 1$, it follows that $\alpha = \Theta(\gamma)$, and hence (17) implies that $C_k^{avg}/M = \mathcal{O}(\gamma^2)$. In conclusion, under the restrictive choice of α in (8), $\gamma_{k+1} = \gamma$ can happen only when the computed average curvature ratio C_k^{avg}/M is $\mathcal{O}(\gamma^2)$. However, choice (8) for α is too conservative in practice. Indeed, it follows from the proof of Lemma 4.5 and the arguments in the paragraph following it that in practice $\alpha \in (0,1)$ can be chosen as $\Theta(1)$ instead of $\Theta(\gamma)$ as above. Clearly, with such

a choice of α, (17) implies that the ratio C_k^{avg}/M is $\mathcal{O}(\gamma)$ instead of $\mathcal{O}(\gamma^2)$ as above. In summary, if $\gamma \in (0,1)$ is relatively small and α is chosen as $(0,1) \ni \alpha = \Theta(1)$ instead of (8), then the chances of having $\gamma_{k+1} = \gamma$ increases. In view of the aforementioned observation, the two AC-ACG variants which are computationally profiled in section 5 relax the choice of α from (8) to one satisfying $(0,1) \ni \alpha = \Theta(1)$.

We now state the main result of the paper which describes how fast one of the iterates $y_1^g, \ldots, y_k^g$ approaches the stationary condition $0 \in \nabla f(y) + \partial h(y)$. A remarkable feature of its convergence rate bound is that it is expressed in terms of M_k rather than a scalar $M \geq \bar{M}$.

THEOREM 2.1. *The following statements hold:*
(a) *for every $k \geq 1$, we have $v_k \in \nabla f(y_k^g) + \partial h(y_k^g)$;*
(b) *for every $k \geq 12$, we have*

$$\min_{1 \leq i \leq k} \|v_i\|^2 = \mathcal{O}\left(\frac{M_k^2 D^2}{\gamma k^2} + \frac{\theta_k \bar{m} M_k D^2}{k}\right),$$

where

$$\theta_k := \max\left\{\frac{M_k}{M_i} : 0 \leq i \leq k\right\} \geq 1. \tag{18}$$

We now make two remarks about Theorem 2.1. First, it immediately leads to a worst-case iteration-complexity bound as follows. In view of the second inequality in (5), the second inequality in (6), the definition of $\bar{M}$ in the paragraph following (A3), and relation (12), it follows that for every $k \geq 0$, $C_k \leq \bar{M}$, and hence that $C_k^{avg} \leq \bar{M}$ in view of (13). The latter inequality, (14), and the fact that $\alpha = \Theta(\gamma)$ (see the line following (17)), then imply that $M/M_{k+1} \leq 1/\gamma$ and

$$\frac{M_{k+1}}{M} = \mathcal{O}\left(\frac{\bar{M}}{\alpha M} + \gamma\right) = \mathcal{O}\left(\frac{\bar{M}}{\gamma M} + \gamma\right) \tag{19}$$

for every $k \geq 0$. These two estimates and the definition of θ_k in (18) then imply that, for some $i \leq k$, we have

$$\theta_k = \frac{M_k}{M}\frac{M}{M_i} = \mathcal{O}\left(\left(\frac{\bar{M}}{\gamma M} + \gamma\right)\frac{1}{\gamma}\right) = \mathcal{O}\left(\frac{\bar{M}}{\gamma^2 M} + 1\right).$$

Moreover, it follows from Theorem 2.1(b) that the iteration-complexity for AC-ACG to obtain a $\hat{\rho}$-approximate stationary pair $(\hat{y}, \hat{v})$ is

$$\mathcal{O}_1\left(\frac{M_k D}{\gamma^{1/2}\hat{\rho}} + \frac{\theta_k \bar{m} M_k D^2}{\hat{\rho}^2}\right) = \mathcal{O}_1\left(M_k\left(\frac{D}{\gamma^{1/2}\hat{\rho}} + \theta_k \frac{\bar{m} D^2}{\hat{\rho}^2}\right)\right)$$

which, in view of (19), the above estimate on θ_k, and the facts that $\gamma < 1$ and $M \geq \bar{M}$ (see step 0 of AC-ACG), is bounded by

$$\mathcal{O}_1\left(\left[\frac{D}{\gamma^{1/2}\hat{\rho}} + \left(\frac{\bar{M}}{\gamma^2 M} + 1\right)\frac{\bar{m} D^2}{\hat{\rho}^2}\right]\left(\frac{\bar{M}}{\gamma} + \gamma M\right)\right) = \mathcal{O}_1\left(\frac{MD}{\gamma^{3/2}\hat{\rho}} + \frac{\bar{m} M D^2}{\gamma^3 \hat{\rho}^2}\right). \tag{20}$$

Hence, for small values of γ, the worst-case iteration-complexity of AC-ACG is high but, if γ is viewed as a constant, i.e., $1/\gamma = \mathcal{O}(1)$, then the above complexity is as good as any other ACG method found in the literature for solving the N-SCO problem as long as the second term in (20) is the dominant one. In particular, in terms of $\hat{\rho}$

only, its worst-case iteration-complexity for solving an N-SCO problem is $\mathcal{O}(1/\hat{\rho}^2)$, which is identical to that of any other known ACG method (see, e.g., [6, 10, 15, 16]).

Second, the dependence of the worst-case iteration-complexity (20) on γ is not good because it is obtained using the conservative estimate (19). We will now examine the iteration-complexity bound under the assumption that $\gamma_{k+1} = M_{k+1}/M = \gamma$, or equivalently, (17) holds, for every $k \geq 0$. In this case, $\theta_k = 1$ for every $k \geq 0$ and hence the convergence rate bound in Theorem 2.1(b) yields the iteration-complexity bound

$$\mathcal{O}_1\left(\frac{\gamma^{1/2}MD}{\hat{\rho}} + \frac{\gamma\bar{m}MD^2}{\hat{\rho}^2}\right)$$

for AC-ACG, which improves as γ decreases. This contrasts with bound (20), which becomes worse as γ decreases.

3. Comparison with other accelerated type methods. This section gives a brief overview of existing ACG methods for solving convex and nonconvex SCO problems. It contains three subsections. The first subsection reviews three ACG variants for solving C-SCO problems. The second one discusses pure ACG variants for solving N-SCO problems, i.e., ACG variants which perform only accelerated steps similar to the ones of the variants of the first subsection. The third one discusses hybrid ACG variants which, in addition to ACG steps, may also perform unaccelerated ones.

3.1. Review of convex ACG methods. This subsection reviews three ACG variants for solving C-SCO problems, i.e., SCO problems of the form (1) where (A1)–(A3) hold with $m = 0$, and hence f is convex. All the ACG methods reviewed here are described in terms of the notation introduced in the AC-ACG method or the ACG framework described below. This approach has the advantage that all the ACG methods are viewed under the same notation and hence their similarities/differences become more apparent.

The accelerated gradient method for solving unconstrained C-SCO problems (i.e., (1) with $h = 0$) were originally developed by Nesterov in his celebrated work [18]. Subsequently, several variants of his method (see, for example, [1, 3, 11, 17, 19, 21, 22, 24]) have been developed for solving C-SCO problems.

Before reviewing ACG variants for solving C-SCO, we first describe a common ACG framework underlying them.

ACG framework

---

0. Let an initial point $y_0 \in \operatorname{dom} h$ be given, and set $x_0 = y_0$, $A_0 = 0$, and $k = 0$;
1. compute a_k, A_{k+1} and $\tilde{x}_k$ as in (9);
2. compute x_{k+1} and y_{k+1} using one of the rules listed below;
3. set $k \leftarrow k + 1$, and go to step 1.

We will now describe three possible rules for computing the iterates x_{k+1} and y_{k+1} in step 2 of the above framework.

(i) (FISTA rule) This rule sets $y^a_{k+1} = y(\tilde{x}_k; M_k)$ where $y(\tilde{x}_k; M_k)$ is defined in (2) and $M_k > 0$ is a good upper curvature of f at $\tilde{x}_k$, then chooses y_{k+1} to be any point satisfying $\phi(y_{k+1}) \leq \phi(y^a_{k+1})$ and computes x_{k+1} as

$$x_{k+1} = y^a_{k+1} + \frac{A_k}{a_k}\left(y^a_{k+1} - y_k\right). \tag{21}$$

The FISTA rule with $y_{k+1} = y^a_{k+1}$ was first introduced by Nesterov when h is the indicator function of a nonempty closed convex set (see, for example, "Constant Step Scheme, III" on pages 83–84 of [19] or "Constant Step Scheme, II. Simple sets" on page 90 of [20]) and was later extended to general composite closed convex functions in [2, 3].

(ii) (AT rule) This rule computes x_{k+1} as (10) and chooses y_{k+1} to be any point satisfying $\phi(y_{k+1}) \leq \phi(y^a_{k+1})$ where

$$y^a_{k+1} = \frac{A_k y_k + a_k x_{k+1}}{A_{k+1}}. \tag{22}$$

This rule with $y_{k+1} = y^a_{k+1}$ was introduced by Auslender and Teboulle in [1], which explains the name "AT" adopted here.

(iii) (LLM rule) This rule sets y_{k+1} as in the FISTA rule and x_{k+1} as in the AT rule. The LLM rule was introduced by Lan, Lu, and Monteiro in [11], which explains the name "LLM" adopted here.

We now make a few remarks on the three ACG variants based on the above three rules. First, the ACG variant based on the LLM rule performs two resolvent evaluations of h per iteration, while the variants based on the AT and FISTA rules perform exactly one resolvent evaluation. Second, two popular choices of an upper curvature sequence $\{M_k\}$ are as follows: (1) for some $M \geq \bar{M}$, $M_k = M$ for every $k \geq 0$; and (2) for every $k \geq 0$, M_k is computed by a backtracking procedure such as the one outlined in the second paragraph of section 1. While [1, 11, 19] consider only the first choice, [3, 22] analyze the FISTA variant for both choices of $\{M_k\}$. Third, the AC-ACG method studied in this paper uses the LLM rule and works with a sequence $\{M_k\}$ such that M_k is not necessarily a good upper curvature of f at $\tilde{x}_k$.

