The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Inexact Proximal Gradient Methods for Non-Convex and Non-Smooth Optimization

Bin Gu,[1] **De Wang,**[2] **Zhouyuan Huo,**[1] **Heng Huang**[1*]
[1]Department of Electrical & Computer Engineering, University of Pittsburgh, USA
[2]Dept. of Computer Science and Engineering, University of Texas at Arlington, USA
big10@pitt.edu, wangdelp@gmail.com, zhouyuan.huo@pitt.edu, heng.huang@pitt.edu

Abstract

In machine learning research, the proximal gradient methods are popular for solving various optimization problems with non-smooth regularization. Inexact proximal gradient methods are extremely important when exactly solving the proximal operator is time-consuming, or the proximal operator does not have an analytic solution. However, existing inexact proximal gradient methods only consider convex problems. The knowledge of inexact proximal gradient methods in the non-convex setting is very limited. To address this challenge, in this paper, we first propose three inexact proximal gradient algorithms, including the basic version and Nesterov's accelerated version. After that, we provide the theoretical analysis to the basic and Nesterov's accelerated versions. The theoretical results show that our inexact proximal gradient algorithms can have the same convergence rates as the ones of exact proximal gradient algorithms in the non-convex setting. Finally, we show the applications of our inexact proximal gradient algorithms on three representative non-convex learning problems. Empirical results confirm the superiority of our new inexact proximal gradient algorithms.

Introduction

Many machine learning problems involve non-smooth regularization, such as the machine learning models with a variety of sparsity-inducing penalties (Bach et al. 2012). Thus, efficiently solving the optimization problem with non-smooth regularization is important for many machine learning applications. In this paper, we focus on the optimization problem of machine learning model with non-smooth regularization as:

$$\min_{x\in\mathbb{R}^N} f(x) = g(x) + h(x) \tag{1}$$

where $g : \mathbb{R}^N \to \mathbb{R}$ corresponding to the empire risk is smooth and possibly non-convex, and $h : \mathbb{R}^N \to \mathbb{R}$ corresponding to the regularization term is non-smooth and possibly non-convex.

Proximal gradient methods are popular for solving various optimization problems with non-smooth regularization. The pivotal step of the proximal gradient method is to solve the proximal operator as following:

$$\mathrm{Prox}_{\gamma h}(y) = \arg\min_{x\in\mathbb{R}^N} \frac{1}{2\gamma}\|x-y\|^2 + h(x) \tag{2}$$

where γ is the stepsize.[1] If the function $h(x)$ is simple enough, we can obtain the solution of the proximal operator analytically. For example, if $h(x) = \|x\|_1$, the solution of the proximal operator can be obtained by a shrinkage thresholding operator (Beck and Teboulle 2009). If the function $h(x)$ is complex such that the corresponding proximal operator does not have an analytic solution, a specific algorithm should be designed for solving the proximal operator. For example, if $h(x) = \|x\|_1 + c\sum_{i<j}\max\{|x_i|,|x_j|\}$ as used in OSCAR (Zhong and Kwok 2012) for the sparse regression with automatic feature grouping, (Zhong and Kwok 2012) proposed an iterative group merging algorithm for exactly solving the proximal operator.

However, it would be expensive to solve the proximal operators when the function $h(x)$ is complex. Once again, take OSCAR as an example, when the data is with high dimensionality (empirically larger than 1,000), the iterative group merging algorithm would become very inefficient. Even worse, there would be no solver for exactly solving the proximal operators when the function $h(x)$ is over complex. For example, (Grave, Obozinski, and Bach 2011) proposed the trace Lasso norm to take into account the correlation of the design matrix to stabilize the estimation in regression. However, due to the complexity of trace Lasso norm, there still have no solver for solving the corresponding proximal operator, to the best of our knowledge.

To address the above issues, (Schmidt, Le Roux, and Bach 2011) first proposed the inexact proximal gradient methods (including the basic version (IPG) and Nesterov's accelerated version (AIPG)), which solves the proximal operator approximately (*i.e.*, tolerating an error in the calculation of the proximal operator). They proved that the inexact proximal gradient methods can have the same convergence rates as the ones of exact proximal gradient methods, provided that the errors in computing the proximal operator decrease at appropriate rates. Independently, (Villa et al. 2013) proposed AIPG algorithm and proved the corresponding convergence rate. In the paper of (Villa et al. 2013),

[*]To whom all correspondence should be addressed.

[1]The stepsize γ is set manually or automatically determined by a backtracking line-search procedure (Beck and Teboulle 2009).

Table 1: Representative (exact and inexact) proximal gradient algorithms. (C and NC are the abbreviations of convex and non-convex respectively.)

Algorithm	Proximal	Accelerated	$g(x)$	$h(x)$	Reference
PG+APG	Exact	Yes	C	C	(Beck and Teboulle 2009)
APG	Exact	Yes	C+NC	C	(Ghadimi and Lan 2016)
PG	Exact	No	NC	NC	(Boţ, Csetnek, and László 2016)
APG	Exact	Yes	C+NC	C+NC	(Li and Lin 2015)
IPG+AIPG	Inexact	Yes	C	C	(Schmidt, Le Roux, and Bach 2011)
AIFB (AIPG)	Inexact	Yes	C	C	(Villa et al. 2013)
IPG+AIPG	Inexact	Yes	C+NC	C+NC	Ours

they called AIPG as the inexact forward-backward splitting method (AIFB) which is well-known in the field of signal processing. We summarize these works in Table 1.

