How to Escape Saddle Points Efficiently

Chi Jin [1] **Rong Ge** [2] **Praneeth Netrapalli** [3] **Sham M. Kakade** [4] **Michael I. Jordan** [1]

Abstract

This paper shows that a perturbed form of gradient descent converges to a second-order stationary point in a number iterations which depends only poly-logarithmically on dimension (i.e., it is almost "dimension-free"). The convergence rate of this procedure matches the well-known convergence rate of gradient descent to first-order stationary points, up to log factors. When all saddle points are non-degenerate, all second-order stationary points are local minima, and our result thus shows that perturbed gradient descent can escape saddle points *almost for free*. Our results can be directly applied to many machine learning applications, including deep learning. As a particular concrete example of such an application, we show that our results can be used directly to establish sharp global convergence rates for matrix factorization. Our results rely on a novel characterization of the geometry around saddle points, which may be of independent interest to the non-convex optimization community.

1. Introduction

Given a function $f : \mathbb{R}^d \to \mathbb{R}$, a gradient descent aims to minimize the function via the following iteration:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \nabla f(\mathbf{x}_t),$$

where $\eta > 0$ is a step size. Gradient descent and its variants (e.g., stochastic gradient) are widely used in machine learning applications due to their favorable computational properties. This is notably true in the deep learning setting, where gradients can be computed efficiently via back-propagation (Rumelhart et al., 1988).

Gradient descent is especially useful in high-dimensional settings because the number of iterations required to reach a point with small gradient is independent of the dimension ("dimension-free"). More precisely, for a function that is ℓ-gradient Lipschitz (see Definition 1), it is well known that gradient descent finds an ϵ-first-order stationary point (i.e., a point $\mathbf{x}$ with $\|\nabla f(\mathbf{x})\| \le \epsilon$) within $\ell(f(\mathbf{x}_0) - f^\star)/\epsilon^2$ iterations (Nesterov, 1998), where $\mathbf{x}_0$ is the initial point and $f^\star$ is the optimal value of f. This bound does not depend on the dimension of $\mathbf{x}$. In convex optimization, finding an ϵ-first-order stationary point is equivalent to finding an approximate global optimum.

In non-convex settings, however, convergence to first-order stationary points is not satisfactory. For non-convex functions, first-order stationary points can be global minima, local minima, saddle points or even local maxima. Finding a global minimum can be hard, but fortunately, for many non-convex problems, it is sufficient to find a local minimum. Indeed, a line of recent results show that, in many problems of interest, all local minima are global minima (e.g., in tensor decomposition (Ge et al., 2015), dictionary learning (Sun et al., 2016a), phase retrieval (Sun et al., 2016b), matrix sensing (Bhojanapalli et al., 2016; Park et al., 2016), matrix completion (Ge et al., 2016), and certain classes of deep neural networks (Kawaguchi, 2016)). Moreover, there are suggestions that in more general deep networks most of the local minima are as good as global minima (Choromanska et al., 2014).

On the other hand, saddle points (and local maxima) can correspond to highly suboptimal solutions in many problems (see, e.g., Jain et al., 2015; Sun et al., 2016b). Furthermore, Dauphin et al. (2014) argue that saddle points are ubiquitous in high-dimensional, non-convex optimization problems, and are thus the main bottleneck in training neural networks. Standard analysis of gradient descent cannot distinguish between saddle points and local minima, leaving open the possibility that gradient descent may get stuck at saddle points, either asymptotically or for a sufficiently long time so as to make training times for arriving at a local minimum infeasible. Ge et al. (2015) showed that by adding noise at each step, gradient descent can escape all saddle points in a polynomial number of iterations, provided that the objective function satisfies the strict saddle property (see Assumption A2). Lee et al. (2016) proved that under similar conditions, gradient descent with random initialization avoids saddle points even without adding

[1]University of California, Berkeley [2]Duke University [3]Microsoft Research India [4]University of Washington. Correspondence to: Chi Jin <chijin@berkeley.edu>.

Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

Algorithm 1 Perturbed Gradient Descent (Meta-algorithm)

```
for $t = 0, 1, \ldots$ do
    if perturbation condition holds then
        $\mathbf{x}_t \leftarrow \mathbf{x}_t + \xi_t, \qquad \xi_t$ uniformly $\sim \mathbb{B}_0(r)$
    $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - \eta \nabla f(\mathbf{x}_t)$
```

noise. However, this result does not bound the number of steps needed to reach a local minimum.

Previous work explains why gradient descent avoids saddle points in the nonconvex setting, but not why it is *efficient*—all of them have runtime guarantees with high polynomial dependency in dimension d. For instance, the number of iterations required in Ge et al. (2015) is at least $\Omega(d^4)$, which is prohibitive in high dimensional setting such as deep learning (typically with millions of parameters). Therefore, we wonder whether gradient descent type of algorithms are fundamentally slow in escaping saddle points, or it is the lack of our theoretical understanding while gradient descent is indeed efficient. This motivates the following question: **Can gradient descent escape saddle points and converge to local minima in a number of iterations that is (almost) dimension-free?**

In order to answer this question formally, this paper investigates the complexity of finding ϵ-second-order stationary points. For ρ-Hessian Lipschitz functions (see Definition 5), these points are defined as (Nesterov & Polyak, 2006):

$$\|\nabla f(\mathbf{x})\| \le \epsilon, \qquad \text{and} \qquad \lambda_{\min}(\nabla^2 f(\mathbf{x})) \ge -\sqrt{\rho\epsilon}.$$

Under the assumption that all saddle points are strict (i.e., for any saddle point $\mathbf{x}_s$, $\lambda_{\min}(\nabla^2 f(\mathbf{x}_s)) < 0$), all second-order stationary points ($\epsilon = 0$) are local minima. Therefore, convergence to second-order stationary points is equivalent to convergence to local minima.

This paper studies a simple variant of gradient descent (with phasic perturbations, see Algorithm 1). For ℓ-smooth functions that are also Hessian Lipschitz, we show that perturbed gradient descent will converge to an ϵ-second-order stationary point in $\tilde{O}(\ell(f(\mathbf{x}_0) - f^\star)/\epsilon^2)$, where $\tilde{O}(\cdot)$ hides polylog factors. This guarantee is almost dimension free (up to polylog(d) factors), answering the above highlighted question affirmatively. Note that this rate is exactly the same as the well-known convergence rate of gradient descent to first-order stationary points (Nesterov, 1998), up to log factors. Furthermore, our analysis admits a maximal step size of up to $\Omega(1/\ell)$, which is the same as that in analyses for first-order stationary points.

