Cubic Regularized ADMM with Convergence to a Local Minimum in Non-convex Optimization

Zai Shi[1] and Atilla Eryilmaz[1]

[1]The Ohio State University, Columbus, OH 43210, USA
[1]Email:{shi.960, eryilmaz.2}@osu.edu

***Abstract*—How to escape saddle points is a critical issue in non-convex optimization. Previous methods on this issue mainly assume that the objective function is Hessian-Lipschitz, which leave a gap for applications using non-Hessian-Lipschitz functions. In this paper, we propose Cubic Regularized Alternating Direction Method of Multipliers (CR-ADMM) to escape saddle points of separable non-convex functions containing a non-Hessian-Lipschitz component. By carefully choosing a parameter, we prove that CR-ADMM converges to a local minimum of the original function with a rate of $O(1/T^{1/3})$ in time horizon T, which is faster than gradient-based methods. We also show that when one or more steps of CR-ADMM are not solved exactly, CR-ADMM can converge to a neighborhood of the local minimum. Through the experiments of matrix factorization problems, CR-ADMM is shown to have a faster rate and a lower optimality gap compared with other gradient-based methods. Our approach can also find applications in other scenarios where regularized non-convex cost minimization is performed, such as parameter optimization of deep neural networks.**

I. Introduction

Non-convex optimization, which minimizes a non-convex objective function, draws more and more attention in machine learning community due to its wide application, such as matrix completion, tensor decomposition, phase retrieval, and deep learning [17]. In general, non-convex optimization is NP-hard [17]. First order stationary points can be global minima, local minima, local maxima and saddle points. Fortunately in most cases, we are satisfied with the local minima of the original problem. For example, in problems like matrix completion [12], tensor decomposition [11], and phase retrieval [27], all local minima are global minima. In deep neural networks, it is found that the main bottleneck in optimization is not due to local minima, but the existence of many saddle points [8]. Therefore, we are more interested in the methods with convergence to local minima for non-convex optimization problems.

The main focus of this paper is to obtain a local minimum of non-convex problems with the following form:

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

where $h(x)$ is non-convex and $f(x)$, $g(x)$ possess different properties that will be explained in more details later. In machine learning practice, this form of problems is very common, such as a case where $f(x)$ is a loss function and $g(x)$ is a regularization term. Broadly speaking, this problem can be attacked in one of two ways.

The first one is to regard $h(x)$ as one function and directly apply a certain method with convergence to local minima. To the best of our knowledge, previous works on this kind of methods mostly assume that $h(x)$ is Hessian-Lipschitz (See details in Section I-A). So when $g(x)$ or $f(x)$ has no such property, there exists no theoretical guarantee of reaching a local minimum using these methods. Recently, Huang et al. [15] proposed a Perturbed Proximal Descent method to escape saddle points for non-convex and non-smooth functions, but its convergence is only guaranteed with a certain probability and one of their assumptions (Assumption 2 in [15]) is hard to check.

The second way to solve (1) is to reformulate it as

$$\begin{aligned}&\min_{x,y} f(x) + g(y)\\ &s.t. \quad x = y\end{aligned}$$

and then apply ADMM [5] to it. Unfortunately, most previous non-convex ADMM methods only converge to the first-order stationary points of the original problem (See details in Section I-A). Meanwhile, Hong et al. [13] proposed the gradient ADMM algorithm which can converge to second-order stationary points with probability one. But they assume that both f and g are Hessian-Lipschitz and that the initial point must be chosen randomly.

We can see that the Hessian-Lipschitz assumption is essential for most of the existing methods. However, this property does not hold in many scenarios whereby either the loss function (e.g., huber loss [16]) or the regularizer (e.g., l_1 regularizer for LASSO [28]) is not twice differentiable everywhere. The reader may refer to Section VI for a specific example.

In this paper, we use the second approach to obtain a local minima of (1) when f or g is not Hessian-Lipschitz. The contributions of the paper are summarized as follows:

- We propose a method called Cubic Regularized ADMM (CR-ADMM) to get a local minimum of (1) globally (i.e., regardless of where the initial point is) without the assumption that f and g are all Hessian-Lipschitz. The algorithm is shown in Section III and its convergence result is proved in Section IV.
- When CR-ADMM is implemented in practice, one or more steps may not be solved exactly. In this case we

This paper is funded by the NSF grant CMMI-SMOR-1562065, CNS-NeTS-1514260, CNS-NeTS-1717045, CNS-ICN-WEN-1719371, and CNS-SpecEES-1824337; ONR Grant N00014-19-1-2621; and the DTRA grant HDTRA1-18-1-0050.

prove that CR-ADMM can converge to a neighborhood of a local minimum under mild conditions in Section V. We also show that the convergence rate of CR-ADMM is in the same order of Nesterov's cubic regularization method [23].
- Through the experiments of matrix factorization problems in Section VI, we demonstrate the advantages of CR-ADMM compared with other gradient-base methods, including its faster rate and smaller optimality gap.

A. Related Works

For general non-convex optimization problems, there are mainly three types of methods to escape saddle points: Hessian-based, Hessian-vector-product-based and gradient-based. We cannot detail all the works due to the fast development of this area, so we only introduce the most classical and related ones to our paper.

Hessian-based methods are the most natural way since strict saddle points can be distinguishable using Hessian information. Cubic regularization method [23] and trust region method [9] are two typical Hessian-based methods requiring the computation of the inverse of the full Hessian per iteration, which can have a large cost.

In order to reduce the cost, a number of papers explore the methods taking advantages of Hessian-vector product instead of the inverse of Hessian. Since the Hessian-vector product $\nabla^2 f(x)v$ can be approximated by $\frac{\nabla f(x+\delta v)-\nabla f(x-\delta v)}{2\delta}$, this kind of methods is sometimes regarded as first-order methods. Among them, Agarwal et al. [3] and Carmon et al. [6] proposed different Hessian-vector-product-based subroutines to solve the subproblem of the cubic regularization method [23]. For the purpose of acceleration, Carmon et al. [7] and Xu et al. [31] combined the accelerated gradient method [24] and negative curvature exploitation to get a faster rate to reach a local minimum.

Gradient-based methods for escaping saddle points is another hot topic because of their low complexity. The basic principle behind these methods is to add perturbation to gradient descent (GD) so that the point can find a descent direction around the saddle point with high probability. Ge et al. [11] first proposed a gradient-based method called noisy gradient descent (NGD) by adding noise to each iteration of gradient descent. Afterwards Levy [20] improved the rate of NGD by normalizing the gradient. Jin et al. [18] achieved a better rate by adding noise to GD periodically only when the gradient is below some threshold. Jin et al. [19] then proved that adding noise to accelerated gradient descent [24] can escape saddle points faster.