We now comment on the monotonicity of the three aforementioned ACG variants. The three ACG variants based on the identity $y_{k+1} = y^a_{k+1}$ are not necessarily monotone (i.e., it satisfies $\phi(y_{k+1}) \leq \phi(y_k)$ for every $k \geq 0$), even if every M_k is a good upper curvature of f at $\tilde{x}_k$. However, they can be made monotone by invoking an idea introduced in [21] which sets $y_{k+1} = \operatorname{argmin}\{\phi(y) : y \in \{y_k, y^a_{k+1}\}\}$, where y^a_{k+1} is as described in each of the rules above. Another alternative way of forcing monotonicity, which requires an extra resolvent evaluation of h, is to choose y_{k+1} as

$$y_{k+1} = \operatorname{argmin}\{\phi(y) : y \in \{y_k, y^a_{k+1}, y^{na}_{k+1}\}\}, \tag{23}$$

where $y^{na}_{k+1} = y(y_k; M^{na}_k)$ and M^{na}_k is a good upper curvature of f at y_k. We remark that y_k can actually be removed from the right-hand side of (23). This is due to the fact that M^{na}_k being a good upper curvature of f at y_k implies that $\phi(y^{na}_{k+1}) \leq \phi(y_k)$ in view of Lemma A.1 in the appendix with $(M_k, \tilde{x}_k, y_{k+1}) = (M^{na}_k, y_k, y^{na}_{k+1})$.

3.2. Pure accelerated variants. This subsection discusses pure ACG variants for solving the N-SCO problem (1). More specifically, we discuss three methods, namely, the AG method proposed in [6], the NC-FISTA of [16], and its adaptive variant ADAP-NC-FISTA, also described in [16]. The iteration-complexity of all three methods is analyzed under the assumption that $\operatorname{dom} h$ is bounded, but in practice all three methods can successfully solve many problems with unbounded $\operatorname{dom} h$.

AG is a direct extension of the ACG variant, based on the LLM rule and the constant choice of M_k, to the N-SCO context. Clearly, AG performs two resolvent evaluations of h per iteration.

NC-FISTA requires as input a pair (M, m) such that $M > \bar{M}$ and $M \geq m \geq \bar{m}$. It is an extension of the version of FISTA with $y_{k+1} = y^a_{k+1}$ from the C-SCO to the

N-SCO context, and it reduces to the latter one when $m = \bar{m} = 0$. More specifically, NC-FISTA sets $y_{k+1} = y(\tilde{x}_k; M_k)$ where $M_k = M + \kappa_0 m/(Ma_k)$ and computes x_{k+1} as in (21) with A_k/a_k replaced by $(\kappa_0 m/M + 1)^{-1}(A_k/a_k)$ where κ_0 is a positive universal constant. In contrast to an iteration of the AG method, every iteration of NC-FISTA performs exactly one resolvent evaluation of h.

One drawback of NC-FISTA is its required input pair (M, m), which is usually hard to obtain or is often poorly estimated. On the other hand, ADAP-NC-FISTA remedies this drawback in that it only requires as input an arbitrary initial pair (M_0, m_0) such that $M_0 \geq m_0 > 0$, which is dynamically updated by means of two separate backtracking search procedures.

3.3. Hybrid accelerated variants. This subsection discusses hybrid ACG variants for solving the N-SCO problem (1). More specifically, we discuss three methods, namely, a nonmonotone variant as well as a monotone one, both described in [13], which we refer to as NM-APG and M-APG, respectively, and UPFAG proposed in [7]. To the best of our knowledge, the convergence of these hybrid ACG variants is guaranteed due to the possibility of performing an extra unaccelerated composite gradient step. Whether their convergence can be shown without this optional step is an open question even for the case in which $\operatorname{dom} h$ is bounded.

M-APG is exactly the instance of the ACG variant based on the FISTA rule in which y_{k+1} is computed by means of (23) which, as already mentioned above, guarantees its monotonicity property due to the fact that M_k^{na} is chosen as a good upper curvature of f at y_k. NM-APG is a variant of M-APG, which either sets $y_{k+1} = y_{k+1}^a$ or computes y_{k+1} as in (23) depending on whether or not, respectively, y_{k+1}^a satisfies a key inequality, which ensures convergence of the method but not necessarily its monotonicity.

UPFAG is an ACG variant based on the AT rule in which the next iterate y_{k+1} is chosen as in (23) except that (M_k^a, M_k^{na}) is computed by line searches so that M_k^a closely approximates a good curvature of f at $\tilde{x}_k$ and M_k^{na} satisfies a relaxed version of the descent condition (50) with $(M_k, \tilde{x}_k, y_{k+1}) = (M_k^{na}, y_k, y_{k+1}^{na})$.

4. Proof of Theorem 2.1. This section presents the proof of Theorem 2.1. We start with the following technical result, which assumes that all sequences start with $k = 0$.

LEMMA 4.1. *The following statements hold:*

(a) *the sequences* $\{x_k\}$, $\{y_k\}$, $\{y_{k+1}^g\}$, $\{y_{k+1}^b\}$, *and* $\{\tilde{x}_k\}$ *are all contained in* $\operatorname{dom} h$;

(b) *for every* $u \in \operatorname{dom} h$ *and* $k \geq 0$, *we have*

$$A_k\|y_k - \tilde{x}_k\|^2 + a_k\|u - \tilde{x}_k\|^2 \leq a_k D^2;$$

(c) *for every* $k \geq 0$, $C_k \leq \bar{M}$ *and* $F_k \leq \bar{M}$, *where*

$$F_k := \mathcal{C}(y_{k+1}; \tilde{x}_k) \tag{24}$$

and $\mathcal{C}(\cdot;\cdot)$ *is defined in* (4);

(d) *for every* $k \geq 0$, *we have*

$$v_{k+1} \in \nabla f(y_{k+1}^g) + \partial h(y_{k+1}^g), \quad \|v_{k+1}\| \leq (M_k + C_k)\|y_{k+1}^g - \tilde{x}_k\|. \tag{25}$$

Proof. (a) The sequences $\{x_k\}$ and $\{y_{k+1}^g\}$ are contained in $\operatorname{dom} h$ in view of (10), (2), and step 0 of AC-ACG. Hence, using step 0 of AC-ACG again, (15), and the

convexity of $\operatorname{dom} h$, we easily see by induction that $\{y_k\}$ and $\{y_{k+1}^b\}$ are contained in $\operatorname{dom} h$. Finally, $\{\tilde{x}_k\} \subset \operatorname{dom} h$ follows from the third identity in (9) and the convexity of $\operatorname{dom} h$.

(b) Let $u \in \operatorname{dom} h$ and $k \geq 0$ be given. First note that for every $A, a \in \mathbb{R}_+$ and $x, y \in \mathbb{R}^n$, we have

$$A\|y\|^2 + a\|x\|^2 = (A+a)\left\|\frac{Ay+ax}{A+a}\right\|^2 + \frac{Aa}{A+a}\|y-x\|^2.$$

Applying the above identity with $A = A_k$, $a = a_k$, $y = y_k - \tilde{x}_k$, and $x = u - \tilde{x}_k$, and using both the second and third identities in (9), we have

$$\begin{aligned} A_k\|y_k - \tilde{x}_k\|^2 + a_k\|u - \tilde{x}_k\|^2 &= A_{k+1}\left\|\frac{A_k y_k + a_k u}{A_{k+1}} - \tilde{x}_k\right\|^2 + \frac{A_k a_k}{A_{k+1}}\|y_k - u\|^2 \\ &= \frac{a_k}{A_{k+1}}\left(a_k\|u - x_k\|^2 + A_k\|u - y_k\|^2\right) \leq a_k D^2, \end{aligned}$$

where the inequality follows from Lemma 4.1(a), the assumption that $u \in \operatorname{dom} h$, the definition of D in (A3), and the second equality in (9).

(c) The conclusion follows from definitions of C_k, F_k, and $\mathcal{C}(\cdot;\cdot)$ in (12), (24), and (4), respectively, and the fact that $\bar{M}$ satisfies both the second inequality in (5) and (6).

(d) The inclusion in (25) follows from the fact $y_{k+1} = y(\tilde{x}_k; M_k)$, the optimality condition of (2) and the definition of v_{k+1} in (11). Moreover, the inequality in (25) follows from definitions of C_k in (12) and v_{k+1} and the triangle inequality. ∎

The next result provides an important recursive formula involving a certain potential function η_k and the quantity $\|y_{k+1} - \tilde{x}_k\|$ that will later be related to the residual vector $\|v_{k+1}\|$ (see the proof of Lemma 4.3(a)).

