Recent Theoretical Advances in Non-Convex Optimization

Marina Danilova[1,2], Pavel Dvurechensky[3,4], Alexander Gasnikov[2,3,4], Eduard Gorbunov[2,4], Sergey Guminov[2,5], Dmitry Kamzolov[2], Innokentiy Shibaev[2,5]

Abstract Motivated by recent increased interest in optimization algorithms for non-convex optimization in application to training deep neural networks and other optimization problems in data analysis, we give an overview of recent theoretical results on global performance guarantees of optimization algorithms for non-convex optimization. We start with classical arguments showing that general non-convex problems could not be solved efficiently in a reasonable time. Then we give a list of problems which can be solved efficiently to find the global minimizer by exploiting the structure of the problem as much as it is possible. Another way to deal with non-convexity is to relax the goal from finding the global minimum to finding a stationary point or a local minimum. For this setting, we first present known results for the convergence rates of deterministic first-order methods, which are then followed by a general theoretical analysis of optimal stochastic and randomized gradient schemes, and an overview of the stochastic first-order methods. After that, we discuss quite general classes of non-convex problems, such as minimization of α-weakly-quasi-convex functions and functions that satisfy Polyak–Łojasiewicz condition, which still allow obtaining theoretical convergence guarantees of first-order methods. Then we consider higher-order and zeroth-order/derivative-free methods and their convergence rates for non-convex optimization problems.

[1]Institute of Control Sciences RAS, Moscow, Russia
[2]Moscow Institute of Physics and Technology, Moscow, Russia
[3]Weierstrass Institute for Applied Analysis and Stochastics, Berlin, Germany
[4]HSE University, Moscow, Russia
[5]Institute for Information Transmission Problems RAS, Moscow, Russia

1 Introduction

In this review we consider non-convex optimization problems in different settings, including stochastic optimization. We are mainly motivated by an increased interest in such problems in connection to applications in machine learning and data analysis and our main focus is on the methods which possess theoretical guarantees for their global convergence rate or complexity. As we explain first by providing classical examples [151, 157], there is no hope to have any theoretical guarantees for finding a global minimizer in a general non-convex optimization problem in reasonable time. Despite the practical performance of classical general purpose methods such as L-BFGS [166, 80] is quite good, and their local superlinear convergence is proved, their global complexity is not well understood.

In the last 20 years theoretical analysis of the global convergence rate or global complexity guarantees has become de facto a standard in the area of numerical optimization. Since convexity of the problem allows for such an analysis, many global complexity and convergence results have been obtained in convex optimization. Recent advances in machine learning, which were made possible by the application of

neural networks, had lead to the optimization community changing focus to non-convex optimization and, especially to stochastic non-convex optimization. In this, non-exhaustive, review we make an attempt to highlight existing results on global performance guarantees of large-scale non-convex optimization methods. Large dimension of the decision variable in such problems motivates the use of first-order methods, which possess a cheap iteration. Moreover, the large amount of data motivates to use randomized methods such as stochastic gradient descent, which does not require to look through the whole dataset to make one step of the optimization procedure, thus making the iteration even cheaper.

Since, in general, non-convex optimization cannot be made efficiently, we consider several ways to relax this challenging goal. The first relaxation consists in finding problems with hidden convexity or in a convex reformulation of the problem. This requires an exploitation of the problem structure as much as it is possible, which limits the generality of the approach, yet leading to a possibility to find a global solution. Another way is to change the goal from finding the global solution to finding a stationary point or a local extremum. In this case it is possible to obtain polynomial dependence of the complexity of first-order methods on the dimension of the problem and desired accuracy. We consider this approach in the setting of deterministic and stochastic optimization. The third way is to define a class of non-convex problems, which is on the one hand quite general, and on the other hand, allows to obtain global performance guarantees of an algorithm. We consider a class of problems with objective satisfying Polyak–Łojasiewicz condition, which leads to global linear convergence rate, and the class of problems with α-weakly-quasi-convex objective, which leads to global sublinear convergence rate. In the above two approaches we first focus on first-order methods. Then, motivated by several settings in machine learning such as reinforcement learning, black-box adversarial attacks on neural networks, as well as simulation optimization, in which the gradient of the objective is not available, we consider zeroth-order or derivative-free methods and their convergence rates for non-convex optimization problems. By no means we claim that our review contains all the important results in this area since the literature is huge and we could miss some recent results. We would like to list here some other books [126, 55] and reviews [221, 108, 56, 49, 53, 200, 239] related to our paper[1].

2 Preliminaries

The main challenges in non-convex optimization are caused either by non-convexity of the feasible set or by non-convexity of the objective function. The first case is tightly connected with discrete optimization, when the decision variable can take only a discrete set of values. In the second case, yet the variable can take a continuum number of values, the non-convexity of the problem does not allow to

[1] See also this webpage with the list of references being updated `https://sunju.org/research/nonconvex/`.

hope for finding a global solution in a reasonable amount of time. We start with two particular examples which illustrate the intractability of non-convex optimization in general. This intractability motivates different kinds of relaxations, such as changing the goal to the one consisting in finding an approximate stationary point instead of a global minimum, or introducing additional assumptions on the problem, or heavily using the structure of the problem, which lead to provable convergence to the global minimizer. Next, we present general non-convex optimization problems and some ways to classify them.

2.1 Global Optimization is NP-hard

Following [151], we consider an example which illustrates that the problem of finding the exact global solution of a non-convex problem, can be shown to be NP-hard. To that end, we consider the minimization problem

$$\min_{x\in\mathbb{R}^n}\left\{ f(x) := \sum_{i=1}^n x_i^4 - \frac{1}{n}\left(\sum_{i=1}^n x_i^2\right)^2 + \left(\sum_{i=1}^n a_i x_i\right)^4 + (1-x_1)^4 \right\},$$

where x_i is the i-th component of the vector x. Let $A = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, where I is the identity matrix of size n and $\mathbf{1}$ is a vector of n ones, and let $[x]^2$ denote a vector with components $[x]_i^2 = x_i^2$. In this notation, the objective takes the form

$$f(x) = \langle A[x]^2, [x]^2\rangle + \left(\sum_{i=1}^n a_i x_i\right)^4 + (1-x_1)^4.$$

Since A is a positive semidefinite matrix, $f(x) \geqslant 0$. One may also note that 0 is an eigenvalue of A with multiplicity 1 and that $\mathbf{1}$ is the corresponding eigenvector. With this in mind, it is not difficult to see that $f(x) = 0$ if and only if x satisfies

$$a_1 + \sum_{i=2}^n a_i x_i = 0, \quad x_i = \pm 1,\ i = 2,\ldots,n.$$

The problem of checking whether this equation has a solution is a form of the subset sum problem, which is known to be NP-complete. Since this problem has a solution if and only if the global minimum in the original optimization problem is exactly zero, this implies that the problem of finding even the value of a global minimum for a non-convex objective is NP-hard.

2.2 Lower Complexity Bound for Global Optimization

Following [157], we now derive a lower bound for the complexity of finding an approximate global minimum of a possibly non-convex objective. Consider the problem

$$\min_{x\in[0,1]^n} f(x),$$

where f is possibly non-convex and Lipschitz-continuous function, i.e., for some $M > 0$ and for all $x, y \in [0,1]^n$

$$|f(y) - f(x)| \leqslant M \|y - x\|_\infty .$$

Such constant exists for all continuous functions $f(x)$, so this assumption is not restrictive. Let us set the desired accuracy in terms of the objective to ε, i.e. our goal is to find a point $\hat{x}$ such that $f(\hat{x}) - f^* \leqslant \varepsilon$, where f^* is the *global* minimum of f on $[0,1]^n$. For simplicity, we assume ε to be equal to $1/N$ for some $N \in \mathbb{N}$. Consider a family of continuous non-convex objectives $f_k(x)$, $k = 1, \ldots, N^n$, constructed as follows: we divide the hypercube $[0,1]^n$ into $(MN/2)^n$ non-intersecting hypercubes C_k with side length $\frac{2}{NM}$ and set

$$f_k(x) = \begin{cases} -M\mathrm{dist}_\infty(x, \partial C_k), & x \in C_k, \\ 0, & x \notin C_k \end{cases},$$

where ∂C_k is the boundary of C_k and $\mathrm{dist}_\infty(x, \partial C_k)$ is the distance between x and ∂C_k in the $\|\cdot\|_\infty$ norm. Each f_k has a minimum value of exactly $-\varepsilon$ attained at the center of C_k, and the Lipschitz constant of f_k is equal to M.

Any minimization method generating its trajectory based on the values of $f(x)$ and its derivatives at the points of the trajectory would need to sample a point from each C_k to find an approximate minimum of each $f_k(x)$. This gives us a lower bound on the number of iterations required: $\Omega((MN)^n) = \Omega(M^n \varepsilon^{-n})$. And this bound is attained by the algorithm which simply samples the objective values at the vertices of a uniform grid and returns the point with the smallest value. This demonstrates that it is practically impossible to solve a high-dimensional non-convex minimization problem with any reasonable accuracy unless some additional assumptions are introduced.

A similar complexity bound is proved in [156] for finding a point $\hat{x}$ such that $\|\nabla f(\hat{x})\|_\infty \leqslant \varepsilon$ and $\|\hat{x}\|_\infty \leqslant R$. More precisely, for nonconvex functions with Lipschitz continuous Hessian, such that there exists at least one point x^* with $\nabla f(x^*) = 0$ and $\|x^*\|_\infty \leqslant R$, the lower complexity bound is $\Omega\left(\left(\frac{MR^2}{4\varepsilon}\right)^{n/2}\right)$.

2.3 Examples of Non-Convex Problems

In this subsection we make a non-extensive overview of non-convex problem formulations and applications where they arise, with a focus on tractable problems. One possible way to classify such non-convex problems is to divide them into

- problems with hidden convexity or analytic solutions;
- problems with provable global solution.

Let us consider formulations of a few concrete problems in each of these classes.

2.3.1 Problems with Hidden Convexity or Analytic Solutions

Firstly, it is worth noting a wide class of classical non-convex problems, that include linear-fractional programs, geometric programs, problems with two quadratic functions, handling convex equality constraints, convexifying constraint sets. Many such problems are equivalent to convex problems via a simple transformation such as convex relaxation and duality [34].

Next, a wide range of tasks in machine learning and statistics is reduced to eigenproblems. Among these problems are the following principal component analysis, classical multidimensional scaling, and other generalized eigenvalue problems [47].

In the context of non-convex optimization problems, one cannot but mention the class of combinatorial optimization problems as graph problems. Basically, most of these problems are NP-complete, but despite this, there are effective approaches and ways to solve them. Let us consider a closer look at the MAX-CUT problem. This is bright example of convex reformulations. In some problems the goal is to find a point with a value as small as possible (or as large as possible in the context of maximization problems), but whether this point is close to the global minimum is not that important. In this case, we can try to approximate the problem with a simpler one and show that the exact solution to the approximate problem corresponds to a good solution of the original problem. We will first illustrate this idea on the MAX-CUT problem

$$\max_{x\in\{-1,1\}^n}\left\{f(x) := \frac{1}{2}\sum_{i,j=1,1}^{n,n} A_{ij}(x_i - x_j)^2\right\},$$

where $A = \left\|A_{ij}\right\|_{i,j=1,1}^{n,n}$ ($A = A^T$). This is a discrete optimization problem. If we are interested only in the value of the functional and not in the cut itself, we can approximate this problem with a computationally tractable one. Let us introduce

$$L = \operatorname{diag}\left\{\sum_{j=1}^{n} A_{ij}\right\}_{i=1}^{n} - A,$$

which allows us to write

$$f(x) = \langle x, Lx\rangle.$$

A simple observation: if ς is a random vector uniformly distributed on the Hamming cube $\{-1,1\}^n$, then

$$\mathbb{E}\langle \varsigma, L\varsigma\rangle \geq 0.5 \max_{x\in\{-1,1\}^n} \langle x, Lx\rangle .$$

In fact, we can do better, and the construction is due to Goemans and Williamson [87]

$$\max_{x\in\{-1,1\}^n} \langle x, Lx\rangle = \max_{x\in\{-1,1\}^n} \left\langle L, xx^T\right\rangle \leqslant \max_{\substack{X \in S_+^n \\ X_{ii}=1,\ i=1,\ldots,n}} \langle L, X\rangle .$$

This is an SDP problem. Let Σ be the solution of this SDP problem and let

$$\xi \in N(0,\Sigma), \quad \varsigma = \operatorname{sign}(\xi).$$

Then

$$E\langle \varsigma, L\varsigma\rangle \geq \alpha_{GW} \max_{x\in\{-1,1\}^n} \langle x, Lx\rangle ,$$

where $\alpha_{GW} \approx 0.878567$, and this constant is unimprovable provided that P $\neq$ NP and the Unique games conjecture is true [118].

Further, we would like to highlight the following subclasses of non-convex problems: nonconvex proximal operators (Hard-thresholding [28], Potts minimization [120]), discrete problems(Binary graph segmentation, Discrete Potts minimization, Nearly optimal K-means), infinite-dimensional problems (Smoothing splines, Locally adaptive regression splines, Reproducing kernel Hilbert spaces) and statistical problems.

Another important practical example we would like to mention in this part is Blind Deconvolution. Convolutional models arise in a wide range of problems in image processing and computer vision. The most basic convolutional data model – blind deconvolution aims to recover a convolution kernel $a_0 \in \mathbb{R}^k$ and signal $x_0 \in \mathbb{R}^m$ from their convolution

$$y = a_0 \circledast x_0,$$

where $y \in \mathbb{R}^m$ and $\circledast$ is some kind of convolution. This problem is ill-posed in general – there are infinitely many (a_0, x_0) that convolve to produce y. To overcome this issue, some low dimensional priors about a_0 and x_0 are necessary. As a result, it is essential to use additional constraints and regularization terms. Different priors produce different non-convex optimization problems: Sparse Blind Deconvolution [174], Multi-channel Sparse Blind Deconvolution [192], Subspace blind deconvolution [133], Convolutional dictionary learning [169].

In many settings in science and engineering, the observed data are admixtures of multiple latent sources. We would typically want to infer the latent sources as well as the admixture distribution given the observations. Non-negative matrix factorization (NMF) is a natural mathematical framework to model many admixture problems. In NMF we are given an observation matrix $M \in \mathbb{R}^n$, where each row of M corresponds to a data-point in $\mathbb{R}^m$. We assume that there are r latent sources, modeled by the unobserved matrix $W \in \mathbb{R}^{r\times m}$, where each row of M characterizes one source. Each observed data-point is a linear combination of the r sources and

the combination weights are encoded in a matrix $A \in \mathbb{R}^{n\times r}$. Moreover, in many natural settings, the sources are non-negative and the combinations are additive. The computational problem is then is to factor a given matrix M as $M = AW$, where all the entries of M, A and W are non-negative. We call r the inner-dimension of the factorization, and the smallest possible r is usually called the nonnegative rank of M.

Finally, in the part devoted to problems with Hidden Convexity or Analytical Solution we would like deal with Compressed Sensing and L1-optimization. A vector is said to be s-sparse if it has at most s non-zero elements. Consider solving $Ax = b$ for x where A is an $n \times d$ matrix with $n < d$. The set of solutions to $Ax = b$ is a subspace. However, if we restrict ourselves to s-sparse solutions, under certain conditions on A there is a unique sparse solution [26]. For instance, suppose that there were two s-sparse solutions x_1 and x_2. Then $x_1 - x_2$ would be a $2s$-sparse solution to the homogeneous system $Ax = 0$, which would imply that some $2s$ columns of A are linearly dependent. Unless A has 2s linearly dependent columns, there can only be one s-sparse solution.

There are many areas in which the problem is to find the unique sparse solution to a linear system. One is in plant breeding [26]. Assume we are given a number of apple trees and the strength of some desirable feature of each tree. If we wish to determine which genes are responsible for the feature, we may formulate a system of linear equations $Ax = b$ in which each row of the matrix A corresponds to a tree and each column corresponds to a position on the genome. The vector b corresponds to the strength of the desired feature in each tree. The solution x tells us the positions on the genome corresponding to the genes that account for the feature.

The problem of finding a sparse solution can be stated as the optimization problem

$$\min_{Ax=b} \|x\|_0 ,$$

where $\|x\|_0$ is just the number of non-zero coordinates of x. This is an NP-hard problem, but it may sometimes be replaced by the convex problem

$$\min_{Ax=b} \|x\|_1 .$$

What are the sufficient conditions for

$$\min_{Ax=b} \|x\|_0 \quad \Leftrightarrow \quad \min_{Ax=b} \|x\|_1 ?$$

A matrix A is said to satisfy *the s-restricted isometry property* if for any s-sparse x there exists δ_s such that

$$(1-\delta_s)\|x\|_2^2 \leqslant \|Ax\|_2^2 \leqslant (1+\delta_s)\|x\|_2^2 .$$

The following theorems give sufficient conditions for the equivalence mentioned above to hold [26, 39].

Theorem 1. *Suppose A satisfies the s-restricted isometry property with* $\delta_{s+1} \leqslant \frac{1}{10\sqrt{s}}$. *Suppose* x_0 *is s-sparse and satisfies* $Ax_0 = b$. *Then* x_0 *is the unique minimum 1-norm solution to* $Ax = b$.

Theorem 2. *Suppose A satisfies the k-restricted isometry property for* $k \in \{s, 2s, 3s\}$ *with* $\delta_s + \delta_{2s} + \delta_{3s} \leqslant 1$. *Suppose* x_0 *is s-sparse and satisfies* $Ax_0 = b$. *Then* x_0 *is the unique minimum 1-norm solution to* $Ax = b$.

Such results demonstrate the importance of matrices satisfying the restricted isometry property for practice. Fortunately, there is an easy way to obtain such matrices [18].

Theorem 3. *Suppose A is an* $d \times n$ *matrix with elements sampled from the Gaussian distribution* $\mathcal{N}(0, 1/d)$. *Then A satisfies the s-restricted isometry property for* $s < d$ *with* $0 < \delta_s < 1$ *with probability* p_s *satisfying*

$$p_s \geqslant 1 - 2(12/\delta_s)^s \exp\left(-\frac{3\delta_s^2 - \delta_s^3}{48} d\right).$$

2.3.2 Problems with Convergence Results

In this section, we would like to give examples of non-convex optimization problems for which there are methods with proven convergence results. And let us start with the **Phase retrieval problem**. The phase retrieval problem has been a topic of study from at least the early 1980s. It is the recovery of a function given the magnitude of its Fourier transform. This problem could be found in various engineering and scientific applications such as optical imaging, electron microscopy, and crystallography, etc. [189]. We recover a d-dimensional signal vector $x^* \in \mathbb{C}^d$ from its phaseless measurements

$$y_k = |\langle a_k, x \rangle|^2, \quad k = 1, \ldots, M,$$

with a_k denoting the measurement vectors. As a result, the phase-retrieval problem can be formulated as the following least squares problem or empirical risk minimization

$$\min_x \sum_{k=1}^{M} \left(y_k - |\langle a_k, x \rangle|^2\right)^2.$$

This problem is well-motivated by practical concerns, but unfortunately, this is a non-convex problem, and it is not clear how to find a global minimum even if one exists. In recent literature, there are various approaches to handle this problem [222, 203, 50], also, algorithms with the provable convergence results were presented in the following papers [37, 229].

In the context of non-convex optimization problems with proven convergence result, one cannot but mention **Low-Rank Matrix Completion**. There are related problems: matrix completion and matrix sensing [24], which are present in big data

problems with incompleteness and other machine learning problems. We would like to draw attention to the exact low-rank matrix completion. Given a matrix $Y \in \mathbb{R}^{n\times n}$, partially observed, over a set of indices $\Omega \subseteq \{1,\dots,n\}^2$. Consider the problem of finding the lowest-rank matrix matching X on the observed set

$$\min_X \operatorname{rank}(X)$$

$$\text{s.t.} \quad X_{ij} = Y_{ij}, \quad (i,j) \in \Omega.$$

This is a nonconvex problem having a natural convex relaxation

$$\min_X \|X\|_{\text{tr}}$$

$$\text{s.t.} \quad X_{ij} = Y_{ij}, \quad (i,j) \in \Omega$$

In the paper [109] the first results of global optimality of alternating minimization were obtained for matrix completion and the related problem of matrix sensing. Proofs of (nearly) linear convergence of gradient descent for Phase retrieval, Matrix completion, Blind deconvolution can be found in the article [143]. Under some assumptions, it can be shown that the solution to the convex problem is exactly equal to the solution to the non-convex problem, with high probability over the sampling model [40, 38]. So, this problem can also be attributed to statistical problems with hidden convexity.

Deep Learning. In the era of AI, training of the deep neural networks [88] is one of the most popular optimization problems. The simplest example of such problem [200] is training fully connected neural network for supervised learning problem

$$\min_{\substack{W=(W_1,\dots,W_L)\\ W_i\in\mathbb{R}^{n_i\times n_{i-1}},i=1,\dots,L}} \left\{ f(W) := \frac{1}{m}\sum_{i=1}^{m} \ell(y_i, f_{x_i}(W)) \right\},$$

where $\{(x_i,y_i)\}_{i=1}^m$, $x_i \in \mathbb{R}^{n_0}$, $y_i \in \mathbb{R}^{n_y}$ are training data points, $W = (W_1,\dots,W_L)$ are weights of the model, L is number of fully connected layers, $\ell(\cdot,\cdot)$ is a loss function, e.g., quadratic loss or logistic loss, and

$$f_{x_i}(W) = W_L\phi\left(W_{L-1}\phi\dots\phi\left(W_2\phi\left(W_1 x_i\right)\right)\right),$$

where ϕ is a scalar[2] function called an activation function.