It must be noted that the above methods all require the objective function to be Hessian-Lipschitz. Meanwhile, by adding noise to proximal gradient descent, Perturbed Proximal Descent [15] proposed by Huang et al. can escape saddle points with a high probability, which has a similar setting to ours but a more complex assumption .

Since our method is related to ADMM, we briefly introduce previous works on non-convex ADMM. For non-convex non-smooth optimization problems, Wang et al. [30] and Liu et al. [21] proposed two different ADMM methods that converge to a first-order stationary point. If both f and g are Hessian-Lipschitz and the initial point is chosen randomly, the gradient ADMM method [13] proposed by Hong et al. can get a second order stationary point with probability one.

II. Preliminaries

A. Notations

We use upper-case letters X to denote a matrix and lower-case letters x to denote a vector. Particularly, I is an identity matrix. $X(i,j)$ is the (i,j)th entry of X and X^T is the transpose of X. We use $||\cdot||$ to denote l_2 norm for vectors and spectral norm for matrices. $||\cdot||_F$ is Frobenius norm for matrices. $\lambda_{\min}(X)$ means the smallest eigenvalue of X. $\langle\cdot,\cdot\rangle$ denotes a vector inner product. ∇f and $\nabla^2 f$ are the gradient and Hessian of the function f respectively. For a multivariate function f, $\partial_x f|_{(x_t,y_t,z_t)}$ and $\partial_x^2 f|_{(x_t,y_t,z_t)}$ denote the first-order and the second-order partial derivatives of f with respect to x at (x_t,y_t,z_t).

B. Definitions

In this subsection we give some definitions that will be used in our paper.

Definition 1. *An extended value function $f:\mathbb{R}^n\to\mathbb{R}\cup\{-\infty,\infty\}$ is called coercive if $f(x)\to\infty$ as $||x||\to\infty$*

Definition 2. *A differentiable function f is L-smooth if for any x and y*

$$||\nabla f(x)-\nabla f(y)||\le L||x-y||$$

Definition 3. *A twice differentiable function f is ρ-Hessian Lipschitz if for any x and y*

$$||\nabla^2 f(x)-\nabla^2 f(y)||\le \rho||x-y||$$

Definition 4. *x is a first-order stationary point of a function f if $\nabla f(x)=0$. y is a second order stationary point of a function f if $\nabla f(y)=0$ and $\nabla^2 f(y)$ is positive semidefinite. z is a saddle point of f if z is a first-order stationary point of f but not a local minimum of f.*

In this definition, a saddle point can be a local maximum. Meanwhile, a second order stationary point can still be a saddle point where $\lambda_{\min}(\nabla^2 f(x))=0$. Here we define a strict saddle point as

Definition 5. *x is a strict saddle point of f if it is a first-order stationary point of f and $\lambda_{\min}(\nabla^2 f(x))<0$.*

It is easy to see that all the second order stationary points will be local minima if all the saddle points are strict.

III. Problem Setup and Algorithm

We consider an optimization problem with the following form:

$$\min_{x\in\mathbb{R}^n} h(x) := f(x)+g(x). \tag{2}$$

Algorithm 1 Cubic Regularized ADMM

1: Initialize $x_0, y_0, \gamma_0, \beta, T$.
2: **for** $t = 0$ to $T-1$ **do**
3: $\quad x_{t+1} = \arg\min_x \bar{f}_t(x)$
4: $\quad y_{t+1} = \arg\min_y \bar{g}_t(y)$
5: $\quad \gamma_{t+1} = \gamma_t + \beta(x_{t+1} - y_{t+1})$
6: **end for**

This form incorporates many problems that appear in machine learning community. In these problems, $f(x)$ is a loss function of a regression problem or a representation function formed by a neural network, and $g(x)$ is a regularization term. Our aim is to find a local minimum of (2) because in many tasks, obtaining a local minimum is a satisfying result as mentioned in Section I. Meanwhile, we have the following assumptions for (2):

Assumption 1. *$f(x) + g(x)$ is lower-bounded and coercive;*

Assumption 2. *$f(x)$ is ρ-Hessian Lipschitz, probably non-convex;*

Assumption 3. *$g(x)$ is L_g-smooth and convex. Meanwhile, $g(x)$ is twice differentiable in some neighborhood of the first-order stationary points of $h(x)$;*

Assumption 4. *$h(x)$ is ζ-strict saddle, i.e., all its saddle points satisfy $\lambda_{\min}(\nabla^2 h(x)) < -\zeta$ for some $\zeta > 0$.*

Among these assumptions, Assumption 1-3 are common ones in optimization literature. Assumption 4 states that the saddle points of h are all strict saddle points, which means the second order stationary points are the local minima of h. Many problems we are interested in satisfy this property [26]. In fact, previous methods to escape saddle points mentioned in Section I-A assume this property, so that Hessian information can distinguish saddle points from local minima or gradient descent with noise can find a descent direction around saddle points in a probability related to ζ.

The separation of h as f and g is also flexible to satisfy these assumptions. For example, if g is non-convex and $\frac{L_g}{2}$-smooth, we can reset $\tilde{f}(x) = f(x) - \frac{L_g}{4}||x - x'||^2$ and $\tilde{g}(x) = g(x) + \frac{L_g}{4}||x - x'||^2$ for some x'. Then $h = \tilde{f} + \tilde{g}$ with Assumption 3 satisfied.

Previous methods mentioned in Section I-A cannot be directly applied to $h(x)$ (at least without theoretical guarantees) when $g(x)$ is not Hessian Lipschitz. To overcome this shortcoming, we proposed a new method called Cubic Regularized ADMM which can converge to a local minimum of (2).