LEMMA 4.2. *For every $k \geq 0$ and $u \in \operatorname{dom} h$, we have*

$$\frac{M_k - F_k}{2} A_{k+1}\|y_{k+1} - \tilde{x}_k\|^2 \leq \eta_k(u) - \eta_{k+1}(u) + \frac{1}{2}\bar{m} a_k D^2,$$

where M_k and F_k are as in (14) *and* (24)*, respectively, and*

$$\eta_k(u) := A_k(\phi(y_k) - \phi(u)) + \frac{1}{2}\|u - x_k\|^2. \tag{26}$$

Proof. Let $k \geq 0$ and $u \in \operatorname{dom} h$ be given and define $\gamma_k(u) := \ell_f(u; \tilde{x}_k) + h(u)$. Using the fact that x_{k+1} is an optimal solution of (10) and γ_k is a convex function, the second and third identities in (9), and relations (15) and (16), we conclude that

$$\begin{aligned} A_k\gamma_k(y_k) + a_k\gamma_k(u) + \frac{1}{2}\|u - x_k\|^2 - \frac{1}{2}\|u - x_{k+1}\|^2 &\geq A_k\gamma_k(y_k) + a_k\gamma_k(x_{k+1}) + \frac{1}{2}\|x_{k+1} - x_k\|^2 \\ &\geq A_{k+1}\gamma_k(y_{k+1}^b) + \frac{1}{2}\frac{A_{k+1}^2}{a_k^2}\|y_{k+1}^b - \tilde{x}_k\|^2 \\ &= A_{k+1}\left[\gamma_k(y_{k+1}^b) + \frac{M_k}{2}\|y_{k+1}^b - \tilde{x}_k\|^2\right]. \end{aligned}$$

Moreover, relations (2), (15), and (24) and the fact that $\{y_k^b\} \subset \operatorname{dom} h$ imply that

$$\gamma_k(y_{k+1}^b) + \frac{M_k}{2}\|y_{k+1}^b - \tilde{x}_k\|^2 \geq \gamma_k(y_{k+1}) + \frac{M_k}{2}\|y_{k+1} - \tilde{x}_k\|^2$$
$$= \phi(y_{k+1}) + \frac{M_k - F_k}{2}\|y_{k+1} - \tilde{x}_k\|^2.$$

Using the above two inequalities, the definition of η_k in (26), and the first inequality in (5), we easily see that

$$\frac{M_k - F_k}{2}A_{k+1}\|y_{k+1} - \tilde{x}_k\|^2 - \eta_k(u) + \eta_{k+1}(u)$$
$$\leq A_k(\gamma_k(y_k) - \phi(y_k)) + a_k(\gamma_k(u) - \phi(u))$$
$$\leq \frac{\bar{m}}{2}\left(A_k\|y_k - \tilde{x}_k\|^2 + a_k\|u - \tilde{x}_k\|^2\right),$$

which, together with Lemma 4.1(b), then immediately implies the lemma. ∎

For the purpose of stating the next results, we define the set of good and bad iterations as

$$\mathcal{G} := \{k \geq 0 : C_k \leq 0.9M_k\}, \quad \mathcal{B} := \{k \geq 0 : C_k > 0.9M_k\}, \tag{27}$$

respectively. The following result specializes the bound derived in Lemma 4.2 to the two exclusive cases in which $k \in \mathcal{G}$ and $k \in \mathcal{B}$. More specifically, it derives a controllable bound on the residual vector v_{k+1} and the potential function difference $\eta_{k+1}(u) - \eta_k(u)$ in the good iterations and a controllable bound only on $\eta_{k+1}(u) - \eta_k(u)$ in the bad iterations.

LEMMA 4.3. *The following statements hold for every $u \in \operatorname{dom} h$ and $k \geq 0$:*
(a) *if $k \in \mathcal{G}$ then*

$$\frac{A_{k+1}}{72.2M_k}\|v_{k+1}\|^2 \leq \eta_k(u) - \eta_{k+1}(u) + \frac{1}{2}\bar{m}a_k D^2; \tag{28}$$

(b) *if $k \in \mathcal{B}$ then*

$$0 \leq \eta_k(u) - \eta_{k+1}(u) + \frac{1}{2}\bar{m}a_k D^2 + \frac{1-\gamma}{2\gamma}D^2. \tag{29}$$

Proof. (a) Let $k \in \mathcal{G}$ be given and note that (27) and (15) imply that $0.9M_k \geq C_k$ and $y_{k+1} = y_{k+1}^g$ where $y_{k+1}^g = y(\tilde{x}_k; M_k)$ is as in (2). Hence, using the inequality in (25), and the definitions of C_k and F_k in (12) and (24), respectively, we conclude that $\|v_{k+1}\| \leq 1.9M_k\|y_{k+1} - \tilde{x}_k\|$ and $F_k \leq C_k \leq 0.9M_k$. The latter two conclusions and Lemma 4.2 then immediately imply that (28) holds.

(b) Let $k \in \mathcal{B}$ be given and note that (15) and (27) imply that $y_{k+1} = y_{k+1}^b$. Using the latter observation, Lemma 4.2, Lemma 4.1(c), the last equality in (9), and relation (16), we conclude that

$$\eta_k(u) - \eta_{k+1}(u) + \frac{1}{2}\bar{m}a_k D^2 \geq \frac{(M_k - F_k)}{2}A_{k+1}\|y_{k+1}^b - \tilde{x}_k\|^2$$
$$= \frac{(M_k - F_k)}{2}A_{k+1}\left\|\frac{A_k y_k + a_k x_{k+1}}{A_{k+1}} - \frac{A_k y_k + a_k x_k}{A_{k+1}}\right\|^2$$
$$= \frac{(M_k - F_k)a_k^2}{2A_{k+1}}\|x_{k+1} - x_k\|^2 = \frac{1}{2}\left(1 - \frac{F_k}{M_k}\right)\|x_{k+1} - x_k\|^2$$
$$\geq \frac{1}{2}\left(1 - \frac{1}{\gamma}\right)\|x_{k+1} - x_k\|^2$$

and hence that (29) holds in view of Lemma 4.1(a) and (A3). ∎

As a consequence, the next lemma provides the result of the summation of inequalities for $k \in \mathcal{G}$ and $k \in \mathcal{B}$ in Lemma 4.3.

LEMMA 4.4. *For every $u \in \operatorname{dom} h$ and $k \geq 1$, we have*

$$\left(\frac{1}{36.1}\sum_{i\in\mathcal{G}_k}\frac{A_{i+1}}{M_i}\right)\min_{1\leq i\leq k}\|v_i\|^2 \leq \|u-x_0\|^2 - 2\eta_k(u) + \bar{m}D^2A_k + \frac{1-\gamma}{\gamma}D^2|\mathcal{B}_k|, \tag{30}$$

where $\mathcal{G}_k$ and $\mathcal{B}_k$ are defined as

$$\mathcal{G}_k = \{i \in \mathcal{G} : i \leq k-1\}, \quad \mathcal{B}_k := \{i \in \mathcal{B} : i \leq k-1\}. \tag{31}$$

Proof. First, note that

$$\sum_{i\in\mathcal{G}_k}\frac{A_{i+1}}{M_i}\|v_{i+1}\|^2 \geq \left(\sum_{i\in\mathcal{G}_k}\frac{A_{i+1}}{M_i}\right)\min_{i\in\mathcal{G}_k}\|v_{i+1}\|^2 \geq \left(\sum_{i\in\mathcal{G}_k}\frac{A_{i+1}}{M_i}\right)\min_{1\leq i\leq k}\|v_i\|^2.$$

The conclusion follows by adding (28) and (29) both with $k = i$ as i varies in $\mathcal{G}_k$ and $\mathcal{B}_k$, respectively, and using the above inequality, the definition of η_k in (26), and the facts that $A_k = A_0 + \sum_{i=0}^{k-1} a_i$ and $A_0 = 0$, which are due to (9) and step 0 of the AC-ACG method, respectively. ∎

Note that the left-hand side of (30) is actually zero when $\mathcal{G}_k = \emptyset$, and hence (30) is meaningless in this case. The result below, which plays a major role in our analysis, uses for the first time the fact that M_k is chosen as in (14) and shows that $\mathcal{G}_k$ is nonempty and well-populated. This fact in turn implies that the term inside the parentheses in the left-hand side of (30) is sufficiently large (see Lemma 4.8 below). The proof of Theorem 2.1 will then follow by combining these observations.

LEMMA 4.5. *For every $k \geq 12$, $|\mathcal{B}_k| \leq k/3$ where $\mathcal{B}_k$ is as defined in* (31).

Proof. Let $k \geq 12$ be given and, for the sake of this proof, define $C_{-1}^{avg} = 0$. In view of (14) and the definition of $\mathcal{B}_k$ in (31), it follows that for every $i \in \mathcal{B}_k$,

$$\frac{\alpha}{0.9}C_i > \alpha M_i \geq C_{i-1}^{avg}$$

and hence that

$$\frac{\alpha}{0.9}\sum_{i\in\mathcal{B}_k} C_i > \sum_{i\in\mathcal{B}_k} C_{i-1}^{avg}. \tag{32}$$

Using Lemma 4.1(c) and the facts that $C_i > 0.9M_i$ for every $i \in \mathcal{B}_k$ and that $M_i \geq \gamma M \geq \gamma\bar{M}$ (see (14) and step 0 of the AC-ACG method) for every $i \geq 0$, we have

$$0.9\gamma\bar{M} \leq C_i \leq \bar{M}, \quad i \in \mathcal{B}_k. \tag{33}$$

Let $l := |\mathcal{B}_k|$ and let $i_1 < \cdots < i_l$ denote the indices in $\mathcal{B}_k$. Clearly, in view of (13) and the fact that $i_j \leq k$ for every $j = 1, \ldots, l$, we have

$$C_{i_1-1}^{avg} \geq 0, \quad C_{i_2-1}^{avg} \geq \frac{1}{k}C_{i_1}, \quad \cdots\cdots, \quad C_{i_l-1}^{avg} \geq \frac{1}{k}\left(C_{i_1} + \cdots + C_{i_{l-1}}\right).$$

Summing these inequalities, we obtain

$$\sum_{i\in\mathcal{B}_k} C_{i-1}^{avg} \geq \frac{1}{k}\sum_{j=1}^{l}(l-j)C_{i_j} \geq \frac{1}{k}\sum_{j=1}^{\lceil l/2\rceil}(l-j)C_{i_j} \geq \frac{1}{k}\left\lfloor \frac{l}{2}\right\rfloor \sum_{j=1}^{\lceil l/2\rceil} C_{i_j}.$$

Combining (32) and the last inequality, we then conclude that

$$\frac{\alpha(S_1+S_2)}{0.9} \geq \frac{1}{k}\left\lfloor \frac{l}{2}\right\rfloor S_1,$$

where

$$S_1 := \sum_{j=1}^{\lceil l/2\rceil} C_{i_j}, \quad S_2 := \sum_{j=\lceil l/2\rceil+1}^{l} C_{i_j}. \tag{34}$$

