Escaping Saddle Points with the Successive Convex Approximation Algorithm

Amrit Singh Bedi¹*, Ketan Rajawat² and Vaneet Aggarwal³
¹,²Dept. of EE, IIT Kanpur, India
³Dept. of IE, Purdue University, USA
{amritbd, ketan}@iitk.ac.in, vaneet@purdue.edu

Abstract

Optimizing non-convex functions is of primary importance in the vast majority of machine learning algorithms. Even though many gradient descent based algorithms have been studied, successive convex approximation based algorithms have been recently empirically shown to converge faster. However, such successive convex approximation based algorithms can get stuck in a first-order stationary point. To avoid that, we propose an algorithm that perturbs the optimization variable slightly at the appropriate iteration. In addition to achieving the same convergence rate results as the non-perturbed version, we show that the proposed algorithm converges to a second order stationary point. Thus, the proposed algorithm escapes the saddle point efficiently and does not get stuck at the first order saddle points.

1 Introduction

Optimization, as a field, has found strong applications in a lot of science and technology areas. Some of them include mechanics, economics and management, electromagnetics, communications, scheduling, control systems, smart grids, and artificial intelligence [Jain et al., 2017]. Most of the algorithms in supervised and unsupervised machine learning involve solving optimization problems. Most of these problems are non-convex optimization problems such as matrix completion [Sun and Luo, 2016], phase retrieval [Sun et al., 2018], tensor decomposition [Ge et al., 2015], tensor completion [Liu et al., 2016] [Wang et al., 2017], dimensionality reduction [Wang et al., 2018], etc., and efficient mechanisms to solve the problems is an important research direction. Even though many of these problems are non-convex, a popular workaround is to relax the problems to a convex optimization problem. However, this approach may be lossy and nevertheless presents significant challenges for large scale optimization. Thus, novel algorithms for solving non-convex optimization problems are of interest in machine learning, and this paper focuses on proposing and analyzing a novel algorithm for solving non-convex optimization problems.

Efficient approaches for solving non-convex optimization problems have been widely studied. Most common approaches use a version of gradient (or sub-gradient) descent [Jin et al., 2017] [Candas et al., 2015] [Sun and Luo, 2016]. In contrast, the authors of [Scutari et al., 2014] [Scutari et al., 2017] [Liu et al., 2018] empirically demonstrated that successive convex approximation (SCA) based approaches can outperform the gradient descent based approaches, even though both obtain the same convergence rate. Thus, we consider an algorithm based on the SCA approach. Recently, [Jin et al., 2017] have proposed an approach in which a perturbed form of gradient descent algorithm is shown to converge to a second order stationary point, and can escape the saddle points. However, there has been no such study for the SCA approach for convergence to second order stationary point (to the best of our knowledge), even though the SCA approaches are more efficient. This paper tackles this problem and shows that the SCA can also avoid the saddle points. Further, the convergence rate guarantees are maintained with perturbation.

To be specific, in this paper we are interested in the question that whether SCA algorithm converges to a local minima in the number of iterations which are almost dimension free? The same question is answered for the gradient descent algorithm in [Jin et al., 2017]. We extend the analysis to SCA algorithms and show that a perturbed successive convex approximation (P-SCA) algorithm converges to $\epsilon$ second order stationary point in $\tilde{O}(\frac{1}{\epsilon^{2}})$ with an extra multiplicative factor of polylog($d$) where $d$ is the underlying dimension. Note that the dependence on the underlying dimension $d$ is better than the $\Omega(d^{4})$ dependence obtained in [Ge et al., 2015]. The convergence rate obtained in this paper is similar to the well-known rate of gradient descent algorithm convergence to a first order stationary point (FOSP) with an additional log factor [Nesterov, 1998]. This comes from the structure of the proposed SCA type algorithm. It is emphasized that the proposed algorithm does not exhibit a gradient descent like update, and hence the analysis of [Jin et al., 2017] cannot be directly extended. To do so, we show that the proposed algorithm is an inexact version of the standard gradient descent algorithm, which is then used to derive

*Contact Author
the convergence rate result [Bertsekas and Tsitsiklis, 2000; So and Zhou, 2017a]. We note that the results in this paper can be generalized to an arbitrary inexact gradient descent algorithm with an additional assumption that the norm of the error in the gradient is bounded.

2 Related Work

In literature, gradient/subgradient descent based approaches are popular for solving the optimization problems in an iterative manner [Spall, 2005]. To handle the non-convexity in the objective function, gradient averaging based methods are proposed in [Ruszczyński, 1980; Bach, 2014]. The other related works in [Netrapalli et al., 2013; Candás et al., 2015; Sun and Luo, 2016a] require smart initialization and have demonstrated faster local convergence.

Apart from the first order methods mentioned above, there are methods such as Majorization-Minimization (MM) [Sun et al., 2017; Mairal, 2013] and SCA [Scutari et al., 2014], which are used for the non-convex optimization. Both of these methods utilize the same technique of solving a sequence of approximate surrogate functions under different conditions. At the point of approximation, MM requires the surrogate function to upper bound the non-convex objective function, while for SCA the surrogate function needs to be convex and the upper bound condition is not required. Therefore, SCA helps to better approximate the non-convex objective and results in better convergence speeds [Liu et al., 2018].

For the approximation based approaches to optimize non-convex objectives, the convergence to FOSP is discussed in literature [Scutari et al., 2017; Liu et al., 2018]. In this paper, we are interested in making these approximation based technique to escape saddle points and converge to the second order stationary point (SOSP).

To escape the saddle point, two popular techniques namely perturbation based [Jin et al., 2017] and second-order information based methods [Nesterov and Polyak, 2006] have been used. A cubic regularization based algorithm is proposed in [Nesterov and Polyak, 2006] with convergence to $\epsilon$ SOSP in $O(\frac{1}{\epsilon^2})$ number of iterations. In practice, the second order information based methods require Hessian inversion at each algorithm iteration, which is computationally expensive. On the other hand, in perturbation based methods, noise from a given distribution (uniform in this paper) is added to the algorithm update (when in the neighborhood of a saddle point) to escape the saddle point which is computationally inexpensive. This work aims to achieve advantages of both the SCA based methods equipped with perturbation to escape the saddle points efficiently and converge to $\epsilon$ SOSP.

3 Preliminaries

In this paper, we are solving a non-convex optimization problem with the help of gradient based methods. To follow the analysis, we describe some useful definitions which will be referred throughout the paper.

Definition 1 For a differentiable function $U(x)$, a point $x$ is $\epsilon$ first order stationary point (FOSP) if $\|\nabla U(x)\| \leq \epsilon$.

Definition 2 (Strict saddle point) For a twice differentiable function, a point $x$ is a strict saddle point if $x$ is a FOSP and $\lambda_{\min}(\nabla^2 U(x)) < 0$. For a strict saddle point, the second order stationary point (SOP) is a local minima of the non-convex objective as defined next.

Definition 3 ($\epsilon$ second order stationary) For a non-convex function $U(x)$, a point $x$ is $\epsilon$ second order stationary if

$$\|\nabla U(x)\| \leq \epsilon \quad \text{and} \quad \lambda_{\min}(\nabla^2 U(x)) \geq -\sqrt{L_2}\epsilon.$$ \hspace{1cm} (1)

where $L_2$ is Hessian Lipschitz constant. Note that some other parameters can also be used to define the upper and lower bounds in (1) but the mentioned values are used to make the first condition for SOSP to be same as that for FOSP as in [Jin et al., 2017].

4 Problem Formulation

In this work, we focus on the minimization of a non-convex objective function $U(x)$ given by

$$\min_{x} U(x) \hspace{1cm} (2)$$

where $x \in \mathbb{R}^n$ is the optimization variable, $U(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ is a twice differentiable non-convex function. For a general non-convex optimization problem, it is well known that an algorithm cannot achieve global minima using first order algorithms [Ge et al., 2015; Jin et al., 2017]. In literature, there are various algorithms proposed which converges to the approximate FOSP of the problem (2). A FOSP could be a local minima, local maxima, or a saddle point. It is sufficient to achieve convergence to FOSP in convex settings because it coincides with the global minima of the function. However, for the non-convex functions, it may be highly suboptimal to achieve FOSP if it denotes the local maxima and we are minimizing the function and vice versa. In addition, the converged FOSP can be a saddle point for the unconstrained optimization problem and algorithm might get stuck there. Hence, it becomes important to look for the convergence to a second order stationary point. In literature, various algorithms are proposed for problem (2) with the guarantee of convergence to a second order stationary point [Curtis et al., 2017; Jin et al., 2017; Levy, 2016]. However, they are based on the standard gradient descent algorithm.