As many real learning problems present strong *local* geometric properties, similar to strong convexity in the global setting (see, e.g. Bhojanapalli et al., 2016; Sun & Luo, 2016; Zheng & Lafferty, 2016), it is important to note that our analysis naturally takes advantage of such local structure. We show that when local strong convexity is present, the ϵ-dependence goes from a polynomial rate, $1/\epsilon^2$, to linear convergence, $\log(1/\epsilon)$. As an example, we show that sharp global convergence rates can be obtained for matrix factorization as a direct consequence of our analysis.

1.1. Our Contributions

This paper presents the first sharp analysis that shows that (perturbed) gradient descent finds an approximate second-order stationary point in at most $polylog(d)$ iterations, thus escaping all saddle points efficiently. Our main technical contributions are as follows:

- For ℓ-gradient Lipschitz, ρ-Hessian Lipschitz functions (possibly non-convex), gradient descent with appropriate perturbations finds an ϵ-second-order stationary point in $\tilde{O}(\ell(f(\mathbf{x}_0) - f^\star)/\epsilon^2)$ iterations. This rate matches the well-known convergence rate of gradient descent to first-order stationary points up to log factors.
- Under a strict-saddle condition (see Assumption A2), the same convergence result applies for local minima. This means that gradient descent can escape all saddle points with only logarithmic overhead in runtime.
- When the function has local structure, such as local strong convexity (see Assumption A3.a), the above results can be further improved to linear convergence. We give sharp rates that are comparable to previous problem-specific local analysis of gradient descent with smart initialization (see Section 1.2).
- All the above results rely on a new characterization of the geometry around saddle points: points from where gradient descent gets stuck at a saddle point constitute a thin "band." We develop novel techniques to bound the volume of this band. As a result, we can show that after a random perturbation the current point is very unlikely to be in the "band"; hence, efficient escape from the saddle point is possible (see Section 5).

1.2. Related Work

Over the past few years, there have been many problem-specific convergence results for non-convex optimization. One line of work requires a smart initialization algorithm to provide a coarse estimate lying inside a local neighborhood, from which popular local search algorithms enjoy fast local convergence (see, e.g., Netrapalli et al., 2013; Candes et al., 2015; Sun & Luo, 2016; Bhojanapalli et al., 2016). While there are not many results that show global convergence for non-convex problems, Jain et al. (2015) show that gradient descent yields global convergence rates for matrix square-root problems. Although these results

Table 1. Oracle models and iteration complexity for convergence to second-order stationary point

Algorithm	Iterations	Oracle
Ge et al. (2015)	$O(\text{poly}(d/\epsilon))$	Gradient
Levy (2016)	$O(d^3\text{poly}(1/\epsilon))$	Gradient
This Work	$O(\log^4(d)/\epsilon^2)$	Gradient
Agarwal et al. (2016)	$O(\log(d)/\epsilon^{7/4})$	Hessian-vector
Carmon et al. (2016)	$O(\log(d)/\epsilon^{7/4})$	Hessian-vector
Carmon & Duchi (2016)	$O(\log(d)/\epsilon^2)$	Hessian-vector
Nesterov & Polyak (2006)	$O(1/\epsilon^{1.5})$	Hessian
Curtis et al. (2014)	$O(1/\epsilon^{1.5})$	Hessian

give strong guarantees, the analyses are heavily tailored to specific problems, and it is unclear how to generalize them to a wider class of non-convex functions.

For general non-convex optimization, there are a few previous results on finding second-order stationary points. These results can be divided into the following three categories, where, for simplicity of presentation, we only highlight dependence on dimension d and ϵ, assuming that all other problem parameters are constant from the point of view of iteration complexity:

Hessian-based: Traditionally, only second-order optimization methods were known to converge to second-order stationary points. These algorithms rely on computing the Hessian to distinguish between first- and second-order stationary points. Nesterov & Polyak (2006) designed a cubic regularization algorithm which converges to an ϵ-second-order stationary point in $O(1/\epsilon^{1.5})$ iterations. Trust region algorithms (Curtis et al., 2014) can also achieve the same performance if the parameters are chosen carefully. These algorithms typically require the computation of the inverse of the full Hessian per iteration, which can be very expensive.

Hessian-vector-product-based: A number of recent papers have explored the possibility of using only Hessian-vector products instead of full Hessian information in order to find second-order stationary points. These algorithms require a Hessian-vector product oracle: given a function f, a point $\mathbf{x}$ and a direction $\mathbf{u}$, the oracle returns $\nabla^2 f(\mathbf{x}) \cdot \mathbf{u}$. Agarwal et al. (2016) and Carmon et al. (2016) presented accelerated algorithms that can find an ϵ-second-order stationary point in $O(\log d/\epsilon^{7/4})$ steps. Also, Carmon & Duchi (2016) showed by running gradient descent as a subroutine to solve the subproblem of cubic regularization (which requires Hessian-vector product oracle), it is possible to find an ϵ-second-order stationary pointin $O(\log d/\epsilon^2)$ iterations. In many applications such an oracle can be implemented efficiently, in roughly the same complexity as the gradient oracle. Also, when the function has a Hessian Lipschitz property such an oracle can be approximated by differentiating the gradients at two very close points (although this may suffer from numerical issues, thus is seldom used in practice).

Gradient-based: Another recent line of work shows that it is possible to converge to a second-order stationary point without any use of the Hessian. These methods feature simple computation per iteration (only involving gradient operations), and are closest to the algorithms used in practice. Ge et al. (2015) showed that stochastic gradient descent could converge to a second-order stationary point in poly(d/ϵ) iterations, with polynomial of order at least four. This was improved in Levy (2016) to $O(d^3 \cdot \text{poly}(1/\epsilon))$ using normalized gradient descent. The current paper improves on both results by showing that perturbed gradient descent can actually find an ϵ-second-order stationary point in $O(\text{polylog}(d)/\epsilon^2)$ steps, which matches the guarantee for converging to first-order stationary points up to polylog factors.

2. Preliminaries

In this section, we will first introduce our notation, and then present some definitions and existing results in optimization which will be used later.

