Accelerated Proximal Gradient Methods for Nonconvex Programming

Authored by:

Zhouchen Lin
Huan Li

Abstract

Nonconvex and nonsmooth problems have recently received considerable attention in signal/image processing, statistics and machine learning. However, solving the nonconvex and nonsmooth optimization problems remains a big challenge. Accelerated proximal gradient (APG) is an excellent method for convex programming. However, it is still unknown whether the usual APG can ensure the convergence to a critical point in nonconvex programming. To address this issue, we introduce a monitor-corrector step and extend APG for general nonconvex and nonsmooth programs. Accordingly, we propose a monotone APG and a non-monotone APG. The latter waives the requirement on monotonic reduction of the objective function and needs less computation in each iteration. To the best of our knowledge, we are the first to provide APG-type algorithms for general nonconvex and nonsmooth problems ensuring that every accumulation point is a critical point, and the convergence rates remain \$O(1/k2)\$ when the problems are convex, in which k is the number of iterations. Numerical results testify to the advantage of our algorithms in speed.

1 Paper Body

In recent years, sparse and low rank learning has been a hot research topic and leads to a wide variety of applications in signal/image processing, statistics and machine learning. l1 -norm and nuclear norm, as the continuous and convex surrogates of l0 -norm and rank, respectively, have been used extensively in the literature. See e.g., the recent collections [1]. Although l1 -norm and nuclear norm have achieved great success, in many cases they are suboptimal as they can promote sparsity and low-rankness only under very limited conditions [2, 3]. To address this issue, many nonconvex regularizers have been proposed, such as lp -norm [4], Capped-l1 penalty [3], Log-Sum Penalty [2], Minimax Concave Penalty [5], Geman Penalty [6], Smoothly Clipped Absolute Deviation [7] and Schatten-p norm [8]. This trend motivates a revived interest in the analysis and design of algorithms for solving nonconvex and nonsmooth problems, which can be formulated as min F (x) = f (x) + g(x),

x?Rn

(1)

where f is differentiable (it can be nonconvex) and g can be both nonconvex and nonsmooth. Accelerated gradient methods have been at the heart of convex optimization research. In a series of celebrated works [9, 10, 11, 12, 13, 14], several accelerated gradient methods are proposed for problem (1) with convex f and g. In these methods, k iterations are sufficient to find a solution within O k12 error from the optimal objective value. Recently, Ghadimi and Lan [15] presented a unified treatment of accelerated gradient method (UAG) for convex, nonconvex and stochastic optimiza1

Table 1: Comparisons of GD (General Descent Method), iPiano, GIST, GDPA, IR, IFB, APG, UAG and our method for problem (1). The measurements include the assumption, whether the methods accelerate for convex programs (CP) and converge for nonconvex programs (NCP). Method name GD [16, 17] iPiano [18] GIST [19] GDPA [20] IR [8, 21] IFB [22] APG [12, 13] UAG [15] Ours

Assumption f + g: KL nonconvex f , convex g nonconvex f , g = g1 ? g2 , g1 , g2 convex nonconvex f , g = g1 ? g2 , g1 , g2 convex special f and g nonconvex f , nonconvex g convex f , convex g nonconvex f , convex g nonconvex f , nonconvex g

Accelerate (CP) No No No No No No Yes Yes Yes

converge (NCP) Yes Yes Yes Yes Yes Yes Unclear Yes Yes

1 tion. They proved that their algorithm converges in nonconvex programming with nonconvex f but

1 convex g and accelerates with an O k2 convergence rate in convex programming for problem (1). Convergence rate about the gradient mapping is also analyzed in [15].

Attouch et al. [16] proposed a unified framework to prove the convergence of a general class of descent methods using the Kurdyka-?ojasiewicz (KL) inequality for problem (1) and Frankel et al. [17] studied the convergence rates of general descent methods under the assumption that the desingularising function ? in KL property has the form of C? t? . A typical example in their framework is the proximal gradient method. However, there is no literature showing that there exists an accelerated gradient method satisfying the conditions in their framework. Other typical methods for problem (1) includes Inertial Forward-Backward (IFB) [22], iPiano [18], General Iterative Shrinkage and Thresholding (GIST) [19], Gradient Descent with Proximal Average(GDPA) [20] and Iteratively Reweighted Algorithms (IR) [8, 21]. Table 1 demonstrates that the existing methods are not ideal. GD and IFB cannot accelerate the convergence for convex programs. GIST and GDPA require that g should be explicitly written as a difference of two convex functions. iPiano demands the convexity of g and IR is suitable for some special cases of problem (1). APG can accelerate the convergence for convex programs, however, it is unclear whether APG can converge to critical points for nonconvex programs. UAG can ensure the convergence for nonconvex programming, however, it requires g to be convex. This restricts the applications of UAG to solving nonconvexly regularized problems, such as sparse and low rank learning. To the best of our knowledge, extending the accelerated gradient method for general nonconvex and nonsmooth programs