Recently, an SCA based algorithm is proposed by [Scutari et al., 2017] to solve the problem of (2) in an iterative manner. We utilize the similar ideas and use SCA to solve the unconstrained version of the problem. The SCA algorithm is shown to outperform the other first order methods in literature [Scutari et al., 2017]. In this method, the non-convex objective function $U(x)$ is approximated by a surrogate convex function $\tilde{U}(x; y)$ approximated at $y$. Utilizing this idea, the standard SCA algorithm to solve the problem in (2) is summarized in Algorithm [1].

In Algorithm [1] at each step $t$, a convex approximation $\tilde{U}(x; x_t)$ of the non-convex objective function $U(x)$ is used to perform the update in step 3. This algorithm converges to the first order stationary point as derived in [Scutari et al., 2017] Theorem 2, which states that for a constant step size $\eta$, the sequence $\{x_t\}$ is bounded and each of
its limit points is stationary point of the optimization problem in (4). But the results presented in [Scutari et al., 2017] are asymptotic in nature and does not provide the number of iterations required to converge to an $\epsilon$ optimal solution. This result is important because it characterizes the convergence behavior of the proposed algorithm. Moreover, as stated earlier, the convergence to FOSP is not a sufficient condition of convergence to local minima. This is because a FOSP could be a saddle point and the proposed algorithm might get stuck there.

In this paper, we are interested in deriving the number of iterations required to achieve an $\epsilon$ FOSP. In addition to that, we want to characterize the number of iterations required to achieve an $\epsilon$ second order stationary point. In literature there are different techniques proposed to escape the saddle points and converge to SOSP. For instant, some of the approaches include methods utilizing second order information [Nesterov, 1998], and random perturbation based methods [Jin et al., 2017]. Motivated from the advantages of perturbation based methods as discussed in [Jin et al., 2017] and advantages of SCA methods [Scutari et al., 2017], we use the perturbed method to make the SCA algorithm escape saddle points effectively. A perturbed successive convex approximation (P-SCA) algorithm is summarized in Algorithm 2.

**Algorithm 1:** Successive convex approximation (SCA)

1: Set $t = 0$, initialize $\eta = (0, 1]$,
2: STOP if $x_t$ is a FOSP
3: Solve the following optimization problem to get $\hat{x}(x_t)$
   \[
   \hat{x}(x_t) := \arg \min_{x \in \mathbb{R}^n} \hat{U}(x; x_t),
   \]
4: Set $x_{t+1} = x_t + \eta(\hat{x}(x_t) - x_t)$
5: Set $t = t + 1$ and go to step 2

In Algorithm 2 the standard SCA algorithm updates are used to reach FOSP. If the condition for FOSP are satisfied (which is nothing but the gradient norm is smaller than a threshold value $g_0$), a random perturbation from uniform ball $B_0(r)$ of radius $r$ around $x_t$ is added. This happens at most once every $t_{\text{go}}$ number of iterations. After adding perturbation, if the function value is not decreased by the threshold $(1-s)f_{th}$ where $s \in (0, 1)$ is a constant and defined later in the paper) then $x_{\text{noise}}$ is returned as the output of Algorithm 2. Here $t_{\text{go}}$ denotes the last iteration at which the optimization variable was perturbed with respect to the current iteration of the algorithm. It is proved in [Scutari et al., 2017] that after the addition of this random perturbations, the standard gradient descent iterates escapes the saddle points efficiently. We extend the analysis in [Scutari et al., 2017] and [Jin et al., 2017] papers to prove that the proposed P-SCA algorithm also escapes the saddle points in a similar manner and $x_{\text{noise}}$ will be in the neighborhood of $\epsilon$ SOSP. We note that the steps in Algorithm 2 do not exhibit a gradient descent like update. Hence, it is difficult to analyze the dynamics of the proposed algorithm. To perform the analysis, it becomes important to look at the proposed algorithm from the perspective of inexact gradient descent algorithms [So and Zhou, 2017b; Bedi et al., 2018; Dixit et al., 2019] and So and Zhou, 2017a]. The meaning of term "inexact" means that the gradient value used for the algorithm update is in error. The error may arise due to stochastic nature of the gradient or partial availability of the gradient [Bertsekas and Tsitsiklis, 2000] etc. In the above mentioned references, all the inexact gradient algorithms are considered for convex objective function settings. The analysis for asymptotic convergence to FOSP under non-convex objective is performed in [Bertsekas and Tsitsiklis, 2000] but with diminishing step size. The work in this paper focuses on the more general scenario of non-convex objective function and inexactness of the gradient [Xiao and Zhang, 2014].

Hence, the main idea for the inexact version is to interpret the algorithm as a gradient descent algorithm with error. It is proved in literature that under some regularity conditions on the error term in the gradient, the algorithm converges to (or provide close approximation to) the optimal value under convexity assumption. In order to proceed with analysis for the proposed P-SCA algorithm, we first show that the SCA algorithm is nothing but an inexact version of the standard gradient descent algorithm. From step 5 of the Algorithm 2 we have

\[
\begin{align*}
x_{t+1} &= x_t + \eta(\hat{x}(x_t) - x_t),
\end{align*}
\]

Next, add and subtract the original gradient $\nabla U(x_t)$ as follows

\[
\begin{align*}
x_{t+1} &= x_t + \eta(\nabla U(x_t) - \nabla U(x_t) + \hat{x}(x_t) - x_t).
\end{align*}
\]

Finally, we can write the P-SCA algorithm update in the following alternative form

\[
\begin{align*}
x_{t+1} &= x_t - \eta \nabla U(x_t) + \eta \left[ \nabla U(x_t) + x_t - \hat{x}(x_t) \right].
\end{align*}
\]

The first term in (7) is similar to the standard gradient descent algorithm. The second term in (7) corresponds to the error in the gradient and denoted by $\epsilon_t$. Hence the final gradient used
for the update is $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta (\nabla U(\mathbf{x}_t) + \mathbf{e}_t)$.

(8)

For the further analysis in this paper, we will utilize the algorithm update form as presented in $[7]$. 

5 Convergence analysis

This section details about the convergence rate analysis of the proposed P-SCA algorithm and show that the number of iterations required to converge to a second order stationary point are of the order of $O\left( \left(\text{polylog}(d)\right) \frac{\Delta U(\mathbf{x}_0) - \Delta U(\mathbf{x}^*)}{\epsilon^2} \right)$. This section also establishes the result that the proposed algorithm indeed escapes the saddle point with probability $1 - \delta$ for any $\delta > 0$. Note that the updates in the proposed algorithm utilize a convex approximations $\tilde{U}(\mathbf{x}; \mathbf{x}_t)$ at each step $t$ for the non-convex function $U(\mathbf{x})$ $[6]$. Before discussing the mathematical results, some assumptions are required to hold for the objective function and its convex approximation at each $t$. All the required assumptions are provided next. That will be followed by the main results.

5.1 Required assumptions

The non-convex objective function is required to satisfy some other technical condition as mentioned here.

Assumption 1 The continuously differentiable objective function $U(\mathbf{x})$ and its gradient $\nabla U(\mathbf{x})$ are Lipschitz continuous with parameter $L_0 > 0$ and $L_1 > 0$, respectively. This implies that

$$\|U(\mathbf{x}) - U(\mathbf{y})\| \leq L_0 \|\mathbf{x} - \mathbf{y}\|$$

(9)

and

$$\|\nabla U(\mathbf{x}) - \nabla U(\mathbf{y})\| \leq L_1 \|\mathbf{x} - \mathbf{y}\|$$

(10)

for all $\mathbf{x}$ and $\mathbf{y}$.

Apart from the objective function, the convex surrogate function needs to satisfy the following assumption.

