Scalable nonconvex inexact proximal splitting

Suvrit Sra
Max Planck Institute for Intelligent Systems
72076 Tübigen, Germany
suvrit@tuebingen.mpg.de

Abstract

We study a class of large-scale, nonsmooth, and nonconvex optimization problems. In particular, we focus on nonconvex problems with *composite* objectives. This class includes the extensively studied class of convex composite objective problems as a subclass. To solve composite nonconvex problems we introduce a powerful new framework based on asymptotically *nonvanishing* errors, avoiding the common stronger assumption of vanishing errors. Within our new framework we derive both batch and incremental proximal splitting algorithms. To our knowledge, our work is first to develop and analyze incremental *nonconvex* proximal-splitting algorithms, even if we were to disregard the ability to handle nonvanishing errors. We illustrate one instance of our general framework by showing an application to large-scale nonsmooth matrix factorization.

1 Introduction

This paper focuses on nonconvex *composite objective* problems having the form

$$\text{minimize} \quad \Phi(x) := f(x) + h(x) \qquad x \in \mathcal{X}, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable, $h : \mathbb{R}^m \to \mathbb{R} \cup \{\infty\}$ is lower semi-continuous (lsc) and convex (possibly nonsmooth), and $\mathcal{X}$ is a compact convex set. We also make the common assumption that ∇f is locally (in $\mathcal{X}$) Lipschitz continuous, i.e., there is a constant $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \qquad \text{for all} \quad x, y \in \mathcal{X}. \tag{2}$$

Problem (1) is a natural but far-reaching generalization of *composite objective* convex problems, which enjoy tremendous importance in machine learning; see e.g., [2, 3, 11, 34]. Although, convex formulations are extremely useful, for many difficult problems a nonconvex formulation is natural. Familiar examples include matrix factorization [20, 23], blind deconvolution [19], dictionary learning [18, 23], and neural networks [4, 17].

The primary contribution of this paper is theoretical. Specifically, we present a new algorithmic framework: Nonconvex Inexact Proximal Splitting (NIPS). Our framework solves (1) by "splitting" the task into smooth (gradient) and nonsmooth (proximal) parts. Beyond splitting, the most notable feature of NIPS is that it allows *computational errors*. This capability proves critical to obtaining a scalable, incremental-gradient variant of NIPS, which, to our knowledge, is the first incremental proximal-splitting method for nonconvex problems.

NIPS further distinguishes itself in how it models computational errors. Notably, it *does not* require the errors to vanish in the limit, which is a more realistic assumption as often one has limited to no control over computational errors inherent to a complex system. In accord with the errors, NIPS also *does not* require stepsizes (learning rates) to shrink to zero. In contrast, most incremental-gradient methods [5] and stochastic gradient algorithms [16] *do assume* that the computational errors and stepsizes decay to zero. We do not make these simplifying assumptions, which complicates the convergence analysis a bit, but results in perhaps a more satisfying description.

Our analysis builds on the remarkable work of Solodov [29], who studied the simpler setting of *differentiable* nonconvex problems (which correspond with $h \equiv 0$ in (1)). NIPS is strictly more general: unlike [29] it solves a *non-differentiable* problem by allowing a nonsmooth regularizer $h \not\equiv 0$, and this h is tackled by invoking *proximal-splitting* [8].

Proximal-splitting has proved to be exceptionally fruitful and effective [2, 3, 8, 11]. It retains the simplicity of gradient-projection while handling the nonsmooth regularizer h via its proximity operator. This approach is especially attractive because for several important choices of h, efficient implementations of the associated proximity operators exist [2, 22, 23]. For convex problems, an alternative to proximal splitting is the subgradient method; similarly, for nonconvex problems one may use a generalized subgradient method [7, 12]. However, as in the convex case, the use of subgradients has drawbacks: it fails to exploit the composite structure, and even when using sparsity promoting regularizers it does not generate intermediate sparse iterates [11].

Among batch nonconvex splitting methods, an early paper is [14]. More recently, in his pioneering paper on convex composite minimization, Nesterov [26] also briefly discussed nonconvex problems. Both [14] and [26], however, enforced monotonic descent in the objective value to ensure convergence. Very recently, Attouch et al. [1] have introduced a generic method for nonconvex nonsmooth problems based on Kurdyka-Łojasiewicz theory, but their entire framework too hinges on descent. A method that uses nonmontone line-search to eliminate dependence on strict descent is [13].

In general, the insistence on strict descent and *exact gradients* makes many of the methods unsuitable for incremental, stochastic, or online variants, all of which usually lead to a nonmonotone objective values especially due to *inexact* gradients. Among nonmonotonic methods that apply to (1), we are aware of the generalized gradient-type algorithms of [31] and the stochastic generalized gradient methods of [12]. Both methods, however, are analogous to the usual subgradient-based algorithms that fail to exploit the composite objective structure, unlike proximal-splitting methods.

But proximal-splitting methods do not apply out-of-the-box to (1): nonconvexity raises significant obstructions, especially because nonmonotonic descent in the objective function values is allowed and inexact gradient might be used. Overcoming these obstructions to achieve a scalable non-descent based method that allows inexact gradients is what makes our NIPS framework novel.