Since (33) and the above definitions of S_1 and S_2 immediately imply that $S_2/S_1 \leq 1/(0.9\gamma)$, we then conclude from the above inequality that

$$|\mathcal{B}_k| = l \leq \left(\frac{2\alpha k}{0.9}\right)\left(1+\frac{S_2}{S_1}\right)+1 \leq \left(\frac{2\alpha k}{0.9}\right)\left(1+\frac{1}{0.9\gamma}\right)+1 \tag{35}$$

and hence that $|\mathcal{B}_k| \leq k/4+1 \leq k/3$ in view of the definition of α in (8) and the fact that $k \geq 12$. The last conclusion of the lemma follows straightforwardly from the first one. ∎

We now make some remarks about choosing α more aggressively, i.e., larger than the value in (8) (recall the discussion in the second paragraph following the AC-ACG method). First, in view of their definitions in (34), the quantities S_1 and S_2 are actually quantities that depend on the iteration index k and hence should have been denoted by S_1^k and S_2^k. Second, it follows from the first inequality in (35) that

$$|\mathcal{B}_k| \leq \left(\frac{2\alpha k}{0.9}\right)(1+\bar{\gamma}_k)+1,$$

where $\bar{\gamma}_k := S_2^k/S_1^k$. Third, we have used in the proof of Lemma 4.5 that $\bar{\gamma}_k$ is bounded above by $1/(0.9\gamma)$, which is a very conservative bound for this quantity. In practice though, $\bar{\gamma}_k$ behaves as $\mathcal{O}(1)$ (if not for all k, then at least for a substantial number of iterations). Fourth, in order to conclude that $|B_k| \leq k/3$ as in the proof of Lemma 4.5, it suffices to choose

$$\alpha = \frac{0.9}{8(1+\bar{\gamma})},$$

where $\bar{\gamma} := \max\{\bar{\gamma}_k : k \geq 1\}$. Observe that the above choice of α is $\Theta(1)$ if $\bar{\gamma}$ behaves as $\mathcal{O}(1)$.

Before presenting Lemma 4.8, we first state two technical results about the sequences $\{M_k\}$ and $\{A_k\}$.

LEMMA 4.6. *For every $1 \leq i < k$, we have*

$$M_k \geq \frac{i}{k}M_i.$$

Proof. From the definition of C_k^{avg} in (13), for every $i = 1,\ldots,k-1$, we have

$$kC_{k-1}^{avg} - iC_{i-1}^{avg} = C_i + \cdots + C_{k-1}$$

and thus

$$\frac{C_{k-1}^{avg}}{C_{i-1}^{avg}} = \frac{i}{k} + \frac{C_i + \ldots + C_{k-1}}{kC_{i-1}^{avg}} \geq \frac{i}{k}.$$

The conclusion follows from the above inequality, the definition of M_k in (14), and the fact that $\max\{a, c\} \geq \max\{b, d\}$ for $a, b, c, d \in \mathbb{R}$ such that $a \geq b$ and $c \geq d$. ∎

The following result describes bounds on A_k in terms of the first k elements of the sequence $\{M_i\}$ and also in terms of M_k alone.

LEMMA 4.7. *Consider the sequences* $\{A_k\}$ *and* $\{M_i\}$ *defined in* (9) *and* (14), *respectively. For every* $k \geq 12$, *we have*

$$A_k \leq \left(\sum_{i=0}^{k-1} \frac{1}{\sqrt{M_i}}\right)^2 \leq k \sum_{i=0}^{k-1} \frac{1}{M_i} \leq k^2 \frac{\theta_k}{M_k} \tag{36}$$

and

$$A_k \geq \frac{1}{4}\left(\sum_{i=0}^{k-1} \frac{1}{\sqrt{M_i}}\right)^2 \geq \frac{k^2}{12M_k}, \tag{37}$$

where θ_k *is as in* (18).

Proof. We first establish the inequalities in (36). Using the first two identities in (9) and the fact $\sqrt{b_1 + b_2} \leq \sqrt{b_1} + \sqrt{b_2}$ for any $b_1, b_2 \in \mathbb{R}_+$, we conclude that for any $i \geq 0$,

$$\sqrt{A_{i+1}} = \left(A_i + \frac{1 + \sqrt{1 + 4M_iA_i}}{2M_i}\right)^{\frac{1}{2}} \leq \left(A_i + \frac{1 + \sqrt{M_iA_i}}{M_i}\right)^{\frac{1}{2}} \leq \sqrt{A_i} + \frac{1}{\sqrt{M_i}}.$$

Now, the first inequality in (36) follows by summing the above inequality from $i = 0$ to $k - 1$ and using the assumption that $A_0 = 0$. Moreover, the second and third inequalities in (36) follow straightforwardly from the Cauchy–Schwarz inequality and the definition of θ_k in (18), respectively.

We now establish the inequalities in (37). Using the first two identities in (9), we have

$$\sqrt{A_{i+1}} = \left(A_i + \frac{1 + \sqrt{1 + 4M_iA_i}}{2M_i}\right)^{\frac{1}{2}} \geq \left(A_i + \frac{1 + 2\sqrt{M_iA_i}}{2M_i}\right)^{\frac{1}{2}} \geq \sqrt{A_i} + \frac{1}{2\sqrt{M_i}}.$$

The first inequality in (37) now follows by summing the above inequality from $i = 0$ to $k - 1$ and using the assumption that $A_0 = 0$. For every $k \geq 12$, we have

$$\sum_{i=1}^{k-1} \sqrt{i} \geq \int_0^{k-1} \sqrt{x}dx = \frac{2}{3}(k-1)^{\frac{3}{2}} \geq \frac{2}{3}\left(\frac{11}{12}k\right)^{\frac{3}{2}} \geq 0.58k^{\frac{3}{2}},$$

which, together with Lemma 4.6, then implies that

$$\sum_{i=1}^{k-1} \frac{1}{\sqrt{M_i}} \geq \frac{1}{\sqrt{kM_k}} \sum_{i=1}^{k-1} \sqrt{i} \geq \frac{0.58k}{\sqrt{M_k}}.$$

The second inequality in (37) now follows immediately from the one above. ∎

The following result provides a lower bound on the term inside the parentheses of the left-hand side of (30).

LEMMA 4.8. *For every $k \geq 12$, we have*

$$\sum_{i \in \mathcal{G}_k} \frac{A_{i+1}}{M_i} \geq \frac{k^3}{3402 M_k^2}.$$

Proof. Let $k \geq 12$ be given and define

$$\tilde{\mathcal{G}}_k := \{i \in \mathcal{G}_k : i \geq \lfloor k/3 \rfloor\}, \quad \tilde{\mathcal{B}}_k := \{i \in \mathcal{B}_k : i \geq \lfloor k/3 \rfloor\}. \tag{38}$$

Using Lemma 4.6, the facts that $\tilde{\mathcal{G}}_k \subset \mathcal{G}_k$, $\{A_k\}$ is strictly increasing, and $i/k \geq 2/7$ for any $i \in \tilde{\mathcal{G}}_k$ and $k \geq 12$, and inequality (37), we conclude that

$$\begin{aligned} \sum_{i \in \mathcal{G}_k} \frac{A_{i+1}}{M_i} &\geq \sum_{i \in \mathcal{G}_k} \frac{i A_{i+1}}{k M_k} \geq \sum_{i \in \tilde{\mathcal{G}}_k} \frac{i A_{i+1}}{k M_k} \geq \frac{2|\tilde{\mathcal{G}}_k|}{7 M_k} A_{\lfloor k/3 \rfloor + 1} \\ &\geq \frac{2|\tilde{\mathcal{G}}_k|}{7 M_k} A_{\lceil k/3 \rceil} \geq \frac{|\tilde{\mathcal{G}}_k|(\lceil k/3 \rceil)^2}{42 M_k M_{\lceil k/3 \rceil}} \geq \frac{|\tilde{\mathcal{G}}_k| k^2}{378 M_k M_{\lceil k/3 \rceil}}. \end{aligned} \tag{39}$$

On the other hand, Lemma 4.6 with $i = \lceil k/3 \rceil$ implies that

$$M_k \geq \frac{\lceil k/3 \rceil}{k} M_{\lceil k/3 \rceil} \geq \frac{1}{3} M_{\lceil k/3 \rceil}.$$

Moreover, the definition of $\tilde{\mathcal{G}}_k$ in (38), the fact that $\tilde{\mathcal{B}}_k \subset \mathcal{B}_k$, and Lemma 4.5 imply that

$$|\tilde{\mathcal{G}}_k| = k - \lfloor k/3 \rfloor - |\tilde{\mathcal{B}}_k| \geq k - \lfloor k/3 \rfloor - |\mathcal{B}_k| \geq k/3.$$

The conclusion of the lemma now follows by combining (39) with the last two observations. ∎

We are now ready to prove the main result of our paper.

Proof of Theorem 2.1. (a) The conclusion immediately follows from Lemma 4.1(d).