while keeping the O k12 convergence rate in the convex case remains an open problem. In this paper we aim to extend Beck and Teboulle?s APG [12, 13] to solve general nonconvex and nonsmooth problem (1). APG first extrapolates a point yk by combining the current point and the previous point, then solves a proximal mapping problem. When extending APG to nonconvex programs the chief difficulty lies in the extrapolated point yk . We have little restriction on F (yk) when the convexity is absent. In fact, F (yk) can be arbitrarily larger than F (xk) when yk is a bad extrapolation, especially when F is oscillatory. When xk+1 is computed by a proximal mapping at a bad yk , F (xk+1) may also be arbitrarily larger than F (xk). Beck and Teboulle?s monotone APG [12] ensures F (xk+1) ? F (xk). However, this is not enough to ensure the convergence to critical points. To address this issue, we introduce a monitor satisfying the sufficient descent property to prevent a bad extrapolation of yk and then correct it by this monitor. In summary, our contributions include: 1. We propose APG-type algorithms for general nonconvex and nonsmooth programs (1). We first extend Beck and Teboulle?s monotone APG [12] by replacing their descent condition with sufficient descent condition. This critical change ensures that every accumulation point is a critical point. Our monotone APG satisfies some modified conditions for the framework of [16, 17] and thus stronger results on convergence rate can be obtained under the KL 1 Except for the work under the KL assumption, convergence for nonconvex problems in this paper and the references of this paper means that every accumulation point is a critical point.

2

assumption. Then we propose a nonmonotone APG, which allows for larger stepsizes when line search is used and reduces the average number of proximal mappings in each iteration. Thus it can further speed up the convergence in practice.

2. For our APGs, the convergence rates maintain O k12 when the problems are convex. This result is of great significance when the objective function is locally convex in the neighborhoods of local minimizers even if it is globally nonconvex.

2 2.1

Preliminaries Basic Assumptions

Note that a function g : Rn ? (??, +?] is said to be proper if dom g 6= ?, where dom g = {x ? R : g(x) ¡ +?}. g is lower semicontinuous at point x0 if lim inf x?x0 g(x) ? g(x0). In problem (1), we assume that f is a proper function with Lipschitz continuous gradients and g is proper and lower semicontinuous. We assume that F (x) is coercive, i.e., F is bounded from below and F (x) ? ? when kxk ? ?, where k ? k is the l2 -norm. 2.2

KL Inequality

Definition 1. [23] A function f : Rn ? (??, +?] is said to have the KL property at u ? dom?f := {x ? Rn : ?f (u) 6= ?} if there T exists ? ? (0, +?], a neighborhood U of u and a function ? ? ?? , such that for all u ? U {u ? Rn : f (u) ¡ f (u) ¡ f (u) + ?}, the following inequality holds ?0 (f (u) ? f (u))dist(0, ?f (u)) ¿ 1, (2) where ?? stands for a class of function ? : [0, ?) ? R+ satisfying: (1) ? is concave and C 1 on (0, ?); (2) ? is continuous at 0, ?(0) = 0; and (3) ?0 (x) ¿ 0, ?x ? (0, ?). All semi-algebraic functions and subanalytic functions satisfy the KL property. Specially, the desingularising function ?(t) of semi-algebraic functions can be chosen to be the form of C? t? with ? ? (0, 1]. Typical semi-algebraic functions include real polynomial functions, kxkp with p ? 0, rank(X), the indicator function of PSD cone, Stiefel manifolds and constant rank matrices [23]. 2.3

