Stochastic Variance-Reduced Cubic Regularized Newton Methods

Dongruo Zhou [1] **Pan Xu** [1] **Quanquan Gu** [1]

Abstract

We propose a stochastic variance-reduced cubic regularized Newton method (SVRC) for non-convex optimization. At the core of our algorithm is a novel semi-stochastic gradient along with a semi-stochastic Hessian, which are specifically designed for cubic regularization method. We show that our algorithm is guaranteed to converge to an $(\epsilon, \sqrt{\epsilon})$-approximate local minimum within $\widetilde{O}(n^{4/5}/\epsilon^{3/2})$ second-order oracle calls, which outperforms the state-of-the-art cubic regularization algorithms including subsampled cubic regularization. Our work also sheds light on the application of variance reduction technique to high-order non-convex optimization methods. Thorough experiments on various non-convex optimization problems support our theory.

1 Introduction

We study the following finite-sum optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}), \tag{1.1}$$

where $F(\mathbf{x})$ and each $f_i(\mathbf{x})$ can be non-convex. Such problems are common in machine learning, where each $f_i(\mathbf{x})$ is a loss function on a training example (LeCun et al., 2015). Since $F(\mathbf{x})$ is non-convex, finding its global minimum is generally NP-Hard (Hillar & Lim, 2013). As a result, one possible goal is to find an approximate first-order stationary point ($\epsilon-$stationary point):

$$\|\nabla F(\mathbf{x})\|_2 \leq \epsilon,$$

for some given $\epsilon > 0$. A lot of studies have been devoted to this problem including gradient descent (GD), stochastic gradient descent (SGD) (Robbins & Monro, 1951), and their extensions (Ghadimi & Lan, 2013; Reddi et al., 2016a; Allen-Zhu & Hazan, 2016; Ghadimi & Lan, 2016). Nevertheless, first-order stationary points can be non-degenerate saddle points or even local maximum in non-convex optimization, which are undesirable. Therefore, a more reasonable objective is to find an approximate second-order stationary point (Nesterov & Polyak, 2006), which is also known as an (ϵ_g, ϵ_h)-approximate local minimum of $F(\mathbf{x})$:

$$\|\nabla F(\mathbf{x})\|_2 < \epsilon_g, \lambda_{\min}(\nabla^2 F(\mathbf{x})) \geq -\epsilon_h, \tag{1.2}$$

for some given constant $\epsilon_g, \epsilon_h > 0$. In fact, in some machine learning problems like matrix completion (Ge et al., 2016), one finds that every local minimum is a global minimum, suggesting that finding an approximate local minimum is a better choice than a stationary point, and is good enough in many applications. One of the most popular method to achieve this goal is perhaps cubic regularized Newton method, which was introduced by Nesterov & Polyak (2006), and solves the following kind of subproblems in each iteration:

$$\begin{aligned} \mathbf{h}(\mathbf{x}) &= \mathop{\mathrm{argmin}}_{\mathbf{h}\in\mathbb{R}^d} m(\mathbf{h},\mathbf{x}), \\ m(\mathbf{h},\mathbf{x}) &= \langle \nabla F(\mathbf{x}), \mathbf{h}\rangle + \frac{1}{2}\big\langle \nabla^2 F(\mathbf{x})\mathbf{h}, \mathbf{h}\big\rangle + \frac{\theta}{6}\|\mathbf{h}\|_2^3, \end{aligned} \tag{1.3}$$

where $\theta > 0$ is a regularization parameter. Nesterov & Polyak (2006) proved that fixing a starting point $\mathbf{x}_0$, and performing the updating rule $\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{h}(\mathbf{x}_{t-1})$, the algorithm can output a sequence $\mathbf{x}_i$ that converges to a local minimum provided that the function is Hessian Lipschitz. However, it can be seen that to solve the subproblem (1.3), one needs to calculate the full gradient $\nabla F(\mathbf{x})$ and Hessian $\nabla^2 F(\mathbf{x})$, which is a big overhead in large scale machine learning problem because n is often very large.

Some recent studies presented various algorithms to avoid the calculation of full gradient and Hessian in cubic regularization. Kohler & Lucchi (2017) used subsampling technique to get approximate gradient and Hessian instead of exact ones, and Xu et al. (2017c) also used subsampled Hessian. Both of them can reduce the computational complexity in some circumstance. However, just like other sampling-based algorithm such as subsampled Newton method Erdogdu & Montanari (2015); Xu et al. (2016); Roostakho-

[1]Department of Computer Science, University of California, Los Angeles, CA 90095, USA. Correspondence to: Quanquan Gu <qgu@cs.ucla.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

rasani & Mahoney (2016a;b); Ye et al. (2017), their convergence rates are worse than that of the Newton method, especially when one needs a high-accuracy solution (i.e., the optimization error ϵ is small). This is because the subsampling size one needs to achieve certain accuracy may be even larger than the full sample size n. Therefore, a natural question arises as follows:

When we need a high-accuracy local minimum, is there an algorithm that can output an approximate local minimum with better second-order oracle complexity than cubic regularized Newton method?

In this paper, we give an affirmative answer to the above question. We propose a novel cubic regularization algorithm named Stochastic Variance-Reduced Cubic regularization (**SVR Cubic**, or **SVRC** for short), which incorporates the variance reduction techniques (Johnson & Zhang, 2013; Xiao & Zhang, 2014; Allen-Zhu & Hazan, 2016; Reddi et al., 2016a) into the cubic-regularized Newton method. The key component in our algorithm is a novel semi-stochastic gradient, together with a semi-stochastic Hessian, that are specifically designed for cubic regularization. Furthermore, we prove that, for ρ-Hessian Lipschitz functions, to attain an $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minimum, our proposed algorithm requires $O(n + n^{4/5}/\epsilon^{3/2})$ Second-order Oracle (SO) calls and $O(1/\epsilon^{3/2})$ Cubic Subproblem Oracle (CSO) calls. Here an SO oracle represents an evaluation of triple $(f_i(\mathbf{x}), \nabla f_i(\mathbf{x}), \nabla^2 f_i(\mathbf{x}))$, and a CSO oracle denotes an evaluation of the exact solution (or inexact solution) of the cubic subproblem (1.3). Compared with the original cubic regularization algorithm (Nesterov & Polyak, 2006), which requires $O(n/\epsilon^{3/2})$ SO calls and $O(1/\epsilon^{3/2})$ CSO calls, our proposed algorithm reduces the SO calls by a factor of $\Omega(n^{1/5})$. We also carry out experiments on real data to demonstrate the superior performance of our algorithm.

Our major contributions are summarized as follows:

- We present a novel cubic regularization method with improved oracle complexity. To the best of our knowledge, this is the first algorithm that outperforms cubic regularization without any loss in convergence rate. In sharp contrast, existing subsampled cubic regularization methods (Kohler & Lucchi, 2017; Xu et al., 2017b) suffer from worse convergence rates than cubic regularization.

