Nonconvex proximal splitting with computational errors*

Suvrit Sra suvrit@tuebingen.mpg.de *Max Planck Institute, Tübingen, Germany*

1 Introduction

We study in this chapter large-scale nonconvex optimization problems with *composite objective functions* that are composed of a differentiable possibly nonconvex cost and a nonsmooth but convex regularizer. More precisely, we consider optimization problems of the form

$$\text{minimize} \quad \Phi(x) := f(x) + r(x), \quad \text{s.t.} \quad x \in \mathcal{X}, \tag{1}$$

where $\mathcal{X} \subset \mathbb{R}^n$ is a compact convex set, $f : \mathbb{R}^n \to \mathbb{R}$ is a differentiable cost function and $r : \mathbb{R}^n \to \mathbb{R}$ is a closed convex function. Further, we assume that the gradient ∇f is *Lipschitz continuous* on $\mathcal{X}$ (denoted $f \in C^1_L(\mathcal{X})$), i.e.,

$$\exists L > 0 \quad \text{s.t.} \quad \|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \qquad \text{for all} \quad x, y \in \mathcal{X}. \tag{2}$$

Throughout this chapter, $\|\cdot\|$ denotes the standard Euclidean norm.

Problem (1) generalizes the more thoroughly studied class of *composite convex optimization problems* [30], a class that has witnessed huge interest in machine learning, signal processing, statistics, and other related areas. We refer the interested reader to [2, 3, 21, 37] for several convex examples and recent references. A thread common to existing algorithms for solving composite problems is the remarkably fruitful idea of *proximal-splitting* [9]. Here, nonsmoothness is handled via proximity operators [29], which allows one to treat the nonsmooth objective $f + r$ essentially as a smooth one.

But leveraging proximal-splitting methods is considerably harder for nonconvex problems, especially without compromising scalability. Numerous important problems have appealing nonconvex formulations: matrix factorization [25, 27], blind deconvolution [24], dictionary learning and sparse reconstruction [23, 27], and neural networks [4, 19, 28], to name a few. Regularized optimization within these problems requires handling nonconvex composite objectives, which motivates the material of this chapter.

The focus of this chapter is on a new proximal splitting framework called: **N**onconvex **I**nexact **P**roximal **S**plitting, hereafter NIPS. The NIPS framework is *inexact* because it allows for computational errors, a feature that helps it scale to large-data problems. In contrast to typical incremental methods [5] and to most stochastic gradient methods [16, 18] that assume vanishing errors, NIPS allows the computational errors to be *nonvanishing*.

NIPS inherits this capability from the remarkable framework of Solodov [33]. But NIPS not only builds on [33], it strictly generalizes it: Unlike [33], NIPS allows $r \neq 0$ in (1). To our knowledge, NIPS is the first nonconvex proximal splitting method that has *both* batch and incremental incarnations; this claim remains true, even if we were to exclude the nonvanishing error capability.[1] We mention some more related work below.

Among batch nonconvex splitting methods an early paper is [14]. Another batch method can be found in the pioneering paper on composite minimization by Nesterov [30], who solves (1) via a splitting-like algorithm. Both [14] and [30] rely on monotonic descent (using line-search or otherwise) to ensure convergence. Very recently, [1] introduced a powerful class of "descent-methods" based on Kurdyka-Łojasiewicz theory. In general, the insistence on descent, while theoretically convenient,

*Chapter in: *"Regularization, Optimization, Kernels, and Support Vector Machines."* (Editors: J. A.K. Suykens, M. Signoretto, A. Argyriou. Mar 2014)

[1] Though very recently, in [18] scalable nonconvex stochastic methods were proposed for smooth nonconvex problems; and even more recently those ideas were extended to cover nonsmooth and accelerated methods [17], though still in the vanishing error framework.

makes it hard to extend these methods to incremental, stochastic, or online variants. The general proximal framework of [40] avoids strict monotonic descent at each step by using a non-monotonic line-search.

There are some incremental and stochastic methods that apply to (1), namely the generalized gradient algorithm of [35] and the stochastic generalized gradient methods of [12, 13], (and the very recent work of [17, 18]). All these approaches are analogous to subgradient methods, and thus face similar practical difficulties (except [18]). For example, it is well-recognized that subgradient methods fail to exploit composite objectives [11, 30]. Moreover, they exhibit the effect of the regularizer only in the limit, which conflicts with early termination heuristics frequently used in practice. If, say the nonsmooth part of the objective is $\|x\|_1$, then with subgradient-style methods sparse solutions are obtained only in the limit and intermediate iterates may be dense. Thus, like the convex case it may be of substantial practical advantage to use proximal splitting even for (1).

2 The NIPS Framework

We rewrite (1) as an unconstrained problem by introducing the function

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

where $\delta(x|\mathcal{X})$ is the *indicator function* for set $\mathcal{X}$. Our problem then becomes:

$$\text{minimize} \quad \Phi(x) := f(x) + g(x) \qquad x \in \mathbb{R}^n. \tag{3}$$

Since we solve (3) via a proximal method, we begin with the definition below.

Definition 1 (Proximity operator)**.** *Let $g : \mathbb{R}^n \to \mathbb{R}$ be lower semicontinuous (lsc) and convex. The* proximity operator *for g, indexed by $\eta > 0$, is the nonlinear map [31, Def. 1.22]:*

$$\text{prox}_{g,\eta} \; : \quad y \mapsto \operatorname*{argmin}_{x \in \mathbb{R}^n} \left(g(x) + \frac{1}{2\eta}\|x - y\|^2 \right). \tag{4}$$

Using the operator (4), the classic forward-backward splitting (FBS) [8] iteration (for suitable η_k and convex f) is written as

$$x^{k+1} = \text{prox}_{g,\eta_k}(x^k - \eta_k \nabla f(x^k)), \quad k = 0, 1, \ldots. \tag{5}$$

The NIPS framework described in this chapter is motivated by the simple form of iteration (5). In particular, for this iteration NIPS introduces two powerful generalizations: (i) it permits a nonconvex f; and (ii) it allows computational errors. More precisely, NIPS performs the iteration

$$x^{k+1} = \text{prox}_{g,\eta_k}(x^k - \eta_k \nabla f(x^k) + \eta_k e(x^k)). \tag{6}$$

The error vector $e(x^k)$ in (6) is the interesting part. It denotes potential error made at step k in the computation of the gradient $\nabla f(x^k)$. It is important to observe that the net error is $\eta_k e(x^k)$, so that the error is scaled by the stepsize. This scaling is made to suggest that the limiting value of the stepsize is what ultimately determines the effective error, and thereby governs convergence.