Review of APG in the Convex Case

We first review APG in the convex case. Bech and Teboulle [13] extend Nesterov?s accelerated gradient method to the nonsmooth case. It is named the Accelerated Proximal Gradient method and consists of the following steps: tk?1 ? 1 (xk ? xk?1), tk xk+1 = prox?k g (yk ? ?k ?f (yk)), p 4(tk)2 + 1 + 1 tk+1 = , 2

yk = xk +

(3) (4) (5)

1 where the proximal mapping is defined as prox?g (x) = argminu g(u) + 2? kx ? uk2 . APG is not a monotone algorithm, which means that F (xk+1) may not be smaller than F (xk). So Beck and Teboulle [12] further proposed a monotone APG, which consists of the following steps:

tk?1 ? 1 tk?1 (zk ? xk) + (xk ? xk?1), tk tk zk+1 = prox?k g (yk ? ?k ?f (yk)), p 4(tk)2 + 1 + 1 tk+1 = , 2

zk+1 , if F (zk+1) ? F (xk), xk+1 = xk , otherwise. yk = xk +

3

(6) (7) (8) (9)

3

APGs for Nonconvex Programs

In this section, we propose two APG-type algorithms for general nonconvex nonsmooth problems.

We establish the convergence in the nonconvex case and the O k12 convergence rate in the convex case. When the KL property is satisfied we also provide stronger results on convergence rate. 3.1

Monotone APG

We give two reasons that result in the difficulty of convergence analysis on the usual APG [12, 13] for nonconvex programs: (1) yk may be a bad extrapolation, (2) in [12] only descent property, F (xk+1) ? F (xk), is ensured. To address these issues, we need to monitor and correct yk when it has the potential to fail, and the monitor should enjoy the property of sufficient descent which is critical to ensure the convergence to a critical point. As is known, proximal gradient methods can make sure sufficient descent [16] (cf. (15)). So we use a proximal gradient step as the monitor. More specially, our algorithm consists of the following steps: tk?1 tk?1 ? 1 yk = xk + (zk ? xk) + (xk ? xk?1), (10) tk tk zk+1 = prox?y g (yk ? ?y ?f (yk)), (11) vk+1 = prox?x g (xk ? ?x ?f (xk)), p 4(tk)2 + 1 + 1 , tk+1 = 2

zk+1 , if F (zk+1) ? F (vk+1), xk+1 = vk+1 , otherwise.

(12) (13) (14)

where ?y and ?x can be fixed constants satisfying ?y ¡ L1 and ?x ¡ L1 , or dynamically computed by backtracking line search initialized by Barzilai-Borwein rule2 . L is the Lipschitz constant of ?f . Our algorithm is an extension of Beck and Teboulle?s monotone APG [12]. The difference lies in the extra v, as the role of monitor, and the correction step of x-update. In (9) F (zk+1) is compared with F (xk), while in (14) F (zk+1) is compared with F (vk+1). A further difference is that Beck and Teboulle?s algorithm only ensures descent while our algorithm makes sure sufficient descent, which means F (xk+1) ? F (xk) ? ?kvk+1 ? xk k2 , (15) where ? ¿ 0 is a small constant. It is not difficult to understand that only the descent property cannot ensure the convergence to a critical point in nonconvex programming. We present our convergence result in the following theorem3 . Theorem 1. Let f be a proper function with Lipschitz continuous gradients and g be proper and lower semicontinuous. For nonconvex f and nonconvex nonsmooth g, assume that F (x) is coercive. Then {xk } and {vk } generated by (10)-(14) are bounded. Let x? be any accumulation point of {xk }, we have 0 ? ?F (x?), i.e., x? is a critical point. A remarkable aspect of our algorithm is that although we have made some modifications on Beck and Teboulle?s algorithm, the O k12 convergence rate in the convex case still holds. Similar to Theorem 5.1 in [12], we have the following theorem on the accelerated convergence in the convex case: Theorem 2. For convex f and g, assume that ?f is Lipschitz continuous, let x? be any global optimum, then {xk } generated by (10)-(14) satisfies 2 F (xN +1) ? F (x?) ? kx0 ? x? k2 , (16) ?y (N + 1)2 When the objective function is locally convex in the neighborhood of local minimizers, Theorem 2 means that APG can ensure to have an O k12 convergence rate when approaching to a local minimizer, thus accelerating the convergence. For better reference, we summarize the proposed monotone APG algorithm in Algorithm 1. 2 3

For the detail of line search with Barzilai-Borwein initializtion please see Supplementary Materials. The proofs in this paper can be found in Supplementary Materials.