In general, training neural networks is NP-complete problem [27]. Deep neural networks have bad local minima both for non-smooth activation functions [202, 181] and smooth ones [136, 230] as well as flat saddles [214]. Nevertheless, there exist positive results about training neural networks. First of all, under different assumptions it was shown that all local minima are global for 1-layer neural networks [198, 101, 78]. Next, one can show that GD/SGD converge under some assumptions to global minimum for linear networks [16, 111, 194] and sufficiently

[2] By $\phi(a)$ where $a = (a_1,\dots,a_n)^\top \in \mathbb{R}^n$ is multidimensional vector we mean vector $(\phi(a_1),\dots,\phi(a_n))^\top$.

wide over-parameterized networks [12]. The detailed summary of recent advances in optimization for deep learning can be found in [200].

2.3.3 Geometry of non-convex optimization problems

In one of the latest review [240], the authors to distinguish a class of tractable non-convex problems, which have certain properties of symmetry. They highlight non-convex optimization problems with rotational symmetry and discrete symmetry. Problems with rotational symmetry include the previously described phase retrieval and related problems in low-rank matrix factorization and recovery. It turns out that the blind deconvolution and tensor decomposition problems have discrete symmetry.

3 Deterministic First-Order Methods

In this section we focus on the following optimization problem

$$\min_{x \in Q \subseteq \mathbb{R}^n} f(x), \tag{1}$$

where Q is a simple, closed, convex, set, and f is continuously differentiable function. The simplest method for this kind of problems is projected gradient descent, which can be motivated by a simple continuous-time dynamics. For simplicity we start with the unconstrained case with $Q = \mathbb{R}^n$.

3.1 Unconstrained Minimization

In the case $Q = \mathbb{R}^n$, the trajectory of the continuous-time gradient method is the solution to the differential equation $\dot{x} = -\nabla f(x(t))$. It is easy to see that $W(x) = f(x(t))$ is a Lyapunov function for this dynamical system. Indeed,

$$\frac{dW(x(t))}{dt} = \left\langle \nabla f(x(t)), \frac{dx(t)}{dt} \right\rangle = \langle \nabla f(x(t)), -\nabla f(x(t)) \rangle = -\|\nabla f(x(t))\|_2^2 \leqslant 0.$$

This implies the convergence of the continuous-time gradient descent method to a stationary point.

The classic gradient descent method is then the Euler discretization of the above dynamics and has the form [172]

$$x^{k+1} = x^k - h_k \nabla f\left(x^k\right),$$

where $h_k \geq 0$ is the step size of the method. One of the main assumptions in this setting is that the function f is L-smooth, or, which is the same, its gradient is Lipschitz-continuous, i.e., for some starting point x^0,

$$\forall x, y \in \left\{x \in \mathbb{R}^n : \ f(x) \leqslant f\left(x^0\right)\right\} \quad \|\nabla f(y) - \nabla f(x)\|_2 \leqslant L\|y - x\|_2 .$$

Then the step size $h = 1/L$ guarantees

$$f\left(x^{k+1}\right) \leqslant f\left(x^k\right) - \frac{1}{2L}\left\|\nabla f\left(x^k\right)\right\|_2^2.$$

Summing up these inequalities, we obtain

$$f(x^N) - f(x^0) \leqslant -\frac{1}{2L}\sum_{k=0}^{N-1}\|\nabla f(x^k)\|_2^2 \leqslant -\frac{N}{2L}\min_{k=0,\ldots,N-1}\|\nabla f(x^k)\|_2^2.$$

Define $f^* = \inf\limits_{x\in\mathbb{R}^n} f(x)$ and assume that this value is finite. Then

$$\min_{k=0,\ldots,N-1}\|\nabla f(x^k)\|_2^2 \leqslant \frac{2L(f(x^0) - f^*)}{N}. \tag{2}$$

This proves that the complexity of finding an approximate stationary point, i.e. a point $\hat{x}$ such that $\|\nabla f(\hat{x})\|_2 \leqslant \varepsilon$ is $O\left(\frac{L(f(x_0)-f^*)}{\varepsilon^2}\right)$. This iteration complexity of finding an ε-stationary point $N \sim \varepsilon^{-2}$ is unimprovable in terms of its dependence on ε and L for an arbitrary first-order method applied to minimization of an L-smooth objective.

On the one hand this bound is much better than the exponential in the dimension bound for finding the global minimum, which was derived in Subsection 2.2. On the other hand we can guarantee only an approximate stationary point, which could be a saddle-point or even a maximum. This can be illustrated by the example of minimization of the following objective [157]

$$f(x_1, x_2) = \frac{1}{2}(x_1)^2 + \frac{1}{2}(x_2)^4 - \frac{1}{2}(x_2)^2 .$$

If we set $x^0 = (1,0)^T$, then x^k converges to $(0,0)^T$ as $k \to \infty$, which is a saddle-point. The good news here is that gradient descent can be perturbed by adding some noise in the iterates in such a way that it converged to a local minimum for almost all initial points and escapes saddle-points [112].

It is important to note that, under additional smoothness assumptions that higher-order derivatives of the objective are Lipschitz continuous, i.e.

$$\forall x, y \in \left\{x \in \mathbb{R}^n : \ f(x) \leqslant f\left(x^0\right)\right\} \quad \|\nabla^p f(y) - \nabla^p f(x)\|_2 \leqslant L_p\|y - x\|_2 ,$$

[46, 45] obtain several lower complexity bounds for finding an approximate stationary point. If this inequality holds for $p \in \{1,2\}$, the lower bound becomes $\varepsilon^{-\frac{12}{7}}$,

and the additional assumption that the same holds for $p = 3$ gives the lower bound to $\varepsilon^{-\frac{8}{5}}$. Surprisingly, Lipschitz continuity of derivatives of order 4 and higher gives the same lower complexity bound.

3.2 Incorporating Simple Constraints

It is possible to generalize gradient method for the setting of composite optimization with simple convex constraints, i.e. for the problem

$$\min_{x \in Q} \{F(x) := f(x) + \psi(x)\},$$

where Q is a closed convex set, $\psi(x)$ is a simple convex function, e.g. $\|x\|_1$, and f is L-smooth function. The standard approach for such problems uses *prox-function* $d(x)$ which is continuously differentiable and strongly convex on Q, i.e. $d(y) - d(x) - \langle \nabla d(x), y - x \rangle \geq \frac{1}{2}\|y - x\|^2$ for any $x, y \in Q$. We define also the corresponding *Bregman divergence* $V[z](x) = d(x) - d(z) - \langle d'(z), x - z \rangle$, $x, z \in Q$. Then the step of the gradient method from a point x with stepsize h is generalized [157, 85] to

$$x^+ = \arg\min_{u \in Q} \left\{ \langle \nabla f(x), u \rangle + \frac{1}{h} V[x](u) + \psi(u) \right\},$$

which in the simplest case $\psi(x) \equiv 0$, $d(x) = \frac{1}{2}\|x\|_2^2$, $V[z](x) = \frac{1}{2}\|x - z\|_2^2$, $Q = \mathbb{R}^n$ coincides with the step of the gradient method. This generalized gradient step leads to a generalized gradient, which is usually referred to as gradient mapping [157, 85] $g_Q(x) = \frac{1}{h}(x - x^+)$. In this setting, the authors of [85] prove that

$$\min_{k=0,\ldots,N-1} \|g_Q(x^k)\|_2^2 \leqslant \frac{2L(F(x^0) - F^*)}{N}$$

if $h = 1/L$. Here F^* is a lower bound for $F(x)$. In the described above simple situation this bound coincides with the bound (2). The authors of [58] prove that if $\|g_Q(x)\| \leqslant \varepsilon$, then x^+ is an approximately stationary point of the problem. More precisely, there exist $p \in \partial\psi(x^+)$ such that

$$\nabla f(x^+) + p \in -\mathcal{N}_Q(x^+) + \mathcal{B}((1 + L(d)\varepsilon),$$

where $\mathcal{N}_Q(x^+)$ is the normal cone of Q at the point x^+, $B(r) = \{v \in \mathbb{R}^n : \|v\|_* \leqslant r\}$ – ball in the dual space defined by the conjugate norm, and it is assumed that d is $L(d)$-smooth. Note that there is no contradiction with the exponential lower bound given in the end of Subsection 2.2 since non-necessarily the obtained point x^+ has small norm of the gradient.

This approach was further generalized in [29, 72, 81] for the case of optimization with inexact oracle for the function f.

Definition 1 *We say that a function $f(x)$ is equipped with an inexact first-order oracle on a set X if there exists $\delta_u > 0$ and at any point $x \in X$ for any number $\delta_c > 0$ there exists a constant $L(\delta_c) \in (0, +\infty)$ and one can calculate $\tilde{f}(x, \delta_c, \delta_u) \in \mathbb{R}$ and $\tilde{g}(x, \delta_c, \delta_u) \in \mathbb{R}^n$ satisfying*

$$|f(x) - \tilde{f}(x, \delta_c, \delta_u)| \leqslant \delta_c + \delta_u,$$
$$f(y) - (\tilde{f}(x, \delta_c, \delta_u) - \langle \tilde{g}(x, \delta_c, \delta_u), y - x \rangle) \leqslant \frac{L(\delta_c)}{2}\|x - y\|^2 + \delta_c + \delta_u, \quad \forall y \in Q.$$

In this definition, δ_c represents the error of the oracle, which we can control and make as small as we would like to. On the opposite, δ_u represents the error, which we can not control. The proposed for this setting method in [72] is adaptive to the constant L, works under inexact calculation of the point x^+, and covers several different settings. In particular, smooth functions with Hölder-continuous, i.e. satisfying, for some $\nu \in [0,1]$, $\|\nabla f(x) - \nabla f(y)\|_* \leqslant L_\nu \|x - y\|^\nu, \forall x, y \in Q$ gradient satisfy this definition with $\delta_u = 0$ and

$$L(\delta_c) = \left(\frac{1-\nu}{1+\nu} \cdot \frac{2}{\delta_c}\right)^{\frac{1-\nu}{1+\nu}} L_\nu^{\frac{2}{1+\nu}}.$$

As a corollary of the general method, [72] propose a universal method for such problems, which does not require the knowledge of the constants ν, L_ν and gives the following convergence rate

$$\min_{k=0,\ldots,N-1} \|g_Q(x_k)\|^2 \leqslant 2^{\frac{1+3\nu}{2\nu}} \left(\frac{1-\nu}{1+\nu} \cdot \frac{40}{\varepsilon}\right)^{\frac{1-\nu}{2\nu}} L_\nu^{\frac{1}{\nu}} \left(\frac{F(x^0) - F^*}{N}\right) + \frac{\varepsilon}{2},$$

or the following complexity estimate $\frac{L_\nu^{\frac{1}{\nu}}(F(x^0) - F^*)}{\varepsilon^{\frac{1+\nu}{2\nu}}}$ to find $\left\|g_Q(x^k)\right\|^2 \leqslant \varepsilon$.

3.3 Incorporating Momentum for Acceleration

The considered above dynamical system $\dot{x} = -\nabla f(x(t))$ does not have any mechanical intuition behind it. In [173] the author proposed to consider the following dynamics

$$\mu \ddot{x}(t) = -\nabla f(x(t)) - p\dot{x}(t).$$

One of the ways to discretize it gives the so called heavy-ball method

$$x^{k+1} = x^k - h\nabla f(x^k) + \beta(x^k - x^{k-1}),$$

where $h > 0$ is the stepsize and $\beta > 0$ is the momentum parameter. Due to the momentum term $\beta\left(x^k - x^{k-1}\right)$ the method avoids zigzagging for ill-conditioned problems, which leads to significant efficiency in practice, especially in training neural networks. Despite practical efficiency, the theoretical guarantee for this method is

no better than for the gradient method. In particular, [97] considers the dynamical system

$$\mu(t)\ddot{x}(t) = -\nabla f(x(t)) - p(t)\dot{x}(t),$$

where $\mu(t) \sim (f(x(t)) - c)$, c is an upper bound on the global minimum of $f(x)$, and $p(t) = F(\nabla f(x(t)))$. With a special choice of $F(\cdot)$, they show that $x(t)$ converges to a local minimizer x^{loc} such that $f\left(x^{loc}\right) \leqslant c$ as $t \to +\infty$. In [66] it is shown that for a discretization of a further generalization of the heavy-ball method one may guarantee

$$\min_{k=1,\ldots,N} \|\nabla f(x^k)\|_2^2 \leqslant \frac{2L(f(x^0) - f^*)}{N},$$

which coincides with the bound (2) for the gradient method.

A different type of momentum was proposed in [154] for convex optimization, which led to the Nesterov's accelerated gradient method

$$x^1 = x^0 - h\nabla f\left(x^0\right),$$

$$x^{k+1} = x^k - h\nabla f(x^k + \beta_k(x^k - x^{k-1})) + \beta_k(x^k - x^{k-1}).$$

The difference with the heavy-ball method is that the gradient is calculated in the extrapolated point. This idea has been very fruitful and allowed to obtain many accelerated algorithms for convex optimization. A variant of this method with a special choice of the stepsize h and momentum term β_k was shown in [84] to have the same convergence rate (2) as the gradient method. This was further extended in [86] for the case of objective with Hölder-continuous gradients to obtain a bound $\frac{L_\nu^{\frac{1}{\nu}}(F(x^0)-F^*)}{\varepsilon^{\frac{1+\nu}{2\nu}}}$ to find $\left\|g_Q(x^k)\right\|^2 \leqslant \varepsilon$ in the general setting of composite optimization with simple constraints. Importantly, this method is universal and uniform, which means that it has best possible convergence rates for convex and non-convex problems without knowing whether the problem is convex or not and without knowing its smoothness parameters such as Hölder exponent and Hölder constant.

It is possible to combine this idea with the idea of line-search, i.e. minimization in the direction of the step. The papers [100, 158] propose a modification of the accelerated gradient method which is listed as Algorithm 1. Instead of explicitly defining the stepsize h and the momentum term β, this method uses full one-dimensional relaxation and local information. This makes this method parameter-free and uniform for convex and non-convex smooth optimization by providing optimal complexity bound for the convex and non-convex case. At the same time, inexact line-search is possible and its sufficient accuracy for achieving the desired accuracy is estimated. This method shares some similarities with nonlinear conjugate gradient methods which were analyzed in [153].

The above idea was further extended in [98] where an accelerated alternating minimization method was proposed and analyzed for convex and non-convex problems. The main assumption is that the set of coordinates is divided into N disjoint subsets (blocks) I_p, $p \in \{1,\ldots,N\}$ and minimization in each block when the other variables are freezed can be made explicitly. The resulting accelerated alternating

Algorithm 1 Accelerated Gradient Method with Small-Dimensional Relaxation (AGMsDR)

Ensure: x^k

1: Set $k=0$, $A_0=0$, $x^0=v^0$, $\psi_0(x)=V[x^0](x)$
2: **for** $k \geqslant 0$ **do**
3:
$$\beta_k = \arg\min_{\beta\in[0,1]} f\left(v^k+\beta(x^k-v^k)\right), \quad y^k = v^k+\beta_k(x^k-v^k).$$
4: Let $(\nabla f(y^k))^{\#}$ be such that $\langle \nabla f(y^k), (\nabla f(y^k))^{\#}\rangle = \|\nabla f(y^k)\|_*^2$ and $\|(\nabla f(y^k))^{\#}\|^2 = 1$.
$$h_{k+1} = \arg\min_{h\geqslant 0} f\left(y^k - h(\nabla f(y^k))^{\#}\right), \quad x^{k+1} = y^k - h_{k+1}(\nabla f(y^k))^{\#}.$$
Find a_{k+1} from equation $f(y^k) - \frac{a_{k+1}^2}{2(A_k+a_{k+1})}\|\nabla f(y^k)\|_*^2 = f(x^{k+1})$.
5: Set $A_{k+1} = A_k + a_{k+1}$.
6: Set $\psi_{k+1}(x) = \psi_k(x) + a_{k+1}\{f(y^k) + \langle \nabla f(y^k), x-y^k\rangle\}$.
7: $v^{k+1} = \arg\min_{x\in\mathbb{R}^n} \psi_{k+1}(x)$, $k = k+1$
8: **end for**

minimization algorithm is listed as Algorithm 2. This method is also parameter-free and uniform for convex and non-convex smooth optimization with optimal complexity bound for the convex and non-convex case.

Algorithm 2 Accelerated Alternating Minimization (AAM)

Require: Starting point x^0.
Ensure: x^k

1: Set $A_0=0$, $x^0=v^0$.
2: **for** $k \geqslant 0$ **do**
3: Set $\beta_k = \arg\min_{\beta\in[0,1]} f\left(x^k+\beta(v^k-x^k)\right)$
4: Set $y^k = x^k+\beta_k(v^k-x^k)$
5: Choose $i_k = \arg\max_{i\in\{1,\dots,n\}} \|\nabla_i f(y^k)\|_2^2$
6: Set $x^{k+1} = \arg\min_{x\in S_{i_k}(y^k)} f(x)$, i.e. minimize f in the corresponding block.
7: Find a_{k+1}, $A_{k+1} = A_k + a_{k+1}$ from
$$f(y^k) - \frac{a_{k+1}^2}{2A_{k+1}}\|\nabla f(y^k)\|_2^2 = f(x^{k+1})$$
8: Set $v^{k+1} = v^k - a_{k+1}\nabla f(y^k)$
9: **end for**

By exploiting the idea of Nesterov's acceleration and combining it with the notion of negative curvature, the authors of [43] manage to accelerate first-order methods for non-convex optimization under additional assumptions that second and third derivatives are Lipschitz continuous. More precisely, if L-smooth function has also Lipschitz continuous Hessian, they obtain complexity $O\left(\varepsilon^{-7/4}\log(1/\varepsilon)\right)$ to find a

point $\hat{x}$ such that $\|\nabla f(\hat{x})\|_2 \leqslant \varepsilon$. Assuming additionally that the third derivative is Lipschitz, this bound is improved to $O\left(\varepsilon^{-5/3}\log(1/\varepsilon)\right)$.

4 Stochastic First-Order Methods

In this section, we consider the same problem as in Section 3:

$$\min_{x\in\mathbb{R}^n} f(x), \tag{3}$$

where function f is a general non-convex L-smooth function with the uniform lower bound f_*, i.e., it is differentiable and

$$f(x) \geq f_* \quad \forall x \in \mathbb{R}^n,$$
$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x-y\|_2 \quad \forall x, y \in \mathbb{R}^n.$$

We are interested in two particular cases: expectation minimization

$$f(x) = \mathbb{E}_{\xi}[f(x,\xi)], \tag{6}$$

and finite-sum minimization

$$f(x) = \frac{1}{m}\sum_{i=1}^{m} f_i(x). \tag{7}$$

Such problems usually arise in applications of (deep) machine learning [88, 200] and mathematical statistics [199], and typically they are solved via stochastic first-order methods.

In general, the best one can expect to achieve is an approximate stationary point [213, 15]. To be specific, for this class of problems stochastic first-order methods in the worst case can only find such point $\hat{x}$ that

$$\mathbb{E}\left[\|\nabla f(\hat{x})\|_2^2\right] \leq \varepsilon^2 \tag{8}$$

For simplicity, we will call the point $\hat{x}$ as ε-stationary point, but mean by this that inequality (8) holds.

Below we summarize recent results about finding ε-stationary point using stochastic first-order methods. We start with presenting the general and unified approach to analyze optimal deterministic and stochastic first-order methods for objectives of types (6) and (7) in the general settings. After that, we consider 3 big classes of stochastic first-order methods with convergence guarantees: SGD and its variants, variance reduced methods, and adaptive stochastic methods.

4.1 General View on Optimal Deterministic and Stochastic First-Order Methods for Non-Convex Optimization

Assume that at each point x, we have access to the estimator $g(x)$ of the gradient $\nabla f(x)$. For now, it is not important to specify what properties $g(x)$ satisfies. In these settings one can use Algorithm 3 in order to find ε-stationary point.

Algorithm 3 General scheme of the optimal first-order method for non-convex optimization

Require: learning rates $\{h_k\}_{k\geq 0}$ satisfying $h_k \leq \frac{1}{2L}$, starting point $x^0 \in \mathbb{R}^n$, stopping criterion C
for $k = 0, 1, 2, \ldots$ **do**
  Get $g^k = g(x^k)$
  if C holds **then**
    $x^N = x^k$
    break
  else
    $x^{k+1} = x^k - h_k g^k$
  end if
end for
return x^N

Below we derive preliminary inequalities playing the central role in the analysis of optimal (stochastic) first-order algorithms. From L-smoothness of f we have

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) + \langle \nabla f(x^k), x^{k+1} - x^k \rangle + \frac{L}{2}\|x^{k+1} - x^k\|_2^2 \\ &= f(x^k) + \langle g^k, x^{k+1} - x^k \rangle + \langle \nabla f(x^k) - g^k, x^{k+1} - x^k \rangle + \frac{L}{2}\|x^{k+1} - x^k\|_2^2 \\ &\leq f(x^k) - h_k\|g^k\|_2^2 + h_k\|\nabla f(x^k) - g^k\|_2^2 + \left(\frac{1}{4h_k} + \frac{L}{2}\right)\|x^{k+1} - x^k\|_2^2, \end{aligned}$$

where in the last inequality we use Fenchel-Young inequality: $\langle a, b\rangle \leq \frac{1}{2\alpha}\|a\|_2^2 + \frac{\alpha}{2}\|b\|_2^2$ with $a = \nabla f(x^k) - g^k$, $b = x^{k+1} - x^k$ and $\alpha = \frac{1}{2h_k}$. Since $h_k \leq \frac{1}{2L}$ and $x^{k+1} = x^k - h_k g^k$ we can continue our derivations:

$$f(x^{k+1}) \leq f(x^k) - \frac{h_k}{2}\|g^k\|_2^2 + h_k\|\nabla f(x^k) - g^k\|_2^2.$$

Now it is crucial to specify what we need to assume about $g(x)$. We emphasize that all 3 cases considered below are based on the tight bounds for $\|\nabla f(x^k) - g^k\|_2^2$ or its expectation.