From Table 1, we find that the existing inexact proximal gradient methods only consider convex problems. However, a lot of optimization problems in machine learning are non-convex. The non-convexity originates either from the empirical risk function $g(x)$ or the regularization function $h(x)$. First, we investigate the empirical risk function $g(x)$ (*i.e.*, loss function). The correntropy induced loss (Feng et al. 2015) is widely used for robust regression and classification, which is non-convex. The symmetric sigmoid loss on the unlabeled samples is used in semi-supervised SVM (Chapelle, Chi, and Zien 2006) which is non-convex. Second, we investigate the regularization function $h(x)$. Capped-l_1 penalty (Zhang 2010) is used for unbiased variable selection, and the low rank constraint (Jain, Meka, and Dhillon 2010) is widely used for the matrix completion. Both of these regularization functions are non-convex. However, our knowledge of inexact proximal gradient methods is very limited in the non-convex setting.

To address this challenge, in this paper, we first propose three inexact proximal gradient algorithms, including the basic and Nesterov's accelerated versions, which can handle the non-convex problems. Then we give the theoretical analysis to the basic and Nesterov's accelerated versions. The theoretical results show that our inexact proximal gradient algorithms can have the same convergence rates as the ones of exact proximal gradient algorithms. Finally, we provide the applications of our inexact proximal gradient algorithms on three representative non-convex learning problems. The applications on robust OSCAR and link prediction show that, our inexact proximal gradient algorithms could be significantly faster than the exact proximal gradient algorithms. The application on robust Trace Lasso fills the vacancy that there is no proximal gradient algorithm for Trace Lasso.

Contributions. The main contributions of this paper are summarized as follows:

1. We first propose the basic and accelerated inexact proximal gradient algorithms with rigorous convergence guarantees. Specifically, our inexact proximal gradient algorithms can have the same convergence rates as the ones of exact proximal gradient algorithms in the non-convex setting.
2. We provide the applications of our inexact proximal gradient algorithms on three representative non-convex learning problems, *i.e.*, robust OSCAR, link prediction and robust Trace Lasso. The results confirm the superiority of our inexact proximal gradient algorithms.

Related Works

Proximal gradient methods are one of the most important methods for solving various optimization problems with non-smooth regularization. There have been a variety of exact proximal gradient methods.

Specifically, for convex problems, (Beck and Teboulle 2009) proposed basic proximal gradient (PG) method and Nesterov's accelerated proximal gradient (APG) method. They proved that PG has the convergence rate $O(\frac{1}{T})$, and APG has the convergence rate $O(\frac{1}{T^2})$, where T is the number of iterations. For non-convex problems, (Ghadimi and Lan 2016) considered that only $g(x)$ could be non-convex, and proved that the convergence rate of APG method is $O(\frac{1}{T})$. (Boţ, Csetnek, and László 2016) considered that both of $g(x)$ and $h(x)$ could be non-convex, and proved the convergence of PG method. (Li and Lin 2015) considered that both of $g(x)$ and $h(x)$ could be non-convex, and proved that the APG algorithm can converge in a finite number of iterations, in a linear rate or a sublinear rate (*i.e.*, $O(\frac{1}{T})$) at different conditions. We summarize these exact proximal gradient methods in Table 1.

In addition to the above batch exact proximal gradient methods, there are the stochastic and online proximal gradient methods (Duchi and Singer 2009; Xiao and Zhang 2014). Because they are beyond the scope of this paper, we do not review them in this paper.

Preliminaries

In this section, we introduce the Lipschitz smooth, ε-approximate subdifferential and ε-approximate Kurdyka-Łojasiewicz (KL) property, which are critical to the convergence analysis of our inexact proximal gradient methods in the non-convex setting.

Lipschitz smooth: For the smooth functions $g(x)$, we have the Lipschitz constant L for $\nabla g(x)$ (Wood and Zhang 1996) as following.

Assumption 1. *L is the Lipschitz constant for $\nabla g(x)$. Thus, for all x and y, L-Lipschitz smooth can be presented as*

$$\|\nabla g(x) - \nabla g(y)\| \leq L\|x - y\| \tag{3}$$

Equivalently, L-Lipschitz smooth can also be written as the formulation (4).

$$g(x) \leq g(y) + \langle \nabla g(y), x - y \rangle + \frac{L}{2} \|x - y\|^2 \quad (4)$$

ε-approximate subdifferential: Because inexact proximal gradient is used in this paper, an ε-approximate proximal operator may produce an ε-approximate subdifferential. In the following, we give the definition of ε-approximate subdifferential (Bertsekas et al. 2003) which will be used in the ε-approximate KL property (*i.e.*, Definition 2).

Definition 1 (ε-approximate subdifferential)**.** *Given a convex function $h(x) : \mathbb{R}^N \mapsto \mathbb{R}$ and a positive scalar ε, the ε-approximate subdifferential of $h(x)$ at a point $x \in \mathbb{R}^N$ (denoted as $\partial_\varepsilon h(x)$) is*

$$\partial_\varepsilon h(x) = \left\{ d \in \mathbb{R}^N : h(y) \geq h(x) + \langle d, y - x \rangle - \varepsilon \right\} \quad (5)$$

Based on Definition 1, if $d \in \partial_\varepsilon h(x)$, we say that d is an ε-approximate subgradient of $h(x)$ at the point x.

ε-approximate KL property: Originally, KL property is introduced to analyze the convergence rate of exact proximal gradient methods in the non-convex setting (Li and Lin 2015; Boţ, Csetnek, and László 2016). Because this paper focuses on the inexact proximal gradient methods, correspondingly we propose the ε-approximate KL property in Definition 2, where the function $dist(x, S)$ is defined by $dist(x, S) = \min_{y \in S} \|x - y\|$, and S is a subset of $\mathbb{R}^N$.