2.1. Notation

We use bold upper-case letters $\mathbf{A}, \mathbf{B}$ to denote matrices and bold lower-case letters $\mathbf{x}, \mathbf{y}$ to denote vectors. $\mathbf{A}_{ij}$ means the $(i,j)^{\text{th}}$ entry of matrix $\mathbf{A}$. For vectors we use $\|\cdot\|$ to denote the ℓ_2-norm, and for matrices we use $\|\cdot\|$ and $\|\cdot\|_{\text{F}}$ to denote spectral norm and Frobenius norm respectively. We use $\sigma_{\max}(\cdot), \sigma_{\min}(\cdot), \sigma_i(\cdot)$ to denote the largest, the smallest and the i-th largest singular values respectively, and $\lambda_{\max}(\cdot), \lambda_{\min}(\cdot), \lambda_i(\cdot)$ for corresponding eigenvalues.

For a function $f : \mathbb{R}^d \to \mathbb{R}$, we use $\nabla f(\cdot)$ and $\nabla^2 f(\cdot)$ to denote its gradient and Hessian, and $f^\star$ to denote the global minimum of $f(\cdot)$. We use notation $O(\cdot)$ to hide only absolute constants which do not depend on any problem parameter, and notation $\tilde{O}(\cdot)$ to hide only absolute constants and log factors. We let $\mathbb{B}_{\mathbf{x}}^{(d)}(r)$ denote the d-dimensional ball centered at $\mathbf{x}$ with radius r; when it is clear from context, we simply denote it as $\mathbb{B}_{\mathbf{x}}(r)$. We use $\mathcal{P}_{\mathcal{X}}(\cdot)$ to denote projection onto the set $\mathcal{X}$. Distance and projection are always defined in a Euclidean sense.

2.2. Gradient Descent

The theory of gradient descent often takes its point of departure to be the study of convex optimization.

Definition 1. A differentiable function $f(\cdot)$ is **ℓ-smooth (or ℓ-gradient Lipschitz)** if:

$$\forall \mathbf{x}_1, \mathbf{x}_2, \ \|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \le \ell \|\mathbf{x}_1 - \mathbf{x}_2\|.$$

Definition 2. A twice-differentiable function $f(\cdot)$ is **α-strongly convex** if $\forall \mathbf{x}, \ \lambda_{\min}(\nabla^2 f(\mathbf{x})) \ge \alpha$

Such smoothness guarantees imply that the gradient can not change too rapidly, and strong convexity ensures that there is a unique stationary point (and hence a global minimum). Standard analysis using these two properties shows that gradient descent converges linearly to a global optimum $\mathbf{x}^\star$ (see e.g. (Bubeck et al., 2015)).

Theorem 1. *Assume $f(\cdot)$ is ℓ-smooth and α-strongly convex. For any $\epsilon > 0$, if we run gradient descent with step size $\eta = \frac{1}{\ell}$, iterate $\mathbf{x}_t$ will be ϵ-close to $\mathbf{x}^\star$ in iterations:*

$$\frac{2\ell}{\alpha} \log \frac{\|\mathbf{x}_0 - \mathbf{x}^\star\|}{\epsilon}$$

In a more general setting, we no longer have convexity, let alone strong convexity. Though global optima are difficult to achieve in such a setting, it is possible to analyze convergence to first-order stationary points.

Definition 3. For a differentiable function $f(\cdot)$, we say that $\mathbf{x}$ is a **first-order stationary point** if $\|\nabla f(\mathbf{x})\| = 0$; we also say $\mathbf{x}$ is an **ϵ-first-order stationary point** if $\|\nabla f(\mathbf{x})\| \le \epsilon$.

Under an ℓ-smoothness assumption, it is well known that by choosing the step size $\eta = \frac{1}{\ell}$, gradient descent converges to first-order stationary points.

Theorem 2 ((Nesterov, 1998))**.** *Assume that the function $f(\cdot)$ is ℓ-smooth. Then, for any $\epsilon > 0$, if we run gradient descent with step size $\eta = \frac{1}{\ell}$ and termination condition $\|\nabla f(\mathbf{x})\| \le \epsilon$, the output will be ϵ-first-order stationary point, and the algorithm will terminate within the following number of iterations:*

$$\frac{\ell(f(\mathbf{x}_0) - f^\star)}{\epsilon^2}.$$

Note that the iteration complexity does not depend explicitly on intrinsic dimension; in the literature this is referred to as "dimension-free optimization."

Note that a first-order stationary point can be either a local minimum or a saddle point or a local maximum. For minimization problems, saddle points and local maxima are undesirable, and we abuse nomenclature to call both of them "saddle points" in this paper. The formal definition is as follows:

Definition 4. For a differentiable function $f(\cdot)$, we say that $\mathbf{x}$ is a **local minimum** if $\mathbf{x}$ is a first-order stationary point, and there exists $\epsilon > 0$ so that for any $\mathbf{y}$ in the ϵ-neighborhood of $\mathbf{x}$, we have $f(\mathbf{x}) \le f(\mathbf{y})$; we also say $\mathbf{x}$ is a **saddle point** if $\mathbf{x}$ is a first-order stationary point but not a local minimum. For a twice-differentiable function $f(\cdot)$, we further say a saddle point $\mathbf{x}$ is **strict** (or **non-degenerate**) if $\lambda_{\min}(\nabla^2 f(\mathbf{x})) < 0$.

For a twice-differentiable function $f(\cdot)$, we know a saddle point $\mathbf{x}$ must satify $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \le 0$. Intuitively, for saddle point $\mathbf{x}$ to be strict, we simply rule out the undetermined case $\lambda_{\min}(\nabla^2 f(\mathbf{x})) = 0$, where Hessian information alone is not enough to check whether $\mathbf{x}$ is a local minimum or saddle point. In most non-convex problems, saddle points are undesirable.

To escape from saddle points and find local minima in a general setting, we move both the assumptions and guarantees in Theorem 2 one order higher. In particular, we require the Hessian to be Lipschitz:

Definition 5. A twice-differentiable function $f(\cdot)$ is **ρ-Hessian Lipschitz** if:

$$\forall \mathbf{x}_1, \mathbf{x}_2, \ \|\nabla^2 f(\mathbf{x}_1) - \nabla^2 f(\mathbf{x}_2)\| \le \rho \|\mathbf{x}_1 - \mathbf{x}_2\|.$$

That is, Hessian can not change dramatically in terms of spectral norm. We also generalize the definition of first-order stationary point to higher order:

Definition 6. For a ρ-Hessian Lipschitz function $f(\cdot)$, we say that $\mathbf{x}$ is a **second-order stationary point** if $\|\nabla f(\mathbf{x})\| = 0$ and $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \ge 0$; we also say $\mathbf{x}$ is **ϵ-second-order stationary point** if:

$$\|\nabla f(\mathbf{x})\| \le \epsilon, \quad \text{and} \quad \lambda_{\min}(\nabla^2 f(\mathbf{x})) \ge -\sqrt{\rho\epsilon}$$

Second-order stationary points are very important in non-convex optimization because when all saddle points are strict, all second-order stationary points are exactly local minima.