4.1.1 Deterministic Case

In this case we assume that for all $x \in \mathbb{R}^n$ we have an access to such $g(x)$ that

$$\|g(x) - \nabla f(x)\|_2^2 \leq \frac{\varepsilon^2}{10}. \tag{10}$$

In other words, $g(x)$ is good enough approximation of $\nabla f(x)$. Consider the stopping criterion $C = \left\{ \|g^k\|_2^2 \leq \frac{2\varepsilon^2}{5} \right\}$ and let $h_k = \frac{1}{2L}$ for all $k \geq 0$. First of all, if Algorithm 3 stops, then $\|g^N\|_2 \leq \frac{4\varepsilon^2}{10}$ and x^N satisfies

$$\|\nabla f(x^N)\|_2^2 = \|\nabla f(x^N) - g^N + g^N\|_2^2 \leq 2\|\nabla f(x^N) - g^N\|_2^2 + 2\|g^N\|_2^2 \overset{(10)}{\leq} \varepsilon^2.$$

Next, we derive an upper bound for such N that Algorithm 3 stops after N iterations. Assume that, after N iterations the method has not stopped. Then for all $k = 0, 1, \ldots, T$ we have

$$f(x^{k+1}) \overset{(9),(10)}{\leq} f(x^k) - \frac{4h_k\varepsilon^2}{10} + \frac{h_k\varepsilon^2}{10} = f(x^k) - \frac{3\varepsilon^2}{20L}.$$

Unrolling the recurrence we obtain:

$$\begin{aligned} f(x^{N+1}) &\leq f(x^0) - \frac{3\varepsilon^2}{20L}(N+1) \\ &\Downarrow \\ N &\leq \frac{20L(f(x^0) - f(x^{N+1}))}{3\varepsilon^2} - 1 \leq \frac{20L(f(x^0) - f_*)}{3\varepsilon^2} - 1. \end{aligned}$$

Therefore, the methods stops after

$$N \leq \frac{20L(f(x^0) - f_*)}{3\varepsilon^2}$$

iterations. This bound is optimal up to constant factors [46].

4.1.2 Stochastic Case: Uniformly Bounded Variance

In this case, we assume that for all $x \in \mathbb{R}^n$ we have

$$\mathbb{E}\left[g(x) \mid x\right] = \nabla f(x), \quad \mathbb{E}\left[\|g(x) - \nabla f(x)\|_2^2 \mid x\right] \leq \frac{\varepsilon^2}{2}. \tag{11}$$

For example, this situation appears when

$$f(x) = \mathbb{E}_\xi\left[f(x, \xi)\right]$$

where ξ is a random variable with distribution $\mathscr{D}$ and $g(x)$ is formed as

$$g(x) = \frac{1}{r}\sum_{i=1}^{r} \nabla f_i(x, \xi_i) \tag{12}$$

where $\xi_1, \ldots, \xi_r$ are i.i.d. samples from $\mathscr{D}$ and

$$\mathbb{E}_\xi[\nabla f(x,\xi)] = \nabla f(x), \quad \mathbb{E}_\xi\left[\|\nabla f(x,\xi) - \nabla f(x)\|_2^2\right] \le \sigma^2. \tag{13}$$

Indeed, if we choose $r = \max\left\{1, \frac{2\sigma^2}{\varepsilon^2}\right\}$, then due to independence of $\xi_1, \ldots, \xi_r$ we have:

$$\mathbb{E}\left[\|g(x) - \nabla f(x)\|_2^2 \mid x\right] = \frac{1}{r^2}\sum_{i=1}^{r} \mathbb{E}_{\xi_i}\left[\|\nabla f(x,\xi_i) - \nabla f(x)\|_2^2\right] \le \frac{\sigma^2}{r} \le \frac{\varepsilon^2}{2}.$$

Then, taking conditional expectation $\mathbb{E}\left[\cdot \mid x^k\right]$ from the both sides of (9) we derive

$$\begin{aligned}
\mathbb{E}\left[f(x^{k+1}) \mid x^k\right] &\le f(x^k) - \frac{h_k}{2}\mathbb{E}\left[\|g^k\|_2^2 \mid x^k\right] + h_k\mathbb{E}\left[\|g^k - \nabla f(x^k)\|_2^2 \mid x^k\right] \\
&= f(x^k) - \frac{h_k}{2}\|\nabla f(x^k)\|_2^2 - \frac{h_k}{2}\mathbb{E}\left[\|g^k - \nabla f(x^k)\|_2^2 \mid x^k\right] \\
&\quad + h_k\mathbb{E}\left[\|g^k - \nabla f(x^k)\|_2^2 \mid x^k\right] \\
&= f(x^k) - \frac{h_k}{2}\|\nabla f(x^k)\|_2^2 + \frac{h_k}{2}\mathbb{E}\left[\|g^k - \nabla f(x^k)\|_2^2 \mid x^k\right] \\
&\overset{(11)}{\le} f(x^k) - \frac{h_k}{2}\|\nabla f(x^k)\|_2^2 + \frac{h_k\varepsilon^2}{4}.
\end{aligned}$$

After that, we take the full expectation from the both sides of the previous inequality, choose $h_k \equiv \frac{1}{2L}$ and sum up the result for $k = 0, 1, \ldots, N-1$:

$$\begin{aligned}
\frac{1}{N}\sum_{k=0}^{N-1}\mathbb{E}\left[\|\nabla f(x^k)\|_2^2\right] &\le \frac{4L}{N}\sum_{k=0}^{N-1}\left(\mathbb{E}[f(x^k)] - \mathbb{E}[f(x^{k+1})]\right) + \frac{\varepsilon^2}{2} \\
&= \frac{4L\left(f(x^0) - \mathbb{E}[f(x^N)]\right)}{N} + \frac{\varepsilon^2}{2} \\
&\le \frac{4L\left(f(x^0) - f_*\right)}{N} + \frac{\varepsilon^2}{2}.
\end{aligned}$$

Finally, we choose the output of the method $\hat{x}^N$ uniformly at random from $x^0, x^1, \ldots, x^{N-1}$ which implies

$$\mathbb{E}\left[\|\nabla f(\hat{x}^N)\|_2^2\right] \le \frac{4L\left(f(x^0) - f_*\right)}{N} + \frac{\varepsilon^2}{2}.$$

Taking $N = \frac{8L\left(f(x^0)-f_*\right)}{\varepsilon^2}$ we obtain $\mathbb{E}\left[\|\nabla f(\hat{x}^N)\|_2^2\right] \leq \varepsilon^2$. Moreover, the total number of stochastic oracle calls (number of $\nabla f(x,\xi)$-calculations) is

$$\sum_{k=0}^{N-1} r_k = \max\left\{\frac{8L\left(f(x^0)-f_*\right)}{\varepsilon^2}, \frac{16L\left(f(x^0)-f_*\right)\sigma^2}{\varepsilon^4}\right\}.$$

This bound is optimal up to constant factors for the case when the variance is uniformly upper bounded [15].

4.1.3 Stochastic Case: Finite Sum Minimization

In this case we assume that the objective function has a finite sum structure (7) with L-smooth summands. In fact, this smoothness constant L can be significantly larger than the smoothness constant of f. It is essential for providing a fair comparison of different complexity results. It is possible to improve the dependence on L in the final complexity bounds [134] using average smoothness assumption, but for simplicity we consider the case when all summands are L-smooth. Moreover, we assume that there exists constant σ^2 (possibly infinite) such that for ξ taken uniformly at random from $\{1,\ldots,m\}$ and for all $x \in \mathbb{R}^n$

$$\mathbb{E}_\xi\left[\|\nabla f_\xi(x) - \nabla f(x)\|_2^2\right] \leq \sigma^2. \tag{14}$$

We define r_k and g^k in the following way:

$$\begin{aligned}
r_k = r &= \max\left\{1, \frac{20\sigma^2}{\varepsilon^2}\right\},\\
q &= \min\{r, m\},\\
g^k &= \begin{cases}
\frac{1}{r}\sum\limits_{j=1}^{r} \nabla f_{\xi_{k,j}}(x^k), & \text{if } r < m \text{ and } r \text{ divides } k,\\
\nabla f(x^k), & \text{if } m \leq r \text{ and } m \text{ divides } k,\\
\nabla f_{\xi_k}(x^k) - \nabla f_{\xi_k}(x^{k-1}) + g^{k-1}, & \text{otherwise}
\end{cases}\\
h_k = h &= \frac{1}{10L\sqrt{q}}.
\end{aligned}$$

Here, at iteration k random index ξ_k is sampled uniformly at random from $\{1,\ldots,m\}$ if k is not divisible by q and random indices $\xi_{k,1},\ldots,\xi_{k,r}$ are i.i.d. samples from uniform distribution on $\{1,\ldots,m\}$ if $q = r$ and r divides k. As the result, we obtain the variant of SPIDER [74]. We notice that for $k = aq + p$, $p \in \{0,1,\ldots,q-1\}$ iteration k requires 2 calculations of $\nabla f_\xi(x)$ when $p \neq 0$ and q calculations of $\nabla f_\xi(x)$ when $p = 0$. This implies that q iterations of the method requires only $3q$ calculations of $\nabla f_\xi(x)$, so, if $k \geq q$, then the number of stochastic first-order oracle coincides with the number of iterations up to a constant factor 3.

Below we present a simplified approach to analyze SPIDER. As before, our goal is to show that $\mathbb{E}\left[\|g^k - \nabla f(x^k)\|_2^2\right]$ can be upper-bounded by either something small or something that can be controlled by other terms in (9). First of all, if $k = aq$, then

$$\begin{aligned}
\mathbb{E}\left[\|g^k - \nabla f(x^k)\|_2^2\right] &= \begin{cases} 0, & \text{if } q = m, \\ \mathbb{E}\left[\left\|\frac{1}{r}\sum\limits_{j=1}^{r} \nabla f_{\xi_{k,j}}(x^k) - \nabla f(x^k)\right\|_2^2\right], & \text{if } q = r \end{cases} \\
&= \begin{cases} 0, & \text{if } q = m, \\ \frac{1}{r^2}\sum\limits_{j=1}^{r} \mathbb{E}\left[\left\|\nabla f_{\xi_{k,j}}(x^k) - \nabla f(x^k)\right\|_2^2\right], & \text{if } q = r \end{cases} \\
&\overset{(14)}{\leq} \begin{cases} 0, & \text{if } q = m, \\ \frac{\sigma^2}{r}, & \text{if } q = r, \end{cases} \overset{(15)}{\leq} \begin{cases} 0, & \text{if } q = m, \\ \frac{\varepsilon^2}{20}, & \text{if } q = r, \end{cases}
\end{aligned}$$

where $\mathbb{E}\left[\left\|\frac{1}{r}\sum\limits_{j=1}^{r} \nabla f_{\xi_{k,j}}(x^k) - \nabla f(x^k)\right\|_2^2\right] = \frac{1}{r^2}\sum\limits_{j=1}^{r} \mathbb{E}\left[\left\|\nabla f_{\xi_{k,j}}(x^k) - \nabla f(x^k)\right\|_2^2\right]$ due to independence of $\xi_{k,1}, \ldots, \xi_{k,r}$ and in the third inequality we applied the tower property: $\mathbb{E}[\cdot] = \mathbb{E}\left[\mathbb{E}\left[\cdot \mid x^k\right]\right]$. Secondly, if $k = aq + p$ with $p \in \{1, \ldots, q-1\}$ we have

$$\begin{aligned}
\mathbb{E}\left[\left\|g^k - \nabla f(x^k)\right\|_2^2\right] &\overset{(17)}{=} \mathbb{E}\left[\left\|\nabla f_{\xi_k}(x^k) - \nabla f_{\xi_k}(x^{k-1}) + g^{k-1} - \nabla f(x^k)\right\|_2^2\right] \\
&= \mathbb{E}\left[\left\|\nabla f_{\xi_k}(x^k) - \nabla f_{\xi_k}(x^{k-1}) - \nabla f(x^k) + \nabla f(x^{k-1})\right\|_2^2\right] \\
&\quad + \mathbb{E}\left[\left\|g^{k-1} - \nabla f(x^{k-1})\right\|_2^2\right]
\end{aligned}$$

where we use the variance decomposition[3] $\mathbb{E}_{\xi_k}\left[\|\eta\|_2^2\right] = \mathbb{E}_{\xi_k}\left[\|\eta - \mathbb{E}_{\xi_k}[\eta]\|_2^2\right] + \left\|\mathbb{E}_{\xi_k}[\eta]\right\|$ for random vector $\eta = \nabla f_{\xi_k}(x^k) - \nabla f_{\xi_k}(x^{k-1}) + g^{k-1} - \nabla f(x^k)$ together with the tower property $\mathbb{E}[\cdot] = \mathbb{E}\left[\mathbb{E}_{\xi_k}[\cdot]\right]$. Using the inequality above together with $\|a+b\|_2^2 \leq 2\|a\|_2^2 + 2\|b\|_2^2$, $a, b \in \mathbb{R}^n$ and L-smoothness of $f_1, \ldots, f_m, f$ we get

[3] Here $\mathbb{E}_{\xi_k}[\cdot]$ is a mathematical expectation conditioned on everything despite ξ_k, i.e. expectation is taken w.r.t. the randomness coming only from ξ_k.

$$\begin{aligned}\mathbb{E}\left[\left\|g^k-\nabla f(x^k)\right\|_2^2\right] &\le 2\mathbb{E}\left[\left\|\nabla f_{\xi_k}(x^k)-\nabla f_{\xi_k}(x^{k-1})\right\|_2^2\right]+2\mathbb{E}\left[\left\|\nabla f(x^k)-\nabla f(x^{k-1})\right\|_2^2\right]\\ &\quad+\mathbb{E}\left[\left\|g^{k-1}-\nabla f(x^{k-1})\right\|_2^2\right]\\ &\le 4L^2\mathbb{E}\left[\|x^k-x^{k-1}\|_2^2\right]+\mathbb{E}\left[\left\|g^{k-1}-\nabla f(x^{k-1})\right\|_2^2\right]\\ &= 4L^2h^2\mathbb{E}\left[\|g^{k-1}\|_2^2\right]+\mathbb{E}\left[\left\|g^{k-1}-\nabla f(x^{k-1})\right\|_2^2\right]\\ &\le 8L^2h^2\mathbb{E}\left[\|\nabla f(x^{k-1})\|_2^2\right]+\left(1+8L^2h^2\right)\mathbb{E}\left[\left\|g^{k-1}-\nabla f(x^{k-1})\right\|_2^2\right].\end{aligned}$$

Unrolling the recurrence we derive

$$\begin{aligned}\mathbb{E}\left[\left\|g^k-\nabla f(x^k)\right\|_2^2\right] &\le 8L^2h^2\sum_{l=1}^{p}\left(1+8L^2h^2\right)^{l-1}\mathbb{E}\left[\|\nabla f(x^{k-l})\|_2^2\right]\\ &\quad+\left(1+8L^2h^2\right)^{p}\mathbb{E}\left[\|g^{aq}-\nabla f(x^{aq})\|_2^2\right]\\ &\overset{(19),p\le q}{\le} \left(1+8L^2h^2\right)^{q}\sum_{l=1}^{p}8L^2h^2\mathbb{E}\left[\|\nabla f(x^{aq+l})\|_2^2\right]\\ &\quad+\left(1+8L^2h^2\right)^{q}\begin{cases}0, & \text{if } q=m,\\ \frac{\varepsilon^2}{20}, & \text{if } q=r\end{cases}\\ &\overset{(1+x)^q\le e^{qx}}{\le} \exp\left(8L^2h^2q\right)\sum_{l=1}^{p}8L^2h^2\mathbb{E}\left[\|\nabla f(x^{aq+l})\|_2^2\right]\\ &\quad+\exp\left(8L^2h^2q\right)\begin{cases}0, & \text{if } q=m,\\ \frac{\varepsilon^2}{20}, & \text{if } q=r.\end{cases}\end{aligned}$$

Next, using the choice of the stepsize $h = 1/(10L\sqrt{q})$ we obtain

$$\mathbb{E}\left[\left\|g^k-\nabla f(x^k)\right\|_2^2\right] \le \sum_{l=1}^{p}9L^2h^2\mathbb{E}\left[\|\nabla f(x^{aq+l})\|_2^2\right]+\frac{11\varepsilon^2}{200}.$$

Finally, we put all the inequalities together. We start with modifying (9):

$$\begin{aligned}f(x^{k+1}) &\le f(x^k)-\frac{h_k}{2}\|g^k\|_2^2+h_k\|\nabla f(x^k)-g^k\|_2^2\\ &\le f(x^k)-\frac{h}{4}\|\nabla f(x^k)\|_2^2+\frac{3h}{2}\|\nabla f(x^k)-g^k\|_2^2,\end{aligned}$$

where we used that inequality $\|a+b\|_2^2 \ge \frac{1}{2}\|a\|_2^2-\|b\|_2^2$ holds for all $a,b\in\mathbb{R}^n$ (in particular, we use $a=\nabla f(x^k)$ and $b=g^k-\nabla f(x^k)$). Next, we take the full mathematical expectation from the both sides of previous inequality (taking into account that $k=aq+p$):

$$\begin{aligned}\mathbb{E}[f(x^{aq+p+1})] &\leq \mathbb{E}[f(x^{aq+p})] - \frac{h}{4}\mathbb{E}\left[\|\nabla f(x^{aq+p})\|_2^2\right] + \frac{3h}{2}\mathbb{E}\left[\|g^{aq+p} - \nabla f(x^{aq+p})\|_2^2\right] \\ &\leq \mathbb{E}[f(x^{aq+p})] - \frac{h}{4}\mathbb{E}\left[\|\nabla f(x^{aq+p})\|_2^2\right] + \frac{3h}{2}\sum_{l=1}^{p} 9L^2h^2\mathbb{E}\left[\|\nabla f(x^{aq+l})\|_2^2\right] \\ &\quad + \frac{33h\varepsilon^2}{400}.\end{aligned}$$

We notice that this inequality holds for all integers $a \geq 0$ and $p \in \{0,\ldots,q-1\}$. Summing up these inequalities for $p = 0,\ldots,P$ and taking $a = A$ where $N = Aq+P$, $P \in \{0,\ldots,q-1\}$ we get

$$\begin{aligned}0 \quad &\leq \quad \sum_{p=0}^{P}\left(\mathbb{E}[f(x^{Aq+p})] - \mathbb{E}[f(x^{Aq+p+1})]\right) - \frac{h}{4}\sum_{p=0}^{P}\mathbb{E}\left[\|\nabla f(x^{Aq+p})\|_2^2\right] \\ &\qquad + \frac{27L^2h^3}{2}\sum_{p=0}^{P}\sum_{l=1}^{p}\mathbb{E}\left[\|\nabla f(x^{Aq+l})\|_2^2\right] + \frac{33h\varepsilon^2(P+1)}{400} \\ &\overset{P\leq q-1}{\leq} \mathbb{E}\left[f(x^{Aq})\right] - \mathbb{E}\left[f(x^{Aq+P+1})\right] - h\left(\frac{1}{4} - \frac{27L^2h^2q}{2}\right)\sum_{p=0}^{P}\mathbb{E}\left[\|\nabla f(x^{Aq+p})\|_2^2\right] \\ &\qquad + \frac{33h\varepsilon^2(P+1)}{400} \\ &\overset{(18)}{=} \mathbb{E}\left[f(x^{Aq})\right] - \mathbb{E}\left[f(x^{Aq+P+1})\right] - \frac{23h}{200}\sum_{p=0}^{P}\mathbb{E}\left[\|\nabla f(x^{Aq+p})\|_2^2\right] + \frac{33h\varepsilon^2(P+1)}{400},\end{aligned}$$

hence

$$\frac{23h}{200}\sum_{p=0}^{P}\mathbb{E}\left[\|\nabla f(x^{Aq+p})\|_2^2\right] \leq \mathbb{E}\left[f(x^{Aq})\right] - \mathbb{E}\left[f(x^{Aq+P+1})\right] + \frac{33h\varepsilon^2(P+1)}{400}.$$

These inequalities hold for all A and P. Then we can sum up these inequalities for $(A,P) = (0,q-1),(1,q-1),\ldots,(\hat{A},\hat{P})$ and get that for $\hat{N} = \hat{A}q+\hat{P}$ and divide the result by $\frac{23h(\hat{N}+1)}{200}$ and get

$$\begin{aligned}\frac{1}{\hat{N}+1}\sum_{k=0}^{\hat{N}}\mathbb{E}\left[\|\nabla f(x^k)\|_2^2\right] &\leq \frac{200\left(f(x^0) - \mathbb{E}\left[f(x^{\hat{N}+1})\right]\right)}{23h(\hat{N}+1)} + \frac{33\varepsilon^2}{46} \\ &\overset{(18)}{\leq} \frac{2000L\sqrt{q}\left(f(x^0) - f_*\right)}{23(\hat{N}+1)} + \frac{33\varepsilon^2}{46}.\end{aligned}$$

Finally, taking $\hat{x}^{\hat{N}}$ uniformly at random from $x^0,\ldots,x^{\hat{N}}$ we get

$$\mathbb{E}\left[\|\nabla f(\hat{x}^{\hat{N}})\|_2^2\right] \leq \frac{2000L\sqrt{q}\left(f(x^0) - f_*\right)}{23(\hat{N}+1)} + \frac{33\varepsilon^2}{46}.$$

This implies that after

$$\begin{aligned}\hat{N} &= \frac{4000L\sqrt{q}(f(x^0)-f(x^*))}{13\varepsilon^2} \\ &\overset{(15),(16)}{=} \frac{4000L(f(x^0)-f(x^*))}{13\varepsilon^2}\min\left\{\sqrt{m},\max\left\{1,\frac{\sqrt{20}\sigma}{\varepsilon}\right\}\right\}\end{aligned}$$

iterations we reach $\mathbb{E}\left[\|\nabla f(\hat{x}^{\hat{N}})\|_2^2\right] \leq \varepsilon^2$. Moreover, it requires

$$O\left(\frac{L(f(x^0)-f(x^*))}{\varepsilon^2}\min\left\{\sqrt{m},\max\left\{1,\frac{\sigma}{\varepsilon}\right\}\right\}+\min\left\{m,\max\left\{1,\frac{\sigma^2}{\varepsilon^2}\right\}\right\}\right)$$

calculations of $\nabla f_\xi(x)$ which is optimal up to constant factors [74].