Definition 2 (ε-KL property)**.** *A function $f(x) = g(x) + h(x) : \mathbb{R}^N \to (-\infty, +\infty]$ is said to have the ε-KL property at $\overline{u} \in \{u \in \mathbb{R}^N : \nabla g(u) + \partial_\varepsilon h(u)) \neq \emptyset\}$, if there exists $\eta \in (0, +\infty]$, a neighborhood U of $\overline{u}$ and a function $\varphi \in \Phi_\eta$, such that for all $u \in U \cap \{u \in \mathbb{R}^N : f(\overline{u}) < f(u) < f(\overline{u}) + \eta\}$, the following inequality holds*

$$\varphi'(f(u) - f(\overline{u})) dist(\boldsymbol{0}, \nabla g(u) + \partial_\varepsilon h(u))) \geq 1 \quad (6)$$

where Φ_η stands for a class of functions $\varphi : [0, \eta) \to \mathbb{R}^+$ satisfying:

1. *φ is concave and continuously differentiable function on $(0, \eta)$;*
2. *φ is continuous at 0, $\varphi(0) = 0$;*
3. *and $\varphi'(x) > 0$, $\forall x \in (0, \eta)$.*

Inexact Proximal Gradient Algorithms

In this section, we first propose the basic inexact proximal gradient algorithm for the non-convex optimization, and then propose two accelerated inexact proximal gradient algorithms.

Basic Version

As shown in Table 1, for the convex problems (*i.e.*, both the functions $g(x)$ and $h(x)$ are convex), (Schmidt, Le Roux, and Bach 2011) proposed a basic inexact proximal gradient (IPG) method. We follow the framework of IPG in (Schmidt, Le Roux, and Bach 2011), and give our IPG algorithm for the non-convex optimization problems (*i.e.*, either the function $g(x)$ or the function $h(x)$ is non-convex).

Specifically, our IPG algorithm is presented in Algorithm 1. Similar with the exact proximal gradient algorithm, the pivotal step of our IPG (*i.e.*, Algorithm 1) is to compute an inexact proximal operator $x \in \text{Prox}_{\gamma h}^{\varepsilon}(y)$ as following.

$$x \in \text{Prox}_{\gamma h}^{\varepsilon}(y) = \Big\{ z \in \mathbb{R}^N : \frac{1}{2\gamma} \|z - y\|^2 + h(z) \leq \varepsilon + \min_x \frac{1}{2\gamma} \|x - y\|^2 + h(x) \Big\}$$

where ε denotes an error in the calculation of the proximal operator. As discussed in (Tappenden, Richtárik, and Gondzio 2016), there are several methods to compute the inexact proximal operator. The most popular method is using a primal dual algorithm to control the dual gap (Bach et al. 2012). Based on the dual gap, we can strictly control the error in the calculation of the proximal operator.

Algorithm 1 Basic inexact proximal gradient method (IPG)

Input: m, error ε_k ($k = 1, \cdots, m$), stepsize $\gamma < \frac{1}{L}$.
Output: x_m.

```
Initialize x_0 ∈ R^d.
for k = 1, ··· , m do
    Compute x_k ∈ Prox_{γh}^{ε_k} (x_{k−1} − γ∇g(x_{k−1})).
end for
```

Accelerated Versions

We first propose a Nesterov's accelerated inexact proximal gradient algorithm for non-convex optimization, then give a nonmonotone accelerated inexact proximal gradient algorithm.

As shown in Table 1, for the convex optimization problems, (Beck and Teboulle 2009) proposed a Nesterov's accelerated inexact proximal gradient (*i.e.*, APG) method, and (Schmidt, Le Roux, and Bach 2011) proposed a Nesterov's accelerated inexact proximal gradient (*i.e.*, AIPG) method. Both of APG and AIPG are accelerated by a momentum term. However, as mentioned in (Li and Lin 2015), traditional Nesterov's accelerated method may encounter a bad momentum term for the non-convex optimization. To address the bad momentum term, (Li and Lin 2015) added another proximal operator as a monitor to make the objective function sufficient descent. To make the objective functions generated from our AIPG strictly descent, we follow the framework of APG in (Li and Lin 2015). Thus, we compute two inexact proximal operators $z_{k+1} \in \text{Prox}_{\gamma h}^{\varepsilon_k}(y_k - \gamma \nabla g(y_k))$ and $v_{k+1} \in \text{Prox}_{\gamma h}^{\varepsilon_k}(x_k - \gamma \nabla g(x_k))$, where v_{k+1} is a monitor to make the objective function strictly descent. Specifically, our AIPG is presented in Algorithm 2.