First, we transfer (2) to the equivalent problem with the following form:

$$\begin{aligned} &\min_{x,y} f(x) + g(y). \\ &s.t. \quad x = y \end{aligned} \tag{3}$$

Now we present CR-ADMM in Algorithm 1 based on (3) and its augmented Lagrangian function

$$\mathcal{L}_\beta(x, y, \gamma) = f(x) + g(y) + \gamma^T(x - y) + \frac{\beta}{2}||x - y||^2. \tag{4}$$

where $\gamma \in \mathbb{R}^n$ is the multiplier and $\beta > 0$ is a constant. In this algorithm,

$$\begin{aligned} \bar{f}_t(x) &= \mathcal{L}_\beta(x_t, y_t, \gamma_t) + \langle \nabla f(x_t) + \gamma_t + \beta(x_t - y_t), x - x_t \rangle \\ &+ \frac{1}{2}\left\langle (\nabla^2 f(x_t) + \beta I)(x - x_t), x - x_t \right\rangle + \frac{\rho}{6}||x - x_t||^3 \\ \bar{g}_t(y) &= f(x_{t+1}) + g(y) + \gamma_t^T(x_{t+1} - y) + \frac{\beta}{2}||x_{t+1} - y||^2 \end{aligned}$$

In fact, $\bar{f}_t(x)$ is the upper second order approximation of $\mathcal{L}_\beta(x, y_t, \gamma_t)$ because f is ρ-Hessian Lipschitz [23], and $\bar{g}_t(y) = L_\beta(x_{t+1}, y, \gamma_t)$.

In Step 3, we need to find the global minimum of $\bar{f}_t(x)$. The methods for implementing this step are well developed by several papers. In [23], the authors transformed this step into one-dimensional equation which can be solved by a technique for the needs of trust region methods. In [6], gradient descent is demonstrated to be an efficient method to find the approximate global minimum with high probability. Besides, Agarwal et al. [3] showed an accelerated method by applying fast approximate matrix inverse and eigenvector computations. It is noted that the later two methods only need to compute Hessian-vector products $\nabla^2 f(x)v$.

In Step 4, since $\bar{g}_t(y)$ is strongly convex for $\beta > 0$, we only need to find the first-order stationary point of $\bar{g}_t(y)$, which satisfies,

$$\nabla g(y_{t+1}) - \gamma_t - \beta(x_{t+1} - y_{t+1}) = 0. \tag{5}$$

In practice, we prefer to choose $g(y)$ that makes (5) easily solved. For machine learning problems, we often regard the regularizer as $g(y)$ since it often takes a simple form. If (5) cannot be solved directly, we can use the gradient descent method as a subsolver.

The novelty of CR-ADMM is that by utilizing ADMM, we can tackle the different properties of two components with their suitable methods. Meanwhile, its performance is comparable to Nesterov's cubic regularization method [23], which will be analyzed in the following two sections.

IV. Convergence Analysis

First we give the convergence result of CR-ADMM by the following theorem. Then we prove this theorem by showing that each step of CR-ADMM gives the augmented Lagrangian function (4) a sufficient descent.

Theorem 1. *If Assumption 1, 2, 3 and 4 are satisfied with $2L_g < \zeta$, then for CR-ADMM with $2L_g < \beta < \zeta$, $\{(x_t, y_t, \gamma_t)\}_t$ is a bounded sequence and x_t converges to a local minimum of* (2)*.*

Remark 1. *Sometimes the requirement $2L_g < \zeta$ needs not be strictly satisfied for the convergence to a local minimum. The experiment of Figure 3a in Section VI is such an example and the reason will be explained in the corresponding part.*

To prove this theorem, we will present three lemmas related to Step 3, 4 and 5 of our algorithm. Each lemma gives a bound of the descent of (4) in the corresponding step.

Lemma 1. *For Step 3 of CR-ADMM, we have*

$$\mathcal{L}_\beta(x_t, y_t, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_t, \gamma_t) \geq \frac{\rho}{12}||x_{t+1} - x_t||^3$$

Proof. It directly comes from Lemma 4 of [23] since f is ρ-Hessian Lipschitz. □

Lemma 2. *For Step 4 of CR-ADMM, we have*

$$\begin{aligned}&\mathcal{L}_\beta(x_{t+1}, y_t, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_{t+1}, \gamma_t)\\ &\geq \frac{\beta - L_g}{2}||y_{t+1} - y_t||^2.\end{aligned}$$

Proof. See Proof of Lemma 2 in [1]. □

Lemma 3. *For Step 5 of CR-ADMM, we have*

$$\begin{aligned}&\mathcal{L}_\beta(x_{t+1}, y_{t+1}, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_{t+1}, \gamma_{t+1})\\ &\geq -\frac{L_g^2}{\beta}||y_{t+1} - y_t||^2\end{aligned}$$

Proof. See Proof of Lemma 3 in [1]. □

With these lemmas, we can turn to the proof of Theorem 1.

Proof. First we prove that the sequence $\{\mathcal{L}_\beta(x_t, y_t, \gamma_t)\}_t$ is lower-bounded:

$$\begin{aligned}&\mathcal{L}_\beta(x_t, y_t, \gamma_t)\\ &= f(x_t) + g(y_t) + \gamma_t^T(x_t - y_t) + \frac{\beta}{2}||x_t - y_t||^2\\ &= f(x_t) + g(y_t) + \nabla g(y_t)^T(x_t - y_t) + \frac{\beta}{2}||x_t - y_t||^2 \quad (6)\\ &\geq f(x_t) + g(x_t) + \frac{\beta - L_g}{2}||x_t - y_t||^2 \quad (7)\end{aligned}$$

where (6) is from Proof of Lemma 3, (7) is from L_g-smoothness of g (Lemma 2 of [21]). (7) is lower-bounded because of Assumption 1 and $\beta > 2L_g$.

Denote the lower bound as $\mathcal{L}_\beta^*$. Based on Lemma 1, 2 and 3, we have

$$\begin{aligned}&\mathcal{L}_\beta(x_t, y_t, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_{t+1}, \gamma_{t+1})\\ &\geq \frac{\rho}{12}||x_{t+1} - x_t||^3 + (\frac{\beta - L_g}{2} - \frac{L_g^2}{\beta})||y_{t+1} - y_t||^2 \quad (8)\end{aligned}$$

When $\beta > 2L_g$, (8) is nonnegative, which means $\{\mathcal{L}_\beta(x_t, y_t, \gamma_t)\}_t$ is decreasing and upper-bounded by $\mathcal{L}_\beta(x_0, y_0, \gamma_0)$. Therefore, $f(x_t) + g(x_t) + \frac{\beta - L_g}{2}||x_t - y_t||^2$ is also upper-bounded by (7). From coerciveness of $f(x) + g(x)$ and Step 5 of CR-ADMM, $\{(x_t, y_t, \gamma_t)\}_t$ is a bounded sequence. Now,

$$\begin{aligned}&\mathcal{L}_\beta(x_0, y_0, \gamma_0) - \mathcal{L}_\beta^*\\ &\geq \sum_{t=0}^{\infty}\{\mathcal{L}_\beta(x_t, y_t, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_{t+1}, \gamma_{t+1})\}\end{aligned}$$