2 The NIPS Framework

To simplify presentation, we replace h by the penalty function

$$g(x) := h(x) + \delta(x|\mathcal{X}), \tag{3}$$

where $\delta(\cdot|\mathcal{X})$ is the *indicator function* for $\mathcal{X}$: $\delta(x|\mathcal{X}) = 0$ for $x \in \mathcal{X}$, and $\delta(x|\mathcal{X}) = \infty$ for $x \notin \mathcal{X}$. With this notation, we may rewrite (1) as the *unconstrained* problem:

$$\min_{x\in\mathbb{R}^n} \quad \Phi(x) := f(x) + g(x), \tag{4}$$

and this particular formulation is our primary focus. We solve (4) via a proximal-splitting approach, so let us begin by defining our most important component.

Definition 1 (Proximity operator)**.** Let $g : \mathbb{R}^n \to \mathbb{R}$ be an lsc, convex function. The *proximity operator* for g, indexed by $\eta > 0$, is the nonlinear map [see e.g., 28; Def. 1.22]:

$$\mathscr{P}^g_\eta \;:\quad y \mapsto \operatorname*{argmin}_{x\in\mathbb{R}^n} \big(g(x) + \tfrac{1}{2\eta}\|x-y\|^2\big). \tag{5}$$

The operator (5) was introduced by Moreau [24] (1962) as a generalization of orthogonal projections. It is also key to Rockafellar's classic *proximal point algorithm* [27], and it arises in a host of *proximal-splitting* methods [2, 3, 8, 11], most notably in *forward-backward splitting* (FBS) [8].

FBS is particularly attractive because of its simplicity and algorithmic structure. It minimizes convex composite objective functions by alternating between "forward" (gradient) steps and "backward" (proximal) steps. Formally, suppose f in (4) is convex; for such f, FBS performs the iteration

$$x^{k+1} = \mathscr{P}^g_{\eta_k}(x^k - \eta_k \nabla f(x^k)), \quad k = 0, 1, \ldots, \tag{6}$$

where $\{\eta_k\}$ is a suitable sequence of stepsizes. The usual convergence analysis of FBS is intimately tied to convexity of f. Therefore, to tackle nonconvex f we must take a different approach. As

previously mentioned, such approaches were considered by Fukushima and Mine [14] and Nesterov [26], but both proved convergence by enforcing monotonic descent.

This insistence on descent severely impedes scalability. Thus, the key challenge is: *how to retain the algorithmic simplicity of FBS and allow nonconvex losses, without sacrificing scalability?*

We address this challenge by introducing the following *inexact* proximal-splitting iteration:

$$x^{k+1} = \mathscr{P}^g_{\eta_k}(x^k - \eta_k \nabla f(x^k) + \eta_k e(x^k)), \quad k = 0, 1, \ldots, \tag{7}$$

where $e(x^k)$ models the *computational errors* in computing the gradient $\nabla f(x^k)$. We also assume that for $\eta > 0$ smaller than some stepsize $\bar{\eta}$, the computational error is uniformly *bounded*, that is,

$$\eta \|e(x)\| \leq \bar{\epsilon}, \quad \text{for some fixed error level } \bar{\epsilon} \geq 0, \quad \text{and } \forall x \in \mathcal{X}. \tag{8}$$

Condition (8) is weaker than the typical vanishing error requirements

$$\sum\nolimits_k \eta \|e(x^k)\| < \infty, \qquad \lim_{k\to\infty} \eta \|e(x^k)\| = 0,$$

which are stipulated by most analyses of methods with gradient errors [4, 5]. Obviously, since errors are nonvanishing, exact stationarity cannot be guaranteed. We will, however, show that the iterates produced by (7) do progress towards reasonable *inexact stationary points*. We note in passing that even if we assume the simpler case of vanishing errors, NIPS is still the first nonconvex proximal-splitting framework that does not insist on monotonicity, which complicates convergence analysis but ultimately proves crucial to scalability.

Algorithm 1 Inexact Nonconvex Proximal Splitting (NIPS)

Input: Operator $\mathscr{P}^g_\eta$, and a sequence $\{\eta_k\}$ satisfying

$$c \leq \liminf_k \eta_k, \quad \limsup_k \eta_k \leq \min\{1, 2/L - c\}, \quad 0 < c < 1/L. \tag{9}$$

Output: Approximate solution to (7)

$k \leftarrow 0$; Select arbitrary $x^0 \in \mathcal{X}$
while $\neg$ converged **do**
  Compute approximate gradient $\widetilde{\nabla} f(x^k) := \nabla f(x^k) - e(x^k)$
  Update: $x^{k+1} = \mathscr{P}^g_{\eta_k}(x^k - \eta_k \widetilde{\nabla} f(x^k))$
  $k \leftarrow k + 1$
end while

2.1 Convergence analysis

We begin by characterizing inexact stationarity. A point x^* is a stationary point for (4) if and only if it satisfies the *inclusion*

$$0 \in \partial_C \Phi(x^*) := \nabla f(x^*) + \partial g(x^*), \tag{10}$$

where $\partial_C \phi$ denotes the Clarke subdifferential [7]. A brief exercise shows that this inclusion may be equivalently recast as the fixed-point equation (which augurs the idea of proximal-splitting)

$$x^* = \mathscr{P}^g_\eta(x^* - \eta \nabla f(x^*)), \quad \text{for } \eta > 0. \tag{11}$$

This equation helps us define a measure of inexact stationarity: the *proximal residual*

$$\rho(x) := x - \mathscr{P}^g_1(x - \nabla f(x)). \tag{12}$$

Note that for an exact stationary point x^* the residual norm $\|\rho(x^*)\| = 0$. Thus, we call a point x to be ϵ-*stationary* if for a prescribed error level $\epsilon(x)$, the corresponding residual norm satisfies

$$\|\rho(x)\| \leq \epsilon(x). \tag{13}$$

Assuming the error-level $\epsilon(x)$ (say if $\bar{\epsilon} = \limsup_k \epsilon(x^k)$) satisfies the bound (8), we prove below that the iterates $\{x^k\}$ generated by (7) satisfy an approximate stationarity condition of the form (13), by allowing the stepsize η to become correspondingly small (but strictly bounded away from zero).