4.2 SGD and Its Variants

As it was shown in the previous section, SGD

$$x^{k+1} = x^k - h_k g(x^k), \quad \mathbb{E}[g(x)] = \nabla f(x)$$

in the settings of Section 4.1.2 requires $O\left(\frac{L(f(x^0)-f_*}{\varepsilon^2}\right)$ iterations with batch-size $r = \Theta\left(\max\left\{1,\frac{\sigma^2}{\varepsilon^2}\right\}\right)$ to find an ε-stationary point in expectation. The total number of stochastic first-order oracle calls equals

$$O\left(\frac{L(f(x^0)-f_*)}{\varepsilon^2}\max\left\{1,\frac{\sigma^2}{\varepsilon^2}\right\}\right). \tag{21}$$

We emphasize that we use large batch-size for the sake of simplicity and unification of the results in 3 different cases. In fact, it is possible to obtain the bound (21) using smaller stepsizes and constant batchsizes of the order $O(1)$ [83].

4.2.1 Assumptions on the Stochastic Gradient

In addition to assumption (13), which is quite restrictive, there exist several other assumptions on the stochastic gradient studied in the literature. Recently Khaled and Richtárik [117] proposed a simple and unified way to cover the most popular ones.

Assumption 4.1 (Expected Smoothness; Assumption 2 from [117]) *The second moment of stochastic gradients satisfies*

$$\mathbb{E}\left[\|g(x)\|_2^2\right] \leq 2A\left(f(x)-f_*\right)+B\|\nabla f(x)\|_2^2+C \tag{22}$$

for some $A, B, C \geq 0$ *and for all* $x \in \mathbb{R}^n$.

This assumption generalizes the notion of expected smoothness introduced and adjusted for convex problems in [94]. Moreover, the following assumptions are stronger than Assumption 4.1 or can be seen as special cases of Assumption 4.1 (see more details and formal proofs in [117]).

Uniformly upper-bounded variance (UV) assumption. Indeed, if $A = 0$, $B = 1$ and $C = \sigma^2$, then using variance decomposition inequality (22) implies (13):

$$\mathbb{E}\left[\|g(x) - \nabla f(x)\|_2^2\right] = \mathbb{E}\left[\|g(x)\|_2^2\right] - \|\nabla f(x)\|_2^2 \overset{(22)}{\leq} \sigma^2.$$

Expected strong growth condition (E-SG). When $A = C = 0$ and $B = \alpha \geq 1$ inequality (22) transforms into so-called expected strong growth condition [197, 211]:

$$\mathbb{E}\left[\|g(x)\|_2^2\right] \leq \alpha \|\nabla f(x)\|_2^2. \tag{23}$$

Maximal strong growth condition (M-SG) [208, 184] states that there exists such $\alpha > 0$ that

$$\|g(x)\|_2^2 \leq \alpha \|\nabla f(x)\|_2^2 \text{ almost surely for all } x \in \mathbb{R}^n.$$

This condition implies E-SG (23) while known convergence results in expectation under M-SG assumption have no advantage in comparison with their counterparts under E-SG.

Relaxed growth condition (RG) [33] can be seen as another special case of Assumption 4.1 with $A = 0$, $B = \alpha \geq 1$ and $C = \beta \geq 0$ or as an extension of E-SG:

$$\mathbb{E}\left[\|g(x)\|_2^2\right] \leq \alpha \|\nabla f(x)\|_2^2 + \beta. \tag{24}$$

However, there exist simple problems of type (3)+(6) that fit the settings we are interested in but do not satisfy (24) (see Proposition 1 from [117]).

Gradient confusion condition (GC) [182] was developed for the finite-sum case (7). In particular, it states that there exists such $\eta > 0$ that for all $i, j = 1, \ldots, m$ and for all $x \in \mathbb{R}^n$

$$\langle \nabla f_i(x), \nabla f_j(x) \rangle \geq -\eta. \tag{25}$$

One can show (see Theorem 1, [117]) that inequality (25) implies (24) with $\alpha = m$ and $\beta = \eta(m-1)$, and, as a consequence, it is a special case of Assumption 4.1 with $A = 0$, $B = m$, and $C = \eta(m-1)$.

Sure-smoothness condition (SS) [130] is defined for the case when the objective is represented as an expectation (6) and $g(x) = \nabla f(x, \xi)$ where ξ is sampled independently at each iteration of SGD. That is, sure-smoothness condition means that[4] for all $x, y \in \mathbb{R}^n$

$$\|\nabla f(x, \xi) - \nabla f(y, \xi)\|_2 \leq L\|x - y\|_2 \quad \text{and} \quad f(x, \xi) \geq 0 \quad \text{almost surely in } \xi. \tag{26}$$

[4] In the original paper [130], authors considered more general situation when stochastic realizations $f(x, \xi)$ have Hölder-continuous gradients.

Applying classical corollaries of L-smoothness one can derive inequality (22) with $A = 2L$, $B = 0$, and $C = 2Lf_*$ from (26).

Next, Assumption 4.1 covers **arbitrary sampling** setup and distributed setup with quantization[5]. For simplicity, we mention only **sampling with replacement** as a special case of arbitrary sampling (see more examples in [117]). In particular, consider the finite-sum optimization problem (3)+(7) and assume that f_i is L_i-smooth and bounded from below by $f_{i,*}$ for all $i = 1,\ldots,m$. Moreover, assume that $g(x) = \nabla f_j(x)$ where $j = i$ with probability $p_i \geq 0$, $i = 1,\ldots,m$, $\sum_{i=1}^m p_i = 1$. Then, one can prove [117] that Assumption 4.1 is satisfied in this case with $A = \max_i \frac{L_i}{mp_i}$, $B = 0$, and $C = 2A\Delta_* = \frac{2A}{m}\sum_{i=1}^m (f_* - f_{i,*})$. That is, if we apply **uniform sampling**, i.e., $p_i = \frac{1}{m}$ for all $i = 1,\ldots,m$, then we get $A = \max_i L_i$, $B = 0$, $C = 2\max_i L_i \Delta_*$, and if **importance sampling** with $p_i = \frac{L_i}{\sum_{l=1}^m L_l}$ is applied, then Assumption 4.1 holds with $A = \overline{L} = \frac{1}{m}\sum_{i=1}^m L_i$, $B = 0$, and $C = 2\overline{L}\Delta_*$.

Finally, under Assumption 4.1 Khaled and Richtárik [117] derived the following complexity bound: if $h = \min\left\{\frac{1}{\sqrt{LAN}}, \frac{1}{LB}, \frac{\varepsilon}{2LC}\right\}$, then inequality

$$\min_{0\leq k\leq N-1} \mathbb{E}\left[\|\nabla f(x^k)\|_2\right] \leq \varepsilon \tag{27}$$

is satisfied after

$$N = O\left(\frac{L(f(x^0) - f_*)}{\varepsilon^2}\max\left\{B, \frac{A(f(x^0) - f(x^*))}{\varepsilon^2}, \frac{C}{\varepsilon^2}\right\}\right) \tag{28}$$

iterations of SGD. It is worth to mention that this bound gives the sharpest rates for all known special cases. We summarize some of them in Table 1. We notice that (27) is weaker than (8), but it is easy to obtain the same bound (28) guaranteeing (8) instead of (27) based on the analysis given in [117].

4.2.2 The Choice of the Stepsize

In practice, instead of using the constant stepsize for SGD it is popular to periodically decrease the stepsize by some factor [31, 124, 104] even for non-convex problems. For strongly convex problems such a choice is natural: it is well-known [91] that if the stepsize equals h and strong convexity parameter equals μ, then SGD converges with linear rate $\widetilde{O}((h\mu)^{-1})$ to the neighborhood of the solution with size proportional to h. Surprisingly, SGD enjoys similar behaviour even for non-convex problems which was recently shown in [191].

In the neural networks training, "warmup" [96, 93] and cyclical stepsize [196, 141] schedules are also very popular and useful. The first one refers to the strategy when, during several epochs of training, tiny stepsizes are used, and then they are

[5] This technique is applied in distributed optimization to reduce the overall communication cost (e.g., see [4, 23]). However, methods for distributed optimization are out of scope of our survey.

Problem	Settings	Citation	Complexity
(3)+(6)	**UV** (13)	[83]	$\frac{L\Delta_0}{\varepsilon^2}\max\left\{1, \frac{\sigma^2}{\varepsilon^2}\right\}$
(3)+(6)/(7)	**RG** (24)	[33, 211]	$\frac{L\Delta_0}{\varepsilon^2}\max\left\{\alpha, \frac{\beta}{\varepsilon^2}\right\}$
(3)+(7)	**GC** (25)	[182]	$\frac{L\Delta_0}{\varepsilon^2}\max\left\{m, \frac{\eta(m-1)}{\varepsilon^2}\right\}$
(3)+(7)	**Uniform Sampling**	[117]	$\frac{L\max_i L_i\Delta_0}{\varepsilon^4}\max\{\Delta_0, \Delta_*\}$
(3)+(7)	**Importance Sampling**	[117]	$\frac{\bar{L}L\Delta_0}{\varepsilon^4}\max\{\Delta_0, \Delta_*\}$

Table 1 Summary of the complexity results for SGD under different assumptions on the stochastic gradient. The column "Complexity" contains an overall number of stochastic first-order oracle calls needed to find ε-stationary point neglecting constant factors. Notation: $\Delta_0 = f(x^0) - f(x^*)$, σ^2 = a uniform bound for the variance of the stochastic gradient (13), α, β = relaxed growth condition parameters, η = gradient confusion parameter, $\Delta_* = \frac{1}{m}\sum_{i=1}^m (f_* - f_{i,*})$, $\max_i L_i$ = maximal smoothness constant of f_i in (7), $\bar{L}$ = averaged smoothness constant of f_i in (7).

increased. This technique was successfully applied for several deep learning problems like ResNet [104], large-batch training of Imagenet [96] and natural language problems [210, 65].

Cyclical stepsize schedule means that the stepsize is changing between some lower and upper bounds. There are different modification of this technique including gradual decrease and increase during one epoch [196] and gradual decrease of the stepsize followed by the sudden increase [141]. However, the theoretical understanding of the success of "warmup" and cyclical schedules is very limited.

We also discuss different stepsize policies including adaptive ones (Section 4.4), Armijo line-search under expected strong growth assumption and stochastic Polyak stepsizes under relaxed growth assumption (Section 4.2.3) in the following subsections.

4.2.3 Over-Parameterized Models

In Section 2.3.2, we mentioned that over-parameterization [138, 161, 234, 165, 132, 12, 13], meaning that the last layer has more neurons than the number of samples in the training set, is a good property for neural networks from the optimization and generalization [144, 11, 10] point perspectives, but not a panacea: over-parameterized neural networks have no spurious valleys, but still can have bad local minima [67].

In the papers, focusing mostly on the optimization aspects of over-parameterized models, it was shown that SGD converges with the same (up to the difference in the smoothness constants) rate as GD in terms of the iteration complexity in convex and strongly convex cases [211, 212, 139] under interpolation condition: for the finite-sum optimization problem (3)+(7) there exists such point $x^* \in \mathbb{R}^n$ that

$$\min_{x\in\mathbb{R}^n} f_i(x) = f_i(x^*) \quad \forall i = 1,\ldots,m.$$

Furthermore, in this setting SGD converges with Armijo line-search [212], with stochastic Polyak stepsizes [139], and, if additionally expected strong growth condition (23) holds, SGD can be accelerated [211] and the accelerated version converges as good as Nesterov's method [154] in terms of iteration complexity up to expected strong growth multiplicative factor α from (23).

In the general non-convex case, the following results exist.

Constant stepsizes. In [211], it was shown that SGD with constant stepsize $h = 1/\alpha L$ finds ε-stationary point under expected strong growth condition (23) with the rate $O\left(\alpha L(f(x^0)-f_*)/\varepsilon^2\right)$ matching the iteration complexity of GD up to the factor α.

Armijo line-search. The idea that under interpolation condition/expected strong growth condition SGD and GD have similar properties was then strengthen in [212], where authors showed that SGD with Armijo line-search converges in these settings. In particular, the authors of [212] considered such stepsizes h_k that

$$f_{i_k}(x^k - h_k\nabla f_{i_k}(x^k)) \leq f_{i_k}(x^k) - ch_k\|\nabla f_{i_k}(x^k)\|_2^2, \tag{29}$$

where the index i_k is sampled uniformly at random from the set $\{1,\ldots,m\}$, the stochastic gradient g^k is defined as $g^k = \nabla f_{i_k}(x^k)$, and $c > 0$ is a hyper-parameter. Moreover, it is assumed that $h_k \in (0, h_{\max}]$ for all $k \geq 0$. Then SGD with Armijo line-search (29) with $c > 1 - L_{\max}/(\alpha L)$ and $h_{\max} \leq 2/\alpha L$ finds ε-stationary point under expected strong growth condition (23) with the rate $O\left((f(x^0)-f_*)/(\delta\varepsilon^2)\right)$, where $\delta = (h_{\max} + 2(1-c)/L_{\max}) - \alpha\left(h_{\max} - \frac{2(1-c)}{L_{\max}} + Lh_{\max}^2\right)$, $L_{\max}$ is the maximal smoothness constant of summands f_i, and f is the smoothness constant of f. Authors of [212] also considered the version with samples used for backtracking (29) independent from those used for determining the stochastic gradient, and the version with non-increasing stepsizes under additional assumption that the iterates lie in some ball with radius D. The rates are $O\left(\max\{L_{\max},\alpha L\}(f(x^0)-f_*)/\varepsilon^2\right)$ and $O\left(\max\{L_{\max},\alpha L\}LD^2/\varepsilon^2\right)$ respectively, and both complexity bounds hold with $c = 1/2$ and $h_{\max} = 1/(\alpha L)$. Finally, in the numerical experiments from [212] the authors observed that the method's performance is robust to the choic of c and $h_{\max}$.

Stochastic Polyak stepsizes. Next, SGD under expected strong growth condition converges with stochastic Polyak stepsizes introduced and analyzed in [139]:

$$h_k = \min\left\{\frac{f_{i_k}(x^k) - f_{i_k,*}}{c\|\nabla f_{i_k}(x^k)\|_2}, h_b\right\},$$

where the index i_k is sampled uniformly at random from the set $\{1,\ldots,m\}$, the stochastic gradient g^k is defined as $g^k = \nabla f_{i_k}(x^k)$, $f_{i,*}$ is uniform lower bound for $f_i(x)$, and $c > 0$ is a hyper-parameter. In particular, one can show [139] that SGD in these settings with $c > \alpha L/4L_{\max}$ and $h_b \leq \max\left\{2/(\alpha L), \overline{h}_b\right\}$ finds ε-stationary point under expected strong growth condition (23) with the rate $O\left((f(x^0)-f_*)/(\delta\varepsilon^2)\right)$, where $\delta = (h_b + \beta) - \alpha\left(h_b - \beta + Lh_b^2\right)$, $\beta = \min\{1/(2cL_{\max}), h_b\}$, and

$$\overline{h}_b = \frac{-(\alpha-1)+\sqrt{(\alpha-1)^2+\frac{4L\alpha(\alpha+1)}{2cL_{\max}}}}{2L\alpha}.$$

4.2.4 Proximal Variants

Many complexity results that we mentioned before and will mention in the following subsections have generalizations to the composite optimization problems:

$$\min_{x\in\mathbb{R}^n}\{F(x)=f(x)+R(x)\},$$

where the function f is L-smooth, but, possibly, non-convex, while $R(x)$, i.e., composite term/regularizer, is a proper closed convex function which can be non-smooth. Moreover, function $R(x)$ is often chosen in such a way that the proximal operator

$$\text{prox}_R(x) = \underset{y\in\mathbb{R}^n}{\text{argmin}}\left\{R(y)+\frac{1}{2}\|y-x\|_2^2\right\}$$

can be easily computed, and to make the solution of the problem satisfy certain properties, e.g., sparsity; see [41, 54, 17] for the detailed discussion and examples of regularizers.

In these settings, instead of SGD one can apply prox-SGD defined by the following recurrence:

$$x^{k+1} = \text{prox}_{h_k R}(x^k - h_k g^k).$$

Moreover, to measure the progress of the method the generalized projected stochastic gradient is used: $\widetilde{g}^k = {}^{(x^k-x^{k+1})}/_{h_k}$. When the regularizer $R(x)$ is a constant $\widetilde{g}^k = g^k$. For proximal stochastic methods we say that the iterate x^k is ε-stationary point if

$$\mathbb{E}\left[\|\widetilde{g}^k\|_2^2\right] \le \varepsilon^2.$$

In [85], it was shown that prox-SGD under uniformly upper-bounded variance assumption (13) converges with the rate given in (21). However, the analysis from [85] works only in the large-batch setting, i.e., when batchsizes are of the order $O(\varepsilon^{-2})$. For a long time, there was no analysis establishing the same bound without using $O(\varepsilon^{-2})$ batches, and the problem was recently resolved in [59].

4.2.5 Momentum-SGD

As we already mentioned, SGD is optimal among stochastic first-order methods for finding ε-stationary points under uniformly bounded variance assumption [15]. However, it does not imply that there is no sense in using different methods for such problems. In practice, different additional tricks are applied to improve the convergence of SGD, and, perhaps, the most popular one is momentum [173].

Momentum-SGD/Heavy Ball SGD can be written in different forms. Usually it is written as

$$m^{k+1} = \beta_k m^k + g(x^k),$$
$$x^{k+1} = x^k - h_k g(x^k),$$

where parameter $\beta_k \in [0,1)$ is called momentum parameter. In the convex and strongly convex cases this method has some advantages in comparison to SGD like better last-iterate convergence guarantees [204, 205, 186], but does not have an accelerated rate [119]. In the non-convex case, Momentum-SGD has the same complexity guarantee (21) as SGD under uniformly bounded variance assumption [228, 60]. However, in practice, Momentum-SGD often works much better than SGD especially on computer vision problems [201], and also navigates ravines and escapes saddle points better than SGD.

Among other works on Momentum-SGD we emphasize the recent paper [60] establishing the tight convergence rates for Momentum-SGD in Stochastic Primal Averaging [204] form via Lyapunov functions analysis. In particular, [60] justifies (theoretically and/or empirically) the following important insights about the behavior of Momentum-SGD: (i) Momentum-SGD is provably better than SGD during the early stage of the convergence, (ii) it is better to gradually reduce momentum parameter β_k rather than the stepsize h_k, and (iii) gradual changes of the parameters of Momentum-SGD are preferable than sudden changes.

4.2.6 Random Reshuffling

Before this subsection, we always assumed that stochastic gradients are sampled independently from previous iterations. However, in the context of finite sum optimization (3)+(7), the different sampling strategy called Random Reshuffling (or SGD with Without Replacement sampling) is often used: at each epoch (pass through the dataset) random permutation $\{i_1, i_2, \ldots, i_m\}$ of the set $\{1, 2, \ldots, m\}$ is generated defining the order of gradients computations (see Algorithm 4). This strategy implies that stochastic gradient in RR is biased.

While the superiority of RR to SGD was empirically discovered a long time ago [30, 32], the theoretical justification of this phenomenon was developed only recently [102, 175, 164, 150]. In particular, authors of [164] proved that RR under uniformly bounded gradients assumption,

$$\|f_i(x)\|_2 \leq G \quad \forall i = 1, \ldots, m, \; \forall x \in \mathbb{R}^n,$$

finds ε-stationary point with the rate $O\left(L_{\max} m (f(x^0) - f_*)\left(\varepsilon^{-2} + G\varepsilon^{-3}\right)\right)$, where $L_{\max}$ is the maximal smoothness constant of summands $f_1, \ldots, f_m$. Then, in [150] this result was generalized and tightened: under the assumption

Algorithm 4 Random Reshuffling (RR)

Require: learning rates $\{h_{s,k}\}_{s,k\geq 0}$, starting point $x^0 \in \mathbb{R}^n$, batch size $r \geq 1$, number of epochs S
  Set $x_0^0 = x^0$
  for $s = 0, 1, 2, \ldots K-1$ **do**
    Generate random permutation $\{i_{s,1}, \ldots, i_{s,m}\}$ of the set $\{1, \ldots, m\}$
    Set $l = \lceil m/r \rceil$
    for $k = 0, 1, \ldots, l-1$ **do**
      Set $\hat{r}_s^k = \min\{r, m - kr\}$
      Compute $g_s^k = \frac{1}{\hat{r}_s^k} \sum\limits_{j=1}^{\hat{r}_s^k} \nabla f_{i_{s,kr+j}}(x_s^k)$
      $x_s^{k+1} = x_s^k - h_{s,k} g_s^k$
    end for
    $x_{s+1}^0 = x_s^l$
  end for
  return x_{K-1}^l

$$\frac{1}{m}\sum_{i=1}^{m} \|\nabla f_i(x) - \nabla f(x)\|_2^2 \leq 2A\left(f(x) - f_*\right) + C, \tag{30}$$

which is a special case of (22) with $B = 1$, authors of [150] derived the following bound:

$$O\left(L_{\max}\sqrt{m}(f(x^0) - f_*)\left(\frac{\sqrt{m}}{\varepsilon^2} + \frac{\sqrt{A(f(x^0) - f_*)} + \sqrt{C}}{\varepsilon^3}\right)\right). \tag{31}$$

That is, under uniformly bounded variance assumption (13) this bound transforms ($A = 0$, $C = \sigma^2$) into $O\left(L_{\max}\sqrt{m}(f(x^0) - f_*)\left(\sqrt{m}\varepsilon^{-2} + \sigma\varepsilon^{-3}\right)\right)$ which outperforms the corresponding complexity bound for SGD (21) whenever $L_{\max}\sqrt{m}\varepsilon \leq L\sigma$. Next, one can show that for $L_{\max}$-smooth f_i uniformly lower bounded by $f_{i,*}$, $i = 1, \ldots, m$, (30) holds with $A = L_{\max}$ and $C = 2L_{\max}\Delta_* = \frac{2L_{\max}}{m}\sum_{i=1}^{m}(f_* - f_{i,*})$, and, as a consequence of (31), RR converges with the rate

$$O\left(L_{\max}\sqrt{m}(f(x^0) - f_*)\left(\frac{\sqrt{m}}{\varepsilon^2} + \frac{\sqrt{L_{\max}(f(x^0) - f(x^*))} + \sqrt{L_{\max}\Delta_*}}{\varepsilon^3}\right)\right)$$

which is better than corresponding bound for SGD (see Table 1) when $L\sqrt{f(x^0) - f_*} \geq \varepsilon\sqrt{L_{\max}m}$ and $L\sqrt{\Delta_*} \geq \varepsilon\sqrt{L_{\max}m}$.