- We also extend our algorithm to the case with inexact solution to the cubic regularization subproblem. Similar to previous work (Cartis et al., 2011a; Xu et al., 2017b), we also layout a set of sufficient conditions, under which the output of the inexact algorithm is still guaranteed to have the same convergence rate and oracle complexity as the exact algorithm. This further sheds light on the practical implementation of our algorithm.

- As far as we know, our work is the first to rigorously demonstrates the advantage of variance reduction for second-order optimization algorithms. Although there exist a few studies (Lucchi et al., 2015; Moritz et al., 2016; Rodomanov & Kropotov, 2016) using variance reduction to accelerate Newton method, none of them can deliver faster rates of convergence than standard Newton method.

The remainder of this paper is organized as follows: We first review the most related work in Section 2. Then we present our algorithm and its main theoretical results in Sections 3 and 4 respectively. We discuss the proposed algorithm with inexact oracles in Section 5. In Section 6, we present our experimental results. Finally, we draw the conclusions in Section 7.

Notation We use $[n]$ to denote index set $\{1, 2, \ldots, n\}$. We denote vector Euclidean norm by $\|\mathbf{v}\|_2$. For symmetric matrix $\mathbf{H} \in \mathbb{R}^{d\times d}$, we denote its eigenvalues by $\lambda_1(\mathbf{H}) \leq \ldots \leq \lambda_d(\mathbf{H})$, its spectral norm by $\|\mathbf{H}\|_2 = \max\{|\lambda_1(\mathbf{H})|, |\lambda_d(\mathbf{H})|\}$, and the Schatten r-norm by $\|\mathbf{H}\|_{S_r} = (\sum_{i=1}^d |\lambda_i(\mathbf{H})|^r)^{1/r}$ for $r \geq 1$. We denote $\mathbf{A} \succeq \mathbf{B}$ if $\lambda_1(\mathbf{A}-\mathbf{B}) \geq 0$ for symmetric matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{d\times d}$. Note that $\|\mathbf{A}-\mathbf{B}\|_2 \leq C \Rightarrow \|\mathbf{A}\|_2 \succeq \|\mathbf{B}\|_2 - C\cdot \mathbf{I}, C > 0$. We call ξ a Rademacher random variable if $\mathbb{P}(\xi = 1) = \mathbb{P}(\xi = -1) = 1/2$. We use $f_n = O(g_n)$ to denote that $f_n \leq Cg_n$ for some constant $C > 0$ and use $f_n = \widetilde{O}(g_n)$ to hide the logarithmic terms of g_n.

2 Related Work

In this section, we briefly review the relevant work in the literature.

The most related work to ours is the cubic regularized Newton method, which was originally proposed in Nesterov & Polyak (2006). Cartis et al. (2011a) presented an adaptive framework of cubic regularization, which uses an adaptive estimation of the local Lipschitz constant and approximate solution to the cubic subproblem. To connect cubic regularization with traditional trust region method (Conn et al., 2000; Cartis et al., 2009; 2012; 2013), Blanchet et al. (2016); Curtis et al. (2017); Martínez & Raydan (2017) showed that the trust-region Newton method can achieve the same iteration complexity as the cubic regularization method. To overcome the computational burden of gradient and Hessian matrix evaluations, Kohler & Lucchi (2017); Xu et al. (2017b;c) proposed to use subsampled gradient and Hessian in cubic regularization. On the other hand, in order to solve the cubic subproblem (1.3) more efficiently, Carmon & Duchi (2016) proposed to use gradient descent, while Agarwal et al. (2017) proposed a sophisticated algorithm based on approximate matrix inverse and approximate PCA. Tripuraneni et al. (2017) proposed a refined stochastic cubic regularization algorithm based on above subproblem solver. However, none of the aforementioned variants of cubic reg-

Table 1. Comparisons between different methods to find $(\epsilon, \sqrt{\rho\epsilon})$-local minimum on the second-order oracle (SO) complexity and and the cubic sub-problem oracle (CSO) complexity.

Algorithm	SO calls	CSO calls	Gradient Lipschitz	Hessian Lipschitz
Cubic regularization (Nesterov & Polyak, 2006)	$O(n/\epsilon^{3/2})$	$O(1/\epsilon^{3/2})$	no	yes
Subsampled cubic regularization (Kohler & Lucchi, 2017; Xu et al., 2017c)	$\widetilde{O}(n/\epsilon^{3/2} + 1/\epsilon^{5/2})$[1]	$O(1/\epsilon^{3/2})$	yes	yes
SVRC (this paper)	$\widetilde{O}(n + n^{4/5}/\epsilon^{3/2})$	$O(1/\epsilon^{3/2})$	no	yes

ularization outperforms the original cubic regularization method in terms of oracle complexity.

Another line of related research is the variance reduction method, which has been extensively studied for large-scale finite-sum optimization problems. Variance reduction was first proposed in convex finite-sum optimization (Roux et al., 2012; Johnson & Zhang, 2013; Xiao & Zhang, 2014; Defazio et al., 2014), which uses semi-stochastic gradient to reduce the variance of the stochastic gradient and improves the gradient complexity of both stochastic gradient descent (SGD) and gradient descent (GD). Representative algorithms include Stochastic Average Gradient (SAG) (Roux et al., 2012), Stochastic Variance Reduced Gradient (SVRG) (Johnson & Zhang, 2013) and SAGA (Defazio et al., 2014), to mention a few. Garber & Hazan (2015); Shalev-Shwartz (2016) studied non-convex finite-sum problems where each individual function may be non-convex, but their sum is still convex. Reddi et al. (2016a) and Allen-Zhu & Hazan (2016) extended the SVRG algorithm to the general non-convex finite-sum optimization, which outperforms SGD and GD in terms of gradient complexity as well. However, to the best of our knowledge, it is still an open problem whether variance reduction can improve the oracle complexity of second-order optimization algorithms.

Last but not least is the vast literature of research which aims to escape from nondegenerated saddle points by finding the negative curvature direction. Ge et al. (2015); Jin et al. (2017a) showed that simple (stochastic) gradient descent with perturbation can escape from saddle points. Carmon et al. (2016); Royer & Wright (2017); Allen-Zhu (2017) showed that by calculating the negative curvature using Hessian information, one can find $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minima faster than first-order methods. Recent work (Xu & Yang, 2017; Allen-Zhu & Li, 2017; Jin et al., 2017b) proposed first-order algorithms that can escape from saddle points without using Hessian information. Yu et al. (2017b) proposed the GOSE algorithm to save negative curvature computation and Yu et al. (2017a) improved the gradient complexity by using third-order smoothness. Raginsky et al. (2017); Zhang et al. (2017); Xu et al. (2017a) proved that a family of algorithms based on discretizations of Langevin dynamics can find a neighborhood of the global minimum of nonconvex objective functions.