(b) Letting $x_* \in X_*$ be given and noting that $\eta_k(x_*) \geq 0$ in view of the definition of η_k in (26) and using the above inequality, Lemma 4.4 with $u = x_*$, Lemma 4.5, and relation (36), we conclude that

$$\begin{aligned} \left(\frac{1}{36.1} \sum_{i \in \mathcal{G}_k} \frac{A_{i+1}}{M_i} \right) \min_{1 \leq i \leq k} \|v_i\|^2 &\leq \|x_0 - x_*\|^2 + \bar{m} D^2 A_k + \frac{1-\gamma}{\gamma} D^2 |\mathcal{B}_k| \\ &\leq D^2 + \bar{m} D^2 A_k + \frac{(1-\gamma) D^2 k}{3\gamma} \\ &\leq D^2 + \frac{\bar{m} D^2 k^2 \theta_k}{M_k} + \frac{(1-\gamma) D^2 k}{3\gamma}. \end{aligned}$$

Statement (b) of the theorem now follows by combining the above inequality and Lemma 4.8. ∎

5. Numerical results. This section presents computational results to illustrate the performance of two variants of the AC-ACG method against five other state-of-the-art algorithms on a collection of nonconvex optimization problems that are either in the form of or can be easily reformulated into (1). It contains five subsections, with each one reporting computational results on one of following classes of nonconvex optimization problems: (a) quadratic programming (subsection 5.1); (b) support vector machine (SVM; subsection 5.2); (c) sparse PCA (subsection 5.3); (d) matrix completion (subsection 5.4); and (e) nonnegative matrix factorization (NMF; subsection 5.5). Note that sparse PCA and NMF are problems for which $\operatorname{dom} h$ is unbounded.

We start by describing the two AC-ACG variants considered in our computational experiments, both of which do not impose the restrictive condition (8) on the choice of α and γ. The first variant, which we refer to as ACT throughout this section, preserves all steps in the AC-ACG method except that γ and α are provided as input by the user without necessarily satisfying (8). In our implementation, we set $\gamma = 0.01$ for every problem class listed above except the one in (b) for which γ is set to 0.002. The latter choice of γ prevents the percentage of good iterations from being 100% all the time and instead keeps it within a range of about 65% to 75% (see subsection 5.2). The choice of the scalar α varies per problem class and is described in each one of the subsections below. The second variant, referred to as AC throughout this section, sets $M_0 = 0.01M$ and computes M_{k+1} as in (14) with $\gamma = 10^{-6}$ and C_k as

$$C_k = \max\{\mathcal{C}(y^g_{k+1}; \tilde{x}_k), 0\}, \tag{40}$$

where $\mathcal{C}(\cdot;\cdot)$ is defined in (4). Our implementation of AC sets α to values that depend on the problem class under consideration and are specified in the subsections below. Clearly, among the two variants described above, ACT is the closest to AC-ACG.

We compare the two variants of AC-ACG with five other methods, namely, (i) the AG method proposed in [6]; (ii) the NC-FISTA of [16]; (iii) the ADAP-NC-FISTA also described in [16]; (iv) the NM-APG method proposed in [13]; and (v) the UPFAG method in [7]. We remark that methods (i)–(iii) are the three pure ACG variants that have been outlined in subsection 3.2 and methods (iv) and (v) are two among the three hybrid ACG variants that have been discussed in subsection 3.3. For the sake of simplicity, we use the abbreviations NM, UP, NC, and AD to refer to the NM-APG, UPFAG, NC-FISTA, and ADAP-NC-FISTA methods, respectively, in both the discussions and the tables below.

This paragraph provides details about the three pure ACG variants used in our benchmark. AG was implemented by the authors based on its description provided in Algorithm 1 of [6] where the sequences $\{\alpha_k\}$, $\{\beta_k\}$, and $\{\lambda_k\}$ were chosen as $\alpha_k = 2/(k+1)$, $\beta_k = 0.99/M$, and $\lambda_k = k\beta_k/2$, respectively, and the Lipschitz constant M was computed as described in each of the five subsections below. We note that the choice $\beta_k = 0.99/M$ used in our implementation differs from the one suggested in [6], namely, $\beta_k = 0.5/M$ (see (2.27) of [6]) and consistently improves the practical performance of AG. The NC and AD variants were also implemented by the authors based on their descriptions in [16]. The triple (M, m, A_0) needed as input by NC was set to $(M/0.99, m, 5000)$ where $m \geq \bar{m}$ (see the first inequality in (5)). The triple (M_0, m_0, θ) needed as input by AD was set to $(1, 1000, 1.25)$ in subsections 5.1, 5.2, and 5.5 and $(1, 1, 1.25)$ in subsections 5.3 and 5.4.

This paragraph provides implementation details for the two hybrid ACG variants used in our benchmark. The NM method was implemented by the authors based on its description provided in Algorithm 2 of [13] which does not use line searches to compute M^a_k and M^{na}_k. More specifically, the quadruple $(\alpha_x, \alpha_y, \eta, \delta)$ needed as

input by Algorithm 2 of [13] was set to $(0.99/M, 0.99/M, 0.8, 1)$. The code for UP was made available by the authors of [7], where UP is described (see Algorithm 1 of [7]). In particular, we have used their choice of parameters but have modified the code slightly to accommodate for the termination criterion (7) used in our benchmark. More specifically, the parameters $(\hat{\lambda}_0, \hat{\beta}_0, \gamma_1, \gamma_2, \gamma_3, \delta, \sigma)$ needed as input by UP were set to $(1/M, 1/M, 1, 1, 1, 10^{-3}, 10^{-10})$. Recall that UP computes the good upper curvatures M_k^a and M_k^{na} by line searches (see subsection 3.3). Our implementation of UP initiates these scalars in both line searches by using a Barzilai–Borwein type strategy (see equation (2.12) in [7]).

All seven methods terminate with a pair (z, v) satisfying

$$v \in \nabla f(z) + \partial h(z), \qquad \frac{\|v\|}{\|\nabla f(z_0)\| + 1} \leq \hat{\rho},$$

where $\hat{\rho} = 5 \times 10^{-4}$ in the matrix completion problem and $\hat{\rho} = 10^{-7}$ in all the other problems. All the computational results were obtained using MATLAB R2017b on a MacBook Pro with a quad-core Intel Core i7 processor and 16 GB of memory.

5.1. Quadratic programming. This subsection discusses the performance of the AC-ACG method for solving a class of quadratic programming problems.

More specifically, it considers the problem

$$\min \left\{ f(Z) := -\frac{\alpha_1}{2}\|D\mathcal{B}(Z)\|^2 + \frac{\alpha_2}{2}\|\mathcal{A}(Z) - b\|^2 : Z \in P_n \right\}, \tag{41}$$

where $(\alpha_1, \alpha_2) \in \mathbb{R}^2_{++}$, $b \in \mathbb{R}^l$ is a vector with entries sampled from the uniform distribution $\mathcal{U}[0, 1]$, $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix whose diagonal entries are sampled from the discrete uniform distribution $\mathcal{U}\{1, 1000\}$, $P_n := \{Z \in \mathcal{S}^n_+ : \mathrm{tr}(Z) = 1\}$ denotes the spectraplex, and $\mathcal{A} : \mathcal{S}^n_+ \to \mathbb{R}^l$ and $\mathcal{B} : \mathcal{S}^n_+ \to \mathbb{R}^n$ are linear operators given by

$$[\mathcal{A}(Z)]_i = \langle A_i, Z \rangle_F \quad \forall\, 1 \leq i \leq l,$$
$$[\mathcal{B}(Z)]_j = \langle B_j, Z \rangle_F \quad \forall\, 1 \leq j \leq n,$$

with $A_i \in \mathcal{S}^n_+$ and $B_j \in \mathcal{S}^n_+$ all being sparse matrices having the same density (i.e., percentage of nonzeros) d and nonzero entries uniformly sampled from $[0, 1]$.

The quadratic programming problem (41) is an instance of (1) where h is the indicator function of the spectraplex P_n. For chosen curvature pairs $(M, m) \in \mathbb{R}^2_{++}$, the scalars α_1 and α_2 are chosen so that $\lambda_{\max}(\nabla^2 f) = M$ and $\lambda_{\min}(\nabla^2 f) = -m$ where $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ denote the largest and smallest eigenvalue functions, respectively.

We start all seven methods from the same initial point $Z_0 = I_n/n$ where I_n is an $n \times n$ identity matrix, namely Z_0 is the centroid of P_n. The parameter α is set to 1 in AC and 0.5 in ACT.

Numerical results for the seven methods are given in Tables 1, 3, and 5, with each table addressing a collection of instances with the same dimension pair (l, n) and density d. Specifically, each row of Tables 1, 3, and 5 corresponds to an instance of (41), their first column specifies the pair (M, m) for the corresponding instance, and their second to eighth columns provide numbers of iterations and running times for the seven methods. The best objective function values obtained by all seven methods are not reported since they are essentially the same on all instances. The number of resolvent evaluations is 1 in NC, 2 in AG, AC and ACT, 1 or 2 in NM, 1 on average in AD, and 3 on average in UP. The bold numbers highlight the method that has the best performance in an instance of the problem.

Some statistic measures for AC and ACT to solve the instances in Tables 1, 3, and 5 are given in Tables 2, 4, and 6, respectively. The first column in these tables is the same as that of Tables 1, 3, and 5, the second (resp., fifth) column provides the maximum of all observed curvatures C_k in AC (resp., ACT), the third (resp., sixth) column provides the average of all observed curvatures C_k in AC (resp., ACT), and the fourth (resp., seventh) column gives the percentage of good iterations (see (27)) in AC (resp., ACT).

In Tables 1–2, the density $d = 2.5\%$ and the dimension pair $(l, n) = (50, 200)$.

In Tables 3–4, the density $d = 0.5\%$ and the dimension pair $(l, n) = (50, 400)$.

In Tables 5–6, the density $d = 0.1\%$ and the dimension pair $(l, n) = (50, 800)$.

TABLE 1
Numerical results for AG, NM, UP, NC, AD, ACT, and AC.

M, m	Iteration count/Running time (s)						
	AG	NM	UP	NC	AD	ACT	AC
$10^6, 10^6$	46/1.6	80/2.1	9/0.7	33/0.8	12/0.7	23/1.4	8/**0.6**
$10^6, 10^5$	3089/130	6242/191	2633/261	3384/94	2206/89	1009/57	883/**39**
$10^6, 10^4$	5400/188	10404/328	7203/705	1236/**30**	2591/104	1820/109	1760/73
$10^6, 10^3$	4621/176	11053/360	5429/540	5139/122	2637/109	1712/118	1508/**68**
$10^6, 10^2$	4476/176	11271/312	6891/653	11838/283	2639/116	1610/103	1472/**65**
$10^6, 10$	4461/171	11253/311	6479/613	14851/362	2640/116	1599/155	1485/**66**

TABLE 2
AC and ACT statistics.