Note that the literature sometime defines ϵ-second-order stationary point by two independent error terms; i.e., letting $\|\nabla f(\mathbf{x})\| \le \epsilon_g$ and $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \ge -\epsilon_H$. We instead follow the convention of Nesterov & Polyak (2006) by choosing $\epsilon_H = \sqrt{\rho \epsilon_g}$ to reflect the natural relations between the gradient and the Hessian.

3. Main Result

In this section we show that it possible to modify gradient descent in a simple way so that the resulting algorithm will provably converge quickly to a second-order stationary point.

Algorithm 2 Perturbed Gradient Descent: $\text{PGD}(\mathbf{x}_0, \ell, \rho, \epsilon, c, \delta, \Delta_f)$

```
$\chi \leftarrow 3\max\{\log(\frac{d\ell\Delta_f}{c\epsilon^2\delta}), 4\}, \eta \leftarrow \frac{c}{\ell}, r \leftarrow \frac{\sqrt{c}}{\chi^2}\cdot\frac{\epsilon}{\ell}$
$g_{\text{thres}} \leftarrow \frac{\sqrt{c}}{\chi^2}\cdot\epsilon, f_{\text{thres}} \leftarrow \frac{c}{\chi^3}\cdot\sqrt{\frac{\epsilon^3}{\rho}}, t_{\text{thres}} \leftarrow \frac{\chi}{c^2}\cdot\frac{\ell}{\sqrt{\rho\epsilon}}$
$t_{\text{noise}} \leftarrow -t_{\text{thres}} - 1$
for $t = 0, 1, \ldots$ do
    if $\|\nabla f(\mathbf{x}_t)\| \le g_{\text{thres}}$ and $t - t_{\text{noise}} > t_{\text{thres}}$ then
        $\tilde{\mathbf{x}}_t \leftarrow \mathbf{x}_t, \quad t_{\text{noise}} \leftarrow t$
        $\mathbf{x}_t \leftarrow \tilde{\mathbf{x}}_t + \xi_t, \qquad \xi_t$ uniformly $\sim \mathbb{B}_0(r)$
    if $t - t_{\text{noise}} = t_{\text{thres}}$ and $f(\mathbf{x}_t) - f(\tilde{\mathbf{x}}_{t_{\text{noise}}}) > -f_{\text{thres}}$ then
        return $\tilde{\mathbf{x}}_{t_{\text{noise}}}$
    $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - \eta\nabla f(\mathbf{x}_t)$
```

The algorithm that we analyze is a perturbed form of gradient descent (see Algorithm 2). The algorithm is based on gradient descent with step size η. When the norm of the current gradient is small ($\le g_{\text{thres}}$) (which indicates that the current iterate $\tilde{\mathbf{x}}_t$ is potentially near a saddle point), the algorithm adds a small random perturbation to the gradient. The perturbation is added at most only once every t_{thres} iterations.

To simplify the analysis we choose the perturbation ξ_t to be uniformly sampled from a d-dimensional ball[1]. The use of the threshold t_{thres} ensures that the dynamics are mostly those of gradient descent. If the function value does not decrease enough (by f_{thres}) after t_{thres} iterations, the algorithm outputs $\tilde{\mathbf{x}}_{t_{\text{noise}}}$. The analysis in this section shows that under this protocol, the output $\tilde{\mathbf{x}}_{t_{\text{noise}}}$ is necessarily "close" to a second-order stationary point.

We first state the assumptions that we require.

Assumption A1. Function $f(\cdot)$ is both ℓ-smooth and ρ-Hessian Lipschitz.

The Hessian Lipschitz condition ensures that the function is well-behaved near a saddle point, and the small perturbation we add will suffice to allow the subsequent gradient updates to escape from the saddle point. More formally, we have:

Theorem 3. *Assume that $f(\cdot)$ satisfies A1. Then there exists an absolute constant $c_{\max}$ such that, for any $\delta > 0, \epsilon \le \frac{\ell^2}{\rho}$, $\Delta_f \ge f(\mathbf{x}_0) - f^\star$, and constant $c \le c_{\max}$, PGD$(\mathbf{x}_0, \ell, \rho, \epsilon, c, \delta, \Delta_f)$ will output an ϵ-second-order stationary point, with probability $1-\delta$, and terminate in the following number of iterations:*

$$O\left(\frac{\ell(f(\mathbf{x}_0) - f^\star)}{\epsilon^2}\log^4\left(\frac{d\ell\Delta_f}{\epsilon^2\delta}\right)\right).$$

[1]Note that uniform sampling from a d-dimensional ball can be done efficiently by sampling $U^{\frac{1}{d}} \times \frac{\mathbf{Y}}{\|\mathbf{Y}\|}$ where $U \sim \text{Uniform}([0,1])$ and $\mathbf{Y} \sim \mathcal{N}(0, \mathbf{I}_d)$ (Harman & Lacko, 2010).

Strikingly, Theorem 3 shows that perturbed gradient descent finds a second-order stationary point in almost the same amount of time that gradient descent takes to find first-order stationary point. The step size η is chosen as $O(1/\ell)$ which is in accord with classical analyses of convergence to first-order stationary points. Though we state the theorem with a certain choice of parameters for simplicity of presentation, our result holds even if we vary the parameters up to constant factors.

Without loss of generality, we can focus on the case $\epsilon \le \ell^2/\rho$, as in Theorem 3. In the case $\epsilon > \ell^2/\rho$, standard gradient descent without perturbation—Theorem 2—easily solves the problem. This is because by A1, we always have $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \ge -\ell \ge -\sqrt{\rho\epsilon}$, which means that all ϵ-second-order stationary points are ϵ-first order stationary points.

We believe that the dependence on at least one $\log d$ factor in the iteration complexity is unavoidable in the nonconvex setting, as our result can be directly applied to the principal component analysis problem, for which the best known runtimes (for the power method or Lanczos method) incur a $\log d$ factor due to random initialization. Establishing this formally is still an open question however.