We start by recalling two basic facts, stated without proof as they are standard knowledge.

Lemma 2 (Lipschitz-descent [see e.g., 25; Lemma 2.1.3]). *Let $f \in C^1_L(\mathcal{X})$. Then,*

$$|f(x) - f(y) - \langle \nabla f(y),\, x - y\rangle| \leq \tfrac{L}{2}\|x-y\|^2, \quad \forall\, x, y \in \mathcal{X}. \tag{14}$$

Lemma 3 (Nonexpansivity [see e.g., 9; Lemma 2.4]). *The operator $\mathscr{P}^g_\eta$ is* nonexpansive*, that is,*

$$\|\mathscr{P}^g_\eta(x) - \mathscr{P}^g_\eta(y)\| \leq \|x - y\|, \quad \forall\, x, y \in \mathbb{R}^n. \tag{15}$$

Next we prove a crucial monotonicity property that actually subsumes similar results for projection operators derived by Gafni and Bertsekas [15; Lem. 1], and may therefore be of independent interest.

Lemma 4 (Prox-Monotonicity). *Let $y, z \in \mathbb{R}^n$, and $\eta > 0$. Define the functions*

$$p_g(\eta) \quad := \quad \tfrac{1}{\eta}\|\mathscr{P}^g_\eta(y - \eta z) - y\|, \quad \textit{and} \quad q_g(\eta) \quad := \quad \|\mathscr{P}^g_\eta(y - \eta z) - y\|. \tag{16}$$

Then, $p_g(\eta)$ is a decreasing function of η, and $q_g(\eta)$ an increasing function of η.

Proof. Our proof exploits properties of Moreau-envelopes [28; pp. 19,52], and we present it in the language of proximity operators. Consider the "deflected" proximal objective

$$m_g(x, \eta; y, z) := \langle z,\, x - y\rangle + \tfrac{1}{2\eta}\|x - y\|^2 + g(x), \quad \text{for some } y, z \in \mathcal{X}. \tag{17}$$

Associate to objective m_g the *deflected Moreau-envelope*

$$E_g(\eta) := \inf_{x \in \mathcal{X}}\; m_g(x, \eta; y, z), \tag{18}$$

whose infimum is attained at the unique point $\mathscr{P}^g_\eta(y - \eta z)$. Thus, $E_g(\eta)$ is differentiable, and its derivative is given by $E_g'(\eta) = -\frac{1}{2\eta^2}\|\mathscr{P}^g_\eta(y - \eta z) - y\|^2 = -\frac{1}{2}p(\eta)^2$. Since E_g is convex in η, E_g' is increasing ([28; Thm. 2.26]), or equivalently $p(\eta)$ is decreasing. Similarly, define $\hat{e}_g(\gamma) := E_g(1/\gamma)$; this function is concave in γ as it is a pointwise infimum (indexed by x) of functions linear in γ [see e.g., §3.2.3 in 6]. Thus, its derivative $\hat{e}_g'(\gamma) = \frac{1}{2}\|\mathscr{P}^g_{1/\gamma}(x - \gamma^{-1}y) - x\|^2 = q_g(1/\gamma)$, is a decreasing function of γ. Set $\eta = 1/\gamma$ to conclude the argument about $q_g(\eta)$. □

We now proceed to bound the difference between objective function values from iteration k to $k+1$, by developing a bound of the form

$$\Phi(x^k) - \Phi(x^{k+1}) \geq h(x^k). \tag{19}$$

Obviously, since we do *not* enforce strict descent, $h(x^k)$ may be negative too. However, we show that for sufficiently large k the algorithm makes enough progress to ensure convergence.

Lemma 5. *Let x^{k+1}, x^k, η_k, and $\mathcal{X}$ be as in (7), and that $\eta_k\|e(x^k)\| \leq \epsilon(x^k)$ holds. Then,*

$$\Phi(x^k) - \Phi(x^{k+1}) \quad \geq \quad \tfrac{2 - L\eta_k}{2\eta_k}\|x^{k+1} - x^k\|^2 - \tfrac{1}{\eta_k}\epsilon(x^k)\|x^{k+1} - x^k\|. \tag{20}$$

Proof. For the deflected Moreau envelope (17), consider the directional derivative $\mathrm{d}m_g$ with respect to x in the direction w; at $x = x^{k+1}$, this derivative satisfies the optimality condition

$$\mathrm{d}m_g(x^{k+1}, \eta; y, z)(w) = \langle z + \eta^{-1}(x^{k+1} - y) + s^{k+1},\, w\rangle \geq 0, \quad s^{k+1} \in \partial g(x^{k+1}). \tag{21}$$

Set $z = \nabla f(x^k) - e(x^k)$, $y = x^k$, and $w = x^k - x^{k+1}$ in (21), and rearrange to obtain

$$\langle \nabla f(x^k) - e(x^k),\, x^{k+1} - x^k\rangle \leq \langle \eta^{-1}(x^{k+1} - x^k) + s^{k+1},\, x^k - x^{k+1}\rangle. \tag{22}$$