4

Algorithm 1 Monotone APG Initialize z1 = x1 = x0 , t1 = 1, t0 = 0, ?y ¡ L1 , ?x ¡ for k = 1, 2, 3, ? ? ? do update yk , zk+1 , vk+1 , tk+1 and xk+1 by (10)-(14). end for

3.2

1 L.

Convergence Rate under the KL Assumption

The KL property is a powerful tool and is studied by [16], [17] and [23] for a class of general descent methods. The usual APG in [12, 13] does not satisfy the sufficient descent property, which is crucial to use the KL property, and thus has no conclusions under the KL assumption. On the other hand, due to the intermediate variables yk , vk and zk , our algorithm is more complex than the general descent methods and also does not satisfy the conditions therein. However, due to the monitor-corrector step (12) and (14), some modified conditions4 can be satisfied and we can still get some exciting results under the KL assumption. With the same framework of [17], we have the following theorem. Theorem 3. Let f be a proper function with Lipschitz continuous gradients and g be proper and lower semicontinuous. For nonconvex f and nonconvex nonsmooth g, assume that F (x) is coercive. If we further assume that f and g satisfy the KL property and the desingularising function has the form of ?(t) = C? t? for some C ¿ 0, ? ? (0, 1], then 1. If ? = 1, then there exists k1 such that F (xk) = F ? for all k ¿ k1 and the algorithm terminates in finite steps. 2. If ? ? [21 , 1), then there exists k2 such that for all k ¿ k2 ,

k?k2 d1 C 2 ? F (xk) ? F ? rk2 . 1 + d1 C 2 3. If ? ? (0, 21), then there exists k3 such that for all k ¿ k3 , 1

1?2? C ? F (xk) ? F ? , (k ? k3)d2 (1 ? 2?)

(17)

(18)

where F ? is the same function value at all the accumulation points of {xk }, rk = F (vk) ? 2

n 2??1

o C F ? , d1 = ?1x + L / 2?1 x ? L2 and d2 = min 2d11 C , 1?2? 2 2??2 ? 1 r02??1 When F (x) is a semi-algebraic function, the desingularising function ?(t) can be chosen to be the form of C? t? with ? ? (0, 1] [23]. In this case, as shown in Theorem 3, our algorithm converges in finite iterations when ? = 1, converges with a linear rate when ? ? [12 , 1) and a sublinear rate (at least O(k1)) when ? ? (0, 12) for the gap F (xk) ? F ? . This is the same as the results mentioned in [17], although our algorithm does not satisfy the conditions therein. 3.3

Nonmonotone APG

Algorithm 1 is a monotone algorithm. When the problem is ill-conditioned, a monotone algorithm has to creep along the bottom of a narrow curved valley so that the objective function value does not increase, resulting in short stepsizes or even zigzagging and hence slow convergence [24]. Removing the requirement on monotonicity can improve convergence speed because larger stepsizes can be adopted when line search is used. On the other hand, in Algorithm 1 we need to compute zk+1 and vk+1 in each iteration and use vk+1 to monitor and correct zk+1 . This is a conservative strategy. In fact, we can accept zk+1 as xk+1 directly if it satisfies some criterion showing that yk is a good extrapolation. Then vk+1 is computed only when this criterion is not met. Thus, we can reduce the average number of proximal 4

For the details of difference please see Supplementary Materials.

5

mappings, accordingly the computation cost, in each iteration. So in this subsection we propose a nonmonotone APG to speed up convergence. In monotone APG, (15) is ensured. In nonmonotone APG, we allow xk+1 to make a larger objective function value than F (xk). Specifically, we allow xk+1 to yield an objective function value smaller than ck , a relaxation of F (xk). ck should not be too far from F (xk). So the average of F (xk), F (xk?1), ? ? ? , F (x1) is a good choice. Thus we follow [24] to define ck as a convex combination of F (xk), F (xk?1), ? ? ? , F (x1) with exponentially decreasing weights: Pk k?j F (xj) j=1 ? ck = , (19) Pk k?j j=1 ? where ? ? [0, 1) controls the degree of nonmonotonicity. In practice ck can be efficiently computed by the following recursion: qk+1 = ?qk + 1, ?qk ck + F (xk+1) ck+1 = , qk+1

(20) (21)

where q1 = 1 and c1 = F (x1). According to (14), we can split (15) into two parts by the different choices of xk+1 . Accordingly, in nonmonotone APG we consider the following two conditions to replace (15): F (zk+1) ? ck ? ?kzk+1 ? yk k2 ,

(22)

2

(23)

F (vk+1) ? ck ? ?kvk+1 ? xk k .