To provide some intuition for Theorem 3, consider an iterate $\mathbf{x}_t$ which is not yet an ϵ-second-order stationary point. By definition, either (1) the gradient $\nabla f(\mathbf{x}_t)$ is large, or (2) the Hessian $\nabla^2 f(\mathbf{x}_t)$ has a significant negative eigenvalue. Traditional analysis works in the first case. The crucial step in the proof of Theorem 3 involves handling the second case: when the gradient is small $\|\nabla f(\mathbf{x}_t)\| \le g_{\text{thres}}$ and the Hessian has a significant negative eigenvalue $\lambda_{\min}(\nabla^2 f(\tilde{\mathbf{x}}_t)) \le -\sqrt{\rho\epsilon}$, then adding a perturbation, followed by standard gradient descent for t_{thres} steps, decreases the function value by at least f_{thres}, with high probability. The proof of this fact relies on a novel characterization of geometry around saddle points (see Section 5)

If we are able to make stronger assumptions on the objective function we are able to strengthen our main result. This further analysis is presented in the next section.

3.1. Functions with Strict Saddle Property

In many real applications, objective functions further admit the property that all saddle points are strict (Ge et al., 2015; Sun et al., 2016a;b; Bhojanapalli et al., 2016; Ge et al., 2016). In this case, all second-order stationary points are local minima and hence convergence to second-order stationary points (Theorem 3) is equivalent to convergence to local minima.

To state this result formally, we introduce a robust version of the strict saddle property (cf. Ge et al., 2015):

Assumption A2. Function $f(\cdot)$ is (θ, γ, ζ)-**strict saddle**.

That is, for any $\mathbf{x}$, at least one of following holds:

- $\|\nabla f(\mathbf{x})\| \geq \theta$.
- $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \leq -\gamma$.
- $\mathbf{x}$ is ζ-close to $\mathcal{X}^\star$ — the set of local minima.

Intuitively, the strict saddle assumption states that the $\mathbb{R}^d$ space can be divided into three regions: 1) a region where the gradient is large; 2) a region where the Hessian has a significant negative eigenvalue (around saddle point); and 3) the region close to a local minimum. With this assumption, we immediately have the following corollary:

Corollary 4. *Let $f(\cdot)$ satisfy A1 and A2. Then, there exists an absolute constant $c_{\max}$ such that, for any $\delta > 0, \Delta_f \geq f(\mathbf{x}_0) - f^\star$, constant $c \leq c_{\max}$, and letting $\tilde{\epsilon} = \min(\theta, \gamma^2/\rho)$, PGD$(\mathbf{x}_0, \ell, \rho, \tilde{\epsilon}, c, \delta, \Delta_f)$ will output a point ζ-close to $\mathcal{X}^\star$, with probability $1-\delta$, and terminate in the following number of iterations:*

$$O\left(\frac{\ell(f(\mathbf{x}_0) - f^\star)}{\tilde{\epsilon}^2} \log^4\left(\frac{d\ell\Delta_f}{\tilde{\epsilon}^2\delta}\right)\right).$$

Corollary 4 shows after finding $\tilde{\epsilon}$-second-order stationary point by Theorem 3 where $\tilde{\epsilon} = \min(\theta, \gamma^2/\rho)$, the output is also in the ζ-neighborhood of some local minimum.

Note although Corollary 4 only explicitly asserts that the output will lie within some fixed radius ζ from a local minimum. In many real applications, we further have that ζ can be written as a function $\zeta(\theta)$ which decreases linearly or polynomially depending on θ, while γ will be non-decreasing w.r.t θ. In these cases, the above corollary further gives a convergence rate to a local minimum.

3.2. Functions with Strong Local Structure

The convergence rate in Theorem 3 is polynomial in ϵ, which is similar to that of Theorem 2, but is worse than the rate of Theorem 1 because of the lack of strong convexity. Although global strong convexity does not hold in the non-convex setting that is our focus, in many machine learning problems the objective function may have a favorable local structure in the neighborhood of local minima (Ge et al., 2015; Sun et al., 2016a;b; Sun & Luo, 2016). Exploiting this property can lead to much faster convergence (linear convergence) to local minima. One such property that ensures such convergence is a local form of smoothness and strong convexity:

Assumption A3.a. In a ζ-neighborhood of the set of local minima $\mathcal{X}^\star$, the function $f(\cdot)$ is α-**strongly convex**, and β-smooth.

Here we use different letter β to denote the local smoothness parameter (in contrast to the global smoothness parameter ℓ). Note that we always have $\beta \leq \ell$.

Algorithm 3 Perturbed Gradient Descent with Local Improvement: PGDli$(\mathbf{x}_0, \ell, \rho, \epsilon, c, \delta, \Delta_f, \beta)$

```
$\mathbf{x}_0 \leftarrow$ PGD$(\mathbf{x}_0, \ell, \rho, \epsilon, c, \delta, \Delta_f)$
for $t = 0, 1, \ldots$ do
    $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - \frac{1}{\beta}\nabla f(\mathbf{x}_t)$
```

However, often even local α-strong convexity does not hold. We thus introduce the following relaxation:

Assumption A3.b. In a ζ-neighborhood of the set of local minima $\mathcal{X}^\star$, the function $f(\cdot)$ satisfies a (α, β)-**regularity condition** if for any $\mathbf{x}$ in this neighborhood:

$$\langle \nabla f(\mathbf{x}), \mathbf{x} - \mathcal{P}_{\mathcal{X}^\star}(\mathbf{x})\rangle \geq \frac{\alpha}{2}\|\mathbf{x} - \mathcal{P}_{\mathcal{X}^\star}(\mathbf{x})\|^2 + \frac{1}{2\beta}\|\nabla f(\mathbf{x})\|^2. \tag{1}$$

Here $\mathcal{P}_{\mathcal{X}^\star}(\cdot)$ is the projection on to the set $\mathcal{X}^\star$. Note (α, β)-regularity condition is more general and is directly implied by standard β-smooth and α-strongly convex conditions. This regularity condition commonly appears in low-rank problems such as matrix sensing and matrix completion, and has been used in Bhojanapalli et al. (2016); Zheng & Lafferty (2016), where local minima form a connected set, and where the Hessian is strictly positive only with respect to directions pointing outside the set of local minima.