Remark 1. *We warn the reader against a potential pitfall of the notation for error in (6). That iteration does* not *mean that* NIPS *adds an error vector $e(x^k)$ when iterating, but rather that it iterates*

$$x^{k+1} = \text{prox}_{g,\eta_k}(x^k - \eta_k g^k),$$

where g^k is an erroneous computation of the gradient, which is explicitly depicted in (6) as $g^k = \nabla f(x^k) - e(x^k)$.

But notice that we *do not* impose the following condition

$$\lim_{k\to\infty} \|e(x^k)\| \to 0 \tag{7}$$

on the error vectors, which is typically imposed by stochastic-gradient methods [5]. Since we do not require the errors to vanish in the limit, to make NIPS well-defined we must nevertheless somehow control them. Thus, we impose a mild restriction on the errors: we assume that there is a fixed value $\bar{\eta}$, so that for all stepsizes η smaller than $\bar{\eta}$ the gradient errors satisfy

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

Clearly, condition (8) is weaker than the usual requirement (7).

Remark 2. *We can consider errors in the proximity operator too, i.e., the* $\text{prox}_{g,\eta}$ *computation may also be inexact (for convex optimization inexact proximity operators have been studied since a long-time; two recent references are [32, 39]). With inexact proximity operations iteration* (6) *becomes*

$$x^{k+1} = \text{prox}_{g,\eta_k}(x^k - \eta_k \nabla f(x^k) + \eta_k e(x^k)) + \eta_k p(x^k),$$

where $\eta_k p(x^k)$ *is the error in proximity operator. The dependency on* η_k *highlights that the error should eventually shrink in a manner similar to* (8)*. This error can be easily incorporated into our analysis below, though at the expense of heavier notation. To avoid clutter we omit details and leave them as an exercise for the interested reader.*

Since errors in the gradient computation need not disappear, we cannot ensure exact stationary points; but we can nevertheless hope to ensure *inexact stationary points*. Let us make this more precise. A point $x^* \in \mathbb{R}^n$ is stationary for (3), if and only if it satisfies the inclusion

$$0 \in \partial_C \Phi(x^*) = \nabla f(x^*) + \partial g(x^*), \tag{9}$$

where $\partial_C \Phi(x^*)$ is the Clarke subdifferential [7] at x^*. The optimality condition (9) may be recast as the fixed-point equation

$$x^* = \text{prox}_{g,\eta}(x^* - \eta \nabla f(x^*)), \quad \text{for } \eta > 0, \tag{10}$$

which helps characterize approximate stationarity. Define the *prox-residual*

$$\rho(x) := x - \text{prox}_{g,1}(x - \nabla f(x)); \tag{11}$$

then, for stationary x^* the residual norm $\|\rho(x^*)\|$ vanishes. At a point x, let the total perturbation be given by $\epsilon(x) \geq 0$. We define a point $\bar{x}$ to be ϵ*-stationary* if the residual norm satisfies the condition

$$\|\rho(\bar{x})\| \leq \epsilon(\bar{x}). \tag{12}$$

Since we cannot measure convergence to an accuracy better than the amount of prevalent noise, we require $\epsilon(x) \geq \eta\|e(x)\|$. By letting η become small enough, we may hope to come arbitrarily close to a stationary point.

2.1 Convergence analysis

In this section, we outline a simple convergence analysis for the NIPS iteration (6). Our analysis is structured upon the powerful framework of [33]. But our problem class of composite objectives is more general than the differentiable problems considered in [33] since we allow nonsmooth objective functions. Our analysis leads to the first nonconvex proximal splitting algorithm which allows noisy gradients; also we obtain the first nonconvex incremental proximal splitting algorithm regardless of whether the noise vanishes or not.

For our analysis we make the following standing assumption.

Assumption. The stepsizes η_k satisfy the bounds

$$c \leq \liminf_k \eta_k, \quad \limsup_k \eta_k \leq \min\{1, 2/L - c\}, \quad 0 < c < 1/L. \tag{13}$$

We start our analysis by recalling two well-known facts.

Lemma 2.1 (Descent). *Let f be such that ∇f satisfies (2). Then*

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

Proof. Since $f \in C^1_L$, by Taylor's theorem for $z_t = y + t(x - y)$ we have

$$\begin{aligned} |f(x) - f(y) - \langle \nabla f(y),\, x - y\rangle| &= |\textstyle\int_0^1 \langle \nabla f(z_t) - \nabla f(y),\, x - y\rangle dt|, \\ &\leq \textstyle\int_0^1 \|\nabla f(z_t) - \nabla f(y)\| \cdot \|x - y\| dt \leq L \int_0^1 t\|x - y\|_2^2 dt = \tfrac{L}{2}\|x - y\|_2^2. \end{aligned}$$

We used the triangle-inequality, Cauchy-Schwarz, and that $f \in C^1_L$ above. □

Lemma 2.2. *The operator* $\operatorname{prox}_{g,\eta}$ *is* nonexpansive, *that is,*

$$\|\operatorname{prox}_{g,\eta} x - \operatorname{prox}_{g,\eta} y\| \leq \|x - y\|, \quad \forall\, x, y \in \mathbb{R}^n. \tag{15}$$

Proof. For brevity we drop the subscripted η. After renaming variables, from optimality conditions for the problem (4), it follows that $x - \operatorname{prox}_g x \in \eta \partial g(\operatorname{prox}_g x)$. A similar characterization holds for y. Thus, $x - \operatorname{prox}_g x$ and $y - \operatorname{prox}_g y$ are subgradients of g at $\operatorname{prox}_g x$ and $\operatorname{prox}_g y$, respectively. Thus,

$$\begin{aligned} g(\operatorname{prox}_g x) &\geq g(\operatorname{prox}_g y) + \langle y - \operatorname{prox}_g y,\, \operatorname{prox}_g x - \operatorname{prox}_g y\rangle \\ g(\operatorname{prox}_g y) &\geq g(\operatorname{prox}_g x) + \langle x - \operatorname{prox}_g x,\, \operatorname{prox}_g y - \operatorname{prox}_g x\rangle. \end{aligned}$$

Adding the two inequalities we obtain *firm nonexpansivity*

$$\|\operatorname{prox}_g x - \operatorname{prox}_g y\|^2 \leq \langle \operatorname{prox}_g x - \operatorname{prox}_g y,\, x - y\rangle,$$

from which via Cauchy-Schwarz, we easily obtain (15). □

Next, we prove a crucial monotonicity property of proximity operators.