We choose (22) as the criteria mentioned before. When (22) holds, we deem that yk is a good extrapolation and accept zk+1 directly. Then we do not compute vk+1 in this case. However, (22) does not hold all the time. When it fails, we deem that yk may not be a good extrapolation. In this case, we compute vk+1 by (12) satisfying (23), and then monitor and correct zk+1 by (14). (23) is ensured when ?x ? 1/L. When backtracking line search is used, such vk+1 that satisfies (23) can be found in finite steps5 . Combing (20), (21), (22) and xk+1 = zk+1 we have ck+1 ? ck ?

?kxk+1 ? yk k2 . qk+1

(24)

Similarly, replacing (22) and xk+1 = zk+1 by (23) and xk+1 = vk+1 , respectively, we have ck+1 ? ck ?

?kxk+1 ? xk k2 . qk+1

(25)

This means that we replace the sufficient descent condition of F (xk) in (15) by the sufficient descent of ck . We summarize the nonmonotone APG in Algorithm 26 . Similar to monotone APG, nonmonotone

APG also enjoys the convergence property in the nonconvex case and the O k12 convergence rate in the convex case. We present our convergence result in Theorem 4. Theorem 2 still holds for Algorithm 2 with no modification. So we omit it here. Define ?1 = {k1 , k2 , ? ? ? , kj , ? ? ? } and ?2 = {m1 , m2 , ? ? ? , mj , ? ? ? }, such that in Algorithm 2, (22) holds and xk+1 = zk+1 is executed for all k T= kj ? ?1 . For S all k = mj ? ?2 , (22) does not hold and (14) is executed. Then we have ?1 ?2 = ?, ?1 ?2 = {1, 2, 3, ? ? ? , } and the following theorem holds. Theorem 4. Let f be a proper function with Lipschitz continuous gradients and g be proper and lower semicontinuous. For nonconvex f and nonconvex nonsmooth g, assume that F (x) is coercive. Then {xk }, {vk } and {ykj } where kj ? ?1 generated by Algorithm 2 are bounded, and 1. if ?1 or ?2 is finite, then for any accumulation point {x? } of {xk }, we have 0 ? ?F (x?). 5 6

See Lemma 2 in Supplementary Materials. Please see Supplementary Materials for nonmonotone APG with line search.

6

Algorithm 2 Nonmonotone APG Initialize z1 = x1 = x0 , t1 = 1, t0 = 0, ? ? [0, 1), ? ¿ 0, c1 = F (x1), q1 = 1, ?x ¡ 1 L. for k = 1, 2, 3, ? ? ? do tk?1 ?1 yk = xk + tk?1 (xk ? xk?1), tk (zk ? xk) + tk zk+1 = prox?y g (yk ? ?y ?f (yk)) if F (zk+1) ? ck ? ?kzk+1 ? yk k2 then xk+1 = zk+1 . else vk+1 = prox?x g (xk ? ?x ?f (xk)),

zk+1 , if F (zk+1) ? F (vk+1), xk+1 = vk+1 , otherwise. end if ? 4(tk)2 +1+1 tk+1 = , 2 qk+1 = ?qk + 1, (xk+1) ck+1 = ?qk ckq+F . k+1 end for

1 L , ?y

¡

2. if ?1 and ?2 are both infinite, then for any accumulation point x? of {xkj +1 }, y? of {ykj } where kj ? ?1 and any accumulation point v? of {vmj +1 }, x? of {xmj } where mj ? ?2 , we have 0 ? ?F (x?), 0 ? ?F (y?) and 0 ? ?F (v?).