4.3 Variance-reduced Methods

In this section, we discuss variance reduction for non-convex optimization – a special technique aimed at improving the convergence speed of SGD for finite-sum optimization problems (3)+(7). The typical behaviour of SGD with constant stepsize

h and batch-size $r < m$ is as following: during the first iterations the method converges rapidly to some neighbourhood of the solution or local minimum and then it starts to oscillate in this neighbourhood. Such oscillations of SGD are common even for strongly convex problems meaning that it is not a drawback of the problem. The size of the oscillation region is proportional to $h\sigma^2/r$ and this fact hints two simple and famous remedies: decreasing (gradually or suddenly) or small stepsizes and large enough batch-sizes. However, the first option can make the convergence too slow and the second option dramatically increases the iteration cost.

To remove these drawbacks one can apply variance-reduced methods like SAG [183], SAGA [61], SVRG [115], Finito [63], MISO [145]. In particular, all of the mentioned methods have $O\left((m+L/\mu)\ln\frac{1}{\varepsilon}\right)$ convergence rate in the μ-strongly convex case. What is more, they use constant stepsize and at each iteration (besides each m-th iteration or besides the first one) they require one computation of the stochastic gradient with batch size $r = 1$ in the strongly convex case.

Among variance-reduced methods SAGA and SVRG are the most popular ones (see Algorithm 5 and 6). In previous subsections, we already mentioned that to find

Algorithm 5 SAGA [61, 178]

Require: learning rate $h > 0$, starting point $x^0 \in \mathbb{R}^n$, batch size $r \geq 1$

Set $\phi_j^0 = x^0$ for each $j \in [m]$

$v^0 = \frac{1}{m}\sum_{i=1}^{m} \nabla f_i(\phi_j^0)$

for $k = 0, 1, 2, \ldots$ **do**

Uniformly randomly pick sets I_k, J_k from $\{1, 2, \ldots, m\}$ (with replacement) such that $|I_k| = |J_k| = r$

$g^k = \frac{1}{r}\sum_{i \in I_k} \left(\nabla f_i(x^k) - \nabla f_i(\phi_i^k)\right) + v^k$

$x^{k+1} = x^k - hg^k$

$\phi_j^{k+1} = x^k$ for $j \in J_k$ and $\phi_j^{k+1} = \phi_j^k$ for $j \notin J_k$

$v^{k+1} = v^k - \frac{1}{r}\sum_{j \in J_k} \left(\nabla f_j(\phi_j^k) - \nabla f_j(\phi_j^{k+1})\right)$

end for

Algorithm 6 SVRG [115, 178]

Require: learning rate $h > 0$, epoch length T, starting point $x^0 \in \mathbb{R}^n$, batch size $r \geq 1$

$\phi_0 = x_0^0 = x^0$

for $s = 0, 1, 2, \ldots$ **do**

for $k = 0, 1, 2, \ldots, T-1$ **do**

Uniformly randomly pick set I_k from $\{1, \ldots, m\}$ (with replacement) such that $|I_k| = r$

$g^k = \frac{1}{r}\sum_{i \in I_k} \left(\nabla f_i(x_s^k) - \nabla f_i(\phi_s)\right) + \nabla f(\phi_s)$

$x_s^{k+1} = x_s^k - hg^k$

end for

$\phi_{s+1} = x_{s+1}^0 = x_s^k$

end for

ε-stationary GD and SGD require[6] $O\left(m\varepsilon^{-2}\right)$ and $O(\varepsilon^{-4})$ calculations of the gradients of the summands respectively. Despite the fact that SAGA and SVRG were initially analysed only in strongly convex cases, now their convergence in non-convex case is also well-known due to [178, 176]. Unfortunately, when $r = 1$ both SAGA and SVRG guarantee only $O(m\varepsilon^{-2})$ convergence rate as simple GD. However, if $r = m^{2/3}$, then SAGA and SVRG converges with the rate $O(m^{2/3}\varepsilon^{-2})$ which has $m^{1/3}$ times better dependence on m than the complexity bound for GD.

However, the lower bound is $\Omega\left(\sqrt{m}\varepsilon^{-2}\right)$ [74, 134] and there exist optimal algorithms. Essentially, these methods are variations of SARAH [163]. However, in the original paper on SARAH for non-convex problems authors did not prove complexity bounds for the finite-sum optimization problems. After that, in [74] authors proposed the first lower bounds in the small data regime $m = O(L^2(f(x^0) - f^*)\varepsilon^{-4})$ together with the first optimal method called SPIDER. Despite the theoretical optimality of the method, it requires very small stepsize (proportional to ε^{-1}) that leads to the poor behaviour in practice. Moreover, the original proof of the convergence rate for SPIDER is technically tough and, because of it, it is hard to generalize the method for the composite optimization problems. In recent works [215, 216], much simpler optimal method called SpiderBoost was proposed (see Algorithm 7). Moreover, this method works with big constant stepsizes (of order L^{-1}), can be easily generalized for the composite optimization problems, and works well with heavy-ball momentum.

Algorithm 7 SpiderBoost [215, 216]

Require: learning rate $h > 0$, epoch length T, starting point $x^0 \in \mathbb{R}^n$, batch size $r \geq 1$, number of iterations K
  for $k = 0, 1, 2, \ldots$ **do**
    if $k \mod T = 0$ **then**
      Compute $g^k = \nabla f(x^k)$
    else
      Uniformly randomly pick set I_k from $\{1, \ldots, m\}$ (with replacement) such that $|I_k| = r$
      Compute $g^k = \frac{1}{r} \sum\limits_{i \in I_k} \left(\nabla f_i(x^k) - \nabla f_i(x^{k-1})\right) + g^{k-1}$
    end if
    $x^{k+1} = x^k - hg^k$
  end for
  Pick ξ uniformly at random from $\{0, \ldots, K-1\}$
  return x^ξ

Next, in [134], the same lower bound $\Omega\left(\sqrt{m}\varepsilon^{-2}\right)$ was derived without any assumptions on m. Furthermore, authors of [134] proposed a new optimal method called PAGE (see Algorithm 8) which is a variant of SPIDER with random length of the inner loop making the method easier to analyze.

However, in deep neural networks training, variance-reduced methods work typically worse than SGD or SGD with momentum [62]. This happens often due to the

[6] For simplicity we neglect all parameters except m and ε, see the details in Table 2

Algorithm 8 ProbAbilistic Gradient Estimator (PAGE) Algorithm [134]

Require: initial point x^0, stepsize h, minibatch size r, $r' < r$, probabilities $\{p_k\}_{k\geq 0} \in (0,1]$ of large-batch stochastic gradient computation, number of iterations K

$g^0 = \frac{1}{r}\sum\limits_{i\in I_0} \nabla f_i(x^0)$, where I_0 denotes indices in the minibatch, $|I_0| = r$

for $k = 0,1,2,\ldots,K-1$ **do**

$x^{k+1} = x^k - hg^k$

$$g^{k+1} = \begin{cases} \frac{1}{r}\sum\limits_{i\in I_k} \nabla f_i(x^{k+1}) & \text{with probability } p_k, \\ g^k + \frac{1}{r'}\sum\limits_{i\in I'_k} \left(\nabla f_i(x^{k+1}) - \nabla f_i(x^k)\right) & \text{with probability } 1-p_k, \end{cases} \quad \text{where } |I_k| = r,\ |I'_k| = r'$$

end for

return $\hat{x}^K$ chosen uniformly from $\{x^k\}_{k=0}^K$

bad behaviour of variance-reduced methods with several widespread in deep learning tricks like batch normalization, data augmentation and dropout (see the details in [62]). Moreover, if the model is over-parameterized or, in particular, expected strong growth condition (23) or its relaxed version (24) with small noise level hold, SGD is as fast as GD in terms of iteration complexity, meaning that variance reduction is superfluous. That is, variance reduction trick is often not needed or gives worse rates than the rate of SGD for over-parameterized models from theoretical and practical perspectives. Nevertheless, when the problem is not over-parameterized, it makes sense to use variance-reduced methods.

We summarize the discussed above complexity bounds in Table 2. We also want to mention some papers not presented in Table 2 but being highly relevant. In [135], there was developed the generalization of the approach from [117] providing a unified analysis of different variants of SGD, non-optimal variance-reduced methods like SAGA or L-SVRG [106, 123], and some distributed methods with quantization [4] including DIANA-type variance reduction [149, 107] for non-convex optimization. Next, for the online case (3)+(6) with smooth stochastic trajectories the optimal rate $O(\varepsilon^{-3})$ was shown for STOchastic Recursive Momentum (STORM) method [57], which does not require periodical large-batch stochastic gradient computations and is more robust to the parameters selection, and for its proximal variant [226]. These results shade a light on the role of momentum in the stochastic first-order methods. Finally, it is optimal to generalize SPIDER and get similar rates for composition optimization problems [237, 52].

4.3.1 Convex and Weakly Convex Sums of Non-Convex Functions

There are also several results devoted to the case when the objective function f from (7) is (strongly) convex or almost convex, while the summands f_i are smooth, but can be non-convex. In particular, [243] establishes the lower bounds for the case when **(i)** f is μ-strongly convex with $\mu \geq 0$, **(ii)** f is α-weakly convex

$$f(x) - f(y) - \langle \nabla f(y), x-y\rangle \geq -\frac{\alpha}{2}\|x-y\|_2^2,$$

Method	Citation	Complexity
Lower bound	[74, 134]	$L\Delta_0 \min\{\sigma\varepsilon^{-3}, \sqrt{m}\varepsilon^{-2}\}$
GD		$mL\Delta_0\varepsilon^{-2}$
SGD, bounded var.	[83]	$L\Delta_0 \max\{\varepsilon^{-2}, \sigma^2\varepsilon^{-4}\}$
SGD, unbounded var.	[117]	$\frac{L^2\Delta_0}{\varepsilon^4}\max\{\Delta_0, \Delta_*\}$
SVRG, $r=1$	[178]	$mL\Delta_0\varepsilon^{-2}$
SVRG, $r=\lceil m^{2/3}\rceil$	[178]	$m^{2/3}L\Delta_0\varepsilon^{-2}$
SAGA, $r=1$	[178]	$mL\Delta_0\varepsilon^{-2}$
SAGA, $r=\lceil m^{2/3}\rceil$	[178]	$m^{2/3}L\Delta_0\varepsilon^{-2}$
SpiderBoost	[215, 216]	$m^{1/2}L\Delta_0\varepsilon^{-2}$
SpiderBoost-M	[216]	$m^{1/2}L\Delta_0\varepsilon^{-2}$
SPIDER	[74]	$L\Delta_0 \min\{\sigma\varepsilon^{-3}, \sqrt{m}\varepsilon^{-2}\}$
PAGE	[134]	$L\Delta_0 \min\{\sigma\varepsilon^{-3}, \sqrt{m}\varepsilon^{-2}\}$

Table 2 Overview of the complexity results for different variance-reduced methods applied to solve problem (3)+(7) with L-smooth summands. The column "Complexity" contains an overall number of stochastic first-order oracle calls needed to find ε-stationary point neglecting constant factors. Notation: $\Delta_0 = f(x^0) - f(x^*)$, $\Delta_* = \frac{1}{m}\sum_{i=1}^m (f_* - f_{i,*})$, σ^2 = a uniform bound for the variance of the stochastic gradient (13) (can be ∞ for variance-reduced methods), r = batchsize.

and **(iii)** f_i are α-weakly convex. Due to the additional assumptions on the structure of non-convexity in the problem the proposed lower bounds are tighter in these situations than the lower bound from [74, 134]. Moreover, there exist optimal methods for each case: **(i)** – SDCA without Duality [188], KatyushaX [7], **(ii)** – RepeatSVRG [44, 2], SPIDER [74], SNVRG [246], **(iii)** – Natasha [5], RapGrad [127], StagewiseKatyusha [51].

4.4 Adaptive Methods

One of the most significant issues of the methods described above is that they require tuning of the stepsize and other parameters (e.g., batch-size) when used in practice. It is often challenging and takes a lot of time, especially for training deep neural networks. That is why, in the recent few years, adaptive methods gained a lot of attention. Below we discuss the most popular ones – AdaGrad and Adam – as well as their variants. In fact, all of these methods depend on some parameters, but these algorithms are much more robust than other variants of SGD or variance-reduced methods. Therefore, they are often called adaptive. One can find PyTorch implementation of many popular adaptive first-order methods together with with visualization of their convergence on Rosenbrock and Rastrigin functions in [1].

4.4.1 AdaGrad and Adam

AdaGrad. As we mentioned above, SGD requires the tuning of the stepsize. The first algorithm aiming to remove this drawback of SGD was AdaGrad [68]:

$$x_i^{k+1} = x_i^k - \frac{h}{\sqrt{G_i^k + \delta}} g_i^k,$$

where the subscript i denotes the i-th component of the vector, $G_i^k = \sum_{t=0}^k (g_i^t)^2$, and δ is some small positive number preventing from the division by zero and typically taken of the order 10^{-8}. AdaGrad can be considered as a special case of SGD with different per-coordinate stepsizes.

The main advantage of AdaGrad is in its robustness to the choice of h: in practice, it often works well with the default value $h = 10^{-2}$. Moreover, AdaGrad was shown to work well with sparse data [69]. However, in the dense settings AdaGrad stepsizes rapidly decrease which leads to the slow convergence of the method [220].

Adam. To resolve this issue of AdaGrad one can use exponential moving averages instead of sums G_i^k leading to the method called RMSprop [206]. Then, based on RMSprop authors of [121] proposed one the most popular methods in deep learning – Adam[7]:

$$\begin{aligned} m_i^k &= \beta_1 m_i^{k-1} + (1-\beta_1) g_i^k, \quad \hat{m}_i^k = \frac{m_i^k}{1-(\beta_1)^k}, \\ v_i^k &= \beta_2 v_i^{k-1} + (1-\beta_2)(g_i^k)^2, \quad \hat{v}_i^k = \frac{v_i^k}{1-(\beta_2)^2}, \\ x_i^{k+1} &= x_i^k - \frac{h}{\sqrt{\hat{v}_i^k} + \delta} \hat{m}_i^k, \quad i = 1, \ldots, n, \end{aligned}$$

δ is some small positive number preventing from the division by zero and typically taken of the order 10^{-8}. Default values $\beta_1 = 0.9$ and $\beta_2 = 0.999$ from the original paper [121] often make Adam work well in practice. Adam was initially analyzed in the online convex case, but then authors of [177] found out the flaw in the proof for Adam and proposed a convergent variant of Adam called AMSGrad.

Convergence Guarantees. While the superiority of AdaGrad and Adam in comparison to SGD was noticed in many application [69, 125, 88], the best-known complexity bounds for AdaGrad, Adam, and their modifications are the same or even worse than ones for SGD [48, 245, 232, 219, 64]. Furthermore, these complexity results in non-convex case under more restrictive assumption, e.g., uniformly bounded second moment of the stochastic gradient, than their counterparts for SGD. Among other works providing complexity results for Adam and AdaGrad in the non-convex case we emphasize [64] because of the generality and the simplicity of the proofs. Moreover, the unified analysis of proximal variants of AdaGrad and Adam was proposed in [231].

[7] To distinguish exponents from superindexes we use braces $(\cdot)$ for exponents.

Next, in [236] the theoretical and empirical study why Adam sometimes behaves significantly better than SGD was conducted. The authors of [236] empirically discovered that Adam performs better than SGD when stochastic gradients are heavy-tailed and the reason is that Adam does an "adaptive gradient clipping" [89, 90, 148, 170, 209]. In the same work [236] authors showed that in such situations SGD can fail to converge while clipped-SGD (with general and coordinate-wise clipping operators) provably converges to ε-stationary point. Moreover, in [235] it was shown that Gradient Descent with clipping converges even under weaker assumption than L-smoothness in the non-convex case with the rate $\sim \varepsilon^{-2}$ while Gradient Descent in the same settings can converge arbitrary slower. Then, the bound from [235] was improved in [233]. Finally, it is known [89] that clipped-SGD works better than SGD in the vicinity of extremely steep cliffs. A very similar approach based on the normalization of Gradient Descent was also studied in [103, 131].

4.4.2 Adaptive SGD

The approach described in Section 4.1.2 for general stochastic optimization problem (3) with the objective given as (6) was recently extended in [70, 71] to obtain adaptive methods with Armijo-type line-search for stochastic non-convex optimization. To do that they consider Algorithm 3 with the mini-batch stochastic gradient (12) and mini-batch size $r = \max\{1, 8\sigma_0^2/\varepsilon^2\}$, where $\sigma_0 \geqslant \sigma$. In each iteration k of Algorithm 3 the stepsize is taken as $h_k = 1/L_k := 1/(2^{i_k-1}L_{k-1})$ by increasing $i_k \geqslant 0$ until the inequality

$$f(x^{k+1}) \leq f(x^k) + \left\langle \frac{1}{r}\sum_{l=1}^{r} \nabla f(x,\xi_l), x^{k+1} - x^k \right\rangle + L_k\|x^{k+1} - x^k\|_2^2 + \frac{\varepsilon^2}{32L_k}$$

is satisfied. This inequality is an inexact upper quadratic bound which follows for sufficiently large L_k from the L-smoothness and bounded variance. Thus, L_k plays the role of a guess of the Lipschitz constant L locally between the points x^k and x^{k+1}. The authors of [70, 71] propose also methods for convex problems based on the same idea with the difference that in the convex case the mini-batch size r depends on the iteration counter k. Careful choice of this dependence allows to simultaneously adaptively choose both the stepsize h_k and the mini-batch size r_k. These methods have the same, up to logarithmic factors, iteration complexity and total number of stochastic oracle calls as their non-adaptive counterparts. In particular, for the non-convex case the iteration complexity to obtain ε-stationary point is $\widetilde{O}\left(L(f(x^0)-f_*)/\varepsilon^2\right)$ and the oracle complexity is $\widetilde{O}\left(L\left(f(x^0)-f_*\right)\max\left\{1/\varepsilon^2, \sigma^2/\varepsilon^4\right\}\right)$. Moreover, empirically, the methods designed for convex problems turned out to be more efficient on non-convex problems than the method designed for non-convex problems.

5 First-Order Methods under Additional Assumptions

In the previous parts of the paper, we focused on general non-convex problems. In this section, we consider two subclasses of non-convex objective functions which satisfy assumptions weaker than convexity and, at the same time, strong enough to obtain good global convergence rates of optimization algorithms. For simplicity, we consider an unconstrained optimization problem (1) with $Q = \mathbb{R}^n$.

5.1 Polyak–Łojasiewicz Condition

A function $f(x)$ is said to satisfy the Polyak–Łojasiewicz (PŁ) condition [171, 140] (or to be gradient dominated) if for all $x \in \mathbb{R}^n$

$$f(x) - f(x^*) \leqslant \frac{1}{2\mu} \|\nabla f(x)\|_2^2.$$

This condition implies that any stationary point of $f(x)$ is a global minimum, although it is not necessarily unique. In particular, this property holds for strongly convex functions. It was first shown in [171] that if the objective is also L-smooth, then gradient descent linearly converges to a global minimum, i.e.,

$$f(x^k) - f(x^*) \;\leqslant\; \exp\left(-\frac{\mu}{L}k\right) \left(f\left(x^0\right) - f(x^*)\right).$$

The Polyak–Łojasiewicz condition is naturally satisfied for the problems of solving nonlinear systems of equalities $g(x) = 0$, where $g(x)$ is a vector-valued function. This problem can be equivalently reformulated as

$$\min_{x \in \mathbb{R}^n} \left\{ f(x) = \frac{1}{2} \|g(x)\|_2^2 \right\}.$$

Assuming that, for all $x \in \mathbb{R}^n$

$$\lambda_{\min}\left(J_g(x) J_g^T(x)\right) \geq \mu > 0,$$

where $J_g(x)$ is the Jacobian matrix of $g(x)$, one can show that

$$\|\nabla f(x)\|^2 = \|J_g^T(x) g(x)\|^2 \geqslant \mu \|g(x)\|^2 = 2\mu f(x),$$

which is exactly the Polyak–Łojasiewicz condition since $g(x^*) = 0$. An extensive review of first-order optimization methods under this condition, as well as its relationship with other classes of functions, can be found in [116]. An interesting example of the emergence of PŁ condition in Linear Feedback Control theory was recently described in [76].

Next, consider the convergence of gradient descent under the PŁ condition in terms of relative accuracy $\widetilde{\nabla} f(x)$

$$\|\widetilde{\nabla} f(x) - \nabla f(x)\|_2 \leq \alpha \|\nabla f(x)\|_2,$$

where $\alpha \in [0,1)$. Let the stepsize h in gradient descent

$$x^{k+1} = x^k - h\widetilde{\nabla} f(x^k)$$

be computed using the following formula:

$$h = \frac{1}{L}\frac{1-\alpha}{(1+\alpha)^2}.$$

Combining this with the Lipschitz condition, we obtain

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L}\frac{(1-\alpha)^2}{(1+\alpha)^2}\|\nabla f(x^k)\|_2^2,$$

leading to

$$f(x^N) - f(x^*) \leq \left(1 - \frac{\mu}{L}\frac{(1-\alpha)^2}{(1+\alpha)^2}\right)^N \left(f(x^0) - f(x^*)\right).$$

As a result, we achieve a linear convergence rate for the gradient descent under the PŁ condition.