M, m	AC			ACT		
	Max	Avg	Good	Max	Avg	Good
$10^6, 10^6$	1.88E5	3.04E4	88%	8.38E5	1.53E5	95%
$10^6, 10^5$	4.85E5	8.84E4	64%	7.00E5	9.25E4	98%
$10^6, 10^4$	5.42E5	1.24E5	65%	7.24E5	1.04E5	99%
$10^6, 10^3$	5.48E5	1.20E5	69%	7.27E5	1.16E5	97%
$10^6, 10^2$	5.49E5	1.20E5	68%	7.27E5	1.10E5	99%
$10^6, 10$	5.49E5	1.18E5	70%	7.27E5	1.09E5	99%

TABLE 3
Numerical results for AG, NM, UP, NC, AD, ACT, and AC.

M, m	Iteration count/Running time (s)						
	AG	NM	UP	NC	AD	ACT	AC
$10^6, 10^6$	44/4.4	75/5.1	10/1.9	33/2.1	12/1.8	17/2.6	8/**1.0**
$10^6, 10^5$	1411/134	3151/224	56/**13**	610/39	530/56	403/58	131/16
$10^6, 10^4$	1963/195	5071/373	105/**26**	1212/76	868/93	599/88	237/28
$10^6, 10^3$	1935/193	5172/382	115/**29**	4415/277	900/103	564/95	245/30
$10^6, 10^2$	1934/190	5045/367	119/32	7325/465	904/103	559/91	242/**29**
$10^6, 10$	1934/194	5056/373	113/31	7527/477	904/104	561/92	246/**29**

TABLE 4
AC and ACT statistics.

M, m	AC			ACT		
	Max	Avg	Good	Max	Avg	Good
$10^6, 10^6$	2.40E5	3.22E4	88%	6.32E5	1.67E5	93%
$10^6, 10^5$	1.53E5	1.98E4	76%	4.05E5	5.92E4	99%
$10^6, 10^4$	2.03E5	2.50E4	72%	4.16E5	6.66E4	98%
$10^6, 10^3$	2.07E5	2.55E4	69%	4.17E5	6.58E4	98%
$10^6, 10^2$	2.08E5	2.55E4	71%	4.17E5	6.54E4	98%
$10^6, 10$	2.08E5	2.51E4	68%	4.17E5	6.56E4	98%

TABLE 5
Numerical results for AG, NM, UP, NC, AD, ACT, and AC.

M, m	Iteration count/Running time (s)						
	AG	NM	UP	NC	AD	ACT	AC
$10^6, 10^6$	69/22	117/26	13/8	38/8	11/7	18/13	8/**4**
$10^6, 10^5$	277/119	502/118	9/6	176/36	24/10	31/20	**7**/**3**
$10^6, 10^4$	491/173	1030/246	13/9	786/163	60/24	65/39	11/**5**
$10^6, 10^3$	531/169	1144/259	13/9	1519/313	70/28	67/41	12/**7**
$10^6, 10^2$	535/172	1156/260	13/9	1698/351	71/28	67/43	12/**6**
$10^6, 10$	536/172	1157/266	13/8	1703/352	71/28	67/44	12/**5**

TABLE 6
AC and ACT statistics.

M, m	AC			ACT		
	Max	Avg	Good	Max	Avg	Good
$10^6, 10^6$	1.28E5	1.70E4	88%	3.65E5	5.37E4	94%
$10^6, 10^5$	1.80E4	2.84E3	86%	1.78E5	2.64E4	96%
$10^6, 10^4$	3.26E4	3.89E3	91%	1.78E5	2.99E4	98%
$10^6, 10^3$	3.41E4	3.73E3	92%	1.78E5	2.62E4	98%
$10^6, 10^2$	3.42E4	3.75E3	92%	1.78E5	2.58E4	98%
$10^6, 10$	3.43E4	3.75E3	92%	1.78E5	2.57E4	98%

In summary, computational results demonstrate that (i) the computed average curvature of AC is small compared with M and the computed maximum curvature; (ii) the percentage of good iterations of AC lies in a suitable range; and (iii) AC has the best performance in terms of running time.

5.2. Support vector machine. This subsection presents the performance of AC-ACG for solving an SVM problem. Given data points $\{(x_i, y_i)\}_{i=1}^p$, where $x_i \in \mathbb{R}^n$ is a feature vector and $y_i \in \{-1, 1\}$ denotes the corresponding label, we consider the SVM problem defined as

$$\min_{z\in\mathbb{R}^n} \frac{1}{p}\sum_{i=1}^{p} \ell(x_i, y_i; z) + \frac{\lambda}{2}\|z\|^2 + I_{\mathcal{B}_r}(z) \tag{42}$$

for some $\lambda, r > 0$, where $\ell(x_i, y_i; \cdot) = 1 - \tanh(y_i\langle \cdot, x_i\rangle)$ is a nonconvex sigmoid loss function and $I_{\mathcal{B}_r}(\cdot)$ is the indicator function of the ball $B_r := \{z \in \mathbb{R}^n : \|z\| \le r\}$. The SVM problem (42) is an instance of nonconvex SCO problems (1) where

$$f(z) = \frac{1}{p}\sum_{i=1}^{p} \ell(x_i, y_i; z) + \frac{\lambda}{2}\|z\|^2, \quad h(z) = I_{\mathcal{B}_r}(z).$$

Clearly, f is differentiable everywhere and its gradient is M-Lipschitz continuous where

$$M = \frac{1}{p}\sum_{i=1}^{p} L_i + \lambda, \quad L_i = \frac{4\sqrt{3}}{9}\|x_i\|^2 \ \ \forall i = 1, \ldots, p. \tag{43}$$

Since no sharper $m < M$ satisfying the first inequality in (5) is known, we simply set $m = M$.

We generate synthetic data sets as follows: for each data point (x_i, y_i), x_i is drawn from the uniform distribution on $[0, 1]^n$ and is sparse with 5% nonzero components, and $y_i = \text{sign}(\langle \bar{z}, x_i \rangle)$ for some $\bar{z} \in B_r$. We consider four different problem sizes (n, p), i.e., $(1000, 500)$, $(2000, 1000)$, $(3000, 1000)$, and $(4000, 500)$. We set $\lambda = 1/p$ and $r = 50$.

We start all seven methods from the same initial point z_0 that is chosen randomly from the uniform distribution within the ball B_r. The parameter α is set to 0.5 in both AC and ACT.

Numerical results of the seven methods are given in Table 7 and some statistic measures of AC and ACT are given in Table 8. The explanation of their columns excluding the first one is the same as those of Tables 1–6 (see the two paragraphs preceding Table 1). Their first columns differ from those of Tables 1–6 in that they only list the value of M computed according to (43). The best objective function values obtained by all seven methods are not reported since they are essentially the same on all instances. The number of resolvent evaluations is 1 in NC, 2 in AG, AC, and ACT, 1 or 2 in NM, 1 on average in AD, and 3 on average in UP. The bold numbers highlight the method that has the best performance in an instance of

TABLE 7
Numerical results for AG, NM, UP, NC, AD, ACT, and AC.

M	Iteration count/Running time (s)						
	AG	NM	UP	NC	AD	ACT	AC
13	37384 639	42532 649	130 8	42533 233	12274 188	583 9	546 **6**
25	112562 4419	123551 4486	278 **39**	174845 5833	21127 1836	1017 93	1131 60
38	155503 12636	163197 12101	401 97	500000* 26258*	71991 8957	1208 168	1032 **95**
50	79752 4406	79064 5264	247 44	172535 5503	12450 1033	730 65	615 **39**

TABLE 8
AC and ACT statistics.

M	AC			ACT		
	Max	Avg	Good	Max	Avg	Good
13	0.25	0.05	67%	0.06	0.05	71%
25	0.47	0.06	65%	0.08	0.06	69%
38	0.34	0.07	63%	0.10	0.07	66%
50	0.18	0.07	71%	0.11	0.07	74%

the problem. The numbers marked with * indicate that the maximum number of iterations has been reached.

In summary, computational results demonstrate that (i) the computed average curvature of AC is small compared with M and the computed maximum curvature; (ii) the percentage of good iterations of AC lies in a suitable range; and (iii) AC is either the best method or close to the best one in terms of running time.

5.3. Sparse PCA. This subsection considers a penalized version of the sparse PCA problem, namely,

$$\min_{X,Y\in\mathbb{R}^{p\times p}} -\langle\hat{\Sigma},X\rangle_F+\frac{\mu}{2}\|X\|_F^2+Q_{\lambda,b}(Y)+\lambda\|Y\|_1+\frac{\beta}{2}\|X-Y\|_F^2+I_{\mathcal{F}^r}(X), \tag{44}$$

where the dataset consists of an empirical covariance matrix $\hat{\Sigma}\in\mathbb{R}^{p\times p}$, two regularization parameters $\mu>0$ and $\lambda>0$, a penalty parameter $\beta>0$, and two scalars $b>0$ and $r\in\mathbb{N}_+$. Moreover, $\|\cdot\|_1$ and $Q_{\lambda,b}(\cdot)$ are the matrix 1-norm and a decomposable nonconvex penalty function defined as

$$\|Y\|_1:=\sum_{i,j=1}^{p}|Y_{ij}|\,,\quad Q_{\lambda,b}(X):=\sum_{i,j=1}^{p}q_{\lambda,b}\left(X_{ij}\right),$$

where

$$q_{\lambda,b}(t):=\begin{cases}-\frac{t^2}{2b} & \text{if } |t|\le b\lambda;\\ \frac{b\lambda^2}{2}-\lambda|t| & \text{otherwise,}\end{cases}$$

and $I_{\mathcal{F}^r}(\cdot)$ is the indicator function of the Fantope

$$\mathcal{F}^r:=\{X\in\mathcal{S}^n:0\preceq X\preceq I \text{ and } \operatorname{tr}(X)=r\}.$$

Clearly, problem (44) is an instance of the nonconvex SCO problem (1) where

$$f(X,Y)=-\langle\hat{\Sigma},X\rangle_F+\frac{\mu}{2}\|X\|_F^2+Q_{\lambda,b}(Y)+\frac{\beta}{2}\|X-Y\|_F^2,\quad h(X,Y)=I_{\mathcal{F}^r}(X)+\lambda\|Y\|_1.$$

Moreover, it is easy to see that the pair

$$(M,m)=\left(\max\left\{\mu+2\beta,\frac{1}{b}\right\},\frac{1}{b}\right) \tag{45}$$

satisfies assumption (A2).