4

Numerical Results

In this section, we test the performance of our algorithm on the problem of Sparse Logistic Regression (LR)7 .Sparse LR is an attractive extension to LR as it can reduce overfitting and perform feature selection simultaneously. Sparse LR is widely used in areas such as bioinformatics [25] and text categorization [26]. In this subsection, we follow Gong et al. [19] to consider Sparse LR with a nonconvex regularizer: n

min w

1X log(1 + exp(?yi xTi w)) + r(w). n i=1

(26)

We choose r(w) as the capped l1 penalty [3], defined as r(w) = ?

d X

min(—wi —, ?),

? ¿ 0.

(27)

i=1

We compare monotone APG (mAPG) and nonmonotone APG (nmAPG) with monotone GIST8 (mGIST), nonmonotone GIST (nmGIST) [19] and IFB [22]. We test the performance on the real-sim data set9 , which contains 72309 samples of 20958 dimensions. We follow [19] to set ? = 0.0001, ? = 0.1? and the starting point as zero vectors. In nmAPG we set ? = 0.8. In IFB the inertial parameter ? is set at 0.01 and the Lipschitz constant is computed by backtracking. To make a fair comparison, we first run mGIST. The algorithm is terminated when the relative change of two consecutive objective function values is less than 10?5 or the number of iterations exceeds 1000. This termination condition is the same as in [19]. Then we run nmGIST, mAPG, nmAPG and IFB. These four algorithms are terminated when they achieve an equal or smaller objective function value than that by mGIST or the number of iterations exceeds 1000. We randomly choose 90% of the data as training data and the rest as test data. The experiment result is averaged over 10 runs. All algorithms are run on Matlab 2011a and Windows 7 with an Intel Core i3 2.53 GHz CPU and 4GB memory. The result is reported in Table 2. We also plot the curves of objective function values vs. iteration number and CPU time in Figure 1. 7

For the sake of space limitation we leave another experiment, Sparse PCA, in Supplementary Materials. http://www.public.asu.edu/ yje02/Software/GIST 9 http://www.csie.ntu.tw/cjlin/libsvmtools/datasets 8

7

Table 2: Comparisons of APG, GIST and IFB on the sparse logistic regression problem. The quantities include number of iterations, averaged number of line searches in each iteration, computing time (in seconds) and test error. They are averaged over 10 runs. Method #Iter. #Line search Time test error mGIST 994 2.19 300.42 2.94% nmGIST 806 1.69 222.22 2.94% 635 2.59 215.82 2.96% IFB mAPG 175 2.99 133.23 2.93% nmAPG 146 1.01 42.99 2.97%

We have the following observations: (1) APG-type methods need much fewer iterations and less computing time than GIST and IFB to reach the same (or smaller) objective function values. As GIST is indeed a Proximal Gradient method (PG) and IFB is an extension of PG, this verifies that APG can indeed accelerate the convergence in practice. (2) nmAPG is faster than mAPG. We give two reasons: nmAPG avoids the computation of vk in most of the time and reduces the number of line searches in each iteration. We mention that in mAPG line search is performed in both (11) and (12), while in nmAPG only the computation of zk+1 needs line search in every iteration. vk+1 is computed only when necessary. We note that the average number of line searches in nmAPG is nearly one. This means that (22) holds in most of the time. So we can trust that zk can work well in most of the time and only in a few times vk is computed to correct zk and yk . On the other hand, nonmonotonicity allows for larger stepsizes, which results in fewer line searches. ?0.8

?0.8 mGIST nmGIST IFB mAPG nmAPG

?1 ?1.2

?1.2 ?1.4 Function Value

Function Value

?1.4 ?1.6 ?1.8

?1.6 ?1.8

?2

?2

?2.2

?2.2

?2.4

?2.4

?2.6

mGIST nmGIST IFB mAPG nmAPG

?1

0

200

400

600
800
?2.6
1000
Iteration
(a) Objective function value v.s. iteration
0
50
100
150 CPU Time
200
250
300
(b) Objective function value v.s. time
Figure 1: Compare the objective function value produced by APG, GIST and IFB.
5
Conclusions
In this paper, we propose two APG-type algorithms for efficiently solving general nonconvex nonsmooth problems, which are abundant in machine learning. We provide a detailed convergence analysis, showing that every accumulation point is a critical point for general nonconvex nonsmooth

programs and the convergence rate is maintained at O k12 for convex programs. Nonmonotone APG allows for larger stepsizes and needs less computation cost in each iteration and thus is faster than monotone APG in practice. Numerical experiments testify to the advantage of the two algorithms. Acknowledgments Zhouchen Lin is supported by National Basic Research Program of China (973 Program) (grant no. 2015CB352502), National Natural Science Foundation (NSF) of China (grant nos. 61272341 and 61231002), and Microsoft Research Asia Collaborative Research Program. He is the corresponding author.
8