From Lemma 2 it follows that

$$\Phi(x^{k+1}) \leq f(x^k) + \langle \nabla f(x^k),\, x^{k+1} - x^k\rangle + \tfrac{L}{2}\|x^{k+1} - x^k\|^2 + g(x^{k+1}), \tag{23}$$

whereby upon adding and subtracting $e(x^k)$, and then using (22) we further obtain

$$\begin{aligned}
&f(x^k) + \langle \nabla f(x^k) - e(x^k),\, x^{k+1} - x^k\rangle + \tfrac{L}{2}\|x^{k+1} - x^k\|^2 + g(x^{k+1}) + \langle e(x^k),\, x^{k+1} - x^k\rangle \\
&\leq f(x^k) + g(x^{k+1}) + \langle s^{k+1},\, x^k - x^{k+1}\rangle + \big(\tfrac{L}{2} - \tfrac{1}{\eta_k}\big)\|x^{k+1} - x^k\|^2 + \langle e(x^k),\, x^{k+1} - x^k\rangle \\
&\leq f(x^k) + g(x^k) - \tfrac{2 - L\eta_k}{2\eta_k}\|x^{k+1} - x^k\|^2 + \langle e(x^k),\, x^{k+1} - x^k\rangle \\
&\leq \Phi(x^k) - \tfrac{2 - L\eta_k}{2\eta_k}\|x^{k+1} - x^k\|^2 + \|e(x^k)\|\|x^{k+1} - x^k\| \\
&\leq \Phi(x^k) - \tfrac{2 - L\eta_k}{2\eta_k}\|x^{k+1} - x^k\|^2 + \tfrac{1}{\eta_k}\epsilon(x^k)\|x^{k+1} - x^k\|.
\end{aligned}$$

The second inequality above follows from convexity of g, the third one from Cauchy-Schwarz, and the last one by assumption on $\epsilon(x^k)$. Now flip signs and apply (23) to conclude the bound (20). □

Next we further bound (20) by deriving two-sided bounds on $\|x^{k+1} - x^k\|$.

Lemma 6. *Let x^{k+1}, x^k, and $\epsilon(x^k)$ be as before; also let c and η_k satisfy* (9)*. Then,*

$$c\|\rho(x^k)\| - \epsilon(x^k) \leq \|x^{k+1} - x^k\| \leq \|\rho(x^k)\| + \epsilon(x^k). \tag{24}$$

Proof. First observe that from Lemma 4 that for $\eta_k > 0$ it holds that

$$\text{if } \; 1 \leq \eta_k \; \text{ then } \; q(1) \leq q_g(\eta_k), \quad \text{ and if } \; \eta_k \leq 1 \; \text{ then } \; p_g(1) \leq p_g(\eta_k) = \tfrac{1}{\eta_k} q_g(\eta_k). \tag{25}$$

Using (25), the triangle inequality, and Lemma 3, we have

$$\begin{aligned} \min\{1, \eta_k\}\, q_g(1) = \min\{1, \eta_k\}\, \|\rho(x^k)\| \quad &\leq \quad \|\mathscr{P}^g_{\eta_k}(x^k - \eta_k \nabla f(x^k)) - x^k\| \\ &\leq \quad \|x^{k+1} - x^k\| + \|x^{k+1} - \mathscr{P}^g_{\eta_k}(x^k - \eta_k \nabla f(x^k))\| \\ &\leq \quad \|x^{k+1} - x^k\| + \|\eta_k e(x^k)\| \quad \leq \quad \|x^{k+1} - x^k\| + \epsilon(x^k). \end{aligned}$$

From (9) it follows that for sufficiently large k we have $\|x^{k+1} - x^k\| \geq c\|\rho(x^k)\| - \epsilon(x^k)$. For the upper bound note that

$$\begin{aligned} \|x^{k+1} - x^k\| &\leq \|x^k - \mathscr{P}^g_{\eta_k}(x^k - \eta_k \nabla f(x^k))\| + \|\mathscr{P}^g_{\eta_k}(x^k - \eta_k \nabla f(x^k)) - x^{k+1}\| \\ &\leq \max\{1, \eta_k\}\, \|\rho(x^k)\| + \|\eta_k e(x^k)\| \quad \leq \quad \|\rho(x^k)\| + \epsilon(x^k). \end{aligned}$$

□

Lemma 5 and Lemma 6 help prove the following crucial corollary.

Corollary 7. *Let x^k, x^{k+1}, η_k, and c be as above and k sufficiently large so that c and η_k satisfy* (9)*. Then, $\Phi(x^k) - \Phi(x^{k+1}) \geq h(x^k)$ holds with $h(x^k)$ given by*

$$h(x^k) := \tfrac{L^2 c^3}{2(2-2Lc)}\|\rho(x^k)\|^2 - \big(\tfrac{L^2 c^2}{2-cL} + \tfrac{1}{c}\big)\|\rho(x^k)\|\epsilon(x^k) - \big(\tfrac{1}{c} - \tfrac{L^2 c}{2(2-cL)}\big)\epsilon(x^k)^2. \tag{26}$$

Proof. Plug in the bounds (24) into (20), invoke (9), and simplify—see [32] for details. □

We now have all the ingredients to state the main convergence theorem.