Assumption 2 The convex approximation function $\tilde{U}(\mathbf{x}; \mathbf{y})$ to non-convex objective function $U(\mathbf{x})$ at $\mathbf{y}$ is continuously differentiable with respect to its first argument such that

B1 $\tilde{U}(\cdot; \mathbf{y})$ is uniformly strongly convex with parameter $C$, which implies that

$$\left(\nabla \tilde{U}(\mathbf{x}; \mathbf{y}) - \nabla \tilde{U}(\mathbf{z}; \mathbf{y})\right)^T(\mathbf{x} - \mathbf{z}) \geq C \|\mathbf{x} - \mathbf{z}\|^2$$

(11)

for all $\mathbf{x}$ and $\mathbf{z}$.

B2 The approximation function gradient $\nabla \tilde{U}(\cdot; \mathbf{y})$ is equal to the original function gradient at the approximation point $\mathbf{y}$

$$\nabla \tilde{U}(\mathbf{y}; \mathbf{y}) = \nabla U(\mathbf{y}).$$

(12)

This property is the key to represent the P-SCA algorithm as an inexact gradient descent algorithm.

Another assumption required for the objective function which upper bounds the rate of change of the Hessian.

Assumption 3 The objective function $U(\mathbf{x})$ is Hessian Lipschitz with parameter $L_2$, which implies that

$$\|\nabla^2 U(\mathbf{x}) - \nabla^2 U(\mathbf{y})\| \leq L_2 \|\mathbf{x} - \mathbf{y}\|$$

(13)

for all $\mathbf{x}$ and $\mathbf{y}$.

All of the above mentioned assumptions are standard and have been considered in literature $[6]$. Assumption $[1]$ states that the non-convex objective function is Lipschitz which means that the gradient of the objective function is smooth and does not changes arbitrarily. This is an important assumption since it provides an upper bound on the gradient at two difference points in terms of the difference between the points itself. Next, assumption $[2]$ is for the convex approximation $\tilde{U}(\mathbf{x}; \mathbf{y})$ to the non-convex objective function $U(\mathbf{x})$ at $\mathbf{y}$, and states that $\tilde{U}(\mathbf{x}; \mathbf{y})$ is strongly convex at given $\mathbf{y}$ with parameter $C$. This assumption assures that the convex optimization problem in $[4]$ has a unique solution. The last assumption $[3]$ states that the gradient of $U(\mathbf{x})$ is smooth and hence the Hessian of the function does not changes abruptly. All of the above mentioned assumptions are utilized to prove the convergence results presented in the following subsection of this paper.

5.2 Main results

In this subsection, we show that it is possible for the SCA algorithm to converge to the second order stationary point with a simple modification. In the SCA algorithm updates, once the gradient norm is less than a threshold value $\delta_{th}$, we add a small random perturbation to current iterate $\mathbf{x}_t$. In other words, a random perturbation is added to the algorithm updates at most once after every $t_{th}$ iterations. To make the analysis simple, similar to $[1]$, we assume that the random perturbation $\xi_t$ is uniformly sampled from a ball $B_0(r)$ of radius $r$ centered around $\mathbf{x}_t$. After adding the perturbation at iteration $t$, if the function value does not decrease by $(1 - s)\delta_{th}$ after $t_{th}$ iterations, then the algorithm stops and return $\mathbf{x}_{th}$. This paper proves that the output $\mathbf{x}_{th}$ is essentially the second order stationary point for the problem in $[4]$. The main result is presented next in Theorem $[1]$.

Theorem 1 Let the assumptions $[1][3]$ are satisfied, there exits a constant $c_{\text{max}}$ such that, for any $\delta > 0$, $\epsilon \leq \frac{L_1}{L_2}$, $\Delta U(\mathbf{x}_0) \geq U(\mathbf{x}_0) - U(\mathbf{x}^*)$, and $\epsilon \leq c_{\text{max}}$, the output of Algorithm $[2]$ is $\epsilon$ second order stationary point with probability $(1 - \delta)$ for problem $[4]$. To achieve the $\epsilon$ second order stationary point, the number of iterations required for the algorithm is given by

$$\mathcal{O}\left(\frac{L_1 \Delta U}{\epsilon^2} \log^4\left(\frac{d L_1 \Delta U}{\epsilon^2 \delta}\right)\right).$$

(14)

It is interesting to note that the convergence rate of the proposed algorithm remains the same as that of the one in $[1]$ Theorem 3. Theorem $[1]$ establishes the fact that the algorithm terminates in finite number of iterations.
Before discussing the proof of Theorem 1 there are some other results which we need to discuss first. The algorithm operation can be divided in two scenarios 1) the iterate is not close to FOSP and the gradient $\nabla U(x_t)$ is large and 2) when gradient is small but the Hessian $\nabla^2 U(x_t)$ has a very small minimum eigenvalue. First, to prove the convergence to FOSP, we need to show that a single iteration of the proposed algorithm results in a descent direction for the objective function $U(x)$ which is stated in Lemma 1.

Lemma 1 Under the Assumptions 1-2, and for a $L_2$ Hessian Lipshitz objective function $U(x)$, one iterate of Algorithm 2 is a descent direction, which implies that

$$U(x_{t+1}) \leq U(x_t) - \eta \|\nabla U(x_t)\|^2$$

where $\eta := \eta C - \eta^2 L_1/2$ with $\eta \leq 2C/L_1$.

Proof: Note that $\tilde{x}(x_t)$ is the solution of strongly convex problem in (4), hence from the first order optimality condition, it holds that

$$(y - \tilde{x}(x_t))^T \nabla \tilde{U}(\tilde{x}(x_t); x_t) \geq 0.$$  (16)

By selecting $y = x_t$ and add subtract $\nabla \tilde{U}(x_t; x_t)$ to the gradient term, we get

$$(x_t - \tilde{x}(x_t))^T (\nabla U(x_t; x_t) - \nabla \tilde{U}(\tilde{x}(x_t; x_t) + \nabla \tilde{U}(x_t; x_t)) \geq 0.$$  (17)

Following the inequality in Assumption, it holds that $\nabla U(x_t) = \nabla \tilde{U}(x_t; x_t)$. This implies

$$(x_t - \tilde{x}(x_t))^T (\nabla U(x_t; x_t) \geq C \|\tilde{x}(x_t) - x_t\|^2.$$  (18)

From the statement of Assumption 1 it holds that the objective function $U(x)$ is Lipschitz continuous gradient with parameter $L_1$, which implies that

$$U(x_{t+1}) \leq U(x_t) + \nabla U(x_t)^T (x_{t+1} - x_t)$$

$$+ \frac{L_1}{2} \|x_{t+1} - x_t\|^2.$$  (19)

From the update for $x_{t+1}$ in Algorithm 2 we get

$$U(x_{t+1}) \leq U(x_t) + \eta \nabla U(x_t)^T (\tilde{x}(x_t) - x_t)$$

$$+ \frac{\eta^2 L_1}{2} \|\tilde{x}(x_t) - x_t\|^2.$$  (20)

Applying the upper bound in (18), we get

$$U(x_{t+1}) \leq U(x_t) - \eta C \|\tilde{x}(x_t) - x_t\|^2$$

$$+ \frac{L_1}{2} \|\eta \tilde{x}(x_t) - x_t\|^2.$$  (21)

$$= U(x_t) + (\eta C - \eta^2 L_1/2) \|\tilde{x}(x_t) - x_t\|^2$$

$$= U(x_t) - \eta \|\tilde{x}(x_t) - x_t\|^2.$$  (22)

where $\eta' := \eta C - \eta^2 L_1/2$. Hence, one run of the Algorithm 2 results in a function value decrement with $\eta < \frac{2C}{L_1}$. The above mentioned analysis helps us to derive an upper bound on the gradient error norm. Consider the additional term in (7), we have

$$\|e_t\| = \|\nabla U(x_t) + x_t - \tilde{x}(x_t)\|$$

$$\leq \|\nabla U(x_t)\| + \|\tilde{x}(x_t) - x_t\|.$$  (23)

From the upper bound in (18) and Lipschitz property of the objective function, we have

$$\|e_t\| \leq L_0 (1 + \frac{1}{C}) =: D.$$  (25)

Before proceeding, let us define the following parameters as specified in [Scutari et al., 2017] which are used for deriving the mathematical results next.

$$F := \frac{\eta L_1}{L_2} \cdot \log^{-3} \left( \frac{d\kappa}{\delta} \right),$$

$$G := \frac{\sqrt{\eta L_1}}{L_2} \cdot \gamma \cdot \log^{-2} \left( \frac{d\kappa}{\delta} \right),$$

where $\gamma$ is the negative eigenvalue and the condition number $\kappa$ is given by $\kappa = \frac{L_1}{L_2} \geq 1$. Once the algorithm iterate $x_t$ is near a FOSP, it holds that $\|\nabla U(x_t)\| \leq g_0$ (gradient is small) and the minimum eigen value of Hesian is largely negative $\lambda_{\text{min}}(\nabla^2 U(x_t)) \leq -\gamma$ (sufficiently negative), then if we perform $x_0 = \tilde{x} + \xi$ where $\xi \in B_\kappa(r)$ with radius $r = \frac{L_1}{\kappa \log(\frac{4\delta}{d\kappa})}$, it holds that

$$U(x_{t+1}) - U(\tilde{x}) \leq -F + 16L_2\eta^3 D^3$$  (27)

for any $T \geq \frac{1}{c_{\text{max}}}$ with probability $1 - \delta$ when the step size is selected as $\eta \leq \frac{c_{\text{max}}}{L_1}$ and $\delta = \frac{d\kappa}{\sqrt{4L\gamma} \exp(-\delta)}$.