Lemma 2.3. *Define $P_\eta \equiv \operatorname{prox}_{g,\eta}$; let $y, z \in \mathbb{R}^n$, and $\eta > 0$; define the functions*

$$p(\eta) := \quad \eta^{-1}\|P_\eta(y - \eta z) - y\|, \tag{16}$$

$$q(\eta) := \quad \|P_\eta(y - \eta z) - y\|. \tag{17}$$

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

Proof. Our proof relies on well-known results about Moreau-envelopes [8, 31]. Consider thus the "deflected" proximal objective

$$m_g(x, \eta; y, z) := \langle z,\, x - y\rangle + \tfrac{1}{2}\eta^{-1}\|x - y\|^2 + g(x), \tag{18}$$

to which we associate its *Moreau-envelope*

$$\mathscr{E}_g(\eta) := \inf_{x \in \mathcal{X}}\; m_g(x, \eta; y, z). \tag{19}$$

Since m_g is strongly convex in x, and $\mathcal{X}$ is compact, the infimum in (19) is attained at a unique point, which is precisely $P_\eta^g(y-\eta z)$. Thus, $\mathscr{E}_g(\eta)$ is a differentiable function of η, and in particular

$$\frac{\partial \mathscr{E}_g(\eta)}{\partial \eta} = -\tfrac{1}{2}\eta^{-2}\|P_\eta(y-\eta z)-y\|^2 = -\tfrac{1}{2}p(\eta)^2.$$

Observe that m_g is jointly convex in (x,η); it follows that $\mathscr{E}_g$ is convex too. Thus, its derivative $\partial\mathscr{E}_g/\partial\eta$ is increasing, whereby $p(\eta)$ is decreasing. Similarly, $\hat{\mathscr{E}}_g(\gamma) := \mathscr{E}_g(1/\gamma)$ is concave in γ (it is a pointwise infimum of linear functions). Thus, its derivative

$$\frac{\partial \hat{\mathscr{E}}_g(\gamma)}{\partial \gamma} = \tfrac{1}{2}\|P_{1/\gamma}(x-\gamma^{-1}y)-x\|^2 = q(1/\gamma),$$

is a decreasing function of γ. Writing $\eta = 1/\gamma$ completes our claim. □

Remark 3. *The monotonicity results* (16) *and* (17) *subsume the monotonicity results for projection operators derived in [15, Lemma 1].*

We now proceed to analyze how the objective function value changes after one step of the NIPS iteration (6). Specifically, we seek to derive an inequality of the form (20) (where Φ is as in (3)):

$$\Phi(x^k) - \Phi(x^{k+1}) \geq h(x^k). \tag{20}$$

Our strategy is to bound the potential function $h(x)$ in terms of prox-residual $\|\rho(x)\|$ and the error level $\epsilon(x)$. It is important to note that the potential $h(x)$ may be negative, because we do not insist on monotonic descent.

To reduce clutter, let us introduce brief notation: $u \equiv x^{k+1}$, $x \equiv x^k$, and $\eta \equiv \eta_k$; therewith the main NIPS update (6) may be rewritten as

$$u = \operatorname{prox}_{g,\eta}(x - \eta\nabla f(x) + \eta e(x)). \tag{21}$$

We are now ready to state the following "descent" theorem.

Theorem 2.4. *Let u, x, η be as in* (21)*; assume $\epsilon(x) \geq \eta\|e(x)\|$. Then,*

$$\Phi(x) - \Phi(u) \quad \geq \quad \tfrac{2-L\eta}{2\eta}\|u-x\|^2 - \tfrac{1}{\eta}\epsilon(x)\|u-x\|. \tag{22}$$

Proof. Let m_g be as in (18); consider its directional derivative $\mathrm{d}m_g$ with respect to x in direction w; at $x = u$ it satisfies the optimality condition

$$\mathrm{d}m_g(u,\eta;y,z)(w) = \langle z + \eta^{-1}(u-y) + s,\, w\rangle \geq 0, \quad s \in \partial g(u). \tag{23}$$

In (23), substitute $z = \nabla f(x) - e(x)$, $y = x$, and $w = x - u$ to obtain

$$\langle \nabla f(x) - e(x),\, u-x\rangle \;\leq\; \langle \eta^{-1}(u-x) + s,\, x-u\rangle. \tag{24}$$

From Lemma 2.1 we know that $\Phi(u) \leq f(x) + \langle \nabla f(x), u-x\rangle + \frac{L}{2}\|u-x\|^2 + g(u)$; now add and subtract $e(x)$ to this and combine with (24) to obtain

$$\begin{aligned}
\Phi(u) &\leq f(x) + \langle \nabla f(x) - e(x),\, u-x\rangle + \tfrac{L}{2}\|u-x\|^2 + g(u) + \langle e(x),\, u-x\rangle \\
&\leq f(x) + \langle \eta^{-1}(u-x) + s,\, x-u\rangle + \tfrac{L}{2}\|u-x\|^2 + g(u) + \langle e(x),\, u-x\rangle \\
&= f(x) + g(u) + \langle s,\, x-u\rangle + \left(\tfrac{L}{2} - \tfrac{1}{\eta}\right)\|u-x\|^2 + \langle e(x),\, u-x\rangle \\
&\leq f(x) + g(x) - \tfrac{2-L\eta}{2\eta}\|u-x\|^2 + \langle e(x),\, u-x\rangle \\
&\leq \Phi(x) - \tfrac{2-L\eta}{2\eta}\|u-x\|^2 + \|e(x)\|\|u-x\|, \\
&\leq \Phi(x) - \tfrac{2-L\eta}{2\eta}\|u-x\|^2 + \tfrac{1}{\eta}\epsilon(x)\|u-x\|.
\end{aligned}$$

The third inequality follows from convexity of g, the fourth one from Cauchy-Schwarz, and the last one from the definition of $\epsilon(x)$.