Theorem 8 (Convergence)**.** *Let $f \in C^1_L(\mathcal{X})$ such that $\inf_{\mathcal{X}} f > -\infty$ and let g be lsc, convex on $\mathcal{X}$. Let $\{x^k\} \subset \mathcal{X}$ be a sequence generated by (7), and let condition (8) on each $\|e(x^k)\|$ hold. There exists a limit point x^* of the sequence $\{x^k\}$, and a constant $K > 0$, such that $\|\rho(x^*)\| \leq K\epsilon(x^*)$. If $\{\Phi(x^k)\}$ converges, then for every limit point x^* of $\{x^k\}$ it holds that $\|\rho(x^*)\| \leq K\epsilon(x^*)$.*

Proof. Lemma 5, 6, and Corollary 7 have done all the hard work. Indeed, they allow us to reduce our convergence proof to the case where the analysis of the differentiable case becomes applicable, and an appeal to the analysis of [29; Thm. 2.1] grants us our claim. □

Theorem 8 says that we can obtain an approximate stationary point for which the norm of the residual is bounded by a linear function of the error level. The statement of the theorem is written in a conditional form, because nonvanishing errors $e(x)$ prevent us from making a stronger statement. In particular, once the iterates enter a region where the residual norm falls below the error threshold, the behavior of $\{x^k\}$ may be arbitrary. This, however, is a small price to pay for having the added flexibility of nonvanishing errors. Under the stronger assumption of vanishing errors (and diminishing stepsizes), we can also obtain guarantees to exact stationary points.

3 Scaling up NIPS: incremental variant

We now apply NIPS to the large-scale setting, where we have composite objectives of the form

$$\Phi(x) := \sum\nolimits_{t=1}^{T} f_t(x) + g(x), \tag{27}$$

where each $f_t : \mathbb{R}^n \to \mathbb{R}$ is a $C^1_{L_t}(\mathcal{X})$ function. For simplicity, we use $L = \max_t L_t$ in the sequel. It is well-known that for such decomposable objectives it can be advantageous to replace the full gradient $\sum_t \nabla f_t(x)$ by an *incremental gradient* $\nabla f_{\sigma(t)}(x)$, where $\sigma(t)$ is some suitable index.

Nonconvex incremental methods for differentiable problems have been extensively analyzed, e.g., backpropagation algorithms [5, 29], which correspond to $g(x) \equiv 0$. However, when $g(x) \neq 0$, the only incremental methods that we are aware of are stochastic generalized gradient methods of [12] or the generalized gradient methods of [31]. As previously mentioned, both of these fail to exploit the composite structure of the objective function, a disadvantage even in the convex case [11].

In stark contrast, we *do* exploit the composite structure of (27). Formally, we propose the following incremental nonconvex proximal-splitting iteration:

$$\begin{aligned} x^{k+1} &= \mathcal{M}\big(x^k - \eta_k \textstyle\sum_{t=1}^{T} \nabla f_t(x^{k,t})\big), \quad k = 0, 1, \ldots, \\ x^{k,1} = x^k, \quad x^{k,t+1} &= \mathcal{O}(x^{k,t} - \eta_k \nabla f_t(x^{k,t})), \quad t = 1, \ldots, T-1, \end{aligned} \tag{28}$$

where $\mathcal{O}$ and $\mathcal{M}$ are appropriate operators, different choices of which lead to different algorithms. For example, when $\mathcal{X} = \mathbb{R}^n$, $g(x) \equiv 0$, $\mathcal{M} = \mathcal{O} = \mathrm{Id}$, and $\eta_k \to 0$, then (28) reduces to the classic incremental gradient method (IGM) [4], and to the IGM of [30], if $\lim \eta_k = \bar{\eta} > 0$. If $\mathcal{X}$ a closed convex set, $g(x) \equiv 0$, $\mathcal{M}$ is orthogonal projection onto $\mathcal{X}$, $\mathcal{O} = \mathrm{Id}$, and $\eta_k \to 0$, then iteration (28) reduces to (projected) IGM [4, 5].

We may consider four variants of (28) in Table 1; to our knowledge, all of these are new. Which of the four variants one prefers depends on the complexity of the constraint set $\mathcal{X}$ and cost to apply $\mathscr{P}_\eta^g$. The analysis of all four variants is similar, so we present details only for the most general case.

$\mathcal{X}$	g	$\mathcal{M}$	$\mathcal{O}$	Penalty and constraints	Proximity operator calls
$\mathbb{R}^n$	$\not\equiv 0$	$\mathscr{P}_\eta^g$	Id	penalized, unconstrained	once every *major* (k) iteration
$\mathbb{R}^n$	$\not\equiv 0$	$\mathscr{P}_\eta^g$	$\mathscr{P}_\eta^g$	penalized, unconstrained	once every *minor* (k,t) iteration
Convex	$h(x) + \delta(\mathcal{X}\|x)$	$\mathscr{P}_\eta^g$	Id	penalized, constrained	once every major (k) iteration
Convex	$h(x) + \delta(\mathcal{X}\|x)$	$\mathscr{P}_\eta^g$	$\mathscr{P}_\eta^g$	penalized, constrained	once every minor (k,t) iteration

Table 1: Different variants of incremental NIPS (28).