Then when $t \to \infty$, $||x_{t+1} - x_t||$ and $||y_{t+1} - y_t||$ will converge to 0 by (8). From Proof of Lemma 3 and L_g smoothness of g, we have

$$||\gamma_{t+1} - \gamma_t|| = ||\nabla g(y_{t+1}) - \nabla g(y_t)|| \leq L_g||y_{t+1} - y_t||$$

Then $||\gamma_{t+1} - \gamma_t||$ will also converge to 0. Define

$$p_t = \partial_x \mathcal{L}_\beta|_{(x_{t+1}, y_t, \gamma_t)} \quad (9)$$

$$q_t = \lambda_{\min}(\partial_x^2 \mathcal{L}_\beta|_{(x_{t+1}, y_t, \gamma_t)}) \quad (10)$$

We have

$$||x_{t+1} - x_t|| \geq \max\left\{\sqrt{\frac{1}{\rho}||p_t||}, -\frac{2}{3\rho}q_t\right\} \to 0 \quad (11)$$

$$||\partial_\gamma \mathcal{L}_\beta|_{(x_t, y_t, \gamma_t)}|| = ||x_t - y_t|| = \frac{1}{\beta}||\gamma_t - \gamma_{t-1}|| \to 0 \quad (12)$$

$$||\partial_y \mathcal{L}_\beta|_{(x_t, y_t, \gamma_t)}|| = ||\nabla g(y_t) - \gamma_t - \beta(x_t - y_t)|| \to 0 \quad (13)$$

where (11) is from Lemma 5 of [23], (12) is from Step 5 of CR-ADMM, and (13) is from Proof of Lemma 3 and (12). Denote (x^*, y^*, γ^*) as the limit point of $\{(x_t, y_t, \gamma_t)\}_t$. Then

$$0 = \partial_\gamma \mathcal{L}_\beta|_{(x^*, y^*, \gamma^*)} = x^* - y^* \quad (14)$$

$$0 = \partial_x \mathcal{L}_\beta|_{(x^*, y^*, \gamma^*)} = \nabla f(x^*) + \gamma^* + \beta(x^* - y^*) \quad (15)$$

$$0 \preceq \partial_x^2 \mathcal{L}_\beta|_{(x^*, y^*, \gamma^*)} = \nabla^2 f(x^*) + \beta I \quad (16)$$

$$0 = \partial_y \mathcal{L}_\beta|_{(x^*, y^*, \gamma^*)} = \nabla g(y^*) - \gamma^* - \beta(x^* - y^*) \quad (17)$$

From (14), (15) and (17), we know that x^* is the first-order stationary point of $h(x)$, i.e., $\nabla f(x^*) + \nabla g(x^*) = 0$. If x^* is a saddle point of $h(x)$, then $\lambda_{\min}(\nabla^2 h(x^*)) < -\zeta$ by Assumption 4. Since g is convex, we have $\lambda_{\min}(\nabla^2 f(x^*)) < -\zeta$ by Weyl's theorem [14]. Since $\beta < \zeta$, we have $\lambda_{\min}(\nabla^2 f(x^*) + \beta I) < 0$, which contradicts (16). Meanwhile, if x^* is a local minimum (i.e., $\nabla^2 f(x^*) + \nabla^2 g(x^*) \succeq 0$), (16) will be satisfied given that g is L_g-smooth (i.e., $\nabla^2 g(x^*) \preceq L_g I$) and $\beta > 2L_g$. So x^* is a local minimum of $h(x)$. □

From the proof, we can see that if Assumption 4 is not satisfied or ζ is unknown, CR-ADMM with $\beta = 2L_g + \delta$ can escape saddle points satisfying $\lambda_{\min}(\nabla^2 h(x)) < -2L_g - \delta$ for any $\delta > 0$. In machine learning practice, the coefficient of the regularizer is usually very small (like 0.25 in [22]), which leads to a small L_g if we treat the regularizer as g.

V. Practical Implementation Issues

In this section, we will discuss the performance of CR-ADMM when Step 3 or 4 is not solved exactly in the implementation of CR-ADMM. It often happens when we use a subsolver for Step 3 or 4, or we make a approximation of f or g. First we give our definition of an *ε-inexact output* for one iteration of CR-ADMM.

Definition 6. *(x_t, y_t, γ_t) is called an ε-inexact output for t-th iteration of CR-ADMM if $\mathcal{L}_\beta(x_t, y_t, \gamma_t)$ is **lower-bounded** by some value Λ and*

$$\begin{aligned}&\mathcal{L}_\beta(x_t, y_t, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_{t+1}, \gamma_{t+1}) \geq \frac{\rho}{12}||x_{t+1} - x_t||^3\\ &+ (\frac{\beta - L_g}{2} - \frac{L_g^2}{\beta})||y_{t+1} - y_t||^2 - \varepsilon \quad (18)\end{aligned}$$

for some $\varepsilon > 0$.

We can see that the above inequality is the same with (8) if $\varepsilon = 0$, which is satisfied for an exact output. We will show later that for a wide range of inexact subsolvers and approximations of CR-ADMM, their outputs satisfy the above definition. But first we present the convergence result of CR-ADMM when each iteration of CR-ADMM has an ε-inexact output.

Theorem 2. *If Assumption 1, 2, 3 and 4 are satisfied with $2L_g < \zeta$ and (x_t, y_t, γ_t) is an ε-inexact output for each t, then for CR-ADMM with $2L_g < \beta < \zeta$, we have the following bounds after T iterations:*

$$||\partial_\gamma \mathcal{L}_\beta|_{(x_{t'}, y_{t'-1}, \gamma_{t'-1})}|| \le \frac{c_1(L_g+\beta)}{\beta}\Pi^{\frac{1}{2}}$$
$$||\partial_y \mathcal{L}_\beta|_{(x_{t'}, y_{t'-1}, \gamma_{t'-1})}|| \le c_1(L_g+\beta)\Pi^{\frac{1}{2}}$$
$$\max\left\{\sqrt{\frac{1}{\rho}||p_{t'-1}||}, -\frac{2}{3\rho}q_{t'-1}\right\} \le c_2\Pi^{\frac{1}{3}},$$

where p_t is defined in (9)*, q_t is defined in* (10)*, and*

$$t' = \arg\min_{1\le t\le T}\theta_t,$$
$$\theta_t = \frac{\rho}{12}||x_t - x_{t-1}||^3 + (\frac{\beta - L_g}{2} - \frac{L_g^2}{\beta})||y_t - y_{t-1}||^2,$$
$$c_1 = \sqrt{\frac{1}{\frac{\beta-L_g}{2} - \frac{L_g^2}{\beta}}},\ c_2 = \sqrt[3]{\frac{12}{\rho}},$$
$$\Pi = \frac{\varepsilon T + (\mathcal{L}_\beta(x_0, y_0, \gamma_0) - \Lambda)}{T}.$$

Proof. See Proof of Theorem 2 in [1] □

From (14), (15), (16), (17) and the above theorem, we know that $x_{t'}$ or $y_{t'-1}$ is in a small neighborhood of a local minimum of $h(x)$ when ε sufficiently small and T sufficiently large. If $\varepsilon = 0$, CR-ADMM will reach a local minimum when $T \to \infty$, as proved in Theorem 1. In this case, $\Pi = O(1/T)$ and we can get the convergence rate of CR-ADMM as follows.