Gradient descent naturally exploits local structure very well. In Algorithm 3, we first run Algorithm 2 to output a point within the neighborhood of a local minimum, and then perform standard gradient descent with step size $\frac{1}{\beta}$. We can then prove the following theorem:

Theorem 5. *Let $f(\cdot)$ satisfy A1, A2, and A3.a (or A3.b). Then there exists an absolute constant $c_{\max}$ such that, for any $\delta > 0, \epsilon > 0, \Delta_f \geq f(\mathbf{x}_0) - f^\star$, constant $c \leq c_{\max}$, and letting $\tilde{\epsilon} = \min(\theta, \gamma^2/\rho)$, PGDli$(\mathbf{x}_0, \ell, \rho, \tilde{\epsilon}, c, \delta, \Delta_f, \beta)$ will output a point that is ϵ-close to $\mathcal{X}^\star$, with probability $1-\delta$, in the following number of iterations:*

$$O\left(\frac{\ell(f(\mathbf{x}_0) - f^\star)}{\tilde{\epsilon}^2} \log^4\left(\frac{d\ell\Delta_f}{\tilde{\epsilon}^2\delta}\right) + \frac{\beta}{\alpha}\log\frac{\zeta}{\epsilon}\right).$$

Theorem 5 says that if strong local structure is present, the convergence rate can be boosted to linear convergence ($\log\frac{1}{\epsilon}$). In this theorem we see that sequence of iterations can be decomposed into two phases. In the first phase, perturbed gradient descent finds a ζ-neighborhood by Corollary 4. In the second phase, standard gradient descent takes us from ζ to ϵ-close to a local minimum. Standard gradient descent and Assumption A3.a (or A3.b) make sure that the iterate never steps out of a ζ-neighborhood in this second phase, giving a result similar to Theorem 1 with linear convergence.

4. Example — Matrix Factorization

As a simple example to illustrate how to apply our general theorems to specific non-convex optimization problems, we consider a symmetric low-rank matrix factorization problem, based on the following objective function:

$$\min_{\mathbf{U}\in\mathbb{R}^{d\times r}} f(\mathbf{U}) = \frac{1}{2}\|\mathbf{U}\mathbf{U}^\top - \mathbf{M}^\star\|_{\mathrm{F}}^2, \qquad (2)$$

where $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$. For simplicity, we assume $\mathrm{rank}(\mathbf{M}^\star) = r$, and denote $\sigma_1^\star := \sigma_1(\mathbf{M}^\star)$, $\sigma_r^\star := \sigma_r(\mathbf{M}^\star)$. Clearly, in this case the global minimum of function value is zero, which is achieved at $\mathbf{V}^\star = \mathbf{T}\mathbf{D}^{1/2}$ where $\mathbf{T}\mathbf{D}\mathbf{T}^\top$ is the SVD of the symmetric real matrix $\mathbf{M}^\star$.

The following two lemmas show that the objective function in Eq. (2) satisfies the geometric assumptions A1, A2,and A3.b. Moreover, all local minima are global minima.

Lemma 6. *For any $\Gamma \ge \sigma_1^\star$, the function $f(\mathbf{U})$ defined in Eq. (2) is 8Γ-smooth and $12\Gamma^{1/2}$-Hessian Lipschitz, inside the region $\{\mathbf{U}|\|\mathbf{U}\|^2 < \Gamma\}$.*

Lemma 7. *For function $f(\mathbf{U})$ defined in Eq. (2), all local minima are global minima. The set of global minima is $\mathcal{X}^\star = \{\mathbf{V}^\star\mathbf{R}|\mathbf{R}\mathbf{R}^\top = \mathbf{R}^\top\mathbf{R} = \mathbf{I}\}$. Furthermore, $f(\mathbf{U})$ is $(\frac{1}{24}(\sigma_r^\star)^{3/2}, \frac{1}{3}\sigma_r^\star, \frac{1}{3}(\sigma_r^\star)^{1/2})$-strict saddle; and satisfies a $(\frac{2}{3}\sigma_r^\star, 10\sigma_1^\star)$-regularity condition in a $\frac{1}{3}(\sigma_r^\star)^{1/2}$-neighborhood of $\mathcal{X}^\star$.*

One caveat is that since the objective function is actually a fourth-order polynomial with respect to $\mathbf{U}$, the smoothness and Hessian Lipschitz parameters from Lemma 6 naturally depend on $\|\mathbf{U}\|$. Fortunately, we can further show that gradient descent (even with perturbation) does not increase $\|\mathbf{U}\|$ beyond $O(\max\{\|\mathbf{U}_0\|, (\sigma_1^\star)^{1/2}\})$. Then, applying Theorem 5 gives:

Theorem 8. *There exists an absolute constant $c_{\max}$ such that the following holds. For the objective function in Eq. (2), for any $\delta > 0$ and constant $c \le c_{\max}$, and for $\Gamma^{1/2} := 2\max\{\|\mathbf{U}_0\|, 3(\sigma_1^\star)^{1/2}\}$, the output of PGDli$(\mathbf{U}_0, 8\Gamma, 12\Gamma^{1/2}, \frac{(\sigma_r^\star)^2}{108\Gamma^{1/2}}, c, \delta, \frac{r\Gamma^2}{2}, 10\sigma_1^\star)$, will be ϵ-close to the global minimum set $\mathcal{X}^\star$, with probability $1-\delta$, after the following number of iterations:*

$$O\left(r\left(\frac{\Gamma}{\sigma_r^\star}\right)^4 \log^4\left(\frac{d\Gamma}{\delta\sigma_r^\star}\right) + \frac{\sigma_1^\star}{\sigma_r^\star}\log\frac{\sigma_r^\star}{\epsilon}\right).$$

Theorem 8 establishes global convergence of perturbed gradient descent from an arbitrary initial point $\mathbf{U}_0$, including exact saddle points. Suppose we initialize at $\mathbf{U}_0 = 0$, then our iteration complexity becomes:

$$O\left(r(\kappa^\star)^4\log^4(d\kappa^\star/\delta) + \kappa^\star\log(\sigma_r^\star/\epsilon)\right),$$

where $\kappa^\star = \sigma_1^\star/\sigma_r^\star$ is the condition number of the matrix $\mathbf{M}^\star$. We see that in the first phase, to move from a neighborhood of the solution, our method requires a number of iterations scaling as $\tilde{O}(r(\kappa^\star)^4)$. We suspect that this strong dependence on condition number arises from our generic assumption that the Hessian Lipschitz is uniformly upper bounded; it may well be the case that this dependence can be reduced in the special case of matrix factorization via a finer analysis of the geometric structure of the problem.