3.1 Convergence analysis

Specifically, we analyze convergence for the case $\mathcal{M} = \mathcal{O} = \mathscr{P}_\eta^g$ by generalizing the differentiable case treated by [30]. We begin by rewriting (28) in a form that matches the main iteration (7):

$$\begin{aligned} x^{k+1} &= \mathscr{P}_\eta^g\big(x^k - \eta_k \textstyle\sum_{t=1}^{T} \nabla f_t(x^{k,t})\big) \\ &= \mathscr{P}_\eta^g\big(x^k - \eta_k \textstyle\sum_{t=1}^{T} \nabla f_t(x^k) + \eta_k\big[\sum_{t=1}^{T} f_t(x^k) - f_t(x^{k,t})\big]\big) \\ &= \mathscr{P}_\eta^g\big(x^k - \eta_k \textstyle\sum_{t} \nabla f_t(x^k) + \eta_k e(x^k)\big). \end{aligned} \tag{29}$$

To show that iteration (29) is well-behaved and actually fits the main NIPS iteration (7), we must ensure that the norm of the error term is bounded. We show this via a sequence of lemmas.

Lemma 9 (Bounded-increment)**.** *Let $x^{k,t+1}$ be computed by* (28)*, and let $s^t \in \partial g(x^{k,t})$. Then,*

$$\|x^{k,t+1} - x^{k,t}\| \leq 2\eta_k \|\nabla f_t(x^{k,t}) + s^t\|. \tag{30}$$

Proof. From the definition of a proximity operator (5), we have the inequality

$$\begin{aligned} & \tfrac{1}{2}\|x^{k,t+1} - x^{k,t} + \eta_k \nabla f_t(x^{k,t})\|^2 + \eta_k g(x^{k,t+1}) \leq \tfrac{1}{2}\|\eta_k \nabla f_t(x^{k,t})\|^2 + \eta_k g(x^{k,t}), \\ \implies \quad & \tfrac{1}{2}\|x^{k,t+1} - x^{k,t}\|^2 \leq \eta_k \langle \nabla f_t(x^{k,t}),\, x^{k,t} - x^{k,t+1}\rangle + \eta_k (g(x^{k,t}) - g(x^{k,t+1})). \end{aligned}$$

Since $s_t \in \partial g(x^{k,t})$, we have $g(x^{k,t+1}) \geq g(x^{k,t}) + \langle s_t,\, x^{k,t+1} - x^{k,t}\rangle$. Therefore,

$$\begin{aligned} \tfrac{1}{2}\|x^{k,t+1} - x^{k,t}\|^2 &\leq \eta_k \langle s^t,\, x^{k,t} - x^{k,t+1}\rangle + \langle \nabla f_t(x^{k,t}),\, x^{k,t} - x^{k,t+1}\rangle \\ &\leq \eta_k \|s_t + \nabla f_t(x^{k,t})\| \|x^{k,t} - x^{k,t+1}\| \\ \implies \quad \|x^{k,t+1} - x^{k,t}\| &\leq 2\eta_k \|\nabla f_t(x^{k,t}) + s^t\|. \end{aligned}$$

□

Lemma 9 proves helpful in bounding the overall error.

Lemma 10 (Bounded error). *If for all $x^k \in \mathcal{X}$, $\|\nabla f_t(x^k)\| \leq M$ and $\|\partial g(x^k)\| \leq G$, then there exists a constant $K_1 > 0$ such that $\|e(x^k)\| \leq K_1$.*

Proof. To bound the error of using $x^{k,t}$ instead of x^k first define the term

$$\epsilon_t := \|\nabla f_t(x^{k,t}) - \nabla f_t(x^k)\|, \quad t = 1, \ldots, T. \tag{31}$$

Then, an inductive argument (see [32] for details) shows that for $2 \leq t \leq T$

$$\epsilon_t \leq 2\eta_k L \sum\nolimits_{j=1}^{t-1} (1 + 2\eta_k L)^{t-1-j} \|\nabla f_j(x^k) + s^j\|. \tag{32}$$

Since $\|e(x^k)\| = \sum_{t=1}^T \epsilon_t$, and $\epsilon_1 = 0$, (32) then leads to the bound

$$\begin{aligned}
\sum\nolimits_{t=2}^T \epsilon_t &\leq 2\eta_k L \sum\nolimits_{t=2}^T \sum\nolimits_{j=1}^{t-1} (1 + 2\eta_k L)^{t-1-j} \beta_j = 2\eta_k L \sum\nolimits_{t=1}^{T-1} \beta_t \Big(\sum\nolimits_{j=0}^{T-t-1} (1 + 2\eta_k L)^j \Big) \\
&\leq \sum\nolimits_{t=1}^{T-1} (1 + 2\eta_k L)^{T-t} \beta_t \quad \leq \quad (1 + 2\eta_k L)^{T-1} \sum\nolimits_{t=1}^{T-1} \|\nabla f_t(x) + s^t\| \\
&\leq C_1 (T-1)(M+G) =: K_1.
\end{aligned}$$

□

Thus, the error norm $\|e(x^k)\|$ is bounded from above by a constant, whereby it satisfies the requirement (8), making the incremental NIPS method (28) a special case of the general NIPS framework. This allows us to invoke the convergence result Theorem 8 for without further ado.