To address the bad momentum term in the non-convex setting, our AIPG (*i.e.*, Algorithm 2) uses a pair of inexact proximal operators to make the objective functions strictly descent. Thus, AIPG is a monotone algorithm. Actually, using two proximal operators is a conservative strategy. As mentioned in (Li and Lin 2015), we can accept

Algorithm 2 Accelerated inexact proximal gradient method (AIPG)

Input: m, error ε_k $(k = 1, \cdots, m)$, $t_0 = 0$, $t_1 = 1$, stepsize $\gamma < \frac{1}{L}$.
Output: x_{m+1}.

```
Initialize x_0 ∈ R^d, and x_1 = z_1 = x_0.
for k = 1, 2, ..., m do
    y_k = x_k + ((t_{k-1})/(t_k))(z_k - x_k) + ((t_{k-1}-1)/(t_k))(x_k - x_{k-1}).
    Compute z_{k+1} such that z_{k+1} ∈ Prox_{γh}^{ε_k}(y_k - γ∇g(y_k)).
    Compute v_{k+1} such that v_{k+1} ∈ Prox_{γh}^{ε_k}(x_k - γ∇g(x_k)).
    t_{k+1} = (sqrt(4t_k^2 + 1) + 1)/2.
    x_{k+1} = { z_{k+1}  if f(z_{k+1}) ≤ f(v_{k+1})
              { v_{k+1}  otherwise
end for
```

z_{k+1} as x_{k+1} directly if it satisfies the criterion $f(z_{k+1}) \leq f(x_k) - \frac{\delta}{2}\|z_{k+1} - y_k\|^2$ which shows that y_k is a good extrapolation. v_{k+1} is computed only when this criterion is not met. Thus, the average number of proximal operators can be reduced. Following this idea, we propose our nonmonotone accelerated inexact proximal gradient algorithm (nmAIPG) in Algorithm 3. Empirically, we find that the nmAIPG with the value of $\delta \in [0.5, 1]$ works good. In our experiments, we set $\delta = 0.6$.

Algorithm 3 Nonmonotone accelerated inexact proximal gradient method (nmAIPG)

Input: m, ε_k $(k = 1, \cdots, m)$, $t_0 = 0$, $t_1 = 1$, stepsize $\gamma < \frac{1}{L}$, $\delta > 0$.
Output: x_{m+1}.

```
Initialize x_0 ∈ R^d, and x_1 = z_1 = x_0.
for k = 1, 2, ..., m do
    y_k = x_k + ((t_{k-1})/(t_k))(z_k - x_k) + ((t_{k-1}-1)/(t_k))(x_k - x_{k-1}).
    Compute z_{k+1} such that z_{k+1} ∈ Prox_{γh}^{ε_k}(y_k - γ∇g(y_k)).
    if f(z_{k+1}) ≤ f(x_k) - (δ/2)||z_{k+1} - y_k||^2 then
        x_{k+1} = z_{k+1}
    else
        Compute v_{k+1} such that v_{k+1} ∈ Prox_{γh}^{ε_k}(x_k - γ∇g(x_k)).
        x_{k+1} = { z_{k+1}  if f(z_{k+1}) ≤ f(v_{k+1})
                  { v_{k+1}  otherwise
    end if
    t_{k+1} = (sqrt(4t_k^2 + 1) + 1)/2.
end for
```

Convergence Analysis

As mentioned before, the convergence analysis of inexact proximal gradient methods for the non-convex problems is still an open problem. This section will address this challenge.

Specifically, we first prove that IPG and AIPG converge to a critical point in the convex or non-convex setting (Theorem 1) if $\{\varepsilon_k\}$ is a decreasing sequence and $\sum_{k=1}^{m} \varepsilon_k < \infty$. Next, we prove that IPG has the convergence rate $O(\frac{1}{T})$ for the non-convex problems (Theorem 2) when the errors decrease at an appropriate rate. Then, we prove the convergence rates for AIPG in the non-convex setting (Theorem 3). The detailed proofs of Theorems 1, 2 and 3 can be found in Appendix.

Convergence of IPG and AIPG

We first prove that IPGA and AIPG converge to a critical point (Theorem 1) if $\{\varepsilon_k\}$ is a decreasing sequence and $\sum_{k=1}^{m} \varepsilon_k < \infty$.

Theorem 1. *With Assumption 1, if $\{\varepsilon_k\}$ is a decreasing sequence and $\sum_{k=1}^{m} \varepsilon_k < \infty$, we have $\mathbf{0} \in \lim_{k\to\infty} \nabla g(x_k) + \partial_{\varepsilon_k} h(x_k)$ for IPG and AIPG in the convex and non-convex optimization.*

Remark 1. *Theorem 1 guarantees that IPG and AIPG converge to a critical point (or called as stationary point) after an infinite number of iterations in the convex or non-convex setting.*

Convergence Rates of IPG

Because both the functions $g(x)$ and $h(x)$ are possibly non-convex, we cannot directly use $f(x_k) - f(x^*)$ or $\|x_k - x^*\|$ for analyzing the convergence rate of IPG, where x^* is an optimal solution of (1). In this paper, we use $\frac{1}{m}\sum_{k=1}^{m} \|x_k - x_{k-1}\|^2$ for analyzing the convergence rate of IPG in the non-convex setting. The detailed reason is provided in Appendix. Theorem 2 shows that IPG has the convergence rate $O(\frac{1}{T})$ for the non-convex optimization when the errors decrease at an appropriate rate, which is exactly the same as the error-free case (see discussion in Remark 2).

Theorem 2. *For $g(x)$ is non-convex, and $h(x)$ is convex or non-convex, we have the following results for IPG:*

1. *If $h(x)$ is convex, we have that*

$$\frac{1}{m}\sum_{k=1}^{m} \|x_k - x_{k-1}\|^2 \leq \frac{1}{m}\left(2A_m + \sqrt{\frac{1}{\frac{1}{\gamma} - \frac{L}{2}}\left(f(x_0) - f(x^*)\right)} + \sqrt{B_m}\right)^2 \tag{7}$$

   where $A_m = \frac{1}{2}\sum_{k=1}^{m} \frac{1}{\frac{1}{\gamma} - \frac{L}{2}}\sqrt{\frac{2\varepsilon_k}{\gamma}}$ *and* $B_m = \frac{1}{\frac{1}{\gamma} - \frac{L}{2}}\sum_{k=1}^{m} \varepsilon_k$.