Corollary 1. *If Assumption 1, 2, 3 and 4 are satisfied with $2L_g < \zeta$, then for CR-ADMM with $2L_g < \beta < \zeta$, after T iterations, we have*

$$||\partial_\gamma \mathcal{L}_\beta|_{(x_{t'}, y_{t'-1}, \gamma_{t'-1})}|| = O(1/T^{1/2})$$
$$||\partial_y \mathcal{L}_\beta|_{(x_{t'}, y_{t'-1}, \gamma_{t'-1})}|| = O(1/T^{1/2})$$
$$\max\left\{\sqrt{\frac{1}{\rho}||p_{t'-1}||}, -\frac{2}{3\rho}q_{t'-1}\right\} = O(1/T^{1/3})$$

with the same notations in Theorem 2.

The $O(1/T^{1/3})$ rate is in the same order with Nesterov's cubic regularization method (please refer to Theorem 1 of [23]), so CR-ADMM conserves the advantages of cubic regularization, which has a faster rate than gradient-based methods [19] .

Now we present some cases where Definition 6 is satisfied.

A. Gradient Descent as a Subsolver

As mentioned in Section III, Step 3 or 4 can be solved via gradient descent. If this subsolver is running for a finite time horizon, we may get an inexact output for the corresponding step. Take Step 3 for example. Carmon et al. [6] proved that gradient descent approximates the global minimum of $\bar{f}_t$ to within ε accuracy in $O(\log(1/\varepsilon))$ steps for small ε. If $\bar{f}_t(x_{t+1}) - \bar{f}_t(x^*_{t+1}) < \varepsilon$ where x_{t+1} is the output of this subslover for Step 3 and x^*_{t+1} is the global minimum of $\bar{f}_t$, then

$$\begin{aligned}
&\mathcal{L}_\beta(x_t, y_t, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_t, \gamma_t)\\
&\ge \mathcal{L}_\beta(x_t, y_t, \gamma_t) - \bar{f}_t(x_{t+1}) \qquad (19)\\
&= \mathcal{L}_\beta(x_t, y_t, \gamma_t) - \bar{f}_t(x^*_{t+1}) + \bar{f}_t(x^*_{t+1}) - \bar{f}_t(x_{t+1})\\
&\ge \frac{\rho}{12}||x_{t+1} - x_t||^3 - \varepsilon \qquad (20)
\end{aligned}$$

where (19) is from the fact that $\bar{f}_t(x_{t+1})$ is the upper second order approximation of $\mathcal{L}_\beta(x_{t+1}, y_t, \gamma_t)$ and (20) is from Lemma 4 of [23]. Meanwhile, the inequalities in Lemma 2 and Lemma 3 remain intact. Then we can check that (18) is satisfied. Using the same proof in (7), we can check that $\mathcal{L}_\beta(x_t, y_t, \gamma_t)$ is lower-bounded. Then Definition 6 can be applied to this case.

B. Stochastic Approximation of $\bar{f}_t$

In machine learning practice, we often encounter the problem where $f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x; \xi_i)$ or $f(x) = \mathbb{E}_{\xi\sim\mathcal{D}}[f(x;\xi)]$ for some distribution $\mathcal{D}$. The former is often called empirical risk and the latter called population risk [4]. In this case, we can use stochastic cubic regularization method proposed in [29] to reduce computation cost, where $\bar{f}_t(x)$ is replaced by

$$\begin{aligned}
\tilde{f}_t(x) &= \mathcal{L}_\beta(x_t, y_t, \gamma_t) + \langle d_t + \gamma_t + \beta(x_t - y_t), x - x_t\rangle\\
&+ \frac{1}{2}\langle (B_t + \beta I)(x - x_t), x - x_t\rangle + \frac{\rho}{6}||x - x_t||^3
\end{aligned}$$

where $d_t = \frac{1}{|S_1|}\sum_{\xi_i\in S_1}\nabla f_i(x;\xi_i)$, $B_t = \frac{1}{|S_2|}\sum_{\xi_i\in S_2}\nabla^2 f_i(x;\xi_i)$, and S_1, S_2 are two independent minibatch samples from data points at each iteration. It can be proved that when $|S_1|$, $|S_2|$ are sufficiently large, (20) holds in high probability with small ε by a similar proof in Section 4.1 of [29]. Then same with Section V-A, we can prove that Definition 6 is satisfied for this case in high probability.

C. Smooth Approximation of g

In some applications such as LASSO [28], the regularizer (often regarded as g) is non-smooth. In this case, we can use a smooth approximation of g to run our algorithm. We call a non-smooth function $g(x)$ (a, b)-*smoothable*, if we can find a smooth approximation $g_\mu(x)$ with the following properties:

$$g_\mu(x) \le g(x) \le g_\mu(x) + b\mu, \forall x; \qquad (21)$$
$$g_\mu(x) \text{ is } \frac{a}{\mu}\text{-smooth.} \qquad (22)$$

For example, for l_1 regularizer $g(x) = \sum_{i=1}^n |x_i|$, we can approximate it by $g_\mu(x) = \sum_{i=1}^n l(x_i)$ where $l(\cdot)$ is called a huber function [16] formulated as

$$l(x_i) = \begin{cases} x_i^2/(2\mu) & \text{if } |x_i| < \mu \\ |x_i| - \mu/2 & \text{else} \end{cases} \tag{23}$$

For this approximation, we have $g_\mu(x) \le g(x) \le g_\mu(x) + n\mu/2$ and g_μ is $\frac{1}{\mu}$-smooth. So l_1 regularizer is $(1, n/2)$-smoothable. The readers may refer to [2] for more examples.