□

To further analyze (22), we derive two-sided bounds on $\|x-u\|$ below.

Lemma 2.5. *Let x, u, and η be as in Theorem 2.4, and c as in (13). Then,*

$$c\|\rho(x)\| - \epsilon(x) \leq \|x-u\| \leq \|\rho(x)\| + \epsilon(x). \tag{25}$$

Proof. Lemma 2.3 implies that for $\eta > 0$ we have the crucial bounds

$$1 \leq \eta \implies q(1) \leq q(\eta), \quad \text{and} \quad 1 \geq \eta \implies p(1) \leq p(\eta). \tag{26}$$

Let $y \leftarrow x, z \leftarrow \nabla f(x)$. Note that $q(1) = \|\rho(x)\| \leq \|P_\eta(x - \eta\nabla f(x)) - x\|$ if $\eta \geq 1$, while if $\eta \leq 1$, we get $\eta p(1) \leq \|P_\eta(x - \eta\nabla f(x)) - x\|$. Compactly, we may therefore write

$$\min\{1,\eta\}\|\rho(x)\| \leq \|P_\eta(x - \eta\nabla f(x)) - x\|.$$

Using the triangle inequality and nonexpansivity of prox we see that

$$\begin{array}{rcl} \min\{1,\eta\}\|\rho(x)\| & \leq & \|P_\eta(x - \eta\nabla f(x)) - x\| \\ & \leq & \|x-u\| + \|u - P_\eta(x - \eta\nabla f(x))\| \\ & \leq & \|x-u\| + \eta\|e(x)\| \quad \leq \quad \|x-u\| + \epsilon(x). \end{array}$$

As $c \leq \liminf_k \eta_k$, for large enough k it holds that $\|x-u\| \geq c\|\rho(x)\| - \epsilon(x)$.

An upper-bound on $\|x-u\|$ may be obtained as follows

$$\begin{array}{rcl} \|x-u\| & \leq & \|x - P_\eta(x - \eta\nabla f(x))\| + \|P_\eta(x - \eta\nabla f(x)) - u\| \\ & \leq & \max\{1,\eta\}\|\rho(x)\| + \eta\|e(x)\| \quad \leq \quad \|\rho(x)\| + \epsilon(x), \end{array}$$

where we again used Lemma 2.3 and nonexpansivity. □

Theorem 2.4 and Lemma 2.5 have done the hard work; they imply the following corollary, which is a key component of the convergence framework of [33] that we ultimately will also invoke.

Corollary 2.6. *Let x, u, η, and c be as above and k sufficiently large so that c and $\eta \equiv \eta_k$ satisfy (13). Then, $\Phi(x) - \Phi(u) \geq h(x)$ holds for*

$$h(x) := \tfrac{L^2c^3}{2(2-2Lc)}\|\rho(x)\|^2 - \left(\tfrac{L^2c^2}{2-cL} + \tfrac{1}{c}\right)\|\rho(x)\|\epsilon(x) - \left(\tfrac{1}{c} - \tfrac{L^2c}{2(2-cL)}\right)\epsilon(x)^2. \tag{27}$$

Proof. We prove (27) by showing that $h(x)$ can be chosen as

$$h(x) := a_1\|\rho(x)\|^2 - a_2\|\rho(x)\|\epsilon(x) - a_3\epsilon(x)^2, \tag{28}$$

where the constants a_1, a_2, and a_3 satisfy

$$a_1 = \tfrac{L^2c^3}{2(2-2Lc)}, \quad a_2 = \tfrac{L^2c^2}{2-cL} + \tfrac{1}{c}, \quad a_3 = \tfrac{1}{c} - \tfrac{L^2c}{2(2-cL)}. \tag{29}$$

Note that by construction the scalars $a_1, a_2, a_3 > 0$. For sufficiently large k, condition (13) implies that

$$c < \eta < \tfrac{2}{L} - c \qquad \implies \qquad \tfrac{1}{\eta} > \tfrac{L}{2-Lc}, \quad \tfrac{1}{\eta} < \tfrac{1}{c}, \quad \text{and} \quad 2 - L\eta > Lc, \tag{30}$$

which in turn shows that

$$\tfrac{2-L\eta}{2\eta} > \tfrac{(2-L\eta)L}{2(2-Lc)} > \tfrac{L^2c}{2(2-Lc)} \qquad \text{and} \qquad -\tfrac{1}{\eta} > -\tfrac{1}{c}.$$

We can plug this into (22) to obtain

$$\Phi(x) - \Phi(u) \geq \tfrac{L^2 c}{2(2-Lc)}\|x-u\|^2 - \tfrac{1}{c}\epsilon(x)\|x-u\|.$$

Apply to this the two-sided bounds (25), so that we get

$$\begin{aligned}\Phi(x) - \Phi(u) \;&\geq\; \tfrac{L^2 c}{2(2-Lc)}\big(c\|\rho(x)\| - \epsilon(x)\big)^2 - \tfrac{1}{c}\epsilon(x)\left(\|\rho(x)\| + \epsilon(x)\right)\\ &= \tfrac{L^2 c^3}{2(2-Lc)}\|\rho(x)\|^2 - \big(\tfrac{L^2c^2}{2-Lc} + \tfrac{1}{c}\big)\|\rho(x)\|\epsilon(x) - \big(\tfrac{1}{c} - \tfrac{L^2 c}{2(2-Lc)}\big)\epsilon(x)^2 \;=: h(x).\end{aligned}$$

All that remains to show is that the said coefficients of $h(x)$ are positive. Since $2 - Lc > 0$ and $c > 0$, positivity of a_1 and a_2 is immediate. Inequality $a_3 = \frac{1}{c} - \frac{L^2 c}{2(2-Lc)} > 0$, holds as long as $0 < c < \frac{\sqrt{5}-1}{L}$, which is obviously true since $c < 1/L$ by assumption (13). Thus, $a_1, a_2, a_3 > 0$. □