The result in above lemma states that adding the perturbation decreases the function values further. By selecting $\eta = \frac{L_1}{\kappa \log(\frac{4\delta}{d\kappa})}$, $\gamma = \sqrt{L_2\gamma}$, and $\delta = \frac{d\kappa}{\sqrt{L_2\gamma} \exp(-\delta)}$, we can restate the result in Lemma 2 follows. If $x_t$ is a FOSP, then after adding perturbation $x_t = \tilde{x}_t + \xi_t$, it holds with probability $1 - \delta$ that

$$U(x_{t+1}) - U(\tilde{x}_t) \leq -f_\theta + 16L_2\eta^3 D^3$$  (28)

with $t_\theta = \frac{\sqrt{\gamma}}{\sqrt{L_2\gamma}}$. Next lemma states that addition of perturbation indeed results in escaping the saddle points.

Lemma 3 Under the conditions similar to Lemma 2 let $g_1$ is the minimum eigen vector of $\nabla^2 U(\tilde{x})$. If we consider two sequences $u_t$ and $w_t$, such that

$$\|u_0 - \tilde{x}\| \leq r, \quad w_0 = u_0 + \mu \gamma g_1; \quad \mu \in [\delta/(2\sqrt{\gamma}), 1],$$

(29)
then it holds that
\[ \min \{ U(u_\tau) - U(u_0), U(w_\tau) - U(w_0) \} \leq -2.5F + 16L23D^3 \]
for any \( \eta \leq \frac{\epsilon_2}{L_1} \) and any \( T \geq \frac{T_{\text{max}}}{c} \).

This lemma states that for two points \( u_0 \) and \( w_0 \), where \( w_0 \) lies in the direction of the minimum eigenvector, one of both sequences \( u_\tau \) and \( w_\tau \) will result in a further decrement of the objective value and hence escapes the saddle point. To prove the statement of Lemma 3 we need another two results stated next. In Lemma 4 we show that if the function value does not decrease after adding perturbation, then all the iterates belongs to a small ball around \( u_0 \). The next Lemma 5 shows that if the iterates starting from \( u_0 \) get stuck, then the sequence \( w_\tau \) will result in decreasing the function value and hence escapes the saddle point. Here, \( w_\tau \) is the sequence obtained after applying SCA algorithm steps to \( w_0 \) by taking a step in the direction of minimum eigenvalue of the Hessian denoted by \( z_1 \) and starting from \( u_0 \).

**Lemma 4** For any constant \( \hat{c} \geq \frac{6}{(12 + \frac{\epsilon_2^2}{L_1} \log \frac{4L}{\epsilon_3})} \), there exists a constant \( c_{\text{max}} \) for any \( \delta \in (0, \frac{\epsilon_2^2}{c} \), \( f(\cdot) \) satisfying the assumptions, and \( \|f(\bar{x})\| \leq G \), for an initial point \( u_0 = \bar{x} + \zeta \) with \( \|u_0 - \bar{x}\| \leq 2r \), let
\[
T := \min \left\{ \inf \left\{ t \mid f(u_\tau) - f(u_0) \leq -3F \right\}, \hat{c}F \right\}.
\]

Then for any \( \eta \leq c_{\text{max}}/L_1 \), it holds that for all \( t < T \) we have \( \|u_\tau - \bar{x}\| \leq Z(L\hat{c}) \), where \( Z := 48 + \frac{\epsilon_2^2}{L_1} \log \left( \frac{4L}{\epsilon_3} \right) \) with \( D = L_0 (1 + \frac{1}{C}) \).

**Lemma 5** There exists \( \hat{c} \), \( c_{\text{max}} \) for any \( \delta \in (0, \frac{\epsilon_2^2}{c} \), \( f(\cdot) \) satisfying the assumptions, and \( \|f(\bar{x})\| \leq G \), for an initial point \( u_0 = \bar{x} + \zeta \) and sequence \{\( u_\tau \), \{\( w_\tau \)\} where \( w_0 = u_0 + \mu \epsilon z_1 \) with \( \mu \in [\frac{1}{2\sqrt{2}}, 1] \), let us define
\[
T = \min \left\{ \inf \left\{ t \mid f(w_\tau) - f(w_0) \leq -3F \right\}, \hat{c}F \right\}.
\]

then for any \( \eta \leq c_{\text{max}}/L_1 \), if \( \|u_\tau - \bar{x}\| \leq Z(L\hat{c}) \) for all \( t < T \), it holds that \( T < \hat{c}F \).

The proof of Lemmas 3 - Lemma 5 are provided in the supplementary material available at [Authors]. After stating all the intermediate results, we are in position to discuss the proof of Theorem 1. The detailed proof is provided next.

**Proof of Theorem 1** Let \( c < \min \{c_{\text{max}}, 2\epsilon\} \), the goal of this proof is to achieve a point \( x \) for which it holds that
\[ \|\nabla U(x)\| \leq g_{\text{th}} = \frac{c\sqrt{c^3}}{L^2} \lambda_{\min}(\nabla^2 U(x)) \geq -\sqrt{cL_2}. \]

Note that \( \epsilon \leq 1 \), \( \chi \geq 1 \) as defined in Algorithm 2 which implies that \( \frac{\epsilon^3}{\chi^2} \leq 1 \) and hence any \( x \) satisfying (33) is an \( \epsilon \) second order stationary point. To proceed with the proof, let us start from \( x_0 \), if \( x_0 \) does not satisfy (33), then there are two possible outcomes

1. \( \|\nabla U(x_0)\| > g_{\text{th}} \). This means that the first order stationary point is not reached yet and hence Algorithm 2 will not perform the perturbation step. From Lemma 1 we have
\[ U(x_1) - U(x_0) \leq \frac{\eta}{2}g_{\text{th}}^2 = \frac{\epsilon^2 c^2}{2\chi^4 L_1}. \]

2. \( \|\nabla U(x_0)\| \leq g_{\text{th}} \). Under this condition, a perturbation step is performed by Algorithm 2, then SCA steps are performed for the next \( t_{\text{th}} \) iterations, and then termination condition is checked. If termination condition is not satisfied, it holds that
\[ U(x_{th}) - U(x_0) \leq -f_{\text{th}} + \frac{16L23^3 D^3}{L_1^2}. \]

where we have substituted step size \( \eta = \frac{\epsilon^2}{L_1} \). Now we select \( \epsilon^3 = \frac{\delta}{16L23^3} \) with \( s \in (0, 1) \) and introduce \( s \) as an adjustment parameter to make \( c < \min \{c_{\text{max}}, 2\epsilon\} \). Note that we cannot make \( s \) arbitrarily small because that would result in a smaller step size \( \eta \) which is proportional to \( c \). Next, we obtain
\[ U(x_{th}) - U(x_0) \leq -(1 - s)f_{\text{th}}. \]

The above results implies that on an average with each step of the algorithm, the function value reduces by
\[ \frac{U(x_{th}) - U(x_0)}{t_{\text{th}}} \leq \frac{c^2 \chi^2}{L^4 L_1}. \]