5. Proof Sketch for Theorem 3

In this section we will present the key ideas underlying the main result of this paper (Theorem 3). We will first argue the correctness of Theorem 3 given two important intermediate lemmas. Then we turn to the main lemma, which establishes that gradient descent can escape from saddle points quickly. We present full proofs of all these results in Appendix A. Throughout this section, we use $\eta, r, g_{\text{thres}}, f_{\text{thres}}$ and t_{thres} as defined in Algorithm 2.

5.1. Exploiting Large Gradient or Negative Curvature

Recall that an ϵ-second-order stationary point is a point with a small gradient, and where the Hessian does not have a significant negative eigenvalue. Suppose we are currently at an iterate $\mathbf{x}_t$ that is not an ϵ-second-order stationary point; i.e., it does not satisfy the above properties. There are two possibilities: (1) The gradient is large: $\|\nabla f(\mathbf{x}_t)\| \ge g_{\text{thres}}$; or (2) Around the saddle point we have $\|\nabla f(\mathbf{x}_t)\| \le g_{\text{thres}}$ and $\lambda_{\min}(\nabla^2 f(\mathbf{x}_t)) \le -\sqrt{\rho\epsilon}$.

The following two lemmas address these two cases respectively. They guarantee that perturbed gradient descent will decrease the function value in both scenarios.

Lemma 9 (Gradient). *Assume that $f(\cdot)$ satisfies A1. Then for gradient descent with stepsize $\eta < \frac{1}{\ell}$, we have $f(\mathbf{x}_{t+1}) \le f(\mathbf{x}_t) - \frac{\eta}{2}\|\nabla f(\mathbf{x}_t)\|^2$.*

Lemma 10 (Saddle). *(informal) Assume that $f(\cdot)$ satisfies A1, If $\mathbf{x}_t$ satisfies $\|\nabla f(\mathbf{x}_t)\| \le g_{thres}$ and $\lambda_{\min}(\nabla^2 f(\mathbf{x}_t)) \le -\sqrt{\rho\epsilon}$, then adding one perturbation step followed by t_{thres} steps of gradient descent, we have $f(\mathbf{x}_{t+t_{thres}}) - f(\mathbf{x}_t) \le -f_{thres}$ with high probability.*

We see that Algorithm 2 is designed so that Lemma 10 can be directly applied. According to these two lemmas, perturbed gradient descent will decrease the function value either in the case of a large gradient, or around strict saddle points. Computing the average decrease in function value yields the total iteration complexity. Since Algorithm 2 only terminate when the function value decreases too slowly, this guarantees that the output must be ϵ-second-order stationary point (see Appendix A for formal proofs).

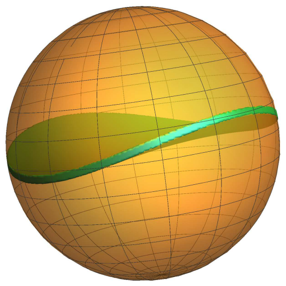

Figure 1. Pertubation ball in 3D and "thin pancake" stuck region

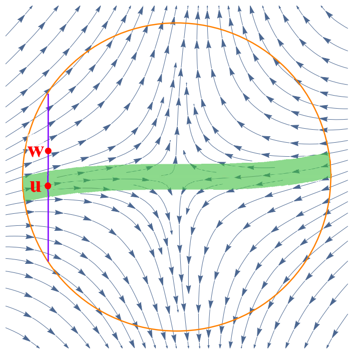

Figure 2. "Narrow band" stuck region in 2D under gradient flow

5.2. Escaping from Saddle Points Quickly

The proof of Lemma 9 is straightforward and follows from traditional analysis. The key technical contribution of this paper is the proof of Lemma 10, which gives a new characterization of the geometry around saddle points.

Consider a point $\tilde{\mathbf{x}}$ that satisfies the the preconditions of Lemma 10 ($\|\nabla f(\tilde{\mathbf{x}})\| \leq g_{\text{thres}}$ and $\lambda_{\min}(\nabla^2 f(\tilde{\mathbf{x}})) \leq -\sqrt{\rho\epsilon}$). After adding the perturbation ($\mathbf{x}_0 = \tilde{\mathbf{x}}+\xi$), we can view $\mathbf{x}_0$ as coming from a uniform distribution over $\mathbb{B}_{\tilde{\mathbf{x}}}(r)$, which we call the **perturbation ball**. We can divide this perturbation ball $\mathbb{B}_{\tilde{\mathbf{x}}}(r)$ into two disjoint regions: (1) an **escaping region** $\mathcal{X}_{\text{escape}}$ which consists of all the points $\mathbf{x} \in \mathbb{B}_{\tilde{\mathbf{x}}}(r)$ whose function value decreases by at least f_{thres} after t_{thres} steps; (2) a **stuck region** $\mathcal{X}_{\text{stuck}} = \mathbb{B}_{\tilde{\mathbf{x}}}(r) - \mathcal{X}_{\text{escape}}$. Our general proof strategy is to show that $\mathcal{X}_{\text{stuck}}$ consists of a very small proportion of the volume of perturbation ball. After adding a perturbation to $\tilde{\mathbf{x}}$, point $\mathbf{x}_0$ has a very small chance of falling in $\mathcal{X}_{\text{stuck}}$, and hence will escape from the saddle point efficiently.

Let us consider the nature of $\mathcal{X}_{\text{stuck}}$. For simplicity, let us imagine that $\tilde{\mathbf{x}}$ is an exact saddle point whose Hessian has only one negative eigenvalue, and $d-1$ positive eigenvalues. Let us denote the minimum eigenvalue direction as $\mathbf{e}_1$. In this case, if the Hessian remains constant (and we have a quadratic function), the stuck region $\mathcal{X}_{\text{stuck}}$ consists of points $\mathbf{x}$ such that $\mathbf{x}-\tilde{\mathbf{x}}$ has a small $\mathbf{e}_1$ component. This is a straight band in two dimensions and a flat disk in high dimensions. However, when the Hessian is not constant, the shape of the stuck region is distorted. In two dimensions, it forms a "narrow band" as plotted in Figure 2 on top of the gradient flow. In three dimensions, it forms a "thin pancake" as shown in Figure 1.