Theorem 2.7 (Convergence). *Let $f \in C^1_L(\mathcal{X})$ such that $\inf_{\mathcal{X}} f > -\infty$ and g be lsc, convex on $\mathcal{X}$. Let $\{x^k\} \subset \mathcal{X}$ be a sequence generated by (6), and let condition (8) hold. Then, there exists a limit point x^* of the sequence $\{x^k\}$, and a constant $K > 0$, such that $\|\rho(x^*)\| \leq K\epsilon(x^*)$. Moreover, if the sequence $\{f(x^k)\}$ converges, then for every limit point x^* of $\{x^k\}$ it holds that $\|\rho(x^*)\| \leq K\epsilon(x^*)$.*

Proof. Theorem 2.4, Lemma 2.5, and Corollary 2.6 have shown that the net change in objective from one step to the next is lower bounded by a quadratic function with suitable positive coefficients, which makes the analysis technique of the differentiable case treated by [33, Thm. 2.1] applicable to setting (the exact nature of the quadratic bound derived above is crucial to the proof which essentially shows that for a large enough iteration count, this bound must be positive, which ensures progress); we omit the details for brevity. □

Theorem 2.7 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 suitable stepsizes), we can also ensure exact stationarity.

3 Scaling up: Incremental proximal splitting

We now apply NIPS to a large-scale setting. Here, the objective function $f(x)$ is assumed to be decomposable, that is

$$f(x) := \textstyle\sum_{t=1}^{T} f_t(x), \tag{31}$$

where $f_t : \mathbb{R}^n \to \mathbb{R}$ is in $C^1_{L_t}(\mathcal{X})$ (set $L \geq L_t$ for all t), and we solve

$$\min \quad f(x) + g(x), \quad x \in \mathcal{X}, \tag{32}$$

where g and $\mathcal{X}$ are as before (3).

It has long been known that for decomposable objectives it can be advantageous to replace the full gradient $\nabla f(x)$ by an *incremental gradient* $\nabla f_{r(t)}(x)$, where $r(t)$ is some suitably chosen index. Nonconvex incremental methods have been extensively analyzed in the setting of backpropagation algorithms [5, 33], which correspond to $g(x) \equiv 0$ in (32). For $g(x) \neq 0$, the stochastic generalized gradient methods of [13] or the perturbed generalized methods of [35] apply. As previously mentioned, these approaches fail to exploit the composite structure of the objective function, which can be a disadvantage already in the convex case [11].

In contrast, we exploit the composite structure of (31), and propose the following incremental nonconvex proximal-splitting method:

$$\begin{aligned} x^{k+1} &= \mathcal{M}\big(x^k - \eta_k \textstyle\sum_{t=1}^{T} \nabla f_t(z^t)\big) \\ z^1 = x^k, \quad z^{t+1} &= \mathcal{O}(z^t - \eta \nabla f_t(z^t)), \quad t = 1, \ldots, T-1. \end{aligned} \tag{33}$$

Here, $\mathcal{O}$ and $\mathcal{M}$ are appropriate nonexpansive maps, choosing which we get different algorithms. For example, when $\mathcal{X} = \mathbb{R}^n$, $g(x) \equiv 0$, and $\mathcal{M} = \mathcal{O} = \mathrm{Id}$, then (33) reduces to the problem class considered in [34]. If $\mathcal{X}$ is a closed convex set, $g(x) \equiv 0$, $\mathcal{M} = \Pi_{\mathcal{X}}$, and $\mathcal{O} = \mathrm{Id}$, then (33) reduces to a method that is essentially implicit in [34]. Note, however, that in this case, the constraints are enforced *only once* every major iteration; the intermediate iterates (z^t) may be infeasible.

Depending on the application, we may implement either of the four variants of (33) in Table 1. Which of these one prefers, depends on the complexity of the constraint set $\mathcal{X}$ and on the cost of applying P_η^g. In the first two examples $\mathcal{X}$ is not bounded, which complicates the convergence analysis; the third variant is also of practical importance but the fourth variant allows a more instructive analysis, so we discuss it only.

$\mathcal{X}$	g	$\mathcal{M}$	$\mathcal{O}$	Penalty	Proximity operator calls
$\mathbb{R}^n$	$\not\equiv 0$	prox_g	Id	P,U	once every *major* (k) iteration
$\mathbb{R}^n$	$\not\equiv 0$	prox_g	prox_g	P,U	once every *minor* (k,t) iteration
CCvx	$h(x) + \delta(x\vert\mathcal{X})$	prox_g	Id	P,C	once every major (k) iteration
CCvx	$h(x) + \delta(x\vert\mathcal{X})$	prox_g	prox_g	P,C	once every minor (k,t) iteration

Table 1: Different variants of incremental Nips (33). 'P' indicates penalized, 'U' indicates 'unconstrained', while 'C' refers to a constrained problem; 'CCvx' signifies 'Compact convex'.

3.1 Convergence

Our analysis is inspired by [34] with the obvious difference that we are dealing with a nonsmooth problem. First, as is usual with incremental methods, we also rewrite (33) in a form that matches the main iteration (6)

$$x^{k+1} = \mathcal{M}\big(x^k - \eta_k \textstyle\sum_{t=1}^{T} \nabla f_t(z^t)\big) \quad = \quad \mathcal{M}\big(x^k - \eta_k \nabla F(x^k) + \eta_k e(x^k)\big).$$

The error term at a general x is then given by $e(x) := \sum_{t=1}^{T}\big(f_t(x) - f_t(z^t)\big)$. Since we wish to reduce incremental Nips to a setting where the analysis of the batch method applies, we must ensure that the norm of the error term is bounded. Lemma 3.3 proves such a bound; but first we need to prove two auxiliary results.