For better comparison of our algorithm with the most related algorithms in terms of SO and CSO oracle complexities, we summarize the results in Table 1. It can be seen from Table 1 that our algorithm (SVRC) achieves the lowest (SO and CSO) oracle complexity compared with the original cubic regularization method (Nesterov & Polyak, 2006) which employs full gradient and Hessian evaluations and the subsampled cubic method (Kohler & Lucchi, 2017; Xu et al., 2017c). In particular, our algorithm reduces the SO oracle complexity of cubic regularization by a factor of $n^{1/5}$ for finding an $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minimum. We will provide more detailed discussion in Section 4.

3 The Proposed Algorithm

In this section, we present a novel algorithm, which utilizes stochastic variance reduction techniques to improve cubic regularization method.

To reduce the computation burden of gradient and Hessian matrix evaluations in the cubic regularization updates in (1.3), subsampled gradient and Hessian matrix have been used in subsampled cubic regularization (Kohler & Lucchi, 2017; Xu et al., 2017c) and stochastic cubic regularization (Tripuraneni et al., 2017). Nevertheless, the stochastic gradient and Hessian matrix have large variances, which undermine the convergence performance. Inspired by SVRG (Johnson & Zhang, 2013), we propose to use a semi-stochastic version of gradient and Hessian matrix, which can control the variances automatically. Specifically, our algorithm has two loops. At the beginning of the s-th iteration of the outer loop, we denote $\widehat{\mathbf{x}}^s = \mathbf{x}_0^s$. We first calculate the full gradient $\mathbf{g}^s = \nabla F(\widehat{\mathbf{x}}^s)$ and Hessian matrix $\mathbf{H}^s = \nabla^2 F(\widehat{\mathbf{x}}^s)$, which are stored for further references in the inner loop. At the t-th iteration of the inner loop, we calculate the following semi-stochastic gradient and Hessian

[1]It is the refined rate proved by Xu et al. (2017c) for the subsampled cubic regularization algorithm proposed in Kohler & Lucchi (2017)

Algorithm 1 Stochastic Variance Reduction Cubic Regularization (**SVRC**)

Input: batch size b_g, b_h, cubic penalty parameter $\{M_{s,t}\}$, epoch number S, epoch length T and starting point $\mathbf{x}_0$.
Initialization $\widehat{\mathbf{x}}^1 = \mathbf{x}_0$
for $s = 1, \ldots, S$ **do**
  $\mathbf{x}_0^s = \widehat{\mathbf{x}}^s$
  $\mathbf{g}^s = \nabla F(\widehat{\mathbf{x}}^s) = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widehat{\mathbf{x}}^s), \mathbf{H}^s = \frac{1}{n}\sum_{i=1}^n \nabla^2 f_i(\widehat{\mathbf{x}}^s)$
  for $t = 0, \ldots, T-1$ **do**
    Sample index set $I_g, I_h, |I_g| = b_g, |I_h| = b_h$;
    $\mathbf{v}_t^s = \frac{1}{b_g}\sum_{i_t \in I_g}\big[\nabla f_{i_t}(\mathbf{x}_t^s) - \nabla f_{i_t}(\widehat{\mathbf{x}}^s)\big] + \mathbf{g}^s - \big(\frac{1}{b_g}\sum_{i_t \in I_g}\nabla^2 f_{i_t}(\widehat{\mathbf{x}}^s) - \mathbf{H}^s\big)(\mathbf{x}_t^s - \widehat{\mathbf{x}}^s)$
    $\mathbf{U}_t^s = \frac{1}{b_h}\sum_{j_t \in I_h}\big[\nabla^2 f_{j_t}(\mathbf{x}_t^s) - \nabla^2 f_{j_t}(\widehat{\mathbf{x}}^s)\big] + \mathbf{H}^s$
    $\mathbf{h}_t^s = \mathrm{argmin}\langle \mathbf{v}_t^s, \mathbf{h}\rangle + \frac{1}{2}\langle \mathbf{U}_t^s\mathbf{h}, \mathbf{h}\rangle + \frac{M_{s,t}}{6}\|\mathbf{h}\|_2^3$,
    $\mathbf{x}_{t+1}^s = \mathbf{x}_t^s + \mathbf{h}_t^s$
  end for
  $\widehat{\mathbf{x}}^{s+1} = \mathbf{x}_T^s$
end for
Output: $\mathbf{x}_{\text{out}} = \mathbf{x}_t^s$, where s, t are uniformly random chosen from $s \in [S]$ and $t \in [T]$.

matrix:

$$\mathbf{v}_t^s = \frac{1}{b_g}\sum_{i_t \in I_g}\big[\nabla f_{i_t}(\mathbf{x}_t^s) - \nabla f_{i_t}(\widehat{\mathbf{x}}^s)\big] + \mathbf{g}^s - \frac{1}{b_g}\sum_{i_t \in I_g}\big(\nabla^2 f_{i_t}(\widehat{\mathbf{x}}^s) - \mathbf{H}^s\big)(\mathbf{x}_t^s - \widehat{\mathbf{x}}^s), \quad (3.1)$$

$$\mathbf{U}_t^s = \frac{1}{b_h}\sum_{j_t \in I_h}\big[\nabla^2 f_{j_t}(\mathbf{x}_t^s) - \nabla^2 f_{j_t}(\widehat{\mathbf{x}}^s)\big] + \mathbf{H}^s, \quad (3.2)$$

where I_g and I_h are batch index sets, and the batch sizes will be decided later. In each inner iteration, we solve the following cubic regularization subproblem:

$$\mathbf{h}_t^s = \mathrm{argmin}\, m_t^s(\mathbf{h}),$$
$$m_t^s(\mathbf{h}) = \langle \mathbf{v}_t^s, \mathbf{h}\rangle + \frac{1}{2}\langle \mathbf{U}_t^s\mathbf{h}, \mathbf{h}\rangle + \frac{M_{s,t}}{6}\|\mathbf{h}\|_2^3, \quad (3.3)$$

where $\{M_{s,t}\}$ are cubic regularization parameter, which may depend on s and t. Then we perform the update $\mathbf{x}_{t+1}^s = \mathbf{x}_t^s + \mathbf{h}_t^s$ in the t-th iteration of the inner loop. The proposed algorithm is displayed in Algorithm 1.

There are two notable features of our "estimator" of the full gradient and Hessian in each inner loop, compared with that used in SVRG (Johnson & Zhang, 2013). The first is that our gradient and Hessian estimators consist of mini-batches of stochastic gradient and Hessian. The second one is that we use second order information when we construct the gradient estimator $\mathbf{v}_t^s$, while classical SVRG only uses first order information to build it. Intuitively speaking, both features are used to make a more accurate estimation of the true gradient and Hessian with affordable oracle calls. Note that similar approximations of the gradient and Hessian matrix have been staged in recent work by Gower et al. (2017) and Wai et al. (2017), where they used this new kind of estimator for traditional SVRG in the convex setting, which radically differs from our setting.

4 Main Theory

We first lay out the following Hessian Lipschitz assumption, which are necessary for our analysis and are widely used in the literature (Nesterov & Polyak, 2006; Xu et al., 2016; Kohler & Lucchi, 2017).