When g is (a, b)-smoothable and g_μ is convex, we can replace $g(y)$ with $g_\mu(y)$ in Step 4 of CR-ADMM. Then Lemma 2 changes to

$$\begin{aligned} &\mathcal{L}_\beta(x_{t+1}, y_t, \gamma_t) - \mathcal{L}_\beta(x_{t+1}, y_{t+1}, \gamma_t) \\ &= g(y_t) - g(y_{t+1}) - \langle \gamma_t, y_t - y_{t+1} \rangle \\ &- \langle y_t - y_{t+1}, \beta(x_{t+1} - y_{t+1}) \rangle + \frac{\beta}{2} ||y_{t+1} - y_t||^2 \\ &\ge g_\mu(y_t) - g_\mu(y_{t+1}) - b\mu - \langle \nabla g_\mu(y_{t+1}), y_t - y_{t+1} \rangle \\ &+ \frac{\beta}{2} ||y_{t+1} - y_t||^2 \qquad (24) \\ &\ge \frac{\beta - a/\mu}{2} ||y_{t+1} - y_t||^2 - b\mu. \qquad (25) \end{aligned}$$

where (24) is from (21) and $\nabla g_\mu(y_{t+1}) = \gamma_t + \beta(x_{t+1} - y_{t+1})$, and (25) is from (22). Meanwhile, Lemma 1 and Lemma 3 remain the same with $L_g = a/\mu$. So (18) is satisfied with $\varepsilon = b\mu$. $f(x_t) + g_\mu(y_t) + \gamma_t^T(x_t - y_t) + \frac{\beta}{2}||x_t - y_t||^2$ is lower-bounded with the same argument of (7), so with (21) we know that $f(x_t) + g(y_t) + \gamma_t^T(x_t - y_t) + \frac{\beta}{2}||x_t - y_t||^2$ is also lower-bounded. Then Definition 6 is satisfied.

VI. Experiments

In this section, we use the example of symmetric matrix factorization to demonstrate the advantages of our algorithm. Symmetric matrix factorization is a standard technique in clustering by providing low-rank decompositions of matrices [10]. With an l_1 regularizer for the purpose of regularization [25], the problem can be formulated as:

$$\min_{X \in \mathbb{R}^{n \times k}} \frac{1}{2} ||XX^T - Z||_F^2 + \lambda \sum_{i=1}^n \sum_{j=1}^k |X(i,j)| \tag{26}$$

where $Z \in \mathbb{R}^{n \times n}$ is a symmetric matrix. We denote the above objective function as $h(X)$. Meanwhile we regard $\frac{1}{2}||XX^T - Z||_F^2$ as $f(X)$ and $\lambda \sum_{i=1}^n \sum_{j=1}^k |X(i,j)|$ as $g(X)$. Since it is not easy to identify saddle points in large scale matrix factorization, we first consider a low dimensional case, where the saddle points can be shown in a 3-dimensional plot. Particularly, we let $X \in \mathbb{R}^2$, and $Z \in \mathbb{R}^{2 \times 2}$ is a matrix with all entries equal to 1. Additionally, we set $\lambda = 0.1$.

Before we use our algorithm, we need to smoothen $g(X)$. As mentioned in Section V-C, we can use the huber function defined in (23) as a smooth approximation and then (26) becomes

$$\min_{X \in \mathbb{R}^{n \times k}} \frac{1}{2} ||XX^T - Z||_F^2 + \lambda \sum_{i=1}^n \sum_{j=1}^k l(X(i,j)) \tag{27}$$

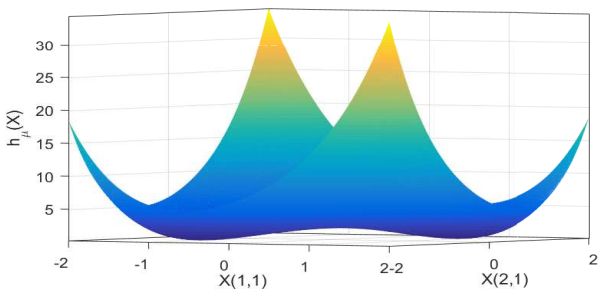
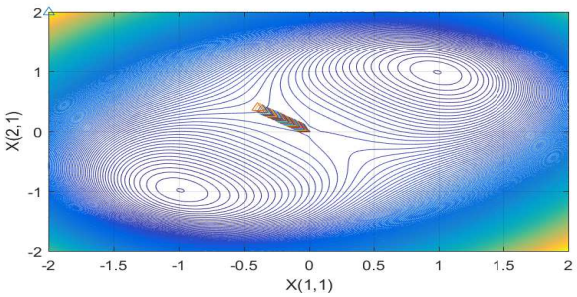


Fig. 1: 3-dimensional plot of $h_\mu(X)$ with $\mu = 0.01$

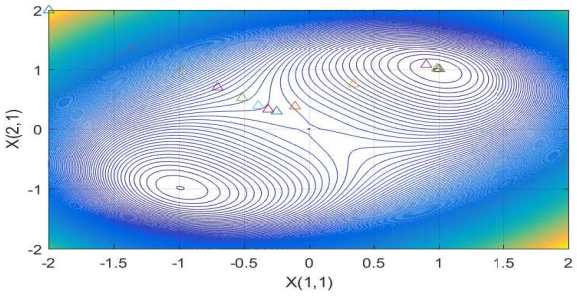
Now we define $h_\mu(X)$ as the above objective function and $g_\mu(X) = \lambda \sum_{i=1}^n \sum_{j=1}^k l(X(i,j))$. For $h_\mu(X)$, it is easy to see that Assumption 1 is satisfied. For Assumption 2, we know that $f(X)$ is $12\Gamma^{\frac{1}{2}}$-Hessian Lipschitz inside the region $\{X | ||X||^2 < \Gamma\}$ when Γ is sufficiently large by Lemma 6 of [18]. Intuitively, Step 3 of our algorithm will constrict $\{X_t\}_t$ inside a certain area such that $||X_t||^2 < \Gamma$ for each t. Since it is not the focus of the paper, we omit the detailed proof. Figure 2 presented later will also justify this claim. It is easy to check Assumption 3 for $g_\mu(X)$. To check whether Assumption 4 are satisfied, we need to identify the local minima and saddle points of $h_\mu(X)$. Figure 1 shows the 3-dimensional plot of $h_\mu(X)$ where g is approximated by the huber function (23) with $\mu = 0.01$. We can see that there exists two local minima $[0.98, 0.98]^T, [-0.98, -0.98]^T$ and one saddle point $[0, 0]^T$. With careful calculations, $h_\mu(X)$ is ζ-strict with $\zeta = 3.8$.