5.1.1 Stochastic First-Order Methods under Polyak–Łojasiewicz Condition

The majority of the methods described in Section 4 are analyzed under PŁ condition as well. That is, one can find the state-of-the-art results for different variants of SGD and non-accelerated variance reduced methods like SVRG and SAGA in [135], accelerated variance reduced methods like PAGE in [134], the tightest known analysis of Random Reshuffling under PŁ condition in [3], and the convergence results for SGD in the over-parameterized case with constant, Armijo-type, and stochastic Polyak's stepsizes in [211], [212], and [139, 95] respectively.

5.2 Star-convexity and α-weak-quasi-convexity

A function $f(x)$ is called star-convex if for some global minimizer x^* and for all $\lambda \in [0,1]$ and $x \in \mathbb{R}^n$

$$f(\lambda x + (1-\lambda)x^*) \leqslant \lambda f(x) + (1-\lambda) f(x^*).$$

While any interval connecting two points on the graph of a convex function lies not lower than the graph, for a star-convex functions this is assumed only for intervals connecting some fixed global minimizer and any other point on the graph. This condition is considerably weaker than convexity, even for functions of one variable. For example, the function $|x|(1-e^{-|x|})$ is a non-convex star-convex function. The authors of [129] analyze a cutting plane method for minimization of this class of functions and obtain a polylogarithmic in ε and polynomial in n complexity bound using only function evaluations. The authors of [100, 158] prove that the same Algorithm 1 possesses the following convergence rate for star-convex L-smooth functions

$$\min_{k=[N/2],\ldots,N} \|\nabla f(y^k)\|_*^2 \leqslant \frac{64L^2V[x^0](x^*)}{N^3},$$
$$f(x^N) - f(x_*) \leqslant \frac{4LV(x_*,x^0)}{N^2}.$$

A more general class of functions is the class of α-weakly-quasi-convex functions satisfying

$$f(x) - f(x^*) \leqslant \frac{1}{\alpha}\langle \nabla f(x), x - x^* \rangle$$

for some $\alpha \in (0,1]$ and some global minimizer x^*. Continuously differentiable 1-weakly-quasi-convex functions are exactly the star-convex functions. The authors of [99] propose an algorithm with iteration complexity $O(\alpha^{-1}L^{1/2}R\varepsilon^{-1/2})$, where R is an upperbound on the initial distance to the point x^*. A slightly worse bound $O(\alpha^{-3/2}L^{1/2}R\varepsilon^{-1/2})$ is obtained in [158] by restarting Algorithm 1. Both approaches require a line search for which the complexity also needs to be estimated. The authors of [105] analyze this complexity and propose an algorithm with $O(\alpha^{-1}L^{1/2}R\varepsilon^{-1/2})$ iteration complexity and the same up to a logarithmic factor in $\alpha^{-1}\varepsilon^{-1}$ number of function and gradient evaluations. Moreover, they provide a similar lower complexity bound, thus proving that their method is optimal. Further, they also consider a class of (α,μ)-strongly quasi-convex functions satisfying

$$f(x) - f(x^*) \leqslant \frac{1}{\alpha}\langle \nabla f(x), x - x^* \rangle - \frac{\mu}{2}\|x - x^*\|^2$$

and provide an algorithm which has iteration complexity

$$O(\alpha^{-1}L^{1/2}\mu^{-1/2}\log(\alpha^{-1}\varepsilon^{-1}))$$

and requires up to a logarithmic factor the same number function and gradient evaluations. Similar optimal complexity bounds for accelerated gradient method for α-weakly-quasi-convex functions and (α,μ)-strongly quasi-convex functions were obtained in [35] by extending the estimating sequence technique.

A more wide class of functions that covers the class of α-weakly-quasi-convex functions referred to as approximately homogeneous functions satisfying the condition

$$N(f(x) - f(x^*)) \leqslant \langle \partial f(x), x - x^* \rangle \leqslant M(f(x) - f(x^*)),$$

where $\partial f(x)$ is a subgradient of $f(x)$ and N, M are some constants. This class of functions was first defined in [195] and discussed in [172].

6 Higher-Order Methods

6.1 Second-Order Methods

Another branch of optimization incremental methods for solving (3) are methods that use the second-order information about the function. This information is very helpful to escape saddle-points by using a negative curvature. Next we define an (ε, δ)-second-order stationary point x^* if

$$\|\nabla f(x^*)\|_2 \leq \varepsilon, \quad \lambda_{\min}\left(\nabla^2 f(x^*)\right) \geq -\delta.$$

Next in this section we suppose that $f(x)$ has L_2-Lipschitz second-order derivative. The basic method for this class of problems is a Cubic Regularized Newton's method [159].

$$x_{Cubic}^{k+1} = x^k + \operatorname*{argmin}_{s \in \mathbb{R}^n} \left[\nabla f(x^k)^\top s + \frac{1}{2} s^\top \nabla^2 f(x^k) s + \frac{\sigma}{6} \|s\|_2^3 \right], \tag{33}$$

where $\sigma \geq 0$. It globally converges to the minimum for convex functions and converges to a (ε, δ)-second-order stationary point for non-convex function within $O(\varepsilon^{-3/2})$ number of iterations. Note, that the subproblem (33) is also non-convex but in [159] authors proposed a method to solve this problem as a convex problem via special choose of σ and line-search for a dual problem. It works very good for small problems, but unfortunately, for many big ML problems it is hard to compute full Hessian and inverse such big matrix. Recent work has therefore explored the use of Hessian-vector products $\nabla^2 f(x) \cdot \mathbf{v}$, which can be computed as efficiently as gradients in many cases including neural networks by using autogradient technique. By this Hessian-vector product we can efficiently find x_{t+1}^{Cubic} by variants of gradient descent [42]. Several algorithms incorporating Hessian-vector products [6, 8] have been shown to achieve faster convergence rates than gradient descent in the non-stochastic setting. However, in the stochastic setting where we only have access to stochastic Hessian-vector products, significantly less progress has been made.

One of the improvement of this method was done in [218]. Authors introduce the momentum step and speed up the convergence speed. This technique widely used to speed up the first order methods and also can speed up the second order method.

Also a second-order methods which have access to the Hessian of f can exploit negative curvature to more effectively escape saddles and arrive at local minima. To show this concept we introduce one of such methods [221]. There are two types of steps: gradient steps and a step in a negative curvature for the Hessian. So

- If $\|\nabla f(x^k)\|_2 > \varepsilon$, we do gradient step.

Algorithm 9 CRm

1: **Input:** Initialization $x^0 = y^0 \in \mathbb{R}^d, \rho < 1, M > L_2$.
2: **for** $k = 0, 1, \ldots$ **do**
3: **Cubic step:**

$$s^{k+1} = \operatorname*{argmin}_s \left[\nabla f(x^k)^\top s + \frac{1}{2} s^\top \nabla^2 f(x^k) s + \frac{\rho}{6} \|s\|_2^3 \right],$$
$$y^{k+1} = x^k + s^{k+1}.$$

4: **Momentum step:**

$$\beta^{k+1} = \min\{\rho, \|\nabla f(y^{k+1})\|_2, \|y^{k+1} - x^k\|_2\},$$
$$v^{k+1} = y^{k+1} + \beta^{k+1}(y^{k+1} - y^k).$$

5: **Monotone Step:**

$$x^{k+1} = \operatorname*{argmin}_{x \in \{y^{k+1}, v^{k+1}\}} f(x).$$

6: **end for**

- Otherwise, if $\lambda_{\min}\left(\nabla^2 f(x^k)\right) < -\delta$, choose p^k to be the eigenvector corresponding to $\lambda_{\min}\left(\nabla^2 f(x^k)\right)$ and do step $x^{k+1} = x^k + \alpha_k p^k$.

There are different policies to α_k and gradient steps. The main idea here is to use the first-order methods as a cheap main method and switch to expensive second-order methods when we reach local stationary point and want to escape it to find a better local minimum. Methods with this idea are still developing. In [82, 112] was proved that gradient methods with additive noise are able to escape from nondegenerate saddle points and find approximate local minima. These ideas lead to the state of art first-order methods to find local minima with Hessian-vector product [44, 180, 8, 227, 9, 114, 74, 162]. In recent works [75, 113, 179] was proved that stochastic gradient descent can escape from saddle point and converges to approximate local minima.

6.2 Stochastic Second-Order Methods

Now we move to stochastic version of problem (3). Firstly, we speak about online version (6), where we minimize expectation of some stochastic function. In the work [207] authors propose a stochastic optimization method that utilizes stochastic gradients and Hessian-vector products to find an (ε, δ)-second-order stationary point using only $O(\varepsilon^{-3.5})$ oracle evaluations. This rate improves upon the $O(\varepsilon^{-4})$ rate of stochastic gradient descent, and matches the best-known result for finding local minima without the need for any delicate acceleration or variance reduction techniques.

Algorithm 10 Stochastic Cubic Regularization

Require: mini-batch sizes n_1, n_2, initialization $\mathbf{x_0}$, number of iterations T_{out}, and final tolerance ε.

for $t = 0, \dots, T_{\text{out}}$ **do**
  Sample $S_1 \leftarrow \{\xi_i\}_{i=1}^{n_1}$, $S_2 \leftarrow \{\xi_i\}_{i=1}^{n_2}$.
  $\mathbf{g}^t \leftarrow \frac{1}{|S_1|}\sum_{\xi_i \in S_1} \nabla f(x^k;\, \xi_i)$
  $\mathbf{B}^t[\cdot] \leftarrow \frac{1}{|S_2|}\sum_{\xi_i \in S_2} \nabla^2 f(x^k, \xi_i)(\cdot)$
  $\Delta, \Delta_m \leftarrow$ Cubic-Subsolver($\mathbf{g}^t,\ \mathbf{B}^t[\cdot], \varepsilon$)
  $x^{t+1} \leftarrow x^t + \Delta$
  if $\Delta_m \geq -\frac{1}{100}\sqrt{\frac{\varepsilon^3}{\rho}}$ **then**
    $\Delta \leftarrow$ Cubic-Finalsolver($\mathbf{g}^t,\ \mathbf{B}^t[\cdot], \varepsilon$)
    $x^* \leftarrow x^t + \Delta$
    brake
  end if
end for

Ensure: x^* if the early termination condition was reached, otherwise the final iterate $x_{T_{\text{out}}+1}$.

This is a general-purpose stochastic cubic regularization meta-algorithm in Algorithm 10, which employs a black-box subroutine to solve stochastic cubic subproblems. At a high level, in order to deal with stochastic gradients and Hessians, we sample two independent minibatches S_1 and S_2 at each iteration. Denoting the average gradient by

$$\mathbf{g}^t = \frac{1}{|S_1|}\sum_{\xi_i \in S_1} \nabla f(x^k, \xi_i)$$

and the average Hessian by

$$\mathbf{B}^t = \frac{1}{|S_2|}\sum_{\xi_i \in S_2} \nabla^2 f(x^k, \xi_i),$$

this implies a *stochastic cubic submodel*:

$$m^k(x) = f(x^k) + (x - x^k)^\top \mathbf{g}^t + \frac{1}{2}(x - x^k)^\top \mathbf{B}^t (x - x^k) + \frac{\sigma}{6}\|x - x^k\|_2^3.$$

Although the subproblem depends on $\mathbf{B}^t$, we note that our meta-algorithm never explicitly formulates this matrix, only providing the subsolver access to $\mathbf{B}^t$ through Hessian-vector products, which we denote by $\mathbf{B}^t[\cdot] : \mathbb{R}^d \to \mathbb{R}^d$. We hence assume that the subsolver performs gradient-based optimization to solve the subproblem, as $\nabla m^k(x)$ depends on $\mathbf{B}^t$ only via $\mathbf{B}^t[x - x^k]$.

After sampling minibatches for the gradient and the Hessian, Algorithm 10 makes a call to a black-box cubic subsolver to optimize the stochastic submodel $m^k(x)$. The subsolver returns a parameter change Δ, i.e., an approximate minimizer of the submodel, along with the corresponding change in submodel value,

$\Delta_m = m^k(x^k + \Delta) - m^k(x^k)$. The algorithm then updates the parameters by adding Δ to the current iterate, and checks whether Δ_m satisfies a stopping condition.

How many Hessians should we take? By concentration inequalities it is possible to show that we need

$$|S_2| = O\left(\varepsilon^{-1}\right).$$

So in total, the method converges with $O\left(\varepsilon^{-3/2}\right)$ iterations and $O\left(\varepsilon^{-5/2}\right)$ Hessian calculations of the function.

In paper [14] this approach is improved by using special variance reduction technique. Authors get method that needs only $O(\varepsilon^{-3})$ gradients and Hessian-vector products for finding second-order stationary point. Also in this article authors prove lower bounds for higher-order stochastic problems.

What is the main advantage of such methods? We make calculate less Hessians than in full CR version and also do it in parallel if we have many cores for computing. The simplicity of the algorithms both with fast rates and escaping from saddle-points lead us to very good optimization methods for non-convex stochastic problems.

Next we go to offline version that works with sum of functions (7). In this regime we have n functions and hence classic CR needs to compute $O(n\varepsilon^{-3/2})$ Hessians. To reduce it in papers [122, 224] authors used subsampled gradient and subsampled Hessian, which achieve $\tilde{O}(n\varepsilon^{-3/2} \wedge \varepsilon^{-7/2})$ gradient complexity and $\tilde{O}(n\varepsilon^{-3/2} \wedge \varepsilon^{-5/2})$ Hessian complexity similarly to the previous section. Next appears many articles with different stochastic variance-reduced cubic(SVRC) methods. To collect this results in one place we add a table with the convergence rates, where $a \wedge b = \min(a;b)$.

Method	Gradient	Hessian
CR [159]	$O\left(n \cdot \varepsilon^{-3/2}\right)$	$O\left(n \cdot \varepsilon^{-3/2}\right)$
SCR [122, 224]	$\tilde{O}\left(n \cdot \varepsilon^{-3/2} \wedge \varepsilon^{-7/2}\right)$	$\tilde{O}\left(n \cdot \varepsilon^{-3/2} \wedge \varepsilon^{-5/2}\right)$
SVRC1 [247]	$\tilde{O}\left(n^{4/5} \cdot \varepsilon^{-3/2}\right)$	$\tilde{O}\left(n^{4/5} \cdot \varepsilon^{-3/2}\right)$
SVRC2 [217, 248]	$\tilde{O}\left(n \cdot \varepsilon^{-3/2}\right)$	$\tilde{O}\left(n^{2/3} \cdot \varepsilon^{-3/2}\right)$
SVRC3 [238]	$\tilde{O}\left(n \cdot \varepsilon^{-3/2} \wedge n^{2/3} \cdot \varepsilon^{-5/2}\right)$	$\tilde{O}\left(n^{2/3} \cdot \varepsilon^{-3/2}\right)$
STR [190]	$\tilde{O}\left(n \cdot \varepsilon^{-3/2} \wedge n^{1/2} \cdot \varepsilon^{-2}\right)$	$\tilde{O}\left(n^{1/2} \cdot \varepsilon^{-3/2} \wedge \varepsilon^{-2}\right)$
SRVRC [244]	$\tilde{O}\left(n \cdot \varepsilon^{-3/2} \wedge n^{1/2} \cdot \varepsilon^{-2} \wedge \varepsilon^{-3}\right)$	$\tilde{O}\left(n^{1/2} \cdot \varepsilon^{-3/2} \wedge \varepsilon^{-2}\right)$

Table 3 Overview of the complexity results (number of computations of gradients and Hessians of functions in (7)

So as a result we have a method that may works efficiently with the big sum by using stochastic nature but also it uses Hessian information to escape saddles more effectively and arrive to better local minimum. This statement is also proved by experiments in [225, 167, 147,]. In this works authors make experiments for different second-order methods and show how in practice they compete with the first-order methods without second-order information. The main conclusions in these papers are that second-order methods find deeper local minima and escape saddle-points, they are more stable to hyperparameters, sub-sampling accelerate computations and

parallelization of such methods. Hence, second-order may be competitive in practice with first-order methods.

6.3 Tensor Methods

Next we present a third-order methods that uses a fourth-order regularized model to find local minima of smooth and non-convex objective functions. One of the motivation for such methods that the second-order method could get stuck at the so-called degenerate saddle point, where Hessian matrix has nonnegative eigenvalues with some eigenvalues equal to 0. In paper [249] it is shown how gradient descent and cubic regularization method stuck in such points for even small problems, like $f(x,y) = x^3 - 3xy^2$ in degenerate saddle point point $(0,0)$. So, we should use third-order information to escape such points. But calculation of third-order derivative for big problem would be very computationally expensive. This problem leads us to stochastic tensor methods.

The main idea of stochastic method that by different concentration inequalities we can compute much less Hessians and third-order derivatives for sum type problem, than gradients. Correct proportions is written in (38). For example if we have 200000 functions in sum, we may compute full gradient, only 10000 Hessians and 100 third-order derivatives and get the same speed as for full Hessian and full third-order derivatives.

First, we lay out some standard assumptions regarding the function f for smoothness. In the following, we will denote the directional derivative of the function f at x along the directions $h_j \in \mathbb{R}^d$, $j = 1, \dots, p$ as

$$\nabla^p f(x)[h^1, \dots, h^p].$$

For instance, $\nabla f(x)[h] = \nabla f(x)^\top h$ and $\nabla^2 f(x)[h]^2 = h^\top \nabla^2 f(x) h$.

The functions f_i for each $p = 0, \dots, 3$ has L_p-Lipschitz-continuous derivatives,

$$\|\nabla^p f_i(x) - \nabla^p f_i(y)\|_2 \leq L_p \|x - y\|_2$$

for all $x, y \in \mathbb{R}^d$.

In papers [25, 46, 45] was proved that tensor p-order method with Taylor approximation is optimal, match lower bounds, and converges with the rate $O(\varepsilon^{-(p+1)/p})$ for non-convex problems, hence for the third order method we get the rate $O(\varepsilon^{-4/3})$. But the main difficulties for using this method is calculation of full Hessian and third-order derivative for big problem. The same as for the second order methods. The good news is that we can calculate inexact derivative and for higher derivative we need smaller batch-size for good convergence. In paper by [142] introduce such method that work with batch tensors and converges as fast as for full-batch methods.

Next we will describe this algorithm. This algorithm uses sub-sampled derivatives instead of exact quantities and its implementation relies on tensor-vector products only. The proposed approach is shown to find an (ε, δ)-second-order critical

point in at most $O\left(\max\left(\varepsilon^{-4/3},\delta^{-2}\right)\right)$ iterations, thereby matching the rate of deterministic approaches.

We construct a surrogate model to optimize f based on a truncated Taylor approximation as well as a power prox function weighted by a sequence $\{\sigma_k\}_k$ that is controlled adaptively according to the fit of the model to the function f. Since the full Taylor expansion of f requires computing high-order derivatives that are expensive, we instead use an inexact model defined as

$$\begin{aligned} m_k(s) &= \phi_k(s) + \frac{\sigma_k}{4}\|s\|_2^4, \\ \phi_k(s) &= f(x^k) + g_k^\top s + \frac{1}{2}s^\top B_k s + \frac{1}{6}T_k[s]^3, \end{aligned} \tag{34}$$

where g_k, B_k and T_k approximate the derivatives $\nabla f(x^k), \nabla^2 f(x^k)$ and $\nabla^3 f(x^k)$ through sampling as follows. Three sample sets S^g, S^b and S^t are drawn and the derivatives are then estimated as

$$\begin{aligned} g_k &= \frac{1}{|S^g|}\sum_{i\in S^g}\nabla f_i(x^k), B_k = \frac{1}{|S^b|}\sum_{i\in S^b}\nabla^2 f_i(x^k), \\ T_k &= \frac{1}{|S^t|}\sum_{i\in \S^t}\nabla^3 f_i(x^k). \end{aligned}$$

Note the implementation of the algorithm we analyze does not require the computation of the Hessian or the third-order tensor – both of which would require significant computational resources – but instead directly compute Tensor-vector products with a complexity of order $O(n)$.

We will make use of the following condition in order to reach an ε-critical point:

For a given ε accuracy, one can choose the size of the sample sets S^g, S^b, S^t for sufficiently small $\kappa_g, \kappa_b, \kappa_t > 0$ such that:

$$\|g_k - \nabla f(x^k)\|_2 \le \kappa_g \varepsilon, \tag{35}$$

$$\|(B_k - \nabla^2 f(x^k))s\|_2 \le \kappa_b \varepsilon^{2/3}\|s\|_2, \ \ \forall s \in \mathbb{R}^d, \tag{36}$$

$$\|T_k[s]^2 - \nabla^3 f(x^k)[s]^2\|_2 \le \kappa_t \varepsilon^{1/3}\|s\|_2^2, \ \ \forall s \in \mathbb{R}^d. \tag{37}$$

Practically we can take the following choice of the size of the sample sets S^g, S^b and S^t:

$$n_g = \tilde{O}\left(\frac{L_0^2}{\kappa_g^2\varepsilon^2}\right), \quad n_b = \tilde{O}\left(\frac{L_1^2}{\kappa_b^2\varepsilon^{4/3}}\right), \quad n_t = \tilde{O}\left(\frac{L_2^2}{\kappa_t^2\varepsilon^{2/3}}\right), \tag{38}$$

where $\tilde{O}$ hides poly-logarithmic factors and a polynomial dependency to d. Here we can see that to get good convergence rate $O\left(\max\left(\varepsilon^{-4/3},\delta^{-2}\right)\right)$ we may use much less computations because of stochastic nature of data and tensor concentration inequalities.