Assumption 4.1 (Hessian Lipschitz)**.** There exists a constant $\rho > 0$, such that for all $\mathbf{x}, \mathbf{y}$ and $i \in [n]$

$$\big\|\nabla^2 f_i(\mathbf{x}) - \nabla^2 f_i(\mathbf{y})\big\|_2 \le \rho\|\mathbf{x} - \mathbf{y}\|_2.$$

In fact, this is the only assumption we need to prove our theoretical results. The Hessian Lipschitz assumption plays a central role in controlling the changing speed of second order information. It is obvious that Assumption 4.1 implies the Hessian Lipschitz assumption of F, which, according to Nesterov & Polyak (2006), is also equivalent to the following lemma.

Lemma 4.2. Let function $F : \mathbb{R}^d \to \mathbb{R}$ satisfy ρ-Hessian Lipschitz assumption, then for any $\mathbf{h} \in \mathbb{R}^d$, it holds that

$$\big\|\nabla^2 F(\mathbf{x}) - \nabla^2 F(\mathbf{y})\big\|_2 \le \rho\|\mathbf{x} - \mathbf{y}\|_2,$$
$$F(\mathbf{x}+\mathbf{h}) \le F(\mathbf{x}) + \langle \nabla F(\mathbf{x}), \mathbf{h}\rangle + \frac{1}{2}\langle \nabla^2 F(\mathbf{x})\mathbf{h}, \mathbf{h}\rangle + \frac{\rho}{6}\|\mathbf{h}\|_2^3,$$
$$\big\|\nabla F(\mathbf{x}+\mathbf{h}) - \nabla F(\mathbf{x}) - \nabla^2 F(\mathbf{x})\mathbf{h}\big\|_2 \le \frac{\rho}{2}\|\mathbf{h}\|_2^2.$$

We then define the following optimal function gap between initial point $\mathbf{x}_0$ and the global minimum of F.

Definition 4.3 (Optimal Gap)**.** For function $F(\cdot)$ and the initial point $\mathbf{x}_0$, let Δ_F be

$$\Delta_F = \inf\{\Delta \in \mathbb{R} : F(\mathbf{x}_0) - F^* \le \Delta\},$$

where $F^* = \inf_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x})$.

W.L.O.G., we assume $\Delta_F < +\infty$ throughout this paper.

Before we present nonasympotic convergence results of Algorithm 1, we define

$$\mu(\mathbf{x}) = \max\left\{\|\nabla F(\mathbf{x})\|_2^{3/2}, -\frac{\lambda_{\min}^3\big(\nabla^2 F(\mathbf{x})\big)}{\rho^{3/2}}\right\}. \quad (4.1)$$

By definition in (4.1), $\mu(\mathbf{x}) \leq \epsilon^{3/2}$ holds if and only if

$$\|\nabla F(\mathbf{x})\|_2 \leq \epsilon, \quad \lambda_{\min}\big(\nabla^2 F(\mathbf{x})\big) > -\sqrt{\rho\epsilon}. \quad (4.2)$$

Therefore, in order to find an $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minimum of the non-convex function F, it suffices to find a point $\mathbf{x}$ which satisfies $\mu(\mathbf{x}) < \epsilon^{3/2}$. Next we define our oracles formally:

Definition 4.4 (Second-order Oracle)**.** Given an index i and a point $\mathbf{x}$, one second-order oracle (SO) call returns such a triple:

$$[f_i(\mathbf{x}), \nabla f_i(\mathbf{x}), \nabla^2 f_i(\mathbf{x})]. \quad (4.3)$$

Definition 4.5 (Cubic Subproblem Oracle)**.** Given a vector $\mathbf{g} \in \mathbb{R}^d$, a Hessian matrix $\mathbf{H}$ and a positive constant θ, one Cubic Subproblem Oracle (CSO) call returns $\mathbf{h}_{\text{sol}}$, where $\mathbf{h}_{\text{sol}}$ can be solved exactly as follows

$$\mathbf{h}_{\text{sol}} = \mathop{\text{argmin}}_{\mathbf{h}\in\mathbb{R}^d} \langle \mathbf{g}, \mathbf{h}\rangle + \frac{1}{2}\langle \mathbf{h}, \mathbf{H}\mathbf{h}\rangle + \frac{\theta}{6}\|\mathbf{h}\|_2^3.$$

Remark 4.6. The second-order oracle is a special form of *Information Oracle* introduced by Nesterov, which returns gradient, Hessian and all high order derivatives of the objective function $F(\mathbf{x})$. Here, our second-order oracle will only returns first and second order information at some point of single objective f_i instead of F. We argue that it is a reasonable adaption because in this paper we focus on finite-sum objective function. The Cubic Subproblem Oracle will return an exact or inexact solution of (3.3), which plays an important role in both theory and practice.

Now we are ready to give a general convergence result of Algorithm 1:

Theorem 4.7. Under Assumption 4.1, suppose that the cubic regularization parameter $M_{s,t}$ of Algorithm 1 satisfies that $M_{s,t} = C_M\rho$, where ρ is the Hessian Lipschitz parameter and $C_M \geq 100$ is a constant. The batch sizes b_g and b_h satisfy that

$$b_g \geq 5T^4,\ b_h \geq 100T^2 \log d, \quad (4.4)$$

where $T \geq 2$ is the length of the inner loop of Algorithm 1 and d is the dimension of the problem. Then the output of Algorithm 1 satisfies

$$\mathbb{E}[\mu(\mathbf{x}_{\text{out}})] \leq \frac{240C_M^2\rho^{1/2}\Delta_F}{ST}. \quad (4.5)$$

Remark 4.8. According to (4.1), to ensure that $\mathbf{x}_{\text{out}}$ is an $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minimum, we can set the right hand side of (4.5) to be less then $\epsilon^{3/2}$. This immediately implies that the total iteration complexity of Algorithm 1 is $ST = O(\Delta_F\rho^{1/2}\epsilon^{-3/2})$, which matches the iteration complexity of cubic regularization (Nesterov & Polyak, 2006).

Remark 4.9. Note that there is a $\log d$ term in the expression of parameter, and it is only related to Hessian batch size b_h. The $\log d$ term comes from matrix concentration inequalities, which is believed to be unavoidable (Tropp et al., 2015). In other words, the batch size of Hessian matrix b_h has a inevitable relation to dimension d, unlike the batch size of gradient b_g.

The complexity result in Theorem 4.7 depends on a series of parameter. In the following corollary, we will show how to choose these parameters in practice to achieve a better oracle complexity.