2 References

[1] F. Yun, editor. Low-rank and sparse modeling for visual analysis. Springer, 2014. 1 [2] E.J. Candes, M.B. Wakin, and S.P. Boyd. Enhancing sparsity by reweighted l1 minimization. Journal of Fourier Analysis and Applications, 14(5):877?905, 2008. 1 [3] T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. The Journal of Machine Learning Rearch, 11:1081?1107, 2010. 1, 7 [4] S. Foucart and M.J. Lai. Sparsest solutions of underdeterminied linear systems via lq minimization for 0 ¡ q ? 1. Applied and Computational Harmonic Analysis, 26(3):395?407, 2009. 1 [5] C.H. Zhang. Nearly unbiased variable selection under minimax concave penalty. The Annals of Statistics, 38(2):894?942, 2010. 1 [6] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. IEEE Transactions on Image Processing,

4(7):932?946, 1995. 1 [7] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of the American Statistical Association, 96(456):1348?1360, 2001. 1 [8] K. Mohan and M. Fazel. Iterative reweighted algorithms for matrix rank minimization. The Journal of Machine Learning Research, 13(1):3441?3473, 2012. 1, 2 [9] Y.E. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence O(1/k2). Soviet Mathematics Doklady, 27(2):372?376, 1983. 1 [10] Y.E. Nesterov. Smooth minimization of nonsmooth functions. Mathematical programming, 103(1):127? 152, 2005. 1 [11] Y.E. Nesterov. Gradient methods for minimizing composite objective functions. Technical report, Center for Operations Research and Econometrics(CORE), Catholie University of Louvain, 2007. 1 [12] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. IEEE Transactions on Image Processing, 18(11):2419?2434, 2009. 1, 2, 3, 4, 5 [13] A. Beck and M. Teboulle. A fast iterative shrinkage thresholding algorithm for linear inverse problems. SIAM J. Imaging Sciences, 2(1):183?202, 2009. 1, 2, 3, 4, 5 [14] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. Technical report, University of Washington, Seattle, 2008. 1 [15] S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. arXiv preprint arXiv:1310.3787, 2013. 1, 2 [16] H. Attouch, J. Bolte, and B.F. Svaier. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. Mathematical Programming, 137:91?129, 2013. 2, 4, 5 [17] P. Frankel, G. Garrigos, and J. Peypouquet. Splitting methods with variable metric for Kurdyka?ojasiewicz functions and general convergence rates. Journal of Optimization Theory and Applications, 165:874?900, 2014. 2, 5 [18] P. Ochs, Y. Chen, T. Brox, and T. Pock. IPiano: Inertial proximal algorithms for nonconvex optimization. SIAM J. Image Sciences, 7(2):1388?1419, 2014. 2 [19] P. Gong, C. Zhang, Z. Lu, J. Huang, and J. Ye. A general iterative shrinkage and thresholding algorithm for nonconvex regularized optimization problems. In ICML, pages 37?45, 2013. 2, 7 [20] W. Zhong and J. Kwok. Gradient descent with proximal average for nonconvex and composite regularization. In AAAI, 2014. 2 [21] P. Ochs, A. Dosovitskiy, T. Brox, and T. Pock. On iteratively reweighted algorithms for non-smooth nonconvex optimization in computer vision. SIAM J. Imaging Sciences, 2014. 2 [22] R.L. Bot, E.R. Csetnek, and S. L?aszl?o. An inertial forward-backward algorithm for the minimization of the sum of two nonconvex functions. Preprint, 2014. 2, 7 [23] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Mathematical Programming, 146(1-2):459?494, 2014. 3, 5 [24] H. Zhang and W.W. Hager. A nonmonotone line search technique and its application to unconstrained optimization. SIAM J. Optimization, 14:1043?1056, 2004. 5, 6 [25] S.K. Shevade and S.S. Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. Bioinformatics, 19(17):2246?2253, 2003. 7 [26] A. Genkin, D.D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. Technometrics, 49(14):291?304, 2007. 7

9