In the following part, four algorithms will be used to solve (27): gradient descent (GD), noisy gradient descent (NGD) [11], normalized noisy gradient descent (NNGD) [20] along with our proposed CR-ADMM. Even though there is no theoretical guarantee of escaping saddle points when h_μ is not Hessian-Lipschitz, we still use NGD and NNGD as a benchmark because of their low complexity. Meanwhile, since $g(X)$ and $g_\mu(X)$ are neither twice differentiable everywhere, we cannot use Hessian-based or Hessian-vector-product-based methods. We choose $X_0 = [-2, 2]^T$ as the initial point for all these four methods. Constant stepsizes are adopted for GD, NGD and NNGD. The parameters (such as β in CR-ADMM, stepsize in gradient descent) of each method are chosen to achieve its best performance. For CR-ADMM, we run the method proposed in [6] as a subsolver to solve Step 3, so we only need the gradient information in CR-ADMM. While two variables X and Y are used in CR-ADMM, we only choose the X value to plot the figures.

First, we plot the trajectories of CR-ADMM and GD in the contour lines of $h_\mu(X)$ with $\mu = 0.01$ to compare their behaviors around the saddle point in Figure 2. We can see that for GD, $\{X_t\}_t$ is trapped at the saddle point, while for CR-ADMM, $\{X_t\}_t$ escapes the saddle point with few iterations. Additionally, because of low gradient values, GD makes $\{X_t\}_t$ move very slowly near the saddle point. CR-ADMM avoids this situation by utilizing the cubic regularization.

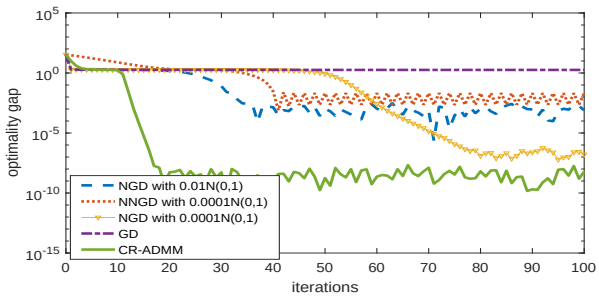


(a) GD in the contour lines of $h_\mu(X)$ with $\mu = 0.01$

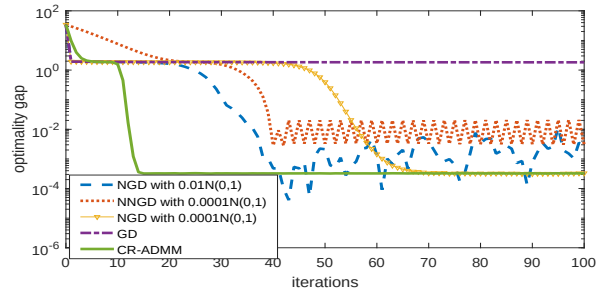


(b) CR-ADMM in the contour lines of $h_\mu(X)$ with $\mu = 0.01$

Fig. 2: Comparison of GD and CR-ADMM around the saddle point



(a) μ=0.01



(b) μ=2

Fig. 3: Optimality gap v.s. iterations for GD, NGD, NNGD and CR-ADMM applied to $h_\mu(X)$ in low dimension

In Figure 3 we compare the rates of CR-ADMM, GD, NGD and NNGD applied to (27). For NGD, we run it two times with different noise levels: $0.01\mathcal{N}(0,1)$ and $0.0001\mathcal{N}(0,1)$, where $\mathcal{N}(0,1)$ is a standard normal random variable. For NNGD, the noise is $0.0001\mathcal{N}(0,1)$. Meanwhile, Figure 3a and 3b correspond to $h_\mu(X)$ with $\mu = 0.01$ and 2 respectively. The optimality gap is defined as $h(X_t) - h(X^*)$ where $h(X^*)$ is the value of the global minimum of (26).

For $\mu = 0.01$, CR-ADMM has the fastest rate and the lowest final optimality gap. Interestingly, $2L_g < \zeta$ is not satisfied as required in Theorem 2 in this case ($\zeta = 3.8$ and $L_g = 10$), but CR-ADMM still convergences to a small neighborhood of a local minimum. It is because in the most area of the trajectory of CR-ADMM, L_g is actually 0 where g_μ is locally a linear function. L_g is large only when any entry of X is close to 0. It happens only in few iterations of CR-ADMM with little impact. For GD, the iterations are trapped at the saddle point of $h_\mu(X)$, resulting in the largest optimality gap. For NGD, we can see that a greater noise level will help escape the saddle point more quickly, but meanwhile yield a greater optimality gap. For NNGD, we can see that it escapes the saddle point more quickly than NGD with the same noise level. But it is unstable with a large optimality gap.

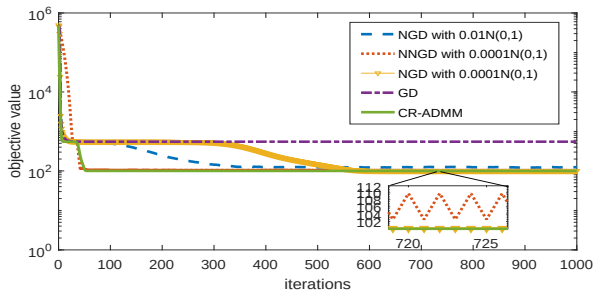
For $\mu = 2$, the final optimality gaps are the same for CR-ADMM and NGD. In fact, as we can see in Figure 2, the trajectories of the algorithms are within the region $\{X||X(1,1)| \leq 2, |X(1,2)| \leq 2\}$. In this region, $h_\mu(X)$ is Hessian Lipschitz when $\mu = 2$, and NGD is proved to have a good performance [11]. Meanwhile, CR-ADMM is still the fastest to achieve the final gap. There is little change for GD and NNGD.

Now we experiment CR-ADMM when Z in (26) has a high dimension. Particularly, we set Z as a 100×100 matrix of rank 10. We aim to find a 100×10 matrix X to minimize $h(X)$ in (26). Similar to the proof of Lemma 7 in [18], we can show that $h(X)$ has multiple local minima and strict saddle points. Same with the experiments in Figure 3, we first smoothen $g(X)$ and then apply CR-ADMM, GD, NNGD and NGD with two noise levels to (27). All the parameters in these algorithms are chosen to achieve their best performance. Meanwhile, the initial point of X is the same for all these algorithms, which is a 100×10 matrix with all entries equal to 1. Since we do not know the local minima they may reach, we use the value of $h(X_t)$ as the performance metric.