Corollary 4.10. Under Assumption 4.1, let the cubic regularization parameter $M_{s,t} = M = C_M\rho$, where $C_M \geq 100$ is a constant. Let the epoch length $T = n^{1/5}$, batch sizes $b_g = 5n^{4/5}$ and $b_h = 100n^{2/5}\log d$, and the number of epochs $S = \max\{1, 240C_M^2\rho^{1/2}\Delta_F n^{-1/5}\epsilon^{-3/2}\}$. Then Algorithm 1 will find an $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minimum $\mathbf{x}_{\text{out}}$ within

$$O\bigg(n + \frac{\Delta_F\sqrt{\rho}n^{4/5}}{\epsilon^{3/2}}\bigg) \text{ SO calls} \quad (4.6)$$

and

$$O\bigg(\frac{\Delta_F\sqrt{\rho}}{\epsilon^{3/2}}\bigg) \text{ CSO calls.} \quad (4.7)$$

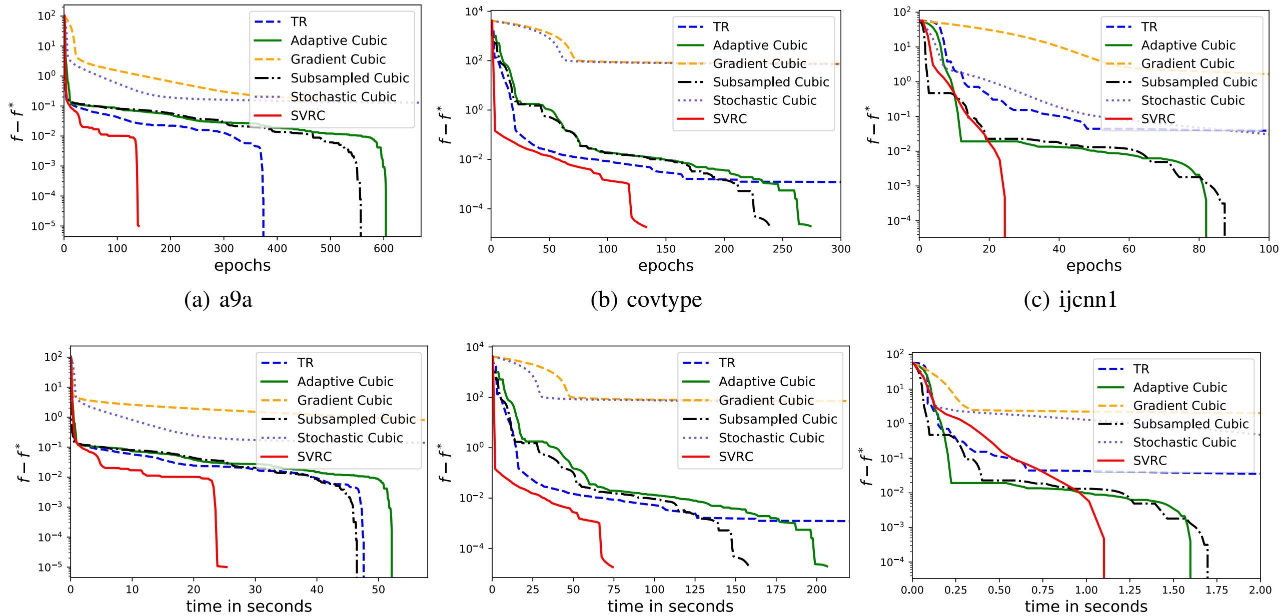
Remark 4.11. Corollary 4.10 states that we can reduce the SO calls by setting the batch size b_g, b_h related to n. In contrast, in order to achieve an $(\epsilon, \sqrt{\rho\epsilon})$ local minimum, original cubic regularization method in Nesterov & Polyak (2006) needs $O(n/\epsilon^{3/2})$ second-order oracle calls, which is by a factor of $n^{1/5}$ worse than ours. And subsampled cubic regularization (Kohler & Lucchi, 2017; Xu et al., 2017c) requires $\widetilde{O}(n/\epsilon^{3/2} + 1/\epsilon^{5/2})$ SO calls, which is also worse than our algorithm.

5 SVRC with Inexact Oracles

In practice, the exact solution to the cubic subproblem (3.3) cannot be obtained. Instead, one can only get an approximate solution by some inexact solver. Thus we replace the CSO oracle in (4.5) with the following inexact CSO oracle

$$\widetilde{\mathbf{h}}_{\text{sol}} \approx \mathop{\text{argmin}}_{\mathbf{h}\in\mathbb{R}^d} \langle \mathbf{g}, \mathbf{h}\rangle + \frac{1}{2}\langle \mathbf{h}, \mathbf{H}\mathbf{h}\rangle + \frac{\theta}{6}\|\mathbf{h}\|_2^3.$$

To analyze the performance of Algorithm 1 with inexact cubic subproblem solver, we relax the exact solver in Line



(a) a9a (b) covtype (c) ijcnn1

(d) a9a (e) covtype (f) ijcnn1

Figure 1. Logarithmic function value gap for nonconvex regularized logistic regression on different datasets. (a), (b) and (c) present the oracle complexity comparison; (d), (e) and (f) present the runtime comparison.

10 of Algorithm 1 with

$$\widetilde{\mathbf{h}}_t^s \approx \operatorname{argmin} m_t^s(\mathbf{h}). \tag{5.1}$$

The ultimate goal of this section is to prove that the theoretical results of our SVRC algorithm still hold with inexact subproblem solvers. To this end, we present the following sufficient condition, under which inexact solution can ensure the same oracle complexity as the exact solution:

Condition 5.1 (Inexact condition)**.** For each s, t and given $\delta > 0$, $\widetilde{\mathbf{h}}_t^s$ satisfies δ-inexact condition if $\widetilde{\mathbf{h}}_t^s$ satisfies

$$\begin{aligned} m_t^s(\widetilde{\mathbf{h}}_t^s) &\le -\frac{M_{s,t}}{12}\|\widetilde{\mathbf{h}}_t^s\|_2^3 + \delta, \\ \|\nabla m_t^s(\widetilde{\mathbf{h}}_t^s)\|_2 &\le M_{s,t}^{1/3}\delta^{2/3}, \\ \big|\|\widetilde{\mathbf{h}}_t^s\|_2 - \|\mathbf{h}_t^s\|_2\big| &\le M_{s,t}^{-1/3}\delta^{1/3}. \end{aligned}$$

Remark 5.2. Similar inexact conditions have been studied in the literature of cubic regularization. For instance, Nesterov & Polyak (2006) presented a practical way to solve the cubic subproblem without termination condition. Cartis et al. (2011a); Kohler & Lucchi (2017) presented termination criteria for approximate solution to cubic subproblem, which is slightly different from Condtion 5.1.

Now we present the convergence result of SVRC with inexact CSO oracles:

Theorem 5.3. Suppose that for each s, t, $\widetilde{\mathbf{h}}_t^s$ is an inexact solver of cubic subproblem $m_t^s(\mathbf{h})$, which satisfies Condition 5.1. Under the same conditions of Theorem 4.7, the output of Algorithm 1 satisfies

$$\mathbb{E}[\mu(\mathbf{x}_{\text{out}})] \le \frac{240C_M^2\rho^{1/2}\Delta_F}{ST} + 480C_M^2\rho^{1/2}\delta. \tag{5.2}$$

Remark 5.4. By the definition of $\mu(\mathbf{x})$, in order to attain an $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minimum, we require $\mathbb{E}[\mu(\mathbf{x}_{\text{out}})] \le \epsilon^{3/2}$ and thus $480C_M^2\rho^{1/2}\delta < \epsilon^{3/2}$, which implies that δ in Condition 5.1 should satisfy $\delta < (480C_M^2\rho^{1/2})^{-1}\epsilon^{3/2}$. Thus the total iteration complexity of Algorithm 1 with inexact oracle is still $O(\Delta_F\rho^{1/2}\epsilon^{-3/2})$.