2. *If $h(x)$ is non-convex, we have that*

$$\frac{1}{m}\sum_{k=1}^{m} \|x_k - x_{k-1}\|^2 \leq \frac{1}{m\left(\frac{1}{2\gamma} - \frac{L}{2}\right)}\left(f(x_0) - f(x^*) + \sum_{k=1}^{m} \varepsilon_k\right) \tag{8}$$

Remark 2. *Theorem 2 implies that IPG has the convergence rate $O(\frac{1}{T})$ for the non-convex optimization without errors. If $\{\sqrt{\varepsilon_k}\}$ is summable and $h(x)$ is convex, we can also have that IPG has the convergence rate $O(\frac{1}{T})$ for the non-convex optimization. If $\{\varepsilon_k\}$ is summable and $h(x)$ is non-convex, we can also have that IPG has the convergence rate $O(\frac{1}{T})$ for the non-convex optimization.*

Convergence Rates of AIPG

In this section, based on the ε-KL property, we prove that AIPG converges in a finite number of iterations, in a linear rate or a sublinear rate at different conditions in the non-convex setting (Theorem 3), which is exactly the same as the error-free case (Li and Lin 2015).

Theorem 3. *Assume that g is a non-convex function with Lipschitz continuous gradients, h is a proper and lower semicontinuous function. If the function f satisfies the ε-KL property, $\varepsilon_k = \alpha \|v_{k+1} - x_k\|^2$, $\alpha \geq 0$, $\frac{1}{2\gamma} - \frac{L}{2} - \alpha \geq 0$ and the desingularising function has the form $\varphi(t) = \frac{C}{\theta} t^\theta$ for some $C > 0$, $\theta \in (0, 1]$, then*

1. *If $\theta = 1$, there exists k_1 such that $f(x_k) = f^*$ for all $k > k_1$ and AIPG terminates in a finite number of steps, where $\lim_{k\to\infty} f(x_k) = f^*$.*
2. *If $\theta \in [\frac{1}{2}, 1)$, there exists k_2 such that for all $k > k_2$*

$$f(x_k) - \lim_{k\to\infty} f(x_k) \leq \left(\frac{d_1 C^2}{1 + d_1 C^2}\right)^{k-k_2} (f(v_k) - f^*)$$

where $d_1 = \frac{\left(\frac{1}{\gamma} + L + \sqrt{\frac{2\alpha}{\gamma}}\right)^2}{\frac{1}{2\gamma} - \frac{L}{2} - \alpha}$.

3. *If $\theta \in (0, \frac{1}{2})$, there exists k_3 such that for all $k > k_3$*

$$f(x_k) - \lim_{k\to\infty} f(x_k) \leq \left(\frac{C}{(k - k_3) d_2 (1 - 2\theta)}\right)^{\frac{1}{1-2\theta}}$$

where

$$d_2 = \min\left\{\frac{1}{2d_1 C}, \frac{C}{1 - 2\theta}\left(2^{\frac{2\theta-1}{2\theta-2}} - 1\right)(f(v_0) - f^*)^{2\theta-1}\right\}$$

Remark 3. *Theorem 3 implies that AIPG converges in a finite number of iterations when $\theta = 1$, in a linear rate when $\theta \in [\frac{1}{2}, 1)$ and at least a sublinear rate when $\theta \in (0, \frac{1}{2})$.*

Experiments

We first give the experimental setup, then present the implementations of three non-convex applications with the corresponding experimental results.

Experimental Setup

Design of Experiments To validate the effectiveness of our inexact proximal gradients methods (*i.e.*, IPG, AIPG and nmAIPG), we apply them to solve three representative non-convex learning problems as follows.

1. **Robust OSCAR:** Robust OSCAR is a robust version of OSCAR method (Zhong and Kwok 2012), which is a feature selection model with the capability to automatically detect feature group structure. The function $g(x)$ is non-convex, and the function $h(x)$ is convex. Zhong and Kwok (Zhong and Kwok 2012) proposed a fast iterative group merging algorithm for exactly solving the proximal operator.
2. **Social Link Prediction:** Given an incomplete matrix M (user-by-user) with each entry $M_{ij} \in \{+1, -1\}$, social link prediction is to recover the matrix M (*i.e.* predict the potential social links or friendships between users) with low-rank constraint. The function $g(x)$ is convex, and the function $h(x)$ is non-convex. The low-rank proximal operator can be solved exactly by the Lanczos method (Larsen 1998).
3. **Robust Trace Lasso:** Robust trace Lasso is a robust version of trace Lasso (Grave, Obozinski, and Bach 2011). The function $g(x)$ is non-convex, and the function $h(x)$ is convex. To the best of our knowledge, there is still no proximal gradient algorithm for solving trace Lasso.

We also summarize these three non-convex learning problems in Table 2. To show the advantages of our inexact proximal gradients methods, we compare the convergence rates and the running time for our inexact proximal gradients methods and the exact proximal gradients methods (*i.e.*, PG, APG and nmAPG).

Table 2: Three typical learning applications. (C, NC and PO are the abbreviations of convex, non-convex and proximal operator, respectively.)