The function value decreases at least by \( \frac{c^3 \chi^2}{L^4 L_1} \) on an average after running for \( t_{\text{th}} \) iterations. Since the maximum amount by which the function value may decrease is upper bound by \( U(x_0) - U^* \), the algorithm must terminate in the following number of iterations
\[ \frac{c^3 \chi^2}{L^4 L_1} = \frac{\chi^4}{c^3} \cdot \frac{L_1(U(x_0) - U^*)}{c^2} = O \left( \frac{L_1(U(x_0) - U^*)}{c^2} \log \left( \frac{dL_1 \Delta U}{c^2 \delta} \right) \right). \]

From Algorithm 2 note that the perturbation is added at iteration \( t \) if the gradient is small or \( \nabla U(x_t) \leq \eta_{\text{th}} \). From Lemma 2 the probability of happening this at each time is at least
\[ 1 - \delta = \frac{dL_{\text{th}}}{\sqrt{L_2^2} \exp(-\chi)}. \]

In other words, the number of times perturbation is added for one run of the algorithm is given by
\[ \frac{1}{t_{\text{th}}} \frac{dL_1 \Delta U}{c^2} \frac{L_1(U(x_0) - U^*)}{c^2} = \frac{\chi^3}{c} \cdot \frac{L_1(U(x_0) - U^*)}{c^2}. \]

Using the union bound, it holds that Lemma 2 holds for each of the iterations after adding perturbations. This result makes sure that the Algorithm 2 converges to the \( \epsilon \) second order stationary point with probability
\[ 1 - \frac{dL_1 \exp(-\chi)}{\sqrt{L_2^2} \cdot \frac{\chi^3}{c} \cdot \frac{L_1(U(x_0) - U^*)}{c^2}} = 1 - \exp(-\chi) \cdot \frac{dL_1(U(x_0) - U^*)}{\exp(-\chi) \cdot c}. \]
Note that we choose \( \mathcal{X} = 3 \max \{ \log \left( \frac{dL_1(\Delta u)}{c^2} \right), 4 \} \), which implies that \( \chi \geq 12 \), and \( \chi^3 \exp(-\chi) \leq \exp(-\chi/3) \). Therefore, we have

\[
\frac{\chi^3 \exp(-\chi)}{c} \frac{dL_1(U(x_0) - U^*)}{\epsilon^2} \leq \frac{\exp(-\chi/3)}{c} \frac{dL_1(U(x_0) - U^*)}{\epsilon^2} \leq \delta. \tag{41}
\]

which completes the proof. \( \blacksquare \)

6 Conclusion and future directions

This paper considers an algorithm for solving non-convex optimization formed by perturbation of successive convex approximation based method. The proposed perturbed successive convex approximation (P-SCA) algorithm is shown to converge to second order stationary point with a rate similar to the vanilla gradient descent (convergence to first order stationary point) up to a constant factor. Hence, the proposed algorithm can escape the saddle point efficiently utilizing the perturbed variant of the successive convex optimization algorithm. To perform the analysis, the proposed algorithm is proved to be a special case of the standard gradient descent algorithm with the gradient in error, which is called inexact gradient descent in literature. The idea of adding perturbation to the algorithms updates is utilized to escape the saddle points. As a future work, we are interested in considering the constrained version of the proposed algorithm which also escapes the saddle points.

References

[Authors, ] Authors. Supplementary material for “Escaping saddle points with the successive convex approximation algorithm”. [https://gofile.io/?c=PDc5QP]

[Bach, 2014] Francis Bach. Adaptivity of averaged stochastic gradient descent to local strong convexity for logistic regression. Journal of Machine Learning Research, 15:595–627, 2014.

[Bedi et al., 2018] A. S. Bedi, P. Sarma, and K. Rajawat. Tracking moving agents via inexact online gradient descent algorithm. IEEE Journal of Selected Topics in Signal Processing, 12(1):202–217, Feb 2018.

[Bertsekas and Tsitsiklis, 2000] Dimitri P Bertsekas and John N Tsitsiklis. Gradient convergence in gradient methods with inexact gradient descent. SIAM Journal on Optimization, 10(3):627–642, 2000.

[Candás et al., 2015] E. J. Candás, X. Li, and M. Soltanolkotabi. Phase retrieval via wirtinger flow: Theory and algorithms. IEEE Transactions on Information Theory, 61(4):1985–2007, April 2015.

[Curtis et al., 2017] Frank E Curtis, Daniel P Robinson, and Mohammadreza 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.

[Dixit et al., 2019] R. Dixit, A. S. Bedi, R. Tripathi, and K. Rajawat. Online learning with inexact proximal online gradient descent algorithms. IEEE Transactions on Signal Processing, 67(5):1338–1352, March 2019.

[Ge et al., 2015] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points — online stochastic gradient for tensor decomposition. In Proceedings of The 28th Conference on Learning Theory, pages 797–842, 2015.

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

[Jin et al., 2017] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M. Kakade, and Michael I. Jordan. How to escape saddle points efficiently. In Proceedings of the 34th International Conference on Machine Learning, pages 1724–1732, 2017.

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

[Liu et al., 2018] An Liu, Vincent Lau, and Borna Kanaanian. Stochastic successive convex approximation for nonconvex constrained stochastic optimization. arXiv preprint arXiv:1801.08266, 2018.

[Mairal, 2013] Julien Mairal. Stochastic majorization-minimization algorithms for large-scale optimization. In Advances in Neural Information Processing Systems, pages 2283–2291, 2013.

[Nesterov and Poljak, 2006] Yuri Nesterov and B.T. Poljak. Cubic regularization of newton method and its global performance. Mathematical Programming, 108(1):177–205, Aug 2006.

[Nesterov, 1998] Yuri Nesterov. Introductory lectures on convex programming volume i: Basic course. Lecture notes, 1998.

[Nesterov, 2013] Yuri Nesterov. Introductory lectures on convex optimization: A basic course, volume 87. Springer Science & Business Media, 2013.

[Netrapalli et al., 2013] Praneeth Netrapalli, Prateek Jain, and Sujay Sanghavi. Phase retrieval using alternating minimization. In Advances in Neural Information Processing Systems, pages 2796–2804, 2013.

[Ruszczyciaski, 1980] Andrzej Ruszczyciaski. Feasible direction methods for stochastic programming problems. Mathematical Programming, 19(1):220–229, Dec 1980.

[Scutari et al., 2014a] G. Scutari, F. Facchinei, P. Song, D. P. Palomar, and J. Pang. Decomposition by partial linearization: Parallel optimization of multi-agent systems. IEEE Transactions on Signal Processing, 62(3):641–656, Feb 2014.

[Scutari et al., 2017] Gesualdo Scutari, Francisco Facchinei, and Lorenzo Lampariello. Parallel and distributed methods for constrained nonconvex optimization: part i: Theory. IEEE Transactions on Signal Processing, 65(8):1929–1944, 2017.

[So and Zhou, 2017a] Anthony Man-Cho So and Zirui Zhou. Non-asymptotic convergence analysis of inexact gradient methods for machine learning without strong convexity. Optimization Methods and Software, 32(4):963–992, 2017.

[So and Zhou, 2017b] Anthony Man-Cho So and Zirui Zhou. Non-asymptotic convergence analysis of inexact gradient methods for machine learning without strong convexity. Optimization Methods and Software, 32(4):963–992, 2017.

[Spall, 2005] James C Spall. Introduction to stochastic search and optimization: estimation, simulation, and control, volume 65. John Wiley & Sons, 2005.
[Sun and Luo, 2016a] R. Sun and Z. Luo. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, Nov 2016.

[Sun and Luo, 2016b] R. Sun and Z. Luo. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, Nov 2016.

[Sun et al., 2017] Y. Sun, P. Babu, and D. P. Palomar. Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Transactions on Signal Processing*, 65(3):794–816, Feb 2017.

[Sun et al., 2018] Ju Sun, Qing Qu, and John Wright. A geometric analysis of phase retrieval. *Foundations of Computational Mathematics*, 18(5):1131–1198, Oct 2018.

[Wang et al., 2017] Wenqi Wang, Vaneet Aggarwal, and Shuchin Aeron. Efficient low rank tensor ring completion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5697–5705, 2017.

[Wang et al., 2018] Wenqi Wang, Vaneet Aggarwal, and Shuchin Aeron. Tensor train neighborhood preserving embedding. *IEEE Transactions on Signal Processing*, 66(10):2724–2732, 2018.