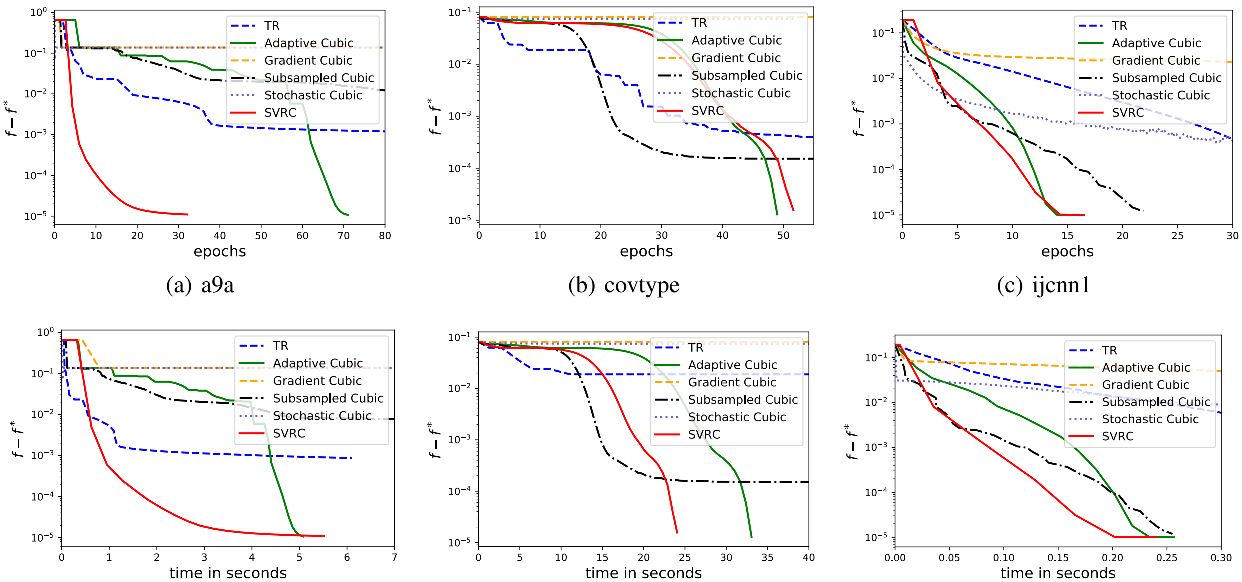
By the same choice of parameters, Algorithm 1 with inexact oracle can achieve a reduction in SO calls.

Corollary 5.5. Suppose that for each s, t, $\widetilde{\mathbf{h}}_t^s$ is an inexact solver of cubic subproblem $m_t^s(\mathbf{h})$, which satisfies Condition 5.1 with $\delta = (960C_M^2)^{-1}\rho^{-1/2}\epsilon^{3/2}$. Under Assumption 4.1, let the cubic regularization parameter $M_{s,t} = M = C_M\rho$, where $C_M \ge 100$ is a constant. Let the epoch length $T = n^{1/5}$, batch sizes $b_g = 5n^{4/5}$ and $b_h = 100n^{2/5}\log d$, and the number of epochs $S = \max\{1, 480C_M^2\rho^{1/2}\Delta_F n^{-1/5}\epsilon^{-3/2}\}$. Then Algorithm 1 will find an $(\epsilon, \sqrt{\rho\epsilon})$-approximate local minimum within

$$O\bigg(n + \frac{\Delta_F\sqrt{\rho}n^{4/5}}{\epsilon^{3/2}}\bigg) \text{ SO calls} \tag{5.3}$$

and

$$O\bigg(\frac{\Delta_F\sqrt{\rho}}{\epsilon^{3/2}}\bigg) \text{ CSO calls.} \tag{5.4}$$



(a) a9a (b) covtype (c) ijcnn1

(d) a9a (e) covtype (f) ijcnn1

Figure 2. Logarithmic function value gap for nonlinear least square on different datasets. (a), (b) and (c) present the oracle complexity comparison; (d), (e) and (f) present the runtime comparison.

Remark 5.6. It is worth noting that even with the inexact CSO oracle satisfying Condition 5.1, the SO and CSO complexities of SVRC remain the same as that of SVRC with exact CSO oracle. Furthermore, this result always holds with any inexact cubic sub-problem solver.

6 Experiments

In this section, we present numerical experiments on different non-convex Empirical Risk Minimization (ERM) problems and on different datasets to validate the advantage of our **SVRC** algorithm in finding approximate local minima.

Baselines: We compare our algorithm with adaptive cubic regularization (**Adaptive Cubic**) (Cartis et al., 2011a), subsampled cubic regularization (**Subsampled Cubic**) (Kohler & Lucchi, 2017), stochastic cubic regularization (**Stochastic Cubic**) (Tripuraneni et al., 2017), gradient cubic regularization (**Gradient Cubic**) (Carmon & Duchi, 2016) and trust region Newton method (**TR**) (Conn et al., 2000). All algorithms are carefully tuned for a fair comparison.

Calculation for SO calls: For **Subsampled Cubic**, each loop takes $(B_g + B_h)$ SO calls, where B_g and B_h are the subsampling sizes of gradient and Hessian. For **Stochastic Cubic**, each loop costs $(n_g + n_h)$ SO calls, where n_g and n_h denote the subsampling sizes of gradient and Hessian-vector operator. **Gradient Cubic**, **Adaptive Cubic** and **TR** cost n SO calls in each loop. We define the amount of epochs to be the amount of SO calls divided by n.