We discuss how synthetic datasets are generated. Let $\Sigma\in\mathbb{R}^{p\times p}$ be an unknown covariance matrix and X^* be the projection matrix onto the r-dimensional principal subspace of Σ. In the sparse PCA problem, we seek an s-sparse approximation X of X^* in the sense that $\|\operatorname{diag}(X)\|_0\le s$, where $s\in\mathbb{N}_+$. We generate four datasets by designing four covariance matrices Σ as described in [9] and list all required parameters in Table 9. For each covariance matrices Σ, we sample $n=80$ independent and identically distributed observations from the normal distribution $\mathcal{N}(0,\Sigma)$ and then calculate the sample covariance matrix $\hat{\Sigma}$.

All seven methods are started from the same initial point (X_0,Y_0) that are chosen as follows. For datasets I and II, we set $X_0=Y_0$ to be a diagonal matrix with the first five diagonal entries equal to 1 and the other entries equal to zero. For datasets III and IV, we set $X_0=Y_0$ with the first diagonal entry being 1 and any other entries

TABLE 9
Synthetic datasets for the sparse PCA problem.

Dataset	s	r	p	b	β	μ	λ
I	10	5	1200	3	0.33	1.67	0.25
II	10	5	1200	3	0.33	3.33	1
III	5	1	1200	3	30	3	5
IV	5	1	1200	3	30	0.67	1

TABLE 10
Numerical results for AG, NM, UP, NC, AD, ACT, and AC.

M	Iteration count/Running time (s)						
	AG	NM	UP	NC	AD	ACT	AC
2.33	21/8.63	18/4.96	7/6.71	15/**4.50**	31/10.70	18/9.70	15/7.33
4	7/10.08	9/**2.73**	8/7.55	13/4.42	12/4.01	9/4.66	7/3.94
63	32/19.91	43/12.06	18/17.61	81/22.54	48/16.05	43/24.08	27/**12.04**
60.67	35/19.01	46/14.28	17/16.97	84/24.31	52/17.05	48/26.70	31/**12.51**

TABLE 11
AC and ACT statistics.

M	AC			ACT		
	Max	Avg	Good	Max	Avg	Good
2.33	2.00	0.72	67%	2.83	0.90	83%
4	3.67	3.41	71%	5.02	5.02	89%
63	44.41	31.12	89%	43.59	43.55	98%
60.67	36.00	28.26	94%	41.55	41.39	98%

being 0. We observe that the initial points were chosen differently so as to guarantee that they are feasible (i.e., lie in $\operatorname{dom} h$) for their respective instances. The parameter α is set to 0.5 in both AC and ACT.

Numerical results of the seven methods are given in Table 10 and some statistic measures of AC and ACT are given in Table 11. The explanation of their columns excluding the first one is the same as those of Tables 7 and 8, respectively. Their first columns differ from those of Tables 7 and 8 in that the value of M is computed according to (45). The best objective function values obtained by all seven methods are not reported since they are essentially the same on all instances. The number of resolvent evaluations is 1 in NC, 2 in AG, AC, and ACT, 1 or 2 in NM, 1 on average in AD, and 3 on average in UP. The bold numbers highlight the method that has the best performance in an instance of the problem.

In summary, computational results demonstrate that (i) the computed average curvature of AC is close to the computed maximum curvature; (ii) the percentage of good iterations of AC lies in a suitable range; and (iii) AC is either the best method or close to the best one in terms of running time.

5.4. Matrix completion. This subsection focuses on a constrained version of the nonconvex low-rank matrix completion problem. Before stating the problem, we first give a few definitions. Let Ω be a subset of $\{1,\dots,l\} \times \{1,\dots,n\}$ and let Π_Ω denote the linear operator that maps a matrix A to the matrix whose entries in Ω have the same values of the corresponding ones in A and whose entries outside of Ω

are all zero. Also, for given parameters $\beta > 0$ and $\theta > 0$, let $p : \mathbb{R} \to \mathbb{R}_+$ denote the log-sum penalty defined as

$$p(t) = p_{\beta,\theta}(t) := \beta \log\left(1 + \frac{|t|}{\theta}\right).$$

The constrained version of the nonconvex low-rank matrix completion problem considered in this subsection is

$$\min_{Z \in \mathbb{R}^{l\times n}} \left\{ \frac{1}{2}\|\Pi_\Omega(Z - O)\|_F^2 + \mu \sum_{i=1}^{r} p(\sigma_i(Z)) : Z \in \mathcal{B}_R \right\}, \tag{46}$$

where R is a positive scalar, $\mathcal{B}_R := \{Z \in \mathbb{R}^{l\times n} : \|Z\|_F \le R\}$, $O \in \mathbb{R}^\Omega$ is an incomplete observed matrix, $\mu > 0$ is a parameter, $r := \min\{l, n\}$, and $\sigma_i(Z)$ is the ith singular value of Z. The above problem differs from the one considered in [25] in that it adds the constraint $\|Z\|_F \le R$ to the latter one.

The matrix completion problem in (46) is equivalent to

$$\min_{Z \in \mathbb{R}^{l\times n}} f(Z) + h(Z), \tag{47}$$

where

$$f(Z) = \frac{1}{2}\|\Pi_\Omega(Z - O)\|_F^2 + \mu \sum_{i=1}^{r} [p(\sigma_i(Z)) - p_0\sigma_i(Z)],$$

$$h(Z) = \mu p_0 \|Z\|_* + I_{\mathcal{B}_R}(Z), \quad p_0 = p'(0) = \frac{\beta}{\theta},$$

and $\|\cdot\|_*$ denotes the nuclear norm defined as $\|\cdot\|_* := \sum_{i=1}^{r} \sigma_i(\cdot)$. Note that the inclusion of the constraint $Z \in \mathcal{B}_R$ in (46) implies that the above composite function h has bounded domain and hence satisfies assumption (A3). It is proved in [25] that the second term in the definition of f, i.e., $\mu \sum_{i=1}^{r} [p(\sigma_i(\cdot)) - p_0\sigma_i(\cdot)]$, is concave and $2\mu\tau$-smooth where $\tau = \beta/\theta^2$, so f is nonconvex and smooth. Since h is convex and nonsmooth, the problem in (47) falls into the general class of nonconvex SCO problems (1). It is easy to see that the pair

$$(M, m) = (\max\{1, 2\mu\tau\}, 2\mu\tau) \tag{48}$$

satisfies assumption (A2).

We use the *MovieLens* dataset[1] to obtain the observed index set Ω and the incomplete observed matrix O. The dataset includes a sparse matrix with 100,000 ratings of {1,2,3,4,5} from 943 users on 1682 movies, namely $l = 943$ and $n = 1682$. The radius R is chosen as the Frobenius norm of the matrix of size 943×1682 containing the same entries as O in Ω and 5 in the entries outside of Ω.

We start all seven methods from the same initial point Z_0 that is sampled from the standard Gaussian distribution and is within $\mathcal{B}_R$. The parameter α is set to 0.5 in AC and 0.1 in ACT.

Numerical results of the seven methods are given in Table 12 and some statistic measures of AC and ACT are given in Table 13. The format of Table 12 is similar to that of Table 10 with the exception that the second to eighth columns also provide

[1] http://grouplens.org/datasets/movielens/.

TABLE 12
Numerical results for AG, NM, UP, NC, AD, ACT, and AC.

M	Function value/Iteration count/Running time (s)						
	AG	NM	UP	NC	AD	ACT	AC
4.4	2257	**1809**	2605	2628	2625	2252	2288
	3856	1036	521	4780	1674	5420	765
	4568	1033	1545	3925	1946	5803	**833**
8.9	3886	**3359**	4261	4246	4203	3846	3884
	9158	1617	576	9751	1794	8726	968
	10251	1605	1621	7901	1930	8806	**1065**
20	4282	**3635**	4637	4641	4582	4282	4267
	22902	2875	676	22259	2209	13031	1079
	29274	2836	1914	15912	2364	13869	**1200**
30	5967	**5237**	6753	6380	6293	5963	5975
	37032	3717	606	32223	1963	18267	1085
	41673	4182	1628	22265	2104	19913	**1214**

TABLE 13
AC and ACT statistics.

M	AC			ACT		
	Max	Avg	Good	Max	Avg	Good
4.4	1.00	0.31	96%	1.00	0.45	99%
8.9	1.00	0.28	94%	1.39	0.48	99%
20	0.99	0.25	91%	2.65	0.72	99%
30	0.97	0.23	89%	4.36	1.13	96%

the function values of (46) at the last iteration and the numbers of iterations for all seven methods. Note that the first columns of Tables 12 and 13 give the value of M computed according to (48). The number of resolvent evaluations is 1 in NC, 2 in AG, AC, and ACT, 1 or 2 in NM, 1 on average in AD, and 3 on average in UP. The bold numbers highlight the method that has the best performance in an instance of the problem.

In summary, computational results demonstrate that (i) the computed average curvature for AC is small compared with M and the computed maximum curvature; (ii) the percentage of good iterations of AC lies in a suitable range; and (iii) AC has the best performance in terms of running time. Although AC uses the least amount of time to terminate, NM finds solutions with the smallest objective function values.