Algorithm 11 Stochastic Tensor Method (STM)

Input:
Starting point $x_0 \in \mathbb{R}^d$ (e.g $x_0 = \mathbf{0}$)
$0 < \gamma_1 < 1 < \gamma_2 < \gamma_3, 1 > \eta_2 > \eta_1 > 0$, and $\sigma_0 > 0, \sigma_{min} > 0$
for $k = 0, 1, \ldots,$ until convergence **do**
  Sample gradient g_k, Hessian B_k and T_k such that Eq. (35), Eq. (36) & Eq. (37) hold.
  Obtain s^k by solving $m_k(s^k)$ (Eq. (34)).
  Compute $f(x^k + s^k)$ and

$$\rho_k = \frac{f(x^k) - f(x^k + s^k)}{f(x^k) - \phi_k(s^k)}.$$

  Set

$$x^{k+1} = \begin{cases} x^k + s^k & \text{if } \rho_k \geq \eta_1 \\ x^k & \text{otherwise.} \end{cases}$$

  Set

$$\sigma_{k+1} = \begin{cases} [\max\{\sigma_{min}, \gamma_1 \sigma_k\}, \sigma_k] & \text{if } \rho_k > \eta_2 \text{ (very successful iteration)} \\ [\sigma_k, \gamma_2 \sigma_k] & \text{if } \eta_2 \geq \rho_k \geq \eta_1 \text{ (successful iteration)} \\ [\gamma_2 \sigma_k, \gamma_3 \sigma_k] & \text{otherwise (unsuccessful iteration).} \end{cases}$$

end for

The optimization algorithm we consider is detailed in Algorithm 11. At iteration step k we sample three sets of datapoints from which we compute stochastic estimates of the derivatives of f. We then obtain the step s^k by solving the problem

$$s^k = \arg\min_{s \in \mathbb{R}^d} m_k(s),$$

either exactly or approximately and update the regularization parameter σ_k depending on ρ_k, which measures how well the model approximates the real objective. This is accomplished by differentiating between different types of iterations. Successful iterations (for which $\rho_k \geq \eta_1$) indicate that the model is, at least locally, an adequate approximation of the objective such that the penalty parameter is decreased in order to allow for longer steps.

For the case where the desired accuracy is high, i.e. $\varepsilon \ll \frac{1}{N}$. Total worst-case complexity (first-order stationarity) for Algorithm 11 with the non-convex AGD variant presented in [43] as subsolver needs at most $\tilde{O}(N\varepsilon^{-3})$ (stochastic) oracle calls to reach an iterate x^* such that $\|\nabla f(x^*)\|_2 \leq \varepsilon$ and $\varepsilon \ll \frac{1}{N}$.

The final total complexity in terms of ε is an improvement over state-of-the-art methods such as NEON + SCSG [227] $O(\varepsilon^{-3.33})$, SCR [207] and Natasha2 [8] $O(\varepsilon^{-3.5})$. We expect that the dependency on N could be further reduced using the variance reduction techniques. Furthermore, we want to point out that the use of a specialised subproblem solver instead of AGD – as was done in [42] for the cubic model – could *significantly* improve the rate. For comparison, while the rate of AGD to reach $\|\nabla f(x)\|_2 \leq \varepsilon$ is $O(\varepsilon^{-5/3})$, the cubic solver from [42] achieves $O(\varepsilon^{-1})$.

7 Zeroth-Order Methods

Gradient free or zeroth-order optimization methods, which use only function values, are becoming increasingly important in machine learning problems, especially in reinforcement learning [146], black-box adversarial attacks on deep neural networks [168] and other problems with structure making gradients difficult or infeasible to obtain.

While there is a class of methods that does not have any connection to the gradient, for example, random search algorithms [185] (which are one of the first methods of zeroth-order optimization, beside grid search), the Nelder–Mead algorithm [152], the model-based methods (see Chapters 2-6 and 10-11 in [55]) or the recent stochastic three points (STP) method [22] and its momentum variant STMP [92] most zeroth-order optimization methods use gradient estimations, such as $g(x) = \sum_{i=1}^{n} \frac{f(x+\mu e_i)-f(x)}{\mu} e_i$ (where e_i are columns of identity matrix I, $i \in \{1,\dots,n\}$), then for good enough functions ($f \in C_L^{1,1}$ i.e. continuously differentiable with Lipschitz-continuous gradient) it can be shown, for example, that $\|g(x) - \nabla f(x)\|_2 \leqslant L\mu\sqrt{n}$. One then can consider some first-order optimization scheme, replace actual gradients with their estimations, and use bounds like this to return to gradients from estimations in proofs, obtaining the results for the zeroth-order case relatively easy.

While such deterministic zeroth-order schemes (like the *GD* with gradient estimation of the same form as above) often suffer from the problem dimensionality because of the number of oracle calls needed to reconstruct the gradient (n for the estimation mentioned above, see also [20] for other examples), in a randomized approach one can use two- or one- point schemes of gradient approximation which makes every iteration simpler, sometimes leading to better results in terms of oracle calls [137]. Another benefit of the stochastic approach is that such methods often have good theoretical properties, for example, the Gaussian smoothing approach [160] that gives a smoothed version of the initial function, for which the convergence of stochastic zeroth-order algorithm can be easily proved, which can be later used to show the convergence of the algorithm for the initial function. And there are setups (for example online learning [36]) where one is limited to use only several (or even one) oracle queries thus being unable to construct the full gradient approximation, so the stochastic approach becomes the only option.

We begin with the formalization of these zeroth-order randomized schemes - we have a problem with the form

$$\min_{x \in Q \subseteq \mathbb{R}^n} f(x)$$

then stochastic zeroth-order methods generate $\{x_k\}$ s.t.

$$x_{k+1} = A\left(\hat{f}, X, P, \{x^i\}_{i=0}^k, \{u^i\}_{i=0}^k\right)$$

so the procedure A gives us x^{k+1} based on function values (obtained via oracle $\hat{f}$), history of $\{x^k\}$, random vectors $\{u^k\}$, and parameters P such as dimension n of X, L_ν and ν – Hölder parameters, etc. Function $\hat{f}$ is not necessarily equal to f, we can, for example, use $\hat{f}(x) = f(x) + \varepsilon(x)$ where $|\varepsilon(x)| \ll |f(x)|$, or $\hat{f}(x,u) = f(x) + \varepsilon(x,u)$ s.t. $\mathbb{E}_u[\hat{f}(x,u)] = f(x)$.

In the subsections, we will discuss the characteristics of several zeroth-order gradient estimations and then the zeroth-order methods for sum-minimization type problems in a non-convex setup. Other information on gradient-free optimization (such as structured objectives) can be found in the recent survey [128].

7.1 Random Directions Gradient Estimations

Let us start with the methods following the standard zeroth-order scheme of using gradient approximation to benefit from the analysis of first-order methods. In this section all methods have a form similar to the classic gradient descent

$$x^{k+1} = x^k - h_k g(x^k, u^k)$$

with only difference that instead of the true gradient we use the gradient approximation $g(x,u)$. One way to build such gradient approximations is to use random directions to compute finite differences in the form

$$g(x^k, u^k) := \frac{\hat{f}(x^k + \mu u^k) - \hat{f}(x^k)}{\mu} \cdot u^k$$

It makes sense to use centrally symmetric distributions for u^k, for example uniformly distributed over the unit Euclidean sphere $S^{n-1} = \{x \in \mathbb{R}^\kappa : \|x\| = 1\}$ (see [79, 73]), or $u^k \sim \mathcal{N}(0, I)$ — so-called Gaussian smoothing introduced in [160]. In this article, the authors proved Gaussian approximation

$$f_\mu(x) = \frac{1}{\kappa} \int\limits_{\mathbb{R}^n} f(x + \mu u) e^{-\frac{1}{2}\|u\|^2} du$$

(there $\kappa = \int\limits_E e^{-\frac{1}{2}\|u\|^2} du = (2\pi)^{n/2}$) to have several good properties, such as convexity preservation (if f is convex then f_μ is convex too), differentiability, and if $f \in C^{0,0}_{L_0}$ or $f \in C^{1,1}_{L_1}$ (i.e. Lipschitz-continuous function with constant L_0 or function with Lipschitz-continuous gradient with L_1 respectively) then the same holds for f_μ with $L_0(f_\mu) \leqslant L_0(f)$ and $L_1(f_\mu) \leqslant L_1(f)$ respectively. It can be also shown that $|f_\mu(x) - f(x)| \leqslant \mu L_0 n^{1/2}$ for the case of $f \in C^{0,0}_{L_0}$.

While in that paper the authors mostly discuss the convex case, there are some results ([160][Section 7]) for a non-convex objective f too. They consider a process $x^{k+1} = x^k - h_k g(x^k, u^k)$, with g defined above, $\hat{f} = f$ and $u^k \sim \mathcal{N}(0, I_n)$, and show

that for the case of $f \in C^{1,1}_{L_1}$ this process converges in the sense of $\mathbb{E}_U \|\nabla f_\mu(x)\|_2$ (where $U = \{u^k\}_{k=0}^{N-1}$):

$$\frac{1}{N}\sum_{k=0}^{N-1} \mathbb{E}_U \left[\|\nabla f_\mu(x^k)\|_2^2\right] \leqslant 8(n+4)L_1 \left[\frac{f_\mu(x^0)-f^*}{N} + \frac{3\mu^2(n+4)}{32}L_1\right]$$

then using the fact that ([160][Lemma 3]) $\|\nabla f_\mu(x) - \nabla f(x)\|_2 \leqslant \frac{\mu L_1}{2}(n+3)^{3/2}$ we obtain (from $\|\nabla f(x)\|_2^2 \leqslant 2\|\nabla f_\mu(x) - \nabla f(x)\|_2^2 + 2\|\nabla f_\mu(x)\|_2^2$)

$$\begin{aligned}\frac{1}{N}\sum_{k=0}^{N-1} \mathbb{E}_U \left[\|\nabla f(x^k)\|_2^2\right] \leqslant & 2\frac{\mu^2 L_1^2}{4}(n+3)^3 \\ & + 16(n+4)L_1 \left[\frac{f_\mu(x^0)-f^*}{N} + \frac{3\mu^2(n+4)}{32}L_1\right]\end{aligned}$$

and choosing $\mu = O\left(\frac{\varepsilon}{n^{3/2}L_1}\right)$ we ensure $\frac{1}{N}\sum\limits_{k=0}^{N-1} \mathbb{E}_U \left[\|\nabla f(x^k)\|_2^2\right] \leqslant \varepsilon^2$ with the upper bound for the expected number of steps $N = O\left(\frac{n}{\varepsilon^2}\right)$.

For the case of $f \in C^{0,0}_{L_0}$

$$\frac{1}{S_N}\sum_{k=0}^{N-1} h_k \mathbb{E}_U \left[\|\nabla f_\mu(x^k)\|_2^2\right] \leqslant \frac{1}{S_N}\left[(f_\mu(x^0)-f^*) + \frac{1}{\mu}n^{1/2}(n+4)^2 L_0^3 \sum_{k=0}^{N-1} h_k^2\right]$$

they show only that this process converges to the stationary point of $f_\mu(x)$ – consider Q with $diam(Q) \leqslant R$, then it can be shown that we need to make

$$N = O\left(\frac{n(n+4)^2 L_0^5 R}{\varepsilon^4 \delta}\right)$$

steps to ensure that $\frac{1}{N}\sum\limits_{k=0}^{N-1} \mathbb{E}_U \left[\|\nabla f_\mu(x^k)\|_2^2\right] \leqslant \varepsilon^2$ keeping functional gap $|f_\mu(x) - f(x)| \leqslant \delta$ small. Authors also mention that with the $h_k \to 0$ and $\mu \to 0$ the convergence in the sense of $\mathbb{E}_U \|\nabla f(x)\|_2$ can be proved too.

These results can be extended [193] to the case of noisy $\hat{f}$ i.e. $|\hat{f} - f(x)| \leqslant \delta$ for f with Hölder continuous gradient ($\|\nabla f(x) - \nabla f(y)\| \leqslant L_\nu \|x-y\|^\nu$) – it can be shown that for a small enough noise δ these convergence rates can be preserved. More specifically, to ensure $\frac{1}{N}\sum\limits_{k=0}^{N-1} \mathbb{E}_U \left[\|\nabla f(x^k)\|_2^2\right] \leqslant \varepsilon^2$ one need to make

$$N = O\left(\frac{n^{2+\frac{1-\nu}{\nu}}}{\varepsilon^{\frac{2}{\nu}}}\right) \text{ steps under the assumption that noise } \delta = O\left(\frac{\varepsilon^{\frac{3+\nu}{2\nu}}}{n^{\frac{3+3\nu}{2\nu}}}\right)$$

where ν is a Hölder parameter. For the convergence in the sense of smoothed function gradient norm $\frac{1}{N}\sum\limits_{k=0}^{N-1}\mathbb{E}_U\left[\|\nabla f_\mu(x^k)\|_2^2\right] \leqslant \varepsilon^2$ it can be shown

$$N = O\left(\frac{n^{\frac{7-3\nu}{2}}}{\varepsilon^{\frac{3-\nu}{1+\nu}}}\right) \text{ with } \delta = O\left(\frac{\varepsilon^{\frac{5-\nu}{1+\nu}}}{n^{\frac{13-3\nu}{4}}}\right)$$

with functional gap $|f_\mu(x) - f(x)| = O\left(\frac{\varepsilon}{n^{\frac{1+\nu}{2}}}\right)$. For the case of $\nu = 1$ (i.e. $f \in C^{1,1}_{L_1}$) these results can be improved to $N = O\left(\frac{n}{\varepsilon^2}\right)$ (n times better) achieving the same rate of convergence as in previous paper [160].

This Gaussian smoothing technique was later used in works [83] (RSGF) and [85] (RSPGF) to obtain complexity guarantees for stochastic zeroth-order optimization. In the first one ([83]), the unconstrained problem $Q = \mathbb{R}^n$ is considered, where $\hat{f} = F(x,\xi)$ s.t. $\mathbb{E}_\xi[F(x,\xi)] = f(x)$ and $F(\cdot,\xi)$ has a Lipschitz-continuous gradient with constant L_1, ξ is a random variable whose distribution P is supported on $\Xi_k \subseteq R^n$. The procedure (7) has a form similar to the one proposed in [160]

$$x^{k+1} = x^k - h_k G(x^k,\xi^k,u^k),\ G(x^k,\xi^k,u^k) := \frac{\hat{f}(x^k+\mu u^k,\xi^k) - \hat{f}(x^k,\xi^k)}{\mu}\cdot u^k,$$

and from $\mathbb{E}_\xi[F(x,\xi)] = f(x)$ it follows that

$$\mathbb{E}_{\xi,u}[G(x,\xi,u)] = \nabla f_\mu(x).$$

The method then chooses the x^k from generated $\{x^k\}_{k=1}^N$ as $k = R$ where R is some random variable with a probability mass function P_R supported on $\{1,\dots,N\}$. The main goal to introduce this random iteration count R is to derive new complexity results for non-convex stochastic optimization case.

For the case of $f \in C^{1,1}_{L_1}$, smoothing parameter μ, $D_f = \left[\frac{2(f(x^1)-f^*)}{L}\right]^{\frac{1}{2}}$, variance σ^2 ($\mathbb{E}_\xi\left[\|\nabla\hat{f}(x,\xi) - \nabla f(x)\|_2^2\right] \leqslant \sigma^2$) and probability mass function $P_R(k) = \frac{h_k - 2L(n+4)h_k^2}{\sum_{i=1}^N (h_i - 2L(n+4)h_i^2)}$ they obtain ([83][Theorem 3.2])

$$\begin{aligned}
&\frac{1}{L}\mathbb{E}\left[\|\nabla f(x^R)\|_2^2\right] \leqslant \\
&\leqslant \frac{D_f^2 + 2\mu^2(n+4)\left(1 + L(n+4)^2\sum\limits_{k=1}^N\left(\frac{h_k}{4} + Lh_k^2\right)\right) + 2(n+4)\sigma^2\sum\limits_{k=1}^N h_k^2}{\sum\limits_{k=1}^N\left[h_k - 2L(n+4)h_k^2\right]}
\end{aligned}$$

where the expectation is taken with respect to R, $\{\xi^k\}$. After choosing specific constant stepsizes $h_k = \frac{1}{\sqrt{n+4}}\min\left\{\frac{1}{4L\sqrt{n+4}}, \frac{\tilde{D}}{\sigma\sqrt{N}}\right\}$ (note that this makes P_R uniform on

$\{1,\dots,N\}$) they get ([83][Corollary 3.3])

$$\frac{1}{L}\mathbb{E}\left[\|\nabla f(x^R)\|_2^2\right] \leqslant \frac{12(n+4)LD_f^2}{N} + \frac{2\sigma\sqrt{n+4}}{\sqrt{N}}\left(\tilde{D} + \frac{D_f^2}{\tilde{D}}\right)$$

where $\tilde{D} > 0$ is our estimation of D_f (for example some upper bound). It can be shown that to ensure $\mathbb{P}\{\|\nabla f(x^R)\|_2^2 \leqslant \varepsilon\} \geqslant 1-\Lambda$ (so-called (ε,Λ)-solution) the total number of calls to the oracle $\hat{f}$ can be bounded as

$$O\left(\frac{nL^2D_f^2}{\Lambda\varepsilon} + \frac{nL^2}{\Lambda^2}\left(\tilde{D} + \frac{D_f^2}{\tilde{D}}\right)^2\frac{\sigma^2}{\varepsilon^2}\right)$$

Another method that is considered in [83] is a two-phase method (2-RSGF), which uses the first one (RSGF) $S = \log(2/\Lambda)$ times as a subroutine producing a list of candidates $\{\bar{x}^k\}_{k=1}^S$ and then the output point $\bar{x}^*$ is chosen in such a way that

$$\|g(\bar{x}^*)\|_2 = \min_{k=1,\dots,S}\|g(\bar{x}^k)\|_2,\ g(\bar{x}^k) := \frac{1}{T}\sum_{i=1}^{T} G(\bar{x}^k,\xi^k,u^k)$$

then it can be shown ([83][Theorem 3.4]) that (ε,Λ)-solution will be achieved after taking

$$O\left(\frac{nL^2D_f^2\log(1/\Lambda)}{\varepsilon} + nL^2\left(\tilde{D} + \frac{D_f^2}{\tilde{D}}\right)^2\frac{\sigma^2}{\varepsilon^2}\log(1/\Lambda) + \frac{n\log^2(1/\Lambda)}{\Lambda}\left(1+\frac{\sigma^2}{\varepsilon}\right)\right)$$

calls to the $\hat{f}$ which is better than the previous one in terms of Λ.

A more general problem $\min_{x\in Q\subseteq\mathbb{R}^n}\Psi(x) = f(x)+h(x)$, where $f\in C_L^{1,1}$ and $h(x)$ is a simple convex and possibly non-smooth function is considered in [85]. They use a mini-batched version of gradient estimation from the previous paper [83] and generalized projection obtaining ([85][Theorem 4, Corollaries 6-7]) similar bounds for the gradient norm.

In [187], the authors use symmetric gradient estimations based on uniform distribution over the sphere to build a less dimension depending method. They consider the minimization problem $\min_{x\in\mathbb{R}^n} f(x) = \mathbb{E}_\xi[F(x,\xi)] = \mathbb{E}_\xi[\hat{f}(x,\xi)]$ (note that in this paper authors consider both $\mathbb{R}^d$ and $\mathbb{R}^n$ with $d \ll n$) where $f(x)$ is L-Lipschitz, and μ-smooth, $|F(x,\xi)| \leqslant \Omega$ and F variance is bounded by V_f. It was shown that using

$$g(x^k,\xi^k,u^k) := n\frac{\hat{f}(x^k+\mu u^k,\xi^k) - \hat{f}(x^k-\mu u^k,\xi^k)}{2\mu}\cdot u^k$$

where $u_k \sim \mathscr{U}\left(S^{n-1}\right)$ (uniform distribution on the unit sphere S^{n-1}) and the process $x^{k+1} = x^k - \alpha g(x^k,\xi^k,u^k)$ after N steps

$$\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\|\nabla f(x^i)\|_2^2\right] = O\left(\frac{n}{N^{1/2}} + \frac{n^{2/3}}{N^{1/3}}\right)$$

Now consider the case when for a given ξ, $F(x,\xi) = g(r(x,\theta^*),\psi^*)$ (there $g(\cdot,\psi)$ and $r(\cdot,\theta)$ are parameterized function classes), where $r(\cdot,\theta^*):\mathbb{R}^n \to \mathbb{R}^d$ where $d \ll n$. To put it simply, the authors consider the case when $F(\cdot,\xi):\mathbb{R}^n \to \mathbb{R}$ while it is actually defined on an d-dimensional manifold $\mathscr{M}$ for all ξ. That means that if one knows the manifold (i.e. θ^*), and g and r are smooth the chain rule can be applied giving $\nabla f(x) = J(x,\theta^*)\nabla_r g(r,\psi)$ (where $J(x,\theta^*) = \partial r(x,\theta^*)/\partial x$) leading to

$$g(x^k,\xi^k,u^k)) := d\frac{\hat{f}(x^k+\mu J_q u^k,\xi^k)) - \hat{f}(x^k-\mu J_q u^k,\xi^k))}{2\mu}\cdot u^k$$

where J_q is the orthonormalized $J(x^k,\theta^{ast})$ and $u_k \sim \mathscr{U}\left(S^{d-1}\right)$, and this gives

$$\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\|\nabla f(x^i)\|_2^2\right] = O\left(\frac{d}{N^{1/2}} + \frac{d^{2/3}}{N^{1/3}}\right)$$

which is much better than the previous one (because $d \ll n$). However, this is impractical due to the fact that it requires the knowledge of θ^*. Authors mix two previous estimations and estimate θ and ψ on every step, obtaining the method that ([187][Theorem 1]) after N steps ensures

$$\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\|\nabla f(x^i)\|_2^2\right] = O\left(\frac{n^{1/2}}{N} + \frac{n^{1/2}+d+dn^{1/2}}{N^{1/2}} + \frac{d^{2/3}+n^{1/2}d^{2/3}}{N^{1/3}}\right)$$

which is better than the initial bound for $d \leqslant n^{1/2}$.