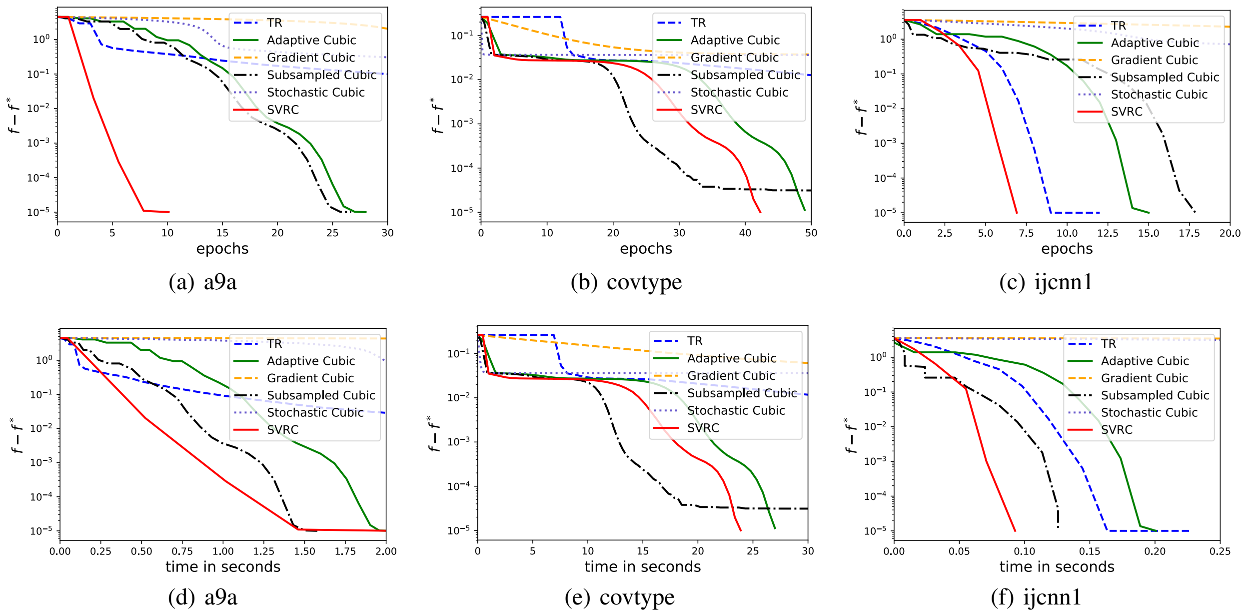
Parameters and subproblem solver: For each algorithm and each dataset, we choose different b_g, b_h, T for the best performance. Meanwhile, we choose $M_{s,t} = \alpha/(1 + \beta)^{(s+t/T)}, \alpha, \beta > 0$ for each iteration. When $\beta = 0$, it has been proved to enjoy good convergence performance. This choice of parameter is similar to the choice of penalty parameter in **Subsampled Cubic** and **Adaptive Cubic**, which sometimes makes some algorithms behave better in our experiment. For the subproblem solover of (3.3) in each loop, we choose the Lanczos-type method (Cartis et al., 2011a).

Datasets: The datasets we use are *a9a, covtype, ijcnn1*, which are common datasets used in ERM problems. The detailed information about these datasets are in Table 2.

Table 2. Overview of the datasets used in our experiments

Dataset	sample size n	dimension d
a9a	32,561	123
covtype	581,012	54
ijcnn1	35,000	22

Non-convex regularized logistic regression: The first non-convex problem we study is a binary logistic regression problem with a non-convex regularizer $\sum_{i=1}^{d} \lambda \mathbf{w}_{(i)}^2/(1 + \mathbf{w}_{(i)}^2)$ (Reddi et al., 2016b). Specifically, suppose we are given training data $\{\mathbf{x}_i, y_i\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$ are feature vectors and labels corresponding to the i-th data



(a) a9a (b) covtype (c) ijcnn1

(d) a9a (e) covtype (f) ijcnn1

Figure 3. Logarithmic function value gap for robust linear regression on different datasets. (a), (b) and (c) present the oracle complexity comparison; (d), (e) and (f) present the runtime comparison.

points. The minimization problem is as follows

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} y_i \log\phi(\mathbf{x}_i^\top\mathbf{w}) + (1-y_i)\log[1-\phi(\mathbf{x}_i^\top\mathbf{w})] + \sum_{i=1}^{d} \lambda\mathbf{w}_{(i)}^2/(1+\mathbf{w}_{(i)}^2),$$

where $\phi(x) = 1/(1+\exp(-x))$ is the sigmoid function. We fix $\lambda = 10$ in our experiments. Recall the definition $M_{s,t} = \alpha/(1+\beta)^{(s+t/T)}, \alpha, \beta > 0$. We set $\alpha = 0.05, \beta = 0$ for *a9a* and *ijcnn1* datasets and $\alpha = 5e3, \beta = 0.15$ for *covtype*. The experiment results are shown in Figure 1.

Nonlinear linear squares: The second problem is a non-linear least squares problem which focuses on the task of binary linear classification (Xu et al., 2017b). Given training data $\{\mathbf{x}_i, y_i\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$ are feature vectors and labels corresponding to the i-th data points. The minimization problem is

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} [y_i - \phi(\mathbf{x}_i^\top\mathbf{w})]^2.$$

Here ϕ is the sigmoid function. We set $\alpha = 0.05, 1e8, 0.003$ and $\beta = 0, 1, 0.5$ for *a9a*, *covtype* and *ijcnn1* datasets respectively. The experiment results are shown in Figure 2.

Robust linear regression: The third problem is a robust linear regression problem where we use a non-convex robust loss function $\log(x^2/2+1)$ (Barron, 2017) instead of square loss in least square regression. Given a training sample $\{\mathbf{x}_i, y_i\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$ are feature vectors and labels corresponding to the i-th data point. The minimization problem is

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} \eta(y_i - \mathbf{x}_i^\top\mathbf{w}),$$

where $\eta(x) = \log(x^2/2+1)$. We set $\alpha = 0.1, 1e9, 2$ and $\beta = 0.1, 1, 0$ for *a9a*, *covtype* and *ijcnn1* datasets respectively. The experimental results are shown in Figure 3.

From Figures 1, 2 and 3, we can see that our algorithm SVRC outperforms all the other baseline algorithms on all the datasets. The only exception happens in the non-linear least square problem and the robust linear regression problem on the *covtype* dataset, where our algorithm behaves a little worse than **Adaptive Cubic** at the high accuracy regime in terms of epoch counts. However, under this setting, our algorithm still outperforms the other baselines in terms of the cpu time.

7 Conclusions

In this paper, we propose a novel second-order algorithm for non-convex optimization called **SVRC**. Our algorithm is the first algorithm which improves the oracle complexity of cubic regularization and its subsampled variants under certain regime using variance reduction techniques. We also show that similar oracle complexity also holds with inexact oracles. Under both settings our algorithm outperforms the state-of-the-art methods.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This research was sponsored in part by the National Science Foundation IIS-1618948 and IIS-1652539. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

References

Agarwal, N., Allenzhu, Z., Bullins, B., Hazan, E., and Ma, T. Finding approximate local minima for nonconvex optimization in linear time. 2017.

Allen-Zhu, Z. Natasha 2: Faster non-convex optimization than sgd. *arXiv preprint arXiv:1708.08694*, 2017.

Allen-Zhu, Z. and Hazan, E. Variance reduction for faster non-convex optimization. In *International Conference on Machine Learning*, pp. 699–707, 2016.

Allen-Zhu, Z. and Li, Y. Neon2: Finding Local Minima via First-Order Oracles. *ArXiv e-prints*, abs/1711.06673, November 2017. Full version available at http://arxiv.org/abs/1711.06673.

Barron, J. T. A more general robust loss function. *arXiv preprint arXiv:1701.03077*, 2017.

Blanchet, J., Cartis, C., Menickelly, M., and Scheinberg, K. Convergence rate analysis of a stochastic trust region method for nonconvex optimization. *arXiv preprint arXiv:1609.07428*, 2016.