5.5. Nonnegative matrix factorization. This subsection focuses on the following NMF problem:

$$\min\left\{f(X,Y) := \frac{1}{2}\|A - XY\|_F^2 : X \geq 0, Y \geq 0\right\}, \tag{49}$$

where $A \in \mathbb{R}^{n\times l}$, $X \in \mathbb{R}^{n\times p}$, and $Y \in \mathbb{R}^{p\times l}$, which have been thoroughly studied in the literature (see, e.g., [8, 12]).

This subsection reports the efficiency of directly using all seven methods to solve (49) without making use of its two-block structure. We use the facial image dataset provided by AT&T Laboratories Cambridge[2] to construct the matrix A. More specifically, this dataset consists of 400 images, and each of those contains 92×112 pixels

[2]https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

TABLE 14
Numerical results for AG, NM, UP, NC, AD, ACT, and AC.

Method	Iteration count	Running time(s)
AG	786	73.03
NM	162	14.91
UP	37	11.12
NC	656	41.67
AD	44	5.21
ACT	41	6.54
AC	36	**4.70**

with 256 gray levels per pixel. It results in an $n \times l = 10{,}304 \times 400$ matrix A whose columns are the vectorized images. The dimension p is set to 20.

We start all methods from the initial point $(X_0, Y_0) = (\mathbf{1}^{n\times p}/(np), \mathbf{1}^{p\times l}/(pl))$, where $\mathbf{1}^{n\times p}$ and $\mathbf{1}^{p\times l}$ are matrices of all ones of sizes $n \times p$ and $p \times l$, respectively. We estimate M in (5) as $M = 100 \times \mathcal{C}((X_0, Y_0), (0, 0))$ where $\mathcal{C}(\cdot, \cdot)$ is defined in (4). Since no sharper $m < M$ satisfying the first inequality in (5) is known, we simply set $m = M$. The parameter α is set to 0.7 in both AC and ACT.

Numerical results for the seven methods are given in Table 14. The bold numbers highlight the method that has the best performance in the problem. The best objective function values obtained by all seven methods are not reported since they are essentially the same.

6. Concluding remarks. This paper presents an average curvature accelerated composite gradient method, namely, the AC-ACG method, for solving the N-SCO problem which is based on the average of all observed curvatures. More specifically, as opposed to other ACG variants, which use a known Lipschitz constant or a backtracking procedure that searches for a good upper curvature M_k, AC-ACG uses the average of all observed curvatures to compute M_k (see (14)) and always accepts the first computed iterate according to (2) no matter whether M_k is good or not. A nice feature of AC-ACG is that its convergence rate bound is expressed in terms of M_k rather than an upper curvature $M \geq \bar{M}$.

We now discuss some possible extensions of this paper. First, numerical results show that the AC variant, which computes C_k as in (40), performs substantially better than previous ACG variants as well as the ACT variant, which is closer to the main method analyzed in this paper, namely, AC-ACG. However, convergence rate analysis of AC (possibly with γ and α satisfying (8)) is an interesting open problem. Second, the AC-ACG method performs two resolvent evaluations of h per iteration. It would be desirable to develop AC-ACG variants which only perform one resolvent evaluation of h per iteration. Third, the analysis of AC-ACG assumes that assumption (A3) holds, i.e., $\operatorname{dom} h$ is bounded. It would be interesting to develop a variant of it with a provable iteration-complexity similar to the one in this paper without assuming (A3).

Appendix A. A technical result. Recall the definition of a good upper curvature of f given above (3).

LEMMA A.1. *If M_k is a good upper curvature of f at $\tilde{x}_k$ and $y_{k+1} = y(\tilde{x}_k; M_k)$ where $y(\cdot;\cdot)$ is defined in* (2)*, then*

$$\phi(y_{k+1}) \leq \phi(\tilde{x}_k) - \frac{M_k}{2}\|y_{k+1} - \tilde{x}_k\|^2. \tag{50}$$

Proof. Using the fact that M_k is a good upper curvature of f at $\tilde{x}_k$ and (3), we have

$$\phi(y_{k+1}) \le \ell_f(y_{k+1};\tilde{x}_k) + h(y_{k+1}) + \frac{M_k}{2}\|y_{k+1}-\tilde{x}_k\|^2. \tag{51}$$

It follows from the definition of y_{k+1}, (2), and the fact that the objective function in (2) is M_k-strongly convex that for every $u \in \operatorname{dom} h$,

$$\begin{aligned}\ell_f(u;\tilde{x}_k) + h(u) + \frac{M_k}{2}\|u-\tilde{x}_k\|^2 &\\ \ge \ell_f(y_{k+1};\tilde{x}_k) + h(y_{k+1}) + \frac{M_k}{2}\|y_{k+1}-\tilde{x}_k\|^2 + \frac{M_k}{2}\|u-y_{k+1}\|^2,&\end{aligned}$$

which together with $u = \tilde{x}_k$ implies that

$$\phi(\tilde{x}_k) \ge \ell_f(y_{k+1};\tilde{x}_k) + h(y_{k+1}) + M_k\|y_{k+1}-\tilde{x}_k\|^2.$$

Now inequality (50) immediately follows from (51) and the above inequality. ∎

Acknowledgments. We are grateful to Guanghui Lan and Saeed Ghadimi for providing the code for the UPFAG method of their paper [7]. We are also grateful to the two anonymous referees and the associate editor Defeng Sun for providing helpful comments on earlier versions of this manuscript.

REFERENCES

[1] A. Auslender and M. Teboulle, *Interior gradient and proximal methods for convex and conic optimization*, SIAM J. Optim., 16 (2006), pp. 697–725.
[2] A. Beck and M. Teboulle, *Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems*, IEEE Trans. Image Process., 18 (2009), pp. 2419–2434.
[3] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sci., 2 (2009), pp. 183–202.
[4] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford, *Accelerated methods for nonconvex optimization*, SIAM J. Optim., 28 (2018), pp. 1751–1772.
[5] D. Drusvyatskiy and C. Paquette, *Efficiency of minimizing compositions of convex functions and smooth maps*, Math. Program., (2018), pp. 1–56.
[6] S. Ghadimi and G. Lan, *Accelerated gradient methods for nonconvex nonlinear and stochastic programming*, Math. Program., 156 (2016), pp. 59–99.
[7] S. Ghadimi, G. Lan, and H. Zhang, *Generalized uniformly optimal methods for nonlinear programming*, J. Sci. Comput., 79 (2019), pp. 1854–1881.
[8] N. Gillis, *The why and how of nonnegative matrix factorization*, Regularization, Optimization, Kernels, and Support Vector Machines, 12 (2014), pp. 257–291.
[9] Q. Gu, Z. Wang, and H. Liu, *Sparse PCA with oracle property*, in Advances in Neural Information Processing Systems, 27, 2014, pp. 1529–1537.
[10] 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 (2019), pp. 2566–2593.
[11] G. Lan, Z. Lu, and R. D. C. Monteiro, *Primal-dual first-order methods with $\mathcal{O}(1/\epsilon)$ iteration-complexity for cone programming*, Math. Program., 126 (2011), pp. 1–29.
[12] D. D. Lee and H. S. Seung, *Learning the parts of objects by non-negative matrix factorization*, Nature, 401 (1999), pp. 788–791.
[13] H. Li and Z. Lin, *Accelerated proximal gradient methods for nonconvex programming*, in Advances in Neural Information Processing Systems, 28, 2015, pp. 379–387.
[14] Q. Li, Y. Zhou, Y. Liang, and P. K. Varshney, *Convergence analysis of proximal gradient with momentum for nonconvex optimization*, in Proceedings of the 34th International Conference on Machine Learning, Vol. 70, 2017, pp. 2111–2119.
[15] J. Liang and R. D. C. Monteiro, *A Doubly Accelerated Inexact Proximal Point Method for Nonconvex Composite Optimization Problems*, https://arxiv.org/abs/1811.11378, 2018.

[16] J. Liang, R. D. C. Monteiro, and C.-K. Sim, *A FISTA-Type Accelerated Gradient Algorithm for Solving Smooth Nonconvex Composite Optimization Problems*, https://arxiv.org/abs/1905.07010, 2019.
[17] R. D. C. Monteiro and B. F. Svaiter, *An accelerated hybrid proximal extragradient method for convex optimization and its implications to second-order methods*, SIAM J. Optim., 23 (2013), pp. 1092–1125.
[18] Y. Nesterov, *A method for unconstrained convex minimization problem with the rate of convergence* $O(1/k^2)$, Doklady AN SSSR, 269 (1983), pp. 543–547.
[19] Y. Nesterov, *Introductory Lectures on Convex Programming*, Lecture notes, 1998.
[20] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, Kluwer, Boston, 2004.
[21] Y. Nesterov, *Smooth minimization of nonsmooth functions*, Math. Program., 103 (2005), pp. 127–152.
[22] Y. Nesterov, *Gradient methods for minimizing composite functions*, Math. Program., (2012), pp. 1–37.
[23] C. Paquette, H. Lin, D. Drusvyatskiy, J. Mairal, and Z. Harchaoui, *Catalyst Acceleration for Gradient-Based Non-Convex Optimization*, https://arxiv.org/abs/1703.10993, 2017.
[24] P. Tseng, *On Accelerated Proximal Gradient Methods for Convex-Concave Optimization*, http://www.mit.edu/~dimitrib/PTseng/papers.html, 2008.
[25] Q. Yao and J. T. Kwok, *Efficient learning with a family of nonconvex regularizers by redistributing nonconvexity.*, J. Mach. Learn. Res., 18 (2017), pp. 179–1.
[26] Q. Yao, J. T. Kwok, F. Gao, W. Chen, and T.-Y. Liu, *Efficient inexact proximal gradient algorithm for nonconvex problems*, in Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 3308–3314.