Application	$g(x)$	$h(x)$	Exact PO
Robust OSCAR	NC	C	Yes
Link Prediction	C	NC	Yes
Robust Trace Lasso	NC	C	No

Datasets Table 3 summarizes the six datasets used in our experiments. Specifically, the Cardiac and Coil20[2] datasets are for the robust OSCAR application. The Cardiac dataset was collected from hospital, which is to predict the area of the left ventricle, and is encouraged to find the homogenous groups of features. The Soc-sign-epinions and Soc-Epinions1 datasets[3] are the social network data for the social link prediction application, where each row corresponds to a node, and each collum corresponds to an edge. The GasSensorArrayDrift and YearPredictionMSD datasets from UCI[4] are for the robust trace Lasso application.

Implementations and Results

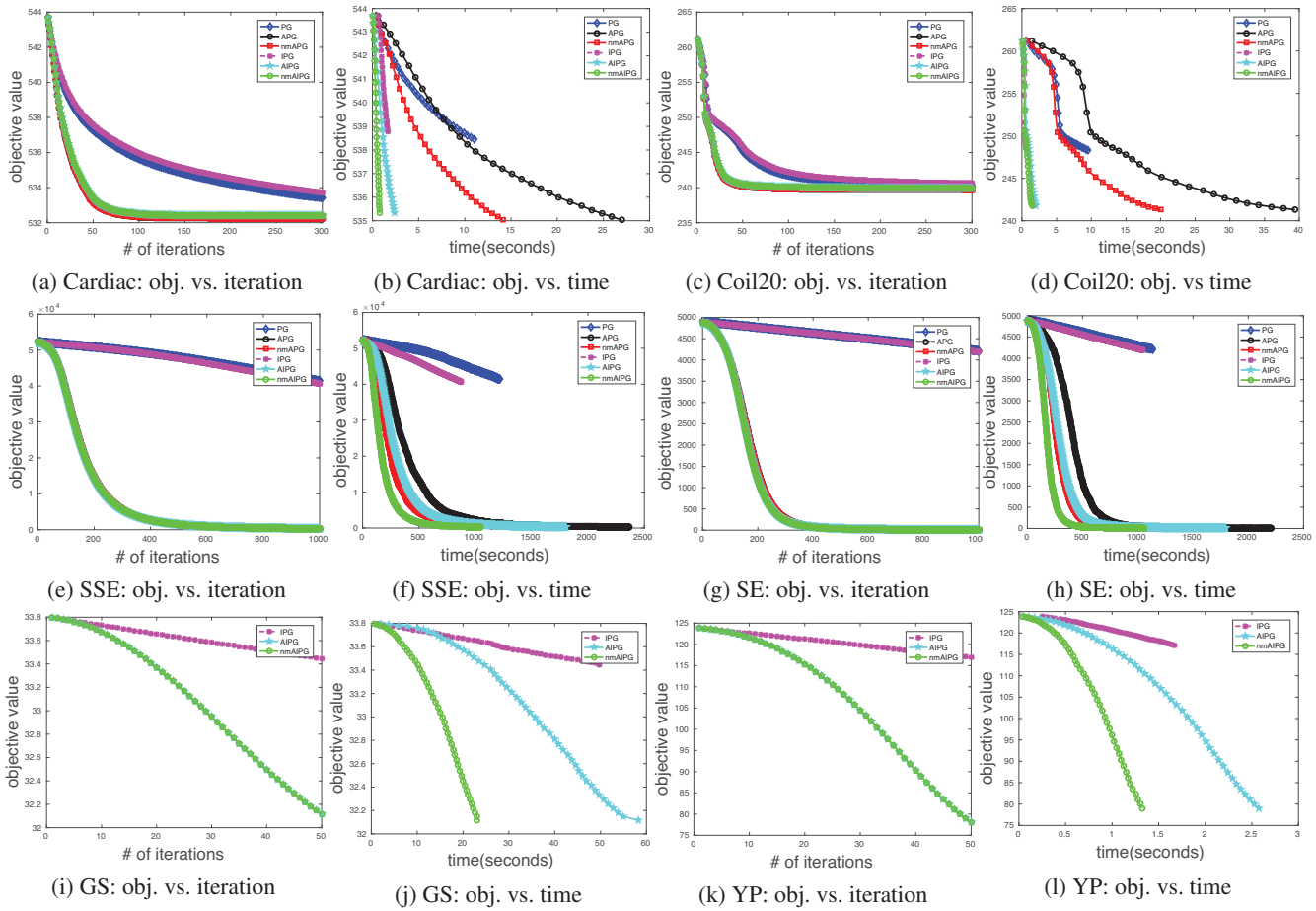
Robust OSCAR For the robust regression, we replace the square loss originally used in OSCAR with the correntropy induced loss (He, Zheng, and Hu 2011). Thus, we consider the robust OSCAR with the functions $g(x)$ and $h(x)$ as follows.

$$g(x) = \frac{\sigma^2}{2} \sum_{i=1}^{l} \left(1 - e^{-\frac{(y_i - X_i^T x)^2}{\sigma^2}}\right), \quad (9)$$

[2]http://www1.cs.columbia.edu/CAVE/software/softlib/coil-20.php

[3]http://snap.stanford.edu/data

[4]https://archive.ics.uci.edu/ml/datasets.html



(a) Cardiac: obj. vs. iteration (b) Cardiac: obj. vs. time (c) Coil20: obj. vs. iteration (d) Coil20: obj. vs time

(e) SSE: obj. vs. iteration (f) SSE: obj. vs. time (g) SE: obj. vs. iteration (h) SE: obj. vs. time

(i) GS: obj. vs. iteration (j) GS: obj. vs. time (k) YP: obj. vs. iteration (l) YP: obj. vs. time

Figure 1: Comparison of convergence speed for different methods. (a-d): Robust OSCAR. (e-h): Link prediction. (i-l) Robust trace LASSO.