Carmon, Y. and Duchi, J. C. Gradient descent efficiently finds the cubic-regularized non-convex newton step. 2016.

Carmon, Y., Duchi, J. C., Hinder, O., and Sidford, A. Accelerated methods for non-convex optimization. 2016.

Cartis, C., Gould, N. I., and Toint, P. L. Trust-region and other regularisations of linear least-squares problems. *BIT Numerical Mathematics*, 49(1):21–53, 2009.

Cartis, C., Gould, N. I., and Toint, P. L. Adaptive cubic regularisation methods for unconstrained optimization. part i: motivation, convergence and numerical results. *Mathematical Programming*, 127(2):245–295, 2011a.

Cartis, C., Gould, N. I. M., and Toint, P. L. *Adaptive cubic regularisation methods for unconstrained optimization. Part II: worst-case function- and derivative-evaluation complexity*. Springer-Verlag New York, Inc., 2011b.

Cartis, C., Gould, N. I., and Toint, P. L. Complexity bounds for second-order optimality in unconstrained optimization. *Journal of Complexity*, 28(1):93–108, 2012.

Cartis, C., Gould, N. I., and Toint, P. L. On the evaluation complexity of cubic regularization methods for potentially rank-deficient nonlinear least-squares problems and its relevance to constrained nonlinear optimization. *SIAM Journal on Optimization*, 23(3):1553–1574, 2013.

Chen, R. Y., Gittens, A., and Tropp, J. A. The masked sample covariance estimator: an analysis using matrix concentration inequalities. *Information and Inference: A Journal of the IMA*, 1(1):2–20, 2012.

Conn, A. R., Gould, N. I., and Toint, P. L. *Trust region methods*. SIAM, 2000.

Curtis, F. E., Robinson, D. P., and Samadi, M. 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.

Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014.

Durrett, R. *Probability: theory and examples*. Cambridge university press, 2010.

Erdogdu, M. A. and Montanari, A. Convergence rates of sub-sampled newton methods. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pp. 3052–3060. MIT Press, 2015.

Garber, D. and Hazan, E. Fast and simple pca via convex optimization. *arXiv preprint arXiv:1509.05647*, 2015.

Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle pointsonline stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pp. 797–842, 2015.

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

Ghadimi, S. and Lan, G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

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

Gower, R. M., Roux, N. L., and Bach, F. Tracking the gradients using the hessian: A new look at variance reducing stochastic methods. *arXiv preprint arXiv:1710.07462*, 2017.

Hillar, C. J. and Lim, L.-H. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):45, 2013.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. 2017a.

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

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

Kohler, J. M. and Lucchi, A. Sub-sampled cubic regularization for non-convex optimization. *arXiv preprint arXiv:1705.05933*, 2017.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature*, 521 (7553):436–444, 2015.

Lucchi, A., McWilliams, B., and Hofmann, T. A variance reduced stochastic newton method. *arXiv preprint arXiv:1503.08316*, 2015.

Mackey, L., Jordan, M. I., Chen, R. Y., Farrell, B., Tropp, J. A., et al. Matrix concentration inequalities via the method of exchangeable pairs. *The Annals of Probability*, 42(3):906–945, 2014.

Martínez, J. M. and Raydan, M. Cubic-regularization counterpart of a variable-norm trust-region method for unconstrained minimization. *Journal of Global Optimization*, 68(2):367–385, 2017.

Moritz, P., Nishihara, R., and Jordan, M. A linearly-convergent stochastic l-bfgs algorithm. In *Artificial Intelligence and Statistics*, pp. 249–258, 2016.

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

Raginsky, M., Rakhlin, A., and Telgarsky, M. Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. *arXiv preprint arXiv:1702.03849*, 2017.

Reddi, S. J., Hefny, A., Sra, S., Poczos, B., and Smola, A. Stochastic variance reduction for nonconvex optimization. pp. 314–323, 2016a.

Reddi, S. J., Sra, S., Póczos, B., and Smola, A. Fast incremental method for smooth nonconvex optimization. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pp. 1971–1977. IEEE, 2016b.

Robbins, H. and Monro, S. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Rodomanov, A. and Kropotov, D. A superlinearly-convergent proximal newton-type method for the optimization of finite sums. In *International Conference on Machine Learning*, pp. 2597–2605, 2016.

Roostakhorasani, F. and Mahoney, M. W. Sub-sampled newton methods i: Globally convergent algorithms. 2016a.

Roostakhorasani, F. and Mahoney, M. W. Sub-sampled newton methods ii: Local convergence rates. 2016b.

Roux, N. L., Schmidt, M., and Bach, F. R. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in Neural Information Processing Systems*, pp. 2663–2671, 2012.

Royer, C. W. and Wright, S. J. Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. *arXiv preprint arXiv:1706.03131*, 2017.

Shalev-Shwartz, S. Sdca without duality, regularization, and individual convexity. In *International Conference on Machine Learning*, pp. 747–754, 2016.

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

Tropp, J. A. The expected norm of a sum of independent random matrices: An elementary approach. In *High Dimensional Probability VII*, pp. 173–202. Springer, 2016.

Tropp, J. A. et al. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2): 1–230, 2015.

Wai, H.-T., Shi, W., Nedic, A., and Scaglione, A. Curvature-aided incremental aggregated gradient method. *arXiv preprint arXiv:1710.08936*, 2017.

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

Xu, P., Yang, J., Roosta-Khorasani, F., R, C., and Mahoney, M. W. Sub-sampled newton methods with non-uniform sampling. 2016.

Xu, P., Chen, J., Zou, D., and Gu, Q. Global convergence of langevin dynamics based algorithms for nonconvex optimization. *arXiv preprint arXiv:1707.06618*, 2017a.

Xu, P., Roosta-Khorasan, F., and Mahoney, M. W. Second-order optimization for non-convex machine learning: An empirical study. *arXiv preprint arXiv:1708.07827*, 2017b.

Xu, P., Roosta-Khorasani, F., and Mahoney, M. W. Newton-type methods for non-convex optimization under inexact hessian information. *arXiv preprint arXiv:1708.07164*, 2017c.

Xu, Y. and Yang, T. First-order stochastic algorithms for escaping from saddle points in almost linear time. *arXiv preprint arXiv:1711.01944*, 2017.

Ye, H., Luo, L., and Zhang, Z. Approximate newton methods and their local convergence. In *International Conference on Machine Learning*, pp. 3931–3939, 2017.

Yu, Y., Xu, P., and Gu, Q. Third-order smoothness helps: Even faster stochastic optimization algorithms for finding local minima. *arXiv preprint arXiv:1712.06585*, 2017a.

Yu, Y., Zou, D., and Gu, Q. Saving gradient and negative curvature computations: Finding local minima more efficiently. *arXiv preprint arXiv:1712.03950*, 2017b.

Zhang, Y., Liang, P., and Charikar, M. A hitting time analysis of stochastic gradient langevin dynamics. *arXiv preprint arXiv:1702.05575*, 2017.