Lemma 3.1 (Bounded increment). *Let z^{t+1} be computed by (33). Then,*

$$\text{if} \quad \mathcal{O} = \mathrm{Id}, \ \text{then} \quad \|z^{t+1} - z^t\| = \eta \|\nabla f_t(z^t)\| \tag{34}$$

$$\text{if} \quad \mathcal{O} = \Pi_{\mathcal{X}}, \ \text{then} \quad \|z^{t+1} - z^t\| \leq \eta \|\nabla f_t(z^t)\| \tag{35}$$

$$\text{if} \quad \mathcal{O} = \mathrm{prox}_g^\eta, \ s^t \in \partial g(z^t), \ \text{then} \quad \|z^{t+1} - z^t\| \leq 2\eta \|\nabla f_t(z^t) + s^t\|. \tag{36}$$

Proof. Relation (34) is obvious, and (35) follows immediately from nonexpansivity of projections. To prove (36), notice that definition (4) implies the inequality

$$\begin{aligned} &\tfrac{1}{2}\|z^{t+1} - z^t + \eta \nabla f_t(z^t)\|^2 + \eta g(z^{t+1}) \leq \tfrac{1}{2}\|\eta \nabla f_t(z^t)\|^2 + \eta g(z^t), \\ &\tfrac{1}{2}\|z^{t+1} - z^t\|^2 \leq \eta \langle \nabla f_t(z^t), z^t - z^{t+1} \rangle + \eta(g(z^t) - g(z^{t+1})). \end{aligned}$$

Since g is convex, $g(z^{t+1}) \geq g(z^t) + \langle s_t, z^{t+1} - z^t \rangle$ for $s_t \in \partial g(z^t)$. Moreover,

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

□

Lemma 3.2 (Incrementality error). *Let $x \equiv x^k$, and define*

$$\epsilon_t := \|\nabla f_t(z^t) - \nabla f_t(x)\|, \quad t = 1, \ldots, T. \tag{37}$$

Then, for each $t \geq 2$, the following bound on the error holds:

$$\epsilon_t \leq 2\eta L \textstyle\sum_{j=1}^{t-1}(1 + 2\eta L)^{t-1-j}\|\nabla f_j(x) + s^j\|, \quad t = 2, \ldots, T. \tag{38}$$

Proof. The proof extends the differentiable case treated in [34]. We proceed by induction. The base case is $t = 2$, for which we have

$$\epsilon_2 = \|\nabla f_2(z^2) - \nabla f_2(x)\| \leq L\|z^2 - x\| = L\|z^2 - z^1\| \overset{(36)}{\leq} 2\eta L\|\nabla f_1(x) + s^1\|.$$

Assume inductively that (38) holds for $t \leq r < T$, and consider $t = r + 1$. Then,

$$\begin{aligned} \epsilon_{r+1} &= \|\nabla f_{r+1}(z^{r+1}) - \nabla f_{r+1}(x)\| \leq L\|z^{r+1} - x\| \\ &= L\left\|\textstyle\sum_{j=1}^{r}(z^{j+1} - z^j)\right\| \leq L\textstyle\sum_{j=1}^{r}\|z^{j+1} - z^j\| \\ &\overset{\text{Lemma 3.1}}{\leq} 2\eta L\textstyle\sum_{j=1}^{r}\|\nabla f_j(z^j) + s^j\|. \end{aligned} \tag{39}$$

To complete the induction, first observe that $\|\nabla f_t(z^t) + s^t\| \leq \|\nabla f_t(x) + s^t\| + \epsilon_t$, so that on invoking the induction hypothesis we obtain for $t = 2, \ldots, r$,

$$\|\nabla f_t(z^t)\| \leq \|\nabla f_t(x)\| + 2\eta L\textstyle\sum_{j=1}^{t-1}(1 + 2\eta L)^{t-1-j}\|\nabla f_j(x) + s^j\|. \tag{40}$$

Combining inequality (40) with (39) we further obtain

$$\epsilon_{r+1} \leq 2\eta L\textstyle\sum_{j=1}^{r}\left(\|\nabla f_j(x) + s^j\| + 2\eta L\sum_{l=1}^{j-1}(1 + L\eta)^{j-1-l}\|\nabla f_l(x) + s^l\|\right).$$

Writing $\beta_j \equiv \|\nabla f_j(x) + s^j\|$ a simple manipulation of the above inequality yields

$$\begin{aligned} \epsilon_{r+1} \leq \quad & 2\eta L\beta_r + \textstyle\sum_{l=1}^{r-1}\left(2\eta L + 4\eta^2 L^2\sum_{j=l+1}^{r}(1 + 2\eta L)^{j-l-1}\right)\beta_l \\ = \quad & 2\eta L\beta_r + \textstyle\sum_{l=1}^{r-1}\left(2\eta L + 4\eta^2 L^2\sum_{j=0}^{r-l-1}(1 + 2\eta L)^j)\right)\beta_l \\ = \quad & 2\eta L\beta_r + \textstyle\sum_{l=1}^{r-1}2\eta L(1 + 2\eta L)^{r-l}\beta_l \quad = \quad 2\eta L\sum_{l=1}^{r}(1 + 2\eta L)^{r-l}\beta_l. \end{aligned}$$

□

Now we are ready to bound the error, which is done by Lemma 3.3 below.

Lemma 3.3 (Bounded error). *If for all $x \in \mathcal{X}$, $\|\nabla f_t(x)\| \leq M$ and $\|\partial g(x)\| \leq G$, then $\|e(x)\| \leq K$ for some constant $K > 0$.*

Proof. If z^{t+1} is computed by (33), $\mathcal{O} = \text{prox}_g$, and $s^t \in \partial g(z^t)$, then

$$\|z^{t+1} - z^t\| \quad \leq \quad 2\eta\|\nabla f_t(z^t) + s^t\|. \tag{41}$$

Using (41) we can bound the error incurred upon using z^t instead of x^k. Specifically, if $x \equiv x^k$, and

$$\epsilon_t := \|\nabla f_t(z^t) - \nabla f_t(x)\|, \quad t = 1, \ldots, T, \tag{42}$$

then Lemma 3.2 shows the following bound

$$\epsilon_t \leq 2\eta L \textstyle\sum_{j=1}^{t-1}(1+2\eta L)^{t-1-j}\|\nabla f_j(x) + s^j\|, \quad t = 2, \ldots, T. \tag{43}$$

Since $\epsilon_1 = 0$, we have

$$\begin{aligned}
\|e(x)\| \leq \textstyle\sum_{t=2}^{T} \epsilon_t &\overset{(43)}{\leq} 2\eta L \textstyle\sum_{t=2}^{T}\sum_{j=1}^{t-1}(1+2\eta L)^{t-1-j}\beta_j \\
&= 2\eta L \textstyle\sum_{t=1}^{T-1} \beta_t \left(\sum_{j=0}^{T-t-1}(1+2\eta L)^j\right) \\
&= \textstyle\sum_{t=1}^{T-1} \beta_t\big((1+2\eta L)^{T-t} - 1\big) \\
&\leq \textstyle\sum_{t=1}^{T-1}(1+2\eta L)^{T-t}\beta_t \\
&\leq (1+2\eta L)^{T-1}\textstyle\sum_{t=1}^{T-1}\|\nabla f_t(x) + s^t\| \\
&\leq C_1(T-1)(M+G) =: K.
\end{aligned}$$

□

Thanks to the error bounds established above, convergence of incremental Nips follows immediately from Theorem 2.7; we omit details for brevity.