Table 3: The datasets used in the experiments.

Application	Dataset	Sample size	Attributes
Robust OSCAR	Cardiac	3,360	800
	Coil20	1,440	1,024
Social Link Prediction	Soc-sign-epinions(SSE)	131,828	841,372
	Soc-Epinions1(SE)	75,879	508,837
Robust Trace Lasso	GasSensorArrayDrift(GS)	13,910	128
	YearPredictionMSD(YP)	51,5345	90

$$h(x) \quad = \quad \lambda_1 \|x\|_1 + \lambda_2 \sum_{i<j} \max\{|x_i|, |x_j|\}, \quad (10)$$

where $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are two regularization parameters. For exact proximal gradients algorithms, Zhong and Kwok (Zhong and Kwok 2012) proposed a fast iterative group merging algorithm for exactly solving the proximal subproblem. We propose a subgradient algorithm to approximately solve the proximal subproblem. Let $o(j) \in \{1, 2, \cdots, N\}$ denote the order of $|x_j|$ [5] among $\{|x_1|, |x_2|, \cdots, |x_N|\}$ such that if $o(j_1) < o(j_2)$, we have $|x_{j_1}| \leq |x_{j_2}|$. Thus, the subgradient of $h(x)$ can be computed as $\partial h(x) = \sum_{j=1}^{N} (\lambda_1 + \lambda_2(o(j) - 1)) \, \partial |x_j|$. The subgradient algorithm is omitted here due to the space limit. We implement our IPG, AIPG and nmAIPG methods for robust OSCAR in Matlab. We also implement PG, APG and nmAPG in Matlab.

[5]Here, x_j denotes the j-th coordinate of the vector x.

Figures 1a and 1c show the convergence rates of the objective value vs. iteration for the exact and inexact proximal gradients methods. The results confirm that exact and inexact proximal gradients methods have the same convergence rates. The convergence rates of exact and inexact accelerated methods are faster than the ones of the basic methods (PG and IPG). Figures 1b and 1d show the convergence rates of the objective value vs. the running time for the exact and inexact proximal gradients methods. The results show that our inexact methods are **significantly faster** than the exact methods. When the dimension increases, we can even achieve more than **100 folds speedup**. This is because our subgradient based algorithm for approximately solving the proximal subproblem is much efficient than the projection algorithm for exactly solving the proximal subproblem (Zhong and Kwok 2012).

Social Link Prediction In social link prediction problem, we hope to predict the new potential social links or friendships between online users. Given an incomplete matrix M (user-by-user) with each entry $M_{ij} \in \{+1, -1\}$, social link prediction is to recover the matrix M with low-rank constraint. Specifically, social link prediction considers the function $f(X)$ as following:

$$\begin{aligned} \min_{X} \quad & \underbrace{\frac{1}{2}\sum_{(i,j)\in\Omega} \log(1+\exp(-X_{ij}M_{ij}))}_{g(X)} \qquad (11) \\ s.t. \quad & \text{rank}(X) \leq r\,, \end{aligned}$$

where Ω is a set of (i, j) corresponding to the entries of M which are observed, λ is a regularization parameter. The proximal operator $\min_{\text{rank}(X)\leq r} \|X - (X_{t-1} - \gamma\nabla g(X_{t-1}))\|^2$ can be solved by the rank-r singular value decomposition (SVD) (Jain, Meka, and Dhillon 2010). The rank-r SVD can be solved exactly by the Lanczos method (Larsen 1998), and also can be solved approximately by the power method (Halko, Martinsson, and Tropp 2011; Journée et al. 2010). We implement our IPG, AIPG and nmAIPG methods for social link prediction in Matlab. We also implement PG, APG and nmAPG in Matlab.

Figures 1e and 1g show the convergence rates of the objective value vs. iteration for the exact and inexact proximal gradients methods. The results confirm that exact and inexact proximal gradients methods have the same convergence rates. Figures 1f and 1h illustrate the convergence rates of the objective value vs. running time for the exact and inexact proximal gradients methods. The results verify that our inexact methods are **faster** than the exact methods.

Robust Trace Lasso Robust trace Lasso is a robust version of trace Lasso (Grave, Obozinski, and Bach 2011; Bach 2008). Same with robust OSCAR, we replace the square loss originally used in trace Lasso with the correntropy induced loss (He, Zheng, and Hu 2011) for robust regression. Thus, we consider the robust trace Lasso with the functions $g(x)$ and $h(x)$ as following:

$$g(x) = \frac{\sigma^2}{2}\sum_{i=1}^{l}\left(1 - e^{-\frac{(y_i - X_i^T x)^2}{\sigma^2}}\right) \qquad (12)$$

$$h(x) = \lambda \|X\text{Diag}(x)\|_* \,, \qquad (13)$$

where $\|\cdot\|_*$ is the trace norm, λ is a regularization parameter. To the best of our knowledge, there is still no algorithm to exactly solve the proximal subproblem. To implement our IPG and AIPG, we propose a subgradient algorithm to approximately solve the proximal subproblem. Specifically, the subgradient of the trace Lasso regularization $\|X\text{Diag}(x)\|_*$ can be computed by Theorem 4 which is originally provided in (Bach 2008). We implement our IPG, AIPG and nmAIPG methods for robust trace Lasso in Matlab.