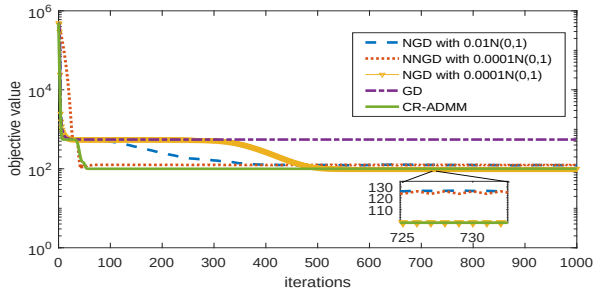
In Figure 4a and 4b, we plot the rates of the algorithms applied to $h_\mu(X)$ with $\mu = 0.01$ and $\mu = 1$. Similar phenomena can be observed compared with Figure 3, except NNGD has a comparable rate with CR-ADMM. But NNGD is not stable and has a larger objective value than CR-ADMM. Note that since the values are in the level of 10^2, the differences are not as obvious as Figure 3, but they are actually much larger than the previous low dimensional case. We can see that CR-ADMM achieves the lowest value among these methods.

VII. Conclusion

We consider the fast solution of a class of non-convex optimization problems that accommodates many machine learning applications, most notably regularized cost minimization. In view of the commonly used regularizers, we released the Hessian-Lipschitz requirement of existing designs



(a) $\mu = 0.01$



(b) $\mu = 1$

Fig. 4: Objective value v.s. iterations for GD, NGD, NNGD and CR-ADMM applied to $h_\mu(X)$ in high dimension

to develop a novel cubic regularized alternating direction method of multipliers (CR-ADMM) algorithm. We proved that our algorithm converges to a local optimum in a rate comparable to Nesterov's cubic regularization method. We also considered an imperfect variation of our algorithm that accommodates errors in its steps, and proved its convergence to a small neighborhood of a local minimum. We tested our algorithms comprehensively for the well-known matrix factorization problem, and observed its fast convergence as well as accuracy compared to the state-of-art gradient-based methods. It remains for future research whether CR-ADMM can be accelerated using Nesterov's technique [24].

REFERENCES

[1] Cubic regularized admm with convergence to a local minimum in non-convex optimization. https://www.dropbox.com/s/b592dnq3orgt860/allertonZai.pdf?dl=0.

[2] Smoothing for nonsmooth optimization. http://www.princeton.edu/~yc5/ele522_optimization/lectures/smoothing.pdf. Accessed: 2018-10-02.

[3] N. Agarwal, Z. Allen-Zhu, B. Bullins, E. Hazan, and T. Ma. Finding approximate local minima for nonconvex optimization in linear time. *arXiv preprint arXiv:1611.01146*, 2016.

[4] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[6] Y. Carmon and J. C. Duchi. Gradient descent efficiently finds the cubic-regularized non-convex newton step. *arXiv preprint arXiv:1612.00547*, 2016.

[7] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Accelerated methods for nonconvex optimization. *SIAM Journal on Optimization*, 28(2):1751–1772, 2018.

[8] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.

[9] F. E. Curtis, D. P. Robinson, and M. Samadi. A trust region algorithm with a worst-case iteration complexity of $O(\epsilon^{-3/2})$for nonconvex optimization. *Mathematical Programming*, 162(1-2):1–32, 2017.

[10] C. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pages 606–610. SIAM, 2005.

[11] R. Ge, F. Huang, C. Jin, and Y. Yuan. Escaping from saddle points-online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842, 2015.

[12] R. Ge, J. D. Lee, and T. Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016.

[13] M. Hong, J. D. Lee, and M. Razaviyayn. Gradient primal-dual algorithm converges to second-order stationary solutions for nonconvex distributed optimization. *arXiv preprint arXiv:1802.08941*, 2018.

[14] R. A. Horn, R. A. Horn, and C. R. Johnson. *Matrix analysis*. Cambridge university press, 1990.

[15] Z. Huang and S. Becker. Perturbed proximal descent to escape saddle points for non-convex and non-smooth objective functions. *arXiv preprint arXiv:1901.08958*, 2019.

[16] P. J. Huber et al. Robust estimation of a location parameter. *The annals of mathematical statistics*, 35(1):73–101, 1964.

[17] P. Jain and P. Kar. Non-convex optimization for machine learning. *Foundations and Trends® in Machine Learning*, 10(3-4):142–336, 2017.

[18] C. Jin, R. Ge, P. Netrapalli, S. M. Kakade, and M. I. Jordan. How to escape saddle points efficiently. *arXiv preprint arXiv:1703.00887*, 2017.

[19] C. Jin, P. Netrapalli, and M. I. Jordan. Accelerated gradient descent escapes saddle points faster than gradient descent. *arXiv preprint arXiv:1711.10456*, 2017.

[20] K. Y. Levy. The power of normalization: Faster evasion of saddle points. *arXiv preprint arXiv:1611.04831*, 2016.

[21] Q. Liu, X. Shen, and Y. Gu. Linearized ADMM for non-convex non-smooth optimization with convergence analysis. *arXiv preprint arXiv:1705.02502*, 2017.

[22] J. Mairal, F. Bach, J. Ponce, et al. Sparse modeling for image and vision processing. *Foundations and Trends® in Computer Graphics and Vision*, 8(2-3):85–283, 2014.

[23] Y. Nesterov and B. T. Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[24] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. In *Dokl. Akad. Nauk SSSR*, volume 269, pages 543–547, 1983.

[25] A. P. Singh and G. J. Gordon. A unified view of matrix factorization models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 358–373. Springer, 2008.

[26] J. Sun, Q. Qu, and J. Wright. When are nonconvex problems not scary? *arXiv preprint arXiv:1510.06096*, 2015.

[27] J. Sun, Q. Qu, and J. Wright. A geometric analysis of phase retrieval. In *Information Theory (ISIT), 2016 IEEE International Symposium on*, pages 2379–2383. IEEE, 2016.

[28] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[29] N. Tripuraneni, M. Stern, C. Jin, J. Regier, and M. I. Jordan. Stochastic cubic regularization for fast nonconvex optimization. *arXiv preprint arXiv:1711.02838*, 2017.

[30] Y. Wang, W. Yin, and J. Zeng. Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, pages 1–35, 2015.

[31] Y. Xu, R. Jin, and T. Yang. NEON+: Accelerated gradient methods for extracting negative curvature for non-convex optimization. *arXiv preprint arXiv:1712.01033*, 2017.