[Xiao and Zhang, 2014] Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.
Supplementary Material for “Escaping Saddle Points with the Successive Convex Approximation Algorithm”

Appendix 1: Proof of Lemma 2

From Lipschitz continuous gradient property, it holds that

$$U(x_0) \leq U(\bar{x}) + \nabla U(\bar{x})^T(x_0 - \bar{x}) + \frac{L_1}{2} \|x_0 - \bar{x}\|^2.$$  \hspace{1cm} (42)

From the statement of Lemma 2 and Cauchy-Schwarz inequality, it holds that

$$U(x_0) - U(\bar{x}) \leq \|\nabla U(\bar{x})\| \|\xi\| + \frac{L_1}{2} \|\xi\|^2$$ \hspace{1cm} (43)

$$\leq Gr + \frac{L_1}{2} r^2$$

$$\leq \frac{\kappa L}{\eta \log \left(\frac{\eta}{\delta}\right)} + \frac{L_1}{2} \left(\frac{\kappa L}{\eta \log \left(\frac{\delta}{\eta}\right)}\right)^2 \leq \frac{3}{2} F.$$  \hspace{1cm} (44)

The above mentioned bound represents the maximum value by which the function value can increase in the worst case after adding the perturbation. Next, following exactly the similar steps as performed in the proof of Lemma 14 in [Scutari et al., 2017], we obtain

$$U(x_T) - U(\bar{x}) = U(x_T) - U(x_0) + U(x_0) - U(\bar{x})$$

$$\leq -2.5 F + 16 L_2 \eta^3 D^3 + 1.5 F$$

$$\leq -F + 16 L_2 \eta^3 D^3.$$  \hspace{1cm} (45)

Appendix 2: Proof of Lemma 3

We consider $\bar{x} = 0$ without loss of generality, $T^* = \bar{c}T$, and $T'$ defined as

$$T' = \inf \{t \mid \bar{U}_{u_i}(u_t) - U(u_0) \leq -3F\}.$$  \hspace{1cm} (46)

In order to prove the lemma statement, let us consider the two cases.

Case 1 ($T' \leq T^*$): From the statement of Lemma 2, we have $\|u_{T'-1}\| \leq O(L)$ which implies that

$$\|u_{T'}\| = \|u_{T'-1}\| + \eta \|\nabla U(u_{T'-1})\| + \eta \|e_{T'-1}\|$$

$$\leq \|u_{T'-1}\| + \eta \|\nabla U(\bar{x})\| + \eta L_1 \|u_{T'-1}\| + \eta \|e_{T'-1}\|$$

$$\leq (1 + \eta L_1)\bar{c} Z L + \eta G + \eta D.$$  \hspace{1cm} (47)

From the equality $\frac{G \log(\frac{\delta}{\eta})}{\gamma} = L$, we can write

$$\|u_{T'}\| \leq Z_1 L + \eta D$$

where $Z_1 := \left(1 + \eta L_1\right)\bar{c} Z + \eta \frac{\gamma}{\log(\frac{\delta}{\eta})}$. By choosing a small $\eta_1$ and $\eta L_1 \leq \eta_1$, for the Hessian Lipschitz function, we have

$$U(u_{T'}) - U(u_0) \leq \nabla U(u_0)^T (u_{T'} - u_0)$$

$$+ \frac{1}{2} (u_{T'} - u_0)^T \nabla^2 U(u_0) (u_{T'} - u_0)$$

$$+ \frac{L_2}{6} \|u_{T'} - u_0\|^3.$$  \hspace{1cm} (48)

Consider the following quadratic approximation

$$\bar{U}_y(x) := U(y) + \nabla U(y)^T (x - y)$$

$$+ \frac{1}{2} (x - y)^T \nabla^2 U(\bar{x})(x - y).$$  \hspace{1cm} (49)
For a Hessian Lipschitz function, it holds that
\[ U(u_T) - U(u_0) \]
\[ \leq \nabla U(u_0)^T (u_{T'} - u_0) + \frac{1}{2} (u_{T'} - u_0)^T \nabla^2 U(u_0) (u_{T'} - u_0) \]
\[ + \frac{L^2}{6} \| u_{T'} - u_0 \|^3 \]
\[ \leq \nabla U(u_0)^T (u_{T'} - u_0) + \frac{1}{2} (u_{T'} - u_0)^T \nabla^2 U(\tilde{x})(u_{T'} - u_0) \]
\[ + \frac{1}{2} (u_{T'} - u_0)^T (\nabla^2 U(u_0) - \nabla^2 U(\tilde{x}))(u_{T'} - u_0) \]
\[ + \frac{L^2}{6} \| u_{T'} - u_0 \|^3 \]
\[ \leq \tilde{U}_{u_0}(u_{T'}) - U(u_0) + \frac{L^2}{2} \| u_{T'} - \tilde{x} \| \| u_{T'} - u_0 \|^2 \]
\[ + \frac{1}{2} (u_{T'} - u_0)^T \nabla^2 U(u_0) (u_{T'} - u_0) + \frac{L^2}{6} \| u_{T'} - u_0 \|^3 \]
\[ \leq -3F + \mathcal{O}(L^2) + 16L_2 \eta^2 \mathcal{D}^3 \]
\[ = -3F + \mathcal{O}(\sqrt{\eta L_1} F) + 16L_2 \eta^2 \mathcal{D}^3 \]
\[ \leq -2.5F + 16L_2 \eta^2 \mathcal{D}^3 \]

by selecting \( c_{max} \leq \min\{1, \frac{1}{4}\} \) and \( \eta < \frac{1}{\mathcal{D}} \). From Lemma 1, it holds that for any \( T \geq \frac{1}{c_{max}} T \geq \hat{T} = T^* \geq T' \), we can write
\[ U(u_T) - U(u_0) \leq U(u_{T'}) - U(u_0) \]
\[ \leq U(u_{T'}) - U(u_0) \leq -2.5F + 16L_2 \eta^2 \mathcal{D}^3. \]

**Case II** \((T' > T^*)\): For this case also, we know that \( \| u_t \| \leq \mathcal{O}(L) + \eta \mathcal{D} \) for all \( t \leq T^* \). Let us define \( T'' \) as
\[ T'' = \inf \{ t \mid \tilde{U}_{w_0}(w_t) - U(w_0) \leq -3F \}. \]

From Lemma 5 we have \( T'' \leq T^* \). Using the similar arguments as in case 1, we conclude that for all \( T \geq \frac{1}{c_{max}} T \), it holds that
\[ U(w_T) - U(w_0) \leq U(w_{T''}) - U(w_0) \leq -2.5F + 16L_2 \eta^2 \mathcal{D}^3. \]

This concludes the proof.

**Appendix 3: Proof of Lemma 4**

**Proof** The idea here is to show that if function value does not decrease after performing SCA iterations then it must be bounded in a small ball. This is performed by analyzing the update dynamics of the proposed algorithm via decomposing \( d \) dimensional update into two components: one along the subspace \( S \) which is span of significantly negative eigenvalues of the Hessian and second to the subspace complement to it.