4 Application to matrix factorization

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*

$$\min_{X, A \geq 0} \quad \tfrac{1}{2}\|Y - XA\|_{\mathrm{F}}^2 + \psi_0(X) + \textstyle\sum_{t=1}^{T} \psi_t(a_t), \tag{44}$$

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 (44) generalizes the well-known nonnegative matrix factorization (NMF) problem of [25] by permitting arbitrary Y (not necessarily nonnegative), and adding regularizers on X and A. A related class of problems was studied in [27], but with a crucial difference: the formulation in [27] *does not* allow nonsmooth regularizers on X. The class of problems studied in [27] is in fact a subset of those covered by Nips. On a more theoretical note, [27] 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 [27] we also rewrite (44) in a form more amenable to Nips. We eliminate A and consider nonnegatively constrained optimization problem

$$\min_X \quad \Phi(X) := \textstyle\sum_{t=1}^{T} f_t(X) + g(X), \quad \text{where} \quad g(X) := \psi_0(X) + \delta(X|\geq 0), \tag{45}$$

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

$$f_t(X) := \min_a \quad \tfrac{1}{2}\|y_t - Xa\|^2 + g_t(a), \tag{46}$$

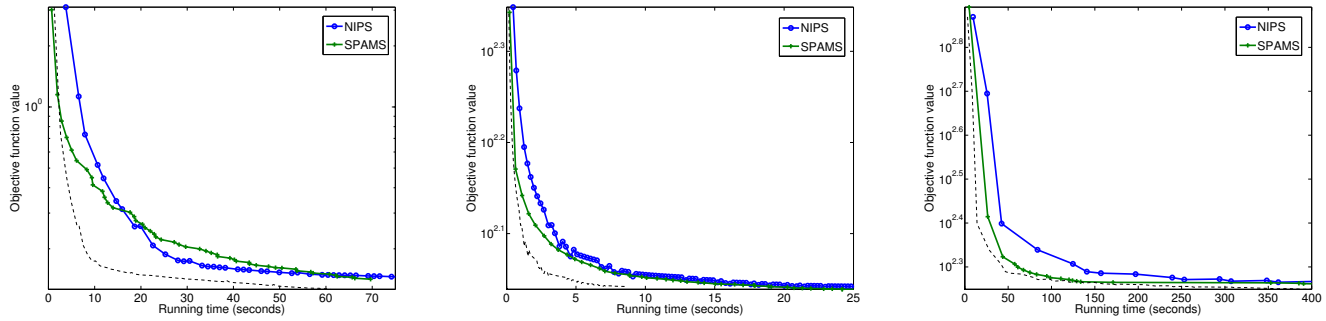

Figure 1: Running times of NIPS (Matlab) versus SPAMS (C++) for NMF on `RAND`, `CBCL`, and `YALE` datasets. Initial objective values and tiny runtimes have been suppressed for clarity.

where $g_t(a) := \psi_t(a) + \delta(a|\geq 0)$. For simplicity, assume that (46) attains its unique[2] 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 (33), and all we need is a subroutine for solving (46).[3]

We present empirical results on the following two variants of (45): (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, (45) 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 [38]; 361×2429; rank-49 factorization;

(iii) `YALE`: Yale B Database [26]; 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; $(\lambda = \gamma = 10^{-6})$.

On the NMF baseline (Fig. 1), we compare NIPS against the well optimized state-of-the-art C++ toolbox SPAMS (version 2.3) [27]. 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 [13] against NIPS, when started at the same point. As is 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 to a much lesser extent than SGGD. As predicted, the solutions returned by NIPS have objective function values lower than SGGD, and have greater sparsity.

5 Other applications

We mention below a few other applications where we have used the NIPS framework successfully. While it lies outside the scope of this chapter to cover the details of these applications, we refer the reader to research articles that include the requisite details.

[2] If not, then at the expense of more notation, we can add a strictly convex perturbation to ensure uniqueness; this error can be absorbed into the overall computational error.

[3] In practice, we use *mini-batches* for all the algorithms.

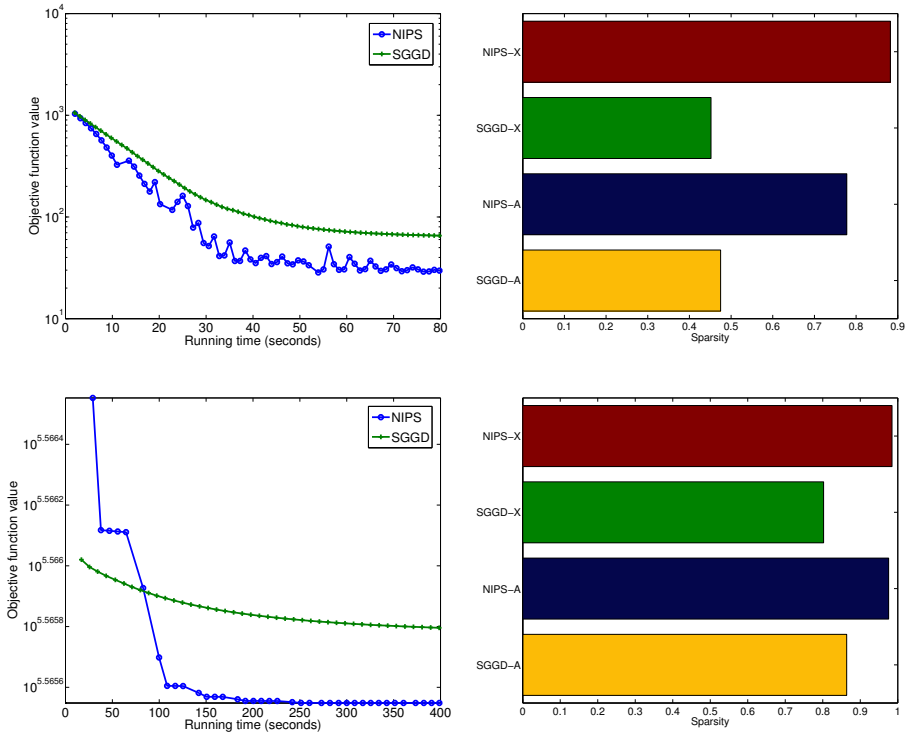

Figure 2: Sparse NMF: Nips versus SGGD. The bar plots show the sparsity (higher is better) of the factors X and A. Left plots for `RAND` dataset; right plots for `WEB`. SGGD yields slightly worse objective function values and significantly less sparse solutions than Nips.