Theorem 4. *Let $U\text{Diag}(s)V^T$ be the singular value decomposition of $X\text{Diag}(x)$. Then, the subgradient of the trace Lasso regularization $\|X\text{Diag}(x)\|_*$ is given by*

$$\begin{aligned} \partial \|X\text{Diag}(x)\|_* = \{&\text{Diag}\left(X^T(UV^T + M)\right) : \qquad (14) \\ &\|M\|_2 \leq 1, U^T M = 0 \ \text{and} \ MV = 0\} \end{aligned}$$

Figures 1i-1l show the convergence rates of the objective value vs. iteration and running time respectively, for our inexact proximal gradient methods. The results demonstrate that we provide efficient proximal gradient algorithms for the robust trace Lasso. More importantly, our IPG, AIPG and nmAIPG algorithms **fill the vacancy** that there is no proximal gradient algorithm for trace Lasso. This is because that, directly solving the proximal subproblem for robust trace Lasso is quite difficult (Grave, Obozinski, and Bach 2011; Bach 2008). Our subgradient based algorithm provides an alternative approach for solving the proximal subproblem.

Summary of the Experimental results Based on the results of three non-convex machine learning applications, our conclusion is that our inexact proximal gradient algorithms can provide **flexible** algorithms to the optimization problems with complex non-smooth regularization. More importantly, our inexact algorithms could be **significantly faster** than the exact proximal gradient algorithms.

Conclusion

Existing inexact proximal gradient methods only consider convex problems. The knowledge of inexact proximal gradient methods in the non-convex setting is very limited. To address this problem, we proposed three inexact proximal gradient algorithms with the theoretical analysis. The theoretical results show that our inexact proximal gradient algorithms can have the same convergence rates as the ones of exact proximal gradient algorithms in the non-convex setting. We also provided the applications of our inexact proximal gradient algorithms on three representative non-convex learning problems. The results confirm the superiority of our inexact proximal gradient algorithms.

Acknowledgement

This work was partially supported by the following grants: NSF-IIS 1302675, NSF-IIS 1344152, NSF-DBI 1356628, NSF-IIS 1619308, NSF-IIS 1633753, NIH R01 AG049371.

References

Bach, F.; Jenatton, R.; Mairal, J.; Obozinski, G.; et al. 2012. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning* 4(1):1–106.

Bach, F. R. 2008. Consistency of trace norm minimization. *Journal of Machine Learning Research* 9(Jun):1019–1048.

Beck, A., and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences* 2(1):183–202.

Bertsekas, D. P.; Nedi, A.; Ozdaglar, A. E.; et al. 2003. Convex analysis and optimization.

Boţ, R. I.; Csetnek, E. R.; and László, S. C. 2016. An inertial forward–backward algorithm for the minimization of the sum of two nonconvex functions. *EURO Journal on Computational Optimization* 4(1):3–25.

Chapelle, O.; Chi, M.; and Zien, A. 2006. A continuation method for semi-supervised svms. In *Proceedings of the 23rd international conference on Machine learning*, 185–192. ACM.

Duchi, J., and Singer, Y. 2009. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research* 10(Dec):2899–2934.

Feng, Y.; Huang, X.; Shi, L.; Yang, Y.; and Suykens, J. A. 2015. Learning with the maximum correntropy criterion induced losses for regression. *Journal of Machine Learning Research* 16:993–1034.

Ghadimi, S., and Lan, G. 2016. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming* 156(1-2):59–99.

Grave, E.; Obozinski, G. R.; and Bach, F. R. 2011. Trace lasso: a trace norm regularization for correlated designs. In *Advances in Neural Information Processing Systems*, 2187–2195.

Halko, N.; Martinsson, P.-G.; and Tropp, J. A. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review* 53(2):217–288.

He, R.; Zheng, W.-S.; and Hu, B.-G. 2011. Maximum correntropy criterion for robust face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33(8):1561–1576.

Hsieh, C.-J., and Olsen, P. A. 2014. Nuclear norm minimization via active subspace selection. In *ICML*, 575–583.

Jain, P.; Meka, R.; and Dhillon, I. S. 2010. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*, 937–945.

Journée, M.; Nesterov, Y.; Richtárik, P.; and Sepulchre, R. 2010. Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research* 11(Feb):517–553.

Larsen, R. M. 1998. Lanczos bidiagonalization with partial reorthogonalization. *DAIMI Report Series* 27(537).

Li, H., and Lin, Z. 2015. Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems*, 379–387.

Lin, Q.; Lu, Z.; and Xiao, L. 2014. An accelerated proximal coordinate gradient method. In *Advances in Neural Information Processing Systems*, 3059–3067.

Schmidt, M.; Le Roux, N.; and Bach, F. 2011. Convergence rates of inexact proximal-gradient methods for convex optimization. *arXiv preprint arXiv:1109.2415*.

Tappenden, R.; Richtárik, P.; and Gondzio, J. 2016. Inexact coordinate descent: complexity and preconditioning. *Journal of Optimization Theory and Applications* 144–176.

Villa, S.; Salzo, S.; Baldassarre, L.; and Verri, A. 2013. Accelerated and inexact forward-backward algorithms. *SIAM Journal on Optimization* 23(3):1607–1633.

Wood, G., and Zhang, B. 1996. Estimation of the lipschitz constant of a function. *Journal of Global Optimization* 8(1):91–103.

Xiao, L., and Zhang, T. 2014. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization* 24(4):2057–2075.

Zhang, T. 2010. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research* 11(Mar):1081–1107.

Zhong, L. W., and Kwok, J. T. 2012. Efficient sparse modeling with automatic feature grouping. *IEEE transactions on neural networks and learning systems* 23(9):1436–1447.