Consider the second order Taylor series approximation of the objective function \( U(x) \) evaluated at \( \tilde{x} \) given by
\[ \hat{U}_{\tilde{x}}(x) := U(\tilde{x}) + \nabla U(\tilde{x})^T (x - \tilde{x}) + \frac{1}{2} (x - \tilde{x})^T \mathcal{H} (x - \tilde{x}) \]

where \( \mathcal{H} = \nabla^2 U(\tilde{x}) \). It is emphasized that the gradient of \( U \) can be approximated by gradient of \( \hat{U}_{\tilde{x}} \) provided \( x \) and \( \tilde{x} \) are close. This result is stated as [Nesterov, 2013]
\[ \| \nabla U(x) - \nabla \hat{U}_{\tilde{x}}(x) \| \leq \frac{L^2}{2} \| x - \tilde{x} \|^2 \]

for a \( L_2 \)-Hessian Lipschitz function \( U \). Set \( u_0 = 0 \), from the SCA steps in (7), we can write
\[ u_{t+1} = u_t - \eta \nabla U(u_t) + \eta e_t \]
\[ = u_t - \eta \nabla U(0) - \eta \int_0^1 \nabla^2 U(\theta u_t) d\theta \] \( u_t + \eta e_t \)
\[ = u_t - \eta \nabla U(0) - \eta (\mathcal{H} + \Delta_t) u_t + \eta e_t \]
\[ = (I - \eta \mathcal{H} - \eta \Delta_t) u_t - \eta (\nabla U(0) - e_t) \]
where $\delta_t := \int_0^1 \nabla^2 U(\theta t_0) d\theta - \mathcal{H}$. Next, from the Hessian Lipschitz in \cite{13}, we have $\|\Delta t\| \leq L_2(\|u_t\| + \|x\|)$. From the smoothness of the gradient in Assumption 1, we have

$$\|\nabla U(0) - e_t\| \leq \|\nabla U(0) - \nabla U(\hat{x}) + \nabla U(\hat{x}) + e_t\|$$

(62)

$$\leq \|\nabla U(\hat{x})\| + L_1 \|\hat{x}\| + D \leq G + L_1 \cdot \frac{L}{\kappa \cdot \log \left(\frac{\gamma}{\delta}\right)} + D \leq 3G + D. \quad (63)$$

Next, we will calculate projections of $u_t$ on to the subspaces $S$ and $S_c$, where $S$ is the subspace of eigen vectors whose corresponding eigen values are less than $-\frac{\log \gamma}{\delta}$. Further, $S_c$ defines the subspace of the remaining eigen vectors.

Next, from the definition of $T$, we know that $U_{u_t}(u_t) - U(0) > -3F$. Let $u_0 = 0$, we get

$$-3F < U_{u_t}(u_t) - U(0) = \nabla U(0)^T u_t + \frac{1}{2} u_t^T \mathcal{H} u_t \quad (65)$$

Now decomposing the vector $u_t$ into $\alpha_t$ (projection on to $S$) and $\beta_t$ (projection on to $S_c$), the updates become

$$\alpha_{t+1} = (I - \eta \mathcal{H}) \alpha_t - \eta P_S \Delta_t u_t - \eta P_S (\nabla U(0) - e_t) \quad (66)$$

$$\beta_{t+1} = (I - \eta \mathcal{H}) \beta_t - \eta P_{S_c} \Delta_t u_t - \eta P_{S_c} (\nabla U(0) - e_t). \quad (67)$$

Utilizing these definitions, we get

$$-3F \leq \nabla U(0)^T u_t - \frac{\gamma}{2} \alpha_t^2 + \frac{1}{2} \beta_t^T \mathcal{H} \beta_t \quad (68)$$

where the negative sign comes for the second term comes from the upper bound on the eigen values in subspace $S$. Note that $\|u_t\|^2 = \|\alpha_t\|^2 + \|\beta_t\|^2$, we get

$$-3F \leq \nabla U(0)^T u_t - \frac{\gamma}{2} \|u_t\|^2 - \|\beta_t\|^2 + \frac{1}{2} \beta_t^T \mathcal{H} \beta_t. \quad (69)$$

After rearranging the terms, we get

$$\|u_t\|^2 \leq \frac{2c \log \frac{\gamma}{\delta}}{\gamma} \left(3F + \nabla U(0)^T u_t + \frac{1}{2} \beta_t^T \mathcal{H} \beta_t\right) \quad (70)$$

$$\leq 4 \max \left\{ \frac{6F \log \frac{\gamma}{\delta}}{\gamma}, \frac{6G \log \frac{\gamma}{\delta}}{\gamma}, \frac{\beta_t^T \mathcal{H} \beta_t \log \frac{\gamma}{\delta}}{\gamma}, \|\beta_t\|^2 \right\}. \quad (71)$$

Taking square root on both sides results in

$$\|u_t\| \leq \frac{\sqrt{6F \log \frac{\gamma}{\delta}}}{\gamma} \sqrt{\frac{6G \log \frac{\gamma}{\delta}}{\gamma}}, \sqrt{\frac{\beta_t^T \mathcal{H} \beta_t \log \frac{\gamma}{\delta}}{\gamma}}, \|\beta_t\|. \quad (72)$$

In order to prove the lemma statement

$$\|u_t\| \leq \delta LZ. \quad (73)$$

we proceed by induction. It holds trivially for $t = 0$ since $u_0 = 0$. Assume that (71) holds for $\tau \leq t$, and then it remains to show that (71) holds for $t + 1 < T$. It is sufficient to bound the last two terms of (71) which are $\beta_t^T \mathcal{H} \beta_t$ and $\|\beta_t\|$. Consider the update equation for $\beta_t$ from (67), we have

$$\beta_t = (I - \eta \mathcal{H}) \beta_t + \eta \delta_t \quad (74)$$

where $\|\delta_t\|$ is bounded as

$$\|\delta_t\| \leq \|\Delta_t\| \|u_t\| + \|\nabla U(0) - e_t\| \quad (75)$$

$$\leq L_2(\|u_t\| + \|x\|) \|u_t\| + \|\nabla U(0) - e_t\| \quad (76)$$

$$\leq L_2 \delta Z \left(\frac{2}{\kappa \cdot \log \left(\frac{\gamma}{\delta}\right)}\right) L_2^2 + 3G + D \leq 4G + D \quad (77)$$

$$\leq 4G + D \quad (78)$$
where the last step follows by choosing $c_{\text{max}} \leq \frac{1}{Z\delta (Z+\eta)}$ and step size $\eta < c_{\text{max}}/L$.

**Bounding $\|\beta_{t+1}\|$**: We have

$$\|\beta_{t+1}\| \leq \left(1 + \frac{\eta \gamma}{c \log \left(\frac{d_k}{\delta}\right)}\right) \|\beta_t\| + \eta (4G + D)$$  

(77)

Applying recursively, we get

$$\|\beta_{t+1}\| \leq \sum_{\tau=0}^{t} (4G + D) \left(1 + \frac{\eta \gamma}{c \log \left(\frac{d_k}{\delta}\right)}\right)^\tau \eta$$

$$\leq 3(4G + D)\eta T$$

(78)

$$\leq (12L + \eta DT)\hat{c}.$$  

(79)

Now, from the definition we have $T \leq \hat{c}F$, so that $\left(1 + \frac{\eta \gamma}{c \log \left(\frac{d_k}{\delta}\right)}\right)^T \leq 3$.

**Bounding $\beta^T_{t+1} \mathcal{H} \beta_{t+1}$**: From (72), we can write

$$\beta_t = \sum_{\tau=0}^{t-1} (I - \eta \mathcal{H})^\tau \delta_{\tau},$$

(80)

which implies that

$$\beta^T_{t+1} \mathcal{H} \beta_{t+1} = \eta^2 \sum_{\tau_1=0}^{t} \sum_{\tau_2=0}^{t} \delta^*_{\tau_1} (I - \eta \mathcal{H})^{\tau_1} \mathcal{H} (I - \eta \mathcal{H})^{\tau_2} \delta_{\tau_2}$$

(81)

$$\leq \eta^2 \sum_{\tau_1=0}^{t} \sum_{\tau_2=0}^{t} \|\delta_{\tau_1}\| \left\| (I - \eta \mathcal{H})^{\tau_1} \mathcal{H} (I - \eta \mathcal{H})^{\tau_2} \right\| \|\delta_{\tau_2}\|$$

(82)

$$\leq (4G + D)^2 \eta^2 \sum_{\tau_1=0}^{t} \sum_{\tau_2=0}^{t} \| (I - \eta \mathcal{H})^{\tau_1} \mathcal{H} (I - \eta \mathcal{H})^{\tau_2} \|.$$  

(83)