4 Illustrative application

The main contribution of our paper is the new NIPS framework, and a specific application is not one of the prime aims of this paper. We do, however, provide an illustrative application of NIPS to a challenging nonconvex problem: *sparsity regularized low-rank matrix factorization*

$$\min_{X, A \geq 0} \quad \tfrac{1}{2}\|Y - XA\|_{\mathrm{F}}^2 + \psi_0(X) + \sum\nolimits_{t=1}^T \psi_t(a_t), \tag{33}$$

where $Y \in \mathbb{R}^{m \times T}$, $X \in \mathbb{R}^{m \times K}$ and $A \in \mathbb{R}^{K \times T}$, with $a_1, \ldots, a_T$ as its columns. Problem (33) generalizes the well-known nonnegative matrix factorization (NMF) problem of [20] by permitting arbitrary Y (not necessarily nonnegative), and adding regularizers on X and A. A related class of problems was studied in [23], but with a crucial difference: the formulation in [23] *does not* allow nonsmooth regularizers on X. The class of problems studied in [23] is in fact a subset of those covered by NIPS. On a more theoretical note, [23] considered stochastic-gradient like methods whose analysis requires computational errors and stepsizes to vanish, whereas our method is deterministic and allows nonvanishing stepsizes and errors.

Following [23] we also rewrite (33) in a form more amenable to NIPS. We eliminate A and consider

$$\min\nolimits_X \quad \phi(X) := \sum\nolimits_{t=1}^T f_t(X) + g(X), \quad \text{where} \quad g(X) := \psi_0(X) + \delta(X|\geq 0), \tag{34}$$

and where each $f_t(X)$ for $1 \leq t \leq T$ is defined as

$$f_t(X) := \min\nolimits_a \quad \tfrac{1}{2}\|y_t - Xa\|^2 + g_t(a), \tag{35}$$

where $g_t(a) := \psi_t(a) + \delta(a|\geq 0)$. For simplicity, assume that (35) attains its unique[1] minimum, say a^*, then $f_t(X)$ is differentiable and we have $\nabla_X f_t(X) = (Xa^* - y_t)(a^*)^T$. Thus, we can instantiate (28), and all we need is a subroutine for solving (35).[2]

We present empirical results on the following two variants of (34): (i) pure unpenalized NMF ($\psi_t \equiv 0$ for $0 \leq t \leq T$) as a baseline; and (ii) sparsity penalized NMF where $\psi_0(X) \equiv \lambda\|X\|_1$ and $\psi_t(a_t) \equiv \gamma\|a_t\|_1$. Note that without the nonnegativity constraints, (34) is similar to sparse-PCA.

We use the following datasets and parameters: ⟦i⟧ `RAND`: 4000×4000 dense random (uniform $[0, 1]$); rank-32 factorization; $(\lambda, \gamma) = (10^{-5}, 10)$; ⟦ii⟧ `CBCL`: CBCL database [33]; 361×2429; rank-49 factorization; ⟦iii⟧ `YALE`: Yale B Database [21]; 32256×2414 matrix; rank-32 factorization; ⟦iv⟧ `WEB`: Web graph from google; sparse 714545×739454 (empty rows and columns removed) matrix; ID: 2301 in the sparse matrix collection [10]); rank-4 factorization; $(\lambda = \gamma = 10^{-6})$.

[1] Otherwise, at the expense of more notation, we can add a small strictly convex perturbation to ensure uniqueness; this perturbation can be then absorbed into the overall computational error.

[2] In practice, it is better to use *mini-batches*, and we used the same sized mini-batches for all the algorithms.

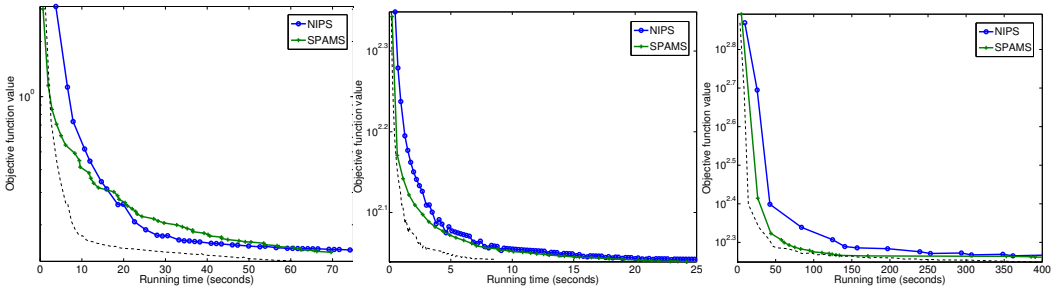

Figure 1: Running times of NIPS (Matlab) versus SPAMS (C++) for NMF on `RAND`, `CBCL`, and `YALE` datasets. Initial objective values and tiny runtimes have been suppressed for clarity of presentation.

On the NMF baseline (Fig. 1), we compare NIPS against the well optimized state-of-the-art C++ toolbox SPAMS (version 2.3) [23]. We compare against SPAMS only on dense matrices, as its NMF code seems to be optimized for this case. Obviously, the comparison is not fair: unlike SPAMS, NIPS and its subroutines are all implemented in MATLAB, and they run equally easily on large sparse matrices. Nevertheless, NIPS proves to be quite competitive: Fig. 1 shows that our MATLAB implementation runs only slightly slower than SPAMS. We expect a well-tuned C++ implementation of NIPS to run at least 4–10 times faster than the MATLAB version—the dashed line in the plots visualizes what such a mere 3X-speedup to NIPS might mean.

Figure 2 shows numerical results comparing the stochastic generalized gradient (SGGD) algorithm of [12] against NIPS, when started at the same point. As in well-known, SGGD requires careful stepsize tuning; so we searched over a range of stepsizes, and have reported the best results. NIPS too requires some stepsize tuning, but substantially lesser than SGGD. As predicted, the solutions returned by NIPS have objective function values lower than SGGD, and have greater sparsity.