The major challenge here is to bound the volume of this high-dimensional non-flat "pancake" shaped region $\mathcal{X}_{\text{stuck}}$. A crude approximation of this "pancake" by a flat "disk" loses polynomial factors in the dimensionalilty, which gives a suboptimal rate. Our proof relies on the following crucial observation: Although we do not know the explicit form of the stuck region, we know it must be very "thin," therefore it cannot have a large volume. The informal statement of the lemma is as follows:

Lemma 11. *(informal) Suppose $\tilde{\mathbf{x}}$ satisfies the precondition of Lemma 10, and let $\mathbf{e}_1$ be the smallest eigendirection of $\nabla^2 f(\tilde{\mathbf{x}})$. For any $\delta \in (0, 1/3]$ and any two points $\mathbf{w}, \mathbf{u} \in \mathbb{B}_{\tilde{\mathbf{x}}}(r)$, if $\mathbf{w}-\mathbf{u} = \mu r \mathbf{e}_1$ and $\mu \geq \delta/(2\sqrt{d})$, then at least one of $\mathbf{w}, \mathbf{u}$ is not in the stuck region $\mathcal{X}_{stuck}$.*

Using this lemma it is not hard to bound the volume of the stuck region: we can draw a straight line along the $\mathbf{e}_1$ direction which intersects the perturbation ball (shown as purple line segment in Figure 2). For any two points on this line segment that are at least $\delta r/(2\sqrt{d})$ away from each other (shown as red points $\mathbf{w}, \mathbf{u}$ in Figure 2), by Lemma 11, we know at least one of them must not be in $\mathcal{X}_{\text{stuck}}$. This implies if there is one point $\tilde{\mathbf{u}} \in \mathcal{X}_{\text{stuck}}$ on this line segment, then $\mathcal{X}_{\text{stuck}}$ on this line can be at most an interval of length $\delta r/\sqrt{d}$ around $\tilde{\mathbf{u}}$. This establishes the "thickness" of $\mathcal{X}_{\text{stuck}}$ in the $\mathbf{e}_1$ direction, which is turned into an upper bound on the volume of the stuck region $\mathcal{X}_{\text{stuck}}$ by standard calculus.

6. Conclusion

This paper presents the first (nearly) dimension-free result for gradient descent in a general non-convex setting. We present a general convergence result and show how it can be further strengthened when combined with further structure such as strict saddle conditions and/or local regularity/convexity.

There are still many related open problems. First, in the presence of constraints, it is worthwhile to study whether gradient descent still admits similar sharp convergence results. Another important question is whether similar techniques can be applied to accelerated gradient descent. We hope that this result could serve as a first step towards a more general theory with strong, almost dimension free guarantees for non-convex optimization.

References

Agarwal, Naman, Allen-Zhu, Zeyuan, Bullins, Brian, Hazan, Elad, and Ma, Tengyu. Finding approximate local minima for nonconvex optimization in linear time. *arXiv preprint arXiv:1611.01146*, 2016.

Bhojanapalli, Srinadh, Neyshabur, Behnam, and Srebro, Nathan. Global optimality of local search for low rank matrix recovery. *arXiv preprint arXiv:1605.07221*, 2016.

Bubeck, Sébastien et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

Candes, Emmanuel J, Li, Xiaodong, and Soltanolkotabi, Mahdi. Phase retrieval via wirtinger flow: Theory and algorithms. *IEEE Transactions on Information Theory*, 61(4):1985–2007, 2015.

Carmon, Yair and Duchi, John C. Gradient descent efficiently finds the cubic-regularized non-convex newton step. *arXiv preprint arXiv:1612.00547*, 2016.

Carmon, Yair, Duchi, John C, Hinder, Oliver, and Sidford, Aaron. Accelerated methods for non-convex optimization. *arXiv preprint arXiv:1611.00756*, 2016.

Choromanska, Anna, Henaff, Mikael, Mathieu, Michael, Arous, Gérard Ben, and LeCun, Yann. The loss surface of multilayer networks. *arXiv:1412.0233*, 2014.

Curtis, Frank E, Robinson, Daniel P, and Samadi, Mohammadreza. A trust region algorithm with a worst-case iteration complexity of\ mathcal {O}(\ epsilon^{-3/2}) for nonconvex optimization. *Mathematical Programming*, pp. 1–32, 2014.

Dauphin, Yann N, Pascanu, Razvan, Gulcehre, Caglar, Cho, Kyunghyun, Ganguli, Surya, and Bengio, Yoshua. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2933–2941, 2014.

Ge, Rong, Huang, Furong, Jin, Chi, and Yuan, Yang. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *COLT*, 2015.

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

Harman, Radoslav and Lacko, Vladimír. On decompositional algorithms for uniform sampling from n-spheres and n-balls. *Journal of Multivariate Analysis*, 101(10): 2297–2304, 2010.

Jain, Prateek, Jin, Chi, Kakade, Sham M, and Netrapalli, Praneeth. Computing matrix squareroot via non convex local search. *arXiv preprint arXiv:1507.05854*, 2015.

Kawaguchi, Kenji. Deep learning without poor local minima. In *Advances In Neural Information Processing Systems*, pp. 586–594, 2016.

Lee, Jason D, Simchowitz, Max, Jordan, Michael I, and Recht, Benjamin. Gradient descent only converges to minimizers. In *Conference on Learning Theory*, pp. 1246–1257, 2016.

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

Nesterov, Yu. Introductory lectures on convex programming volume i: Basic course. *Lecture notes*, 1998.

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

Netrapalli, Praneeth, Jain, Prateek, and Sanghavi, Sujay. Phase retrieval using alternating minimization. In *Advances in Neural Information Processing Systems*, pp. 2796–2804, 2013.

Park, Dohyung, Kyrillidis, Anastasios, Caramanis, Constantine, and Sanghavi, Sujay. Non-square matrix sensing without spurious local minima via the burer-monteiro approach. *arXiv preprint arXiv:1609.03240*, 2016.

Rumelhart, David E, Hinton, Geoffrey E, and Williams, Ronald J. Learning representations by back-propagating errors. *Cognitive modeling*, 5, 1988.

Sun, Ju, Qu, Qing, and Wright, John. Complete dictionary recovery over the sphere i: Overview and the geometric picture. *IEEE Transactions on Information Theory*, 2016a.

Sun, Ju, Qu, Qing, and Wright, John. A geometric analysis of phase retrieval. In *Information Theory (ISIT), 2016 IEEE International Symposium on*, pp. 2379–2383. IEEE, 2016b.

Sun, Ruoyu and Luo, Zhi-Quan. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, 2016.

Zheng, Qinqing and Lafferty, John. Convergence analysis for rectangular matrix completion using burer-monteiro factorization and gradient descent. *arXiv preprint arXiv:1605.07051*, 2016.