If $\{\lambda_i\}$ denotes eigen values of $\mathcal{H}$, then $\lambda_i(1 - \eta \lambda_i^{\tau_1 + \tau_2})$ denotes the corresponding eigen value of $(I - \eta \mathcal{H})^{\tau_1} \mathcal{H} (I - \eta \mathcal{H})^{\tau_2}$. The maxima is achieved for $\lambda^*_i = \frac{1}{(1 + \eta \tau_1)\eta}$. Note that the function $\lambda_i(1 - \eta \lambda_i^{\tau_1 + \tau_2})$ is monotonically increasing between $(-\infty, \lambda^*_i)$. Hence,

$$\left\| (I - \eta \mathcal{H})^{\tau_1} \mathcal{H} (I - \eta \mathcal{H})^{\tau_2} \right\| = \max_{i} \lambda_i(1 - \eta \lambda_i)^{\tau_1 + \tau_2}$$

$$\leq \hat{\lambda}(1 - \eta \hat{\lambda})^{\tau_1 + \tau_2}$$

$$\leq \frac{1}{(1 + \tau_1 + \tau_2)\eta}$$

(84)

where $\hat{\lambda} = \min\{L_1, \lambda^*_i + \tau_2\}$. Hence, we get

$$\beta^T_{t+1} \mathcal{H} \beta_{t+1} \leq 2(4G + D)^2 \eta T$$

$$\leq 4(4G^2 \eta \hat{c}T + \eta \hat{c}T D^2)$$

$$\leq 2(4\hat{c}L^2 \gamma \log^{-1} \left(\frac{d_k}{\delta}\right) + \eta \hat{c}T D^2)$$

(85)

which follows from

$$\sum_{\tau_1=0}^{t} \sum_{\tau_2=0}^{t} \frac{1}{(1 + \tau_1 + \tau_2)\eta} = \sum_{\tau=0}^{2t} \min\{1 + \tau, 2t + 1 - \tau\} \cdot \frac{1}{1 + \tau}$$

$$\leq 2t + 1 < 2T.$$  

(86)

Finally, substituting the upper bounds in (79) and (85) into the right hand side of (70), and further simplifying the bounds, we get

$$\|u_t\| \leq Z L \hat{c}.$$  

(87)

where $Z := 48 + \frac{8D}{\gamma L} \log \left(\frac{d_k}{\delta}\right)$ is a constant and $\hat{c}$ is selected such that $\hat{c} \geq \frac{6}{(12 + \frac{8D}{\gamma L} \log \left(\frac{d_k}{\delta}\right))^2}$. This concludes the proof.
Appendix 4: Proof of Lemma 5

Proof: Consider the update equation for $w_t$, we have

$$\begin{align*}
u_{t+1} + v_{t+1} &= w_{t+1} - \eta \nabla U(w_t) + \eta e_t \\
u_{t+1} &= u_t + v_t - \eta \nabla U(u_t + v_t) \\
u_t &= u_t + v_t - \eta \nabla U(u_t) - \eta \int_0^1 \nabla^2 U(u_t + \theta v_t) d\theta v_t + \eta e_t \\
u_t &= u_t - \eta \nabla U(u_t) + \eta e_t + \eta (I - \eta \mathcal{H} - \Delta') v_t \\
u_t &= (I - \eta \mathcal{H} - \Delta') v_t
\end{align*}$$

From now onwards, the proof is exactly similar to that of Lemma 17 in [Scutari et al., 2017] but provided here for completeness. As in the earlier proofs, we have $||\Delta|| \leq L_2(||u_t|| + ||v_t|| + ||x||)$. Now, we can write the update for $v_{t+1}$ as

$$v_{t+1} = (I - \eta \mathcal{H} - \Delta') v_t$$

Note that we have $||w_0 - \hat{x}|| \leq ||u_0 - \hat{x}|| + ||v_0|| \leq \frac{L}{\kappa \log \frac{2\phi}{\delta}}$, from Lemma 4, we have $||w_t|| \leq Z \hat{c}$ for all $t \leq T$. From the condition in Lemma 5 we note that $||u_t|| \leq Z \hat{c}$ for all $t \leq T$. This implies that

$$||v_t|| \leq ||u_t|| + ||w_t|| \leq 2Z \hat{c}$$

for all $t < T$. From here, we can derive that

$$||\Delta'|| \leq L_2(||u_t|| + ||v_t|| + ||x||)$$

Next, we develop a lower bound on $||v_t||$. Let $\psi_t$ be the norm of the projection of $v_t$ onto $z_1$ direction and $\phi_t$ be the norm of the projection of $v_t$ onto the remaining subspace. From (92), we can write

$$\begin{align*}
\psi_t &\geq (1 + \gamma \eta) \psi_t - \mu \sqrt{\psi_t^2 + \phi_t^2} \\
\phi_t &\leq (1 + \gamma \eta) \phi_t + \mu \sqrt{\psi_t^2 + \phi_t^2}
\end{align*}$$

where $\mu = \eta L_2 (2Z \hat{c} + 1)$. By induction, next we prove that the following inequality holds for all $t < T$

$$\phi_t \leq 4\mu \psi_t$$

For $t = 1$ the base case of the induction holds for $t = 0$ from the definition of $v_0$. Now, let us assume that (99) holds for $t \leq T$, we need to show that it holds for $t + 1 \leq T$, we have

$$\begin{align*}
4\mu(t+1) \psi_t &\geq 4\mu(t+1) \left( (1 + \gamma \eta) \psi_t - \mu \sqrt{\psi_t^2 + \phi_t^2} \right) \\
\phi_{t+1} &\leq 4\mu \psi_t(1 + \gamma \eta) \psi_t + \mu \sqrt{\psi_t^2 + \phi_t^2}.
\end{align*}$$

Next, we only need to show that

$$\begin{align*}
(1 + 4\mu(t+1)) \sqrt{\psi_t^2 + \phi_t^2} &\leq 4(1 + \gamma \eta) \psi_t.
\end{align*}$$

By selecting $\sqrt{\phi_{\text{max}}} \leq \frac{1}{2Z \hat{c} + 1} \min\{ \frac{1}{2\sqrt{2}}, \frac{1}{\mu} \}$ and $\eta \leq \frac{\phi_{\text{max}}}{L_1}$, we get

$$\begin{align*}
4\mu(t+1) &\leq 4\mu T \leq 4\eta L_2 (2Z \hat{c} + 1) \hat{c} T \\
&= 4\sqrt{\eta L_1 (2Z \hat{c} + 1) \hat{c}} \leq 1.
\end{align*}$$

This implies that

$$\begin{align*}
4(1 + \gamma \eta) \psi_t &\geq 4\psi_t \leq 2\sqrt{2\phi_t^2} \geq (1 + 4\mu(t+1)) \sqrt{\psi_t^2 + \phi_t^2}
\end{align*}$$
which proves the induction. From (99), we have \( \phi_t \leq 4\mu \psi_t \leq \psi_t \), which gives

\[
\psi_{t+1} \geq (1 + \gamma \eta) \psi_t - \sqrt{2\mu} \psi_t \geq \left(1 + \frac{\gamma \eta}{2}\right) \psi_t
\]  

(105)

where the last inequality follows from \( \mu = \eta L_L (2ZL\hat{c} + 1) \leq \sqrt{c_{\max}}(2ZL\hat{c} + 1) \gamma \eta \log^{-1} \frac{d\kappa}{\delta} < \frac{\gamma \eta}{2\sqrt{2}}. \)

From (93) and (105), we obtain for all \( t < T \)

\[
2ZL\hat{c} \geq \|v_t\| \geq \psi_t \geq \left(1 + \frac{\gamma \eta}{2}\right)^t \psi_0
\]

\[
= \left(1 + \frac{\gamma \eta}{2}\right)^t \frac{L}{\kappa} \log^{-1} \frac{d\kappa}{\delta}
\]

\[
\geq \left(1 + \frac{\gamma \eta}{2}\right)^t \frac{\delta}{2\sqrt{d\kappa}} \log^{-1} \left(\frac{d\kappa}{\delta}\right).
\]

(106)

The above result implies that

\[
T \leq \frac{1}{2} \frac{\log \left(4ZL\hat{c} \cdot \hat{c} \log \left(\frac{d\kappa}{\delta}\right)\right)}{\log \left(1 + \frac{\gamma \eta}{2}\right)}
\]

\[
\leq \frac{1}{2} \frac{\log \left(4ZL\hat{c} \cdot \hat{c} \log \left(\frac{d\kappa}{\delta}\right)\right)}{\gamma \eta}
\]

\[
\leq (2 + \log(4Z\hat{c})) T.
\]

(107)

The inequality follows from the selection \( \delta \in (0, \frac{d\kappa}{\delta}] \) and we have \( \log \left(\frac{d\kappa}{\delta}\right) \geq 1 \). We choose \( \hat{c} \) large enough such that \( 2 + \log(4Z\hat{c}) \leq \hat{c} \), we get \( T < \hat{c} T \), which concludes the proof.