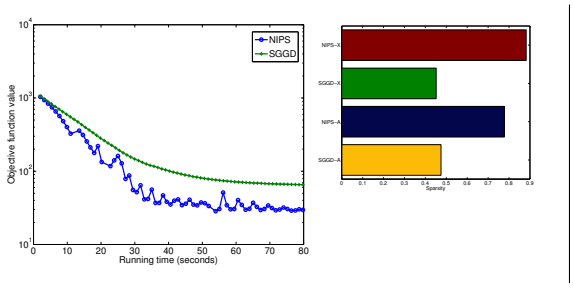

Figure 2: Sparse NMF: NIPS versus SGGD. The bar plots show the sparsity (higher is better) of the factors X and A. Left plots for `RAND` dataset; right plots for `WEB`. As expected, SGGD yields slightly worse objective function values and less sparse solutions than NIPS.

5 Discussion

We presented a new framework called NIPS, which solves a broad class of nonconvex composite objective problems. NIPS permits nonvanishing computational errors, which can be practically useful. We specialized NIPS to also obtain a scalable incremental version. Our numerical experiments on large scale matrix factorization indicate that NIPS is competitive with state-of-the-art methods.

We conclude by mentioning that NIPS includes numerous other algorithms as special cases. For example, batch and incremental convex FBS, convex and nonconvex gradient projection, the proximal-point algorithm, among others. Theoretically, however, the most exciting open problem resulting from this paper is: *extend* NIPS *in a scalable way when even the nonsmooth part is nonconvex*. This case will require very different convergence analysis, and is left to the future.

References

[1] H. Attouch, J. Bolte, and B. F. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Math.*

Programming Series A, Aug. 2011. Online First.

[2] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Convex optimization with sparsity-inducing norms. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.

[3] A. Beck and M. Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM J. Imgaging Sciences*, 2(1):183–202, 2009.

[4] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.

[5] D. P. Bertsekas. Incremental Gradient, Subgradient, and Proximal Methods for Convex Optimization: A Survey. Technical Report LIDS-P-2848, MIT, August 2010.

[6] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[7] F. H. Clarke. *Optimization and nonsmooth analysis*. John Wiley & Sons, Inc., 1983.

[8] P. L. Combettes and J.-C. Pesquet. Proximal Splitting Methods in Signal Processing. *arXiv:0912.3522v4*, May 2010.

[9] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200, 2005.

[10] T. A. Davis and Y. Hu. The University of Florida Sparse Matrix Collection. *ACM Transactions on Mathematical Software*, 2011. To appear.

[11] J. Duchi and Y. Singer. Online and Batch Learning using Forward-Backward Splitting. *J. Mach. Learning Res. (JMLR)*, Sep. 2009.

[12] Y. M. Ermoliev and V. I. Norkin. Stochastic generalized gradient method for nonconvex nonsmooth stochastic optimization. *Cybernetics and Systems Analysis*, 34:196–215, 1998.

[13] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient Projection for Sparse Reconstruction: Application to Compressed Sensing and Other Inverse Problems. *IEEE J. Selected Topics in Sig. Proc.*, 1 (4):586–597, 2007.

[14] M. Fukushima and H. Mine. A generalized proximal point algorithm for certain non-convex minimization problems. *Int. J. Systems Science*, 12(8):989–1000, 1981.

[15] E. M. Gafni and D. P. Bertsekas. Two-metric projection methods for constrained optimization. *SIAM Journal on Control and Optimization*, 22(6):936–964, 1984.

[16] A. A. Gaivoronski. Convergence properties of backpropagation for neural nets via theory of stochastic gradient methods. Part 1. *Optimization methods and Software*, 4(2):117–134, 1994.

[17] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, 1st edition, 1994.

[18] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Computation*, 15:349–396, 2003.

[19] D. Kundur and D. Hatzinakos. Blind image deconvolution. *IEEE Signal Processing Magazine*, 13(3), May 1996.

[20] D. D. Lee and H. S. Seung. Algorithms for Nonnegative Matrix Factorization. In *NIPS*, 2000.

[21] K.C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 27(5):684–698, 2005.

[22] J. Liu and J. Ye. Efficient Euclidean projections in linear time. In *ICML*, Jun. 2009.

[23] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online Learning for Matrix Factorization and Sparse Coding. *JMLR*, 11:10–60, 2010.

[24] J. J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *C. R. Acad. Sci. Paris Sr. A Math.*, 255:2897–2899, 1962.

[25] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2004.

[26] Y. Nesterov. Gradient methods for minimizing composite objective function. Technical Report 2007/76, Universit catholique de Louvain, September 2007.

[27] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM J. Control and Optimization*, 14, 1976.

[28] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*. Springer, 1998.

[29] M. V. Solodov. Convergence analysis of perturbed feasible descent methods. *J. Optimization Theory and Applications*, 93(2):337–353, 1997.

[30] M. V. Solodov. Incremental gradient algorithms with stepsizes bounded away from zero. *Computational Optimization and Applications*, 11:23–35, 1998.

[31] M. V. Solodov and S. K. Zavriev. Error stability properties of generalized gradient-type algorithms. *J. Optimization Theory and Applications*, 98(3):663–680, 1998.

[32] S. Sra. Nonconvex proximal-splitting: Batch and incremental algorithms. Sep. 2012. arXiv:1109.0258v2.

[33] K.-K. Sung. *Learning and Example Selection for Object and Pattern Recognition*. PhD thesis, MIT, 1996.

[34] L. Xiao. Dual averaging method for regularized stochastic learning and online optimization. In *NIPS*, 2009.