- Online multiframe blind deconvolution [20]. In this application, a slightly modified version of NIPS is used for processing a stream of blurry images in an incremental fashion. The end goal is to obtain a sharp reconstruction of a single underlying image. The optimization problem is nonconvex because given observations $y_1, \ldots, y_T$ which are assumed to satisfy the linear model $y_i \approx A_i x$, we need to recover both A_i and x.
- Generalized dictionary learning for positive definite tensors [36]. In this problem, we seek a dictionary whose "atoms" can be sparsely combined to reconstruct a set of matrices. The key difference from ordinary dictionary learning [23] is that the observations are positive definite matrices, so the dictionary atoms must be positive definite too. The problem fits in the NIPS framework (as for NMF, subproblems relied on a nonnegative least-squares solver [22] and a nonsmooth convex solver [21]).
- Denoising signals with spiky (sparse) noise [6]. This application formulates the task of removing spiky noise from signals by formulating it as a nonconvex problem with sparsity regularization, and was hence a suitable candidate for NIPS.

6 Discussion

This chapter discussed a general optimization framework called NIPS that can solve a broad class of nonconvex composite objective (regularized) problems. Our analysis is inspired by [33], and we extend the results of [33] to admit problems that are *strictly* more general by handling nonsmooth components via proximity operators. NIPS permits nonvanishing perturbations, which is a useful practical feature. We exploited the perturbation analysis to derive both batch and incremental versions of NIPS. Finally, experiments with medium to large matrices showed that NIPS is competitive with state-of-the-art methods; NIPS was also seen to outperform the stochastic generalized gradient method.

We conclude by mentioning NIPS includes numerous algorithms and problem settings as special cases. Example are: forward-backward splitting with convex costs, incremental forward-backward splitting (convex), gradient projection (both convex and nonconvex), the proximal-point algorithm, and so on. Thus, it will be valuable to investigate if some of the theoretical results for these methods can be carried over to NIPS.

The most important theoretical question worth pursuing at this point is a less pessimistic convergence analysis for the scalable incremental version of NIPS than implied by Lemma 3.3.

References

[1] H. Attouch, J. Bolte, and B. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Mathematical Programming*, 137(1-2):91–129, 2013.

[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. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.

[6] A. Cherian, S. Sra, and N. Papanikolopoulos. Denoising sparse noise via online dictionary learning. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2011.

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

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

[9] Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer, 2011.

[10] Timothy A Davis and Yifan Hu. The University of Florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):1, 2011.

[11] J. Duchi and Y. Singer. Efficient Online and Batch Learning using Forward-Backward Splitting. *J. Mach. Learning Res. (JMLR)*, 10:2899–2934, Oct. 2009.

[12] Y. M. Ermoliev and V. I. Norkin. Stochastic generalized gradient method with application to insurance risk management. Technical Report IR-97-021, IIASA, Austria, April 1997.

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

[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] Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *arXiv:1310.3787*, 2013.

[18] Saeed Ghadimi and Guanghui Lan. Stochastic First- and Zeroth-order Methods for Nonconvex Stochastic Programming. *arXiv:1309.5549*, 2013.

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

[20] M. Hirsch, S. Harmeling, S. Sra, and B. Schölkopf. Online multi-frame blind deconvolution with super-resolution and saturation correction. *Astronomy & Astrophysics (AA)*, Feb. 2011. 11 pages.

[21] D. Kim, S. Sra, and I. S. Dhillon. A scalable trust-region algorithm with application to mixed-norm regression. In *International Conference on Machine Learning (ICML)*, 2010.

[22] D. Kim, S. Sra, and I. S. Dhillon. A non-monotonic method for large-scale non-negative least squares. *Optimization Methods and Software (OMS)*, Dec. 2011. 28 pages.

[23] 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.

[24] D. Kundur and D. Hatzinakos. Blind image deconvolution. *IEEE Signal Processing Magazine*, 13(3):43–64, may 1996.

[25] D. D. Lee and H. S. Seung. Algorithms for nonnegative matrix factorization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 556–562, 2000.

[26] K.C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):684–698, 2005.

[27] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online Learning for Matrix Factorization and Sparse Coding. *Journal of Machine Learning Research (JMLR)*, 11:10–60, 2010.

[28] O. L. Mangasarian. Mathematical Programming in Neural Networks. *Informs J. Computing*, 5(4):349–360, 1993.

[29] J. J. Moreau. Proximité et dualité dans un espace hilbertien. *Bull. Soc. Math. France*, 93:273–299, 1965.

[30] Yu. Nesterov. Gradient methods for minimizing composite objective function. Technical Report 2007/76, Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), September 2007.

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

[32] M. Schmidt, N. Le Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

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

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

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

[36] S. Sra and A. Cherian. Generalized Dictionary Learning for Symmetric Positive Definite Matrices with Application to Nearest Neighbor Retrieval. In *European Conf. Machine Learning (ECML)*, Sep. 2011.

[37] S. Sra, S. Nowozin, and S. J. Wright, editors. *Optimization for Machine Learning*. MIT Press, 2011.

[38] K.-K. Sung. *Learning and Example Selection for Object and Pattern Recognition*. PhD thesis, MIT, Artificial Intelligence Laboratory and Center for Biological and Computational Learning, Cambridge, MA, 1996.

[39] Silvia Villa, Saverio Salzo, Luca Baldassarre, and Alessandro Verri. Accelerated and inexact forward-backward algorithms. *SIAM Journal on Optimization*, 23(3):1607–1633, 2013.

[40] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Trans. Sig. Proc.*, 57(7):2479–2493, 2009.