While such gradient estimates based on random directions are common it can be shown that in terms of the number of samples required to the approximate gradient to ensure norm condition (or at least ensure it with some probability) random directions based methods lose to standard finite differences [20, 21, 19]. In these papers, authors consider an unconstrained optimization problem $\min_{x\in\mathbb{R}^n} f(x)$ where $\hat{f}(x) = f(x) + \varepsilon(x)$ is computable, the noise ε is bounded uniformly: $|\varepsilon(x)| \leqslant \varepsilon_f$ and $f(x) \in C_L^{1,1}$ or $f(x) \in C_M^{2,2}$ (i.e. twice continuously differentiable function with M-Lipschitz continuous Hessian) .

The main idea in [20] is to compare the number of calls to the oracle ($\hat{f}(x)$) that will be enough to ensure norm condition

$$\|g(x) - \nabla f(x)\|_2 \leqslant \theta\|\nabla f(x)\|_2,\ \theta \in [0,1)$$

for zeroth-order gradient estimation $g(x)$. This condition simplifies the transition from gradient estimations to gradient when proving the convergence of algorithms. One of its implications is that $g(x)$ is a descent direction for the function ϕ. In [21]

the line-search method that uses such gradient approximations, ensuring the norm condition, is shown to converge.

They consider several methods of gradient estimation, deterministic (Forward and Central Finite Differences (FFD and CFD) and Linear Interpolation (LI) as generalization) and stochastic (Gaussian Smoothed Gradients (GSG and its centered version $cGSG$) and Sphere Smoothed Gradients (BSG and $cBSG$)), for the latter authors obtain the number of calls needed to ensure the norm condition with probability $1-\delta$.

Name	Gradient estimation $g(x)$ form	Number of calls N	$\|\nabla f(x)\|_2$
FFD	$\sum_{i=1}^{n} \frac{\hat{f}(x+\mu e_i)-\hat{f}(x)}{\mu} e_i$	n	$\frac{2\sqrt{nL\varepsilon_f}}{\theta}$
CFD	$\sum_{i=1}^{n} \frac{\hat{f}(x+\mu e_i)-\hat{f}(x-\mu e_i)}{2\mu} e_i$	n	$\frac{2\sqrt{n}\sqrt[3]{M\varepsilon_f^2}}{\sqrt[3]{6}\theta}$
LI	$\sum_{i=1}^{n} \frac{\hat{f}(x+\mu u^i)-\hat{f}(x)}{\mu} u^i$, $u_i = [Q]_i$,	n	$\frac{2\|Q^{-1}\|\sqrt{nL\varepsilon_f}}{\theta}$
GSG	$\frac{1}{N}\sum_{i=1}^{N} \frac{\hat{f}(x+\mu u^i)-\hat{f}(x)}{\mu} u^i$, $u^i \sim \mathcal{N}(0, I_n)$	$\frac{12n}{\delta\theta^2} + \frac{n+20}{16\delta}$	$\frac{6n\sqrt{L\varepsilon_f}}{\theta}$
$cGSG$	$\frac{1}{N}\sum_{i=1}^{N} \frac{\hat{f}(x+\mu u^i)-\hat{f}(x-\mu u^i)}{2\mu} u^i$, $u^i \sim \mathcal{N}(0, I_n)$	$\frac{12n}{\delta\theta^2} + \frac{n+30}{144\delta}$	$\frac{12\sqrt[3]{n^{7/2}M\varepsilon_f^2}}{\theta}$
BSG	$\frac{n}{N}\sum_{i=1}^{N} \frac{\hat{f}(x+\mu u^i)-\hat{f}(x)}{\mu} u^i$, $u^i \sim \mathscr{U}\left(S^{n-1}\right)$	$\left[\frac{8n}{\theta^2} + \frac{8n}{3\theta} + \frac{11n+104}{24}\right] \log\frac{n+1}{\delta}$	$\frac{4n\sqrt{L\varepsilon_f}}{\theta}$
$cBSG$	$\frac{n}{N}\sum_{i=1}^{N} \frac{\hat{f}(x+\mu u^i)-\hat{f}(x-\mu u^i)}{2\mu} u^i$, $u^i \sim \mathscr{U}\left(S^{n-1}\right)$	$\left[\frac{8n}{\theta^2} + \frac{8n}{3\theta} + \frac{9n+192}{27}\right] \log\frac{n+1}{\delta}$	$\frac{4\sqrt[3]{n^{7/2}M\varepsilon_f^2}}{\theta}$

Table 4 Bounds on number of $\hat{f}$ calls N, and $\|\nabla f(x)\|_2$ that ensure the norm condition $\|g(x) - \nabla f(x)\|_2 \leqslant \theta\|\nabla f(x)\|_2$. For the GSG, $cGSG$, BSG and $cBSG$ these are the results with probability $1-\delta$. The gradient norm bound (last column) essentially means that for a noisy oracle $\hat{f}$ we can ensure norm condition only for big enough gradients. The LI method is basically FFD with directions given as columns of the nonsingular matrix Q. When Q is orthonormal the $g(x)$ takes a form from the table.

Let us take a look at two of these methods: FFD and GSG. For the first one, the gradient estimation takes the form

$$g(x) := \sum_{i=1}^{n} \frac{\hat{f}(x+\mu e_i) - \hat{f}(x)}{\mu} e_i$$

where e_i are the columns of I_n. It can be shown that for such $g(x)$ the following holds

$$\|g(x) - \nabla f(x)\|_2 \leqslant \frac{\sqrt{n}L\mu}{2} + \frac{2\sqrt{n}\varepsilon_f}{\mu}.$$

If there was no noise ($\varepsilon_f = 0$) we could make this approximation as close to the gradient as we want, so we would be able to ensure the norm condition in n calls to the $\hat{f}$. This is also true for a small enough noise (for example even from this inequality we can take $\varepsilon_f = L\mu^2/4$ obtaining $\|g(x) - \nabla f(x)\|_2 \leqslant \sqrt{n}L\mu$). Authors

provide such noise bound in form of lower bound on $\|\nabla f(x)\|_2$ for which the norm condition can still be ensured

$$2\sqrt{\frac{\varepsilon_f}{L}} \leqslant \mu \leqslant \frac{\theta\|\nabla f(x)\|_2}{\sqrt{n}L} \Rightarrow \frac{2\sqrt{nL\varepsilon_f}}{\theta} \leqslant \|\nabla f(x)\|_2$$

In other words, that means that we can converge to the neighborhood where $\|\nabla f(x)\|_2 \approx \frac{2\sqrt{nL\varepsilon_f}}{\theta}$.

For the *GSG* they consider the mini-batched version of Gaussian smoothing from [160]

$$g(x,\{u^i\}) := \frac{1}{N}\sum_{i=1}^{N}\frac{\hat{f}(x+\mu u^i)-\hat{f}(x)}{\mu}u^i,\ u^i \sim \mathcal{N}(0,I_n)$$

and prove that the norm condition will be ensured with probability $1-\delta$ after

$$N \geqslant \frac{3n}{\delta\theta^2}\frac{n}{(\sqrt{n}-1)^2} + \frac{(n+4)}{16\delta} + \frac{1}{\delta} = \Omega\left(\frac{3n}{\theta^2\delta}\right)$$

calls, which is while linear on n is still worse than the plain n in FFD, because of δ, and additional constants. However, this is a sufficient number of calls, not a necessary, so authors derive the lower bound for N ([20][Section 2.3.1])

$$N \geqslant \frac{1-\sqrt{\delta}}{\theta^2}(n+1)$$

necessary to have probability $\mathbb{P}(\|g(x)-\nabla f(x)\| \leqslant \theta\|\nabla f(x)\|) > 1-\delta$. In their numerical experiments they show that to ensure the norm condition with $\theta < \frac{1}{2}$ with probability of at least $\frac{1}{2}$ more than n oracle calls are needed, so this lower bound is weak.

The sufficient lower bound can be improved using smoothing on a sphere for which they obtain $\Omega\left(\frac{n}{\theta^2}\log\frac{n+1}{\delta}\right)$, yet it is still worse than deterministic variants, and in practice its behavior is very similar to the Gaussian directions based approach.

There are also results for the case of $f(x) \in C_M^{2,2}$ (centered versions of the estimations), they can be found in Table 4.

7.2 Variance-Reduced Zeroth-Order Methods

One special case of the $\min f(x)$ problem is the finite sum minimization which was considered in previous sections for the first-order methods. These problems in zeroth-order setup arise in reinforcement learning [77] (there as a minimization of a long-term cost which is essentially a sum of functions) and non-stationary online optimization problems [241].

Let us start with the ZO-SVRG from [137] – a zeroth-order version of SVRG from [115].

There a non-convex finite-sum problem of the form

$$\min_{x\in\mathbb{R}^n} f(x) = \frac{1}{m}\sum_{i=1}^{m} f_i(x)$$

where $f_i \in C_L^{1,1}$ i.e. $\|\nabla f_i(x) - \nabla f_i(y)\|_2 \leqslant L\|x-y\|_2$ for any $x,y \in \mathbb{R}^n$ and $i \in \{1,\dots,m\}$ is considered. Authors use the standard assumption that the variance of stochastic gradients is bounded

$$\frac{1}{m}\sum_{i=1}^{m} \|\nabla f_i(x) - \nabla f_i(y)\|_2^2 \leqslant \sigma^2$$

and consider several different gradient estimates: two based on random directions on a unit sphere (in notation of [21] these are *BSG* with $N = 1$ and $N = q$ (see Table 4), called RandGradEst and Avg-RandGradEst respectively), and one deterministic coordinate estimation (variant of *CFD* from Table 4 with possibly different μ_j for each direction e_j called CoordGradEst)

$$\text{RandGradEst}: \hat{\nabla} f_i(x) = \frac{n}{\mu}[f_i(x+\mu u^i) - f_i(x)]u^i,$$

$$\text{Avg-RandGradEst}: \hat{\nabla} f_i(x) = \frac{n}{\mu q}\sum_{j=1}^{q}[f_i(x+\mu u^{i,j}) - f_i(x)]u^{i,j},$$

$$\text{CoordGradEst}: \hat{\nabla} f_i(x) = \frac{1}{2\mu}\sum_{j=1}^{n}[f_i(x+\mu_j e_j) - f_i(x-\mu_j e_j)]e_j$$

there $i \in \{1,\dots,m\}$, $\mu > 0$, and $\{e_j\}_{j=1}^n$ are standard basis vectors (columns of I_n).

Algorithm 12 ZO-SVRG [137]

Require: stepsizes $\{h_s^k\}$, epoch length T, starting point $x^0 \in \mathbb{R}^n$, batch size $r \geq 1$, smoothing parameter $\mu > 0$, number of iterations $N = S \cdot T$
  $\phi_0 = x_0^0 = x^0$
  for $s = 0,1,2,\dots,S-1$ **do**
    for $k = 0,1,2,\dots,T-1$ **do**
      Uniformly randomly pick set I_k from $\{1,\dots,m\}$ such that $|I_k| = r$
      $g^k = \frac{1}{r}\sum_{i\in I_k}\left(\hat{\nabla} f_i(x_s^k) - \hat{\nabla} f_i(\phi_s)\right) + \hat{\nabla} f(\phi_s)$
      $x_s^{k+1} = x_s^k - h_s^k g^k$
    end for
    $\phi_{s+1} = x_{s+1}^0 = x_s^k$
  end for
  Pick ξ uniformly at random from $\{0,\dots,N-1\}$
  return x^ξ

For a mini-batch $I \subseteq \{1,\dots,m\}$ of size r, authors denote

$$\hat{\nabla} f_I(x) = \frac{1}{r}\sum_{i\in I}\hat{\nabla} f_i(x)$$

and the algorithm is the same as for SVRG (Algorithm 6), with the only difference that instead of true gradients update

$$x_s^{k+1} = x_s^k - h_s^k v_s^k,\ v_s^k = \nabla f_{I_k}(x_s^k) - \nabla f_{I_k}(x_s^0) + \nabla f(x_s^0)$$

they use gradient estimations

$$x_s^{k+1} = x_s^k - h_s^k \hat{v}_s^k,\ \hat{v}_s^k = \hat{\nabla} f_{I_k}(x_s^k) - \hat{\nabla} f_{I_k}(x_s^0) + \hat{\nabla} f(x_s^0)$$

This estimation $\hat{\nabla} f(x_s^0)$ is no longer unbiased for zeroth-order gradient estimations, and that is the main problem for the convergence analysis of this method. They show that under assumptions mentioned above ZO-SVRG algorithm after $N = S\cdot T$ (there S is a number of epochs) steps ensures that

$$\text{RandGradEst}: \mathbb{E}\left[\|\nabla f(\bar{x})\|_2^2\right] = O\left(\frac{n}{N} + \frac{\delta_n}{r}\right)$$
$$\text{Avg-RandGradEst}: \mathbb{E}\left[\|\nabla f(\bar{x})\|_2^2\right] = O\left(\frac{n}{N} + \frac{\delta_n}{r\min\{n,q\}}\right)$$
$$\text{CoordGradEst}: \mathbb{E}\left[\|\nabla f(\bar{x})\|_2^2\right] = O\left(\frac{n}{N}\right)$$

there n is a dimension, $r = |I|$ – batch size, q is the number of directions used to estimate gradient via Avg-RandGradEst, $\bar{x}$ is uniformly chosen from $\{x_s^k\}_{s,k=0}^{S-1,T-1}$, $N = S\cdot T$ is a total number of steps and

$$\delta_n = \begin{cases} 1, & \text{if } I_k \text{ draws samples from } \{1,\dots,\mathrm{m}\} \text{ with replacement} \\ I(b<n), & \dots \text{ without replacement} \end{cases}$$

where $I(b<n) = 1$ if $b<n$ and $I(b<n) = 0$ otherwise.

Basically, that means that CoordGradEst, the deterministic policy of gradient estimations, achieves the convergence rates of the original SVRG. In their tests, however, in terms of training loss versus function queries ZO-SVRG (the variant without mini-batching and with random directions on the sphere) beats ZO-SVRG-Ave (based on Avg-RandGradEst) and ZO-SVRG-Coord (based on CoordGradEst).

Another discussed above algorithm that can be used in the zeroth-order finite-sum minimization setting is SPIDER [74]. The zeroth-order variant (Algorithm 13) of the algorithm blends stochastic and deterministic gradient estimations, using mini-batched FFD (Table 4) every p steps to reconstruct v^k, which is later updated by mini-batched *GSG*.

Algorithm 13 SpiderSZO [74]

Require: $n_0 \in [1, n^{1/2}/6]$, Lipschitz constant L, epoch length T, starting point $x^0 \in \mathbb{R}^n$, outer batch size $r_1 \geq 1$, inner batch size $r_2 \geq 1$, number of iterations $N = S \cdot T$
for $k = 0, 1, 2, \ldots, N-1$ **do**
  if $k \mod T = 0$ **then**
    Uniformly randomly pick set I_k from $\{1, \ldots, m\}$ (with replacement) such that $|I_k| = r_1$
    Compute $g^k = \sum\limits_{j=1}^{n} \left(\frac{1}{r_1} \sum\limits_{i \in I_k} \frac{[f_i(x^k + \mu e_j) - f_i(x^k)]}{\mu} \right) e_j$
  else
    Create set of pairs $I_k = \{(i, u^i)\}$ where i uniformly randomly picked from $\{1, \ldots, m\}$ (with replacement) and independent $u_i \sim \mathcal{N}(0, I_n)$ such that $|I_k| = r_2$
    Compute $g^k = \frac{1}{r_2} \sum\limits_{(i,u^i) \in I_k} \left(\frac{f_i(x^k + \mu u^i) - f_i(x^k)}{\mu} u^i - \frac{f_i(x^{k-1} + \mu u^i) - f_i(x^{k-1})}{\mu} u^i \right) + g^{k-1}$
  end if
  $x^{k+1} = x^k - h_k g^k$ where $h_k = \min\left(\frac{\varepsilon}{L n_0 \|v^k\|_2}, \frac{1}{2Ln_0} \right)$
end for
Pick ξ uniformly at random from $\{0, \ldots, N-1\}$
return x^ξ

The $h_k = \min\left(\frac{\varepsilon}{L n_0 \|v^k\|_2}, \frac{1}{2Ln_0} \right)$ is a stepsize policy from Normalized Gradient Descent (NGD, [155]), where the stepsize is inverse-proportional to the norm of the gradient.

Authors show, that after $N = O\left(\frac{1}{\varepsilon^2}\right)$ iterations and $O\left(n \min\left(\frac{m^{1/2}}{\varepsilon^2}, \frac{1}{\varepsilon^3}\right)\right)$ (there n is a dimension and m is a number of functions) IZO calls (i.e. calls of the oracle that returns the value of $f_i(x)$ given x and i) this algorithm ensures

$$\mathbb{E}[\|\nabla f(\bar{x})\|_2] \leqslant 6\varepsilon$$

where $\bar{x}$ is uniformly chosen from $\{x^k\}_{k=0}^{N-1}$. This result is better than what follows directly from [160], at least by the factor of $m^{1/2}$ (the direct application of the results from [160] requires m calls on every step, and gives $\mathbb{E}[\|\nabla f(\bar{x})\|_2] \leqslant \varepsilon$ in $O\left(\frac{n}{\varepsilon^2}\right)$ steps so the number of IZO calls would be $O\left(\frac{nm}{\varepsilon^2}\right)$).

The results of two previously discussed papers [137, 74] were improved in the recent work [110]. Authors show that ZO-SVRG-Coord actually has a better convergence rate ([110][Theorem 2]) of $\mathbb{E}\left[\|\nabla f(\bar{x})\|_2^2\right] = O\left(\frac{1}{N}\right)$ (n times better than the previous analysis). At first they consider an intermediate variant of ZO-SVRG-Coord and ZO-SVRG-Ave called ZO-SVRG-Coord-Rand, that uses *CFD* and *BSG* (Table 4) for $\hat{\nabla} f(\phi_s)$ and $\hat{\nabla} f_i(x_s^k) - \hat{\nabla} f_i(\phi_s)$ parts of

$$g^k = \frac{1}{r} \sum_{i \in I_k} \left(\hat{\nabla} f_i(x_s^k) - \hat{\nabla} f_i(\phi_s) \right) + \hat{\nabla} f(\phi_s)$$

(from Algorithm 12) respectively, while variants in [137] used only one type of gradient estimation at once. Then authors proof ([110][Corollary 1]) the convergence

rate $\mathbb{E}\left[\|\nabla f(\bar{x})\|_2^2\right] = O\left(\frac{1}{N}\right)$ and show ([110][Lemmas 1-2]) that although the replacement of *BSG* with *CFD* requires n more oracle calls it achieves more accurate gradient estimation so the convergence rate stays the same for the ZO-SVRG-Coord.

Another part of this work is devoted to SPIDER. Authors construct a new algorithm (called ZO-SPIDER-Coord) in a way similar to the previous one – they use *CFD* instead of *GSG* in Algorithm 13 and show that it has the same rate of convergence, but with bigger stepsize $h_k = \frac{1}{4L}$ (that doesn't depend on ε), which is better in practice.

One particular case of finite-sum minimization is considered in [241]. In this paper, authors consider non-stationary online optimization problems, when the objective function being queried is time-varying, so one is limited to the use of one-point estimators.

Such estimators can be constructed easily in the stochastic zeroth-order case. For example we can consider *GSG* (Table 4) with $N = 1$ then

$$\mathbb{E}_u(g(x)) = \mathbb{E}_u\left[\frac{f(x+\mu u) - f(x)}{\mu}u\right] = \mathbb{E}_u\left[\frac{f(x+\mu u)}{\mu}u\right] = \nabla f_\mu(x)$$

so we can chose $g(x) = \frac{f(x+\mu u)}{\mu}u$ and obtain a reasonable one-point estimation. The problem is that the variance of such estimations explodes as $\mu \to 0$ (see [21]).

In this work, authors consider the residual feedback estimator

$$\tilde{g}_k(x^k) = \frac{u^k}{\mu}\left(f_k(x^k + \mu u^k) - f_k(x^{k-1} + \mu u^{k-1})\right)$$

where $u^k, u^{k-1} \sim \mathcal{N}(0, I_n)$. They show that (Lemma 2.4)

$$\mathbb{E}[\tilde{g}_k(x^k)] = \nabla f_{\mu,k}(x^k),\ \forall x^k \in X \text{ and } k$$

(there $\nabla f_{\mu,k}$ is a gradient of smoothed f_k). They consider the online bandit problem with regret function

$$R_{g,\mu}^T = \sum_{k=0}^{T-1} \mathbb{E}\left[\|\nabla f_{\mu,k}(x^k)\|_2^2\right]$$

and show ([241][Theorem 4.2]) that for $x^{k+1} = \Pi_X\left(x^k - \eta \tilde{g}_k(x^k)\right)$ (where Π_X is the projection operator onto set X) if $f \in C_{L_0}^{0,0}$

$$R_{g,\mu}^T = O\left(\frac{n^{3/2}L_0^2}{\varepsilon_f^{3/2}}\left(W_T + \tilde{W}_T T^{-1}\right)T^{1/2} + n^{3/2}L_0\varepsilon_f^{1/2}T^{1/2}\right)$$

and if additionally $f \in C_{L_1}^{1,1}$ ([241][Theorem 4.3])

$$R_g^T = \sum_{k=0}^{T-1} \mathbb{E}\left[\|\nabla f_k(x^k)\|_2^2\right] = O\left(n^{4/3} L_0 W_T T^{1/2} + n^{4/3} L_1 L_0^{-1} \tilde{W}_T\right)$$

where W_T and $\tilde{W}_T$ are constants s.t.

$$\sum_{k=1}^{T} \mathbb{E}\left[f_k(x) - f_{k-1}(x)\right] \leqslant W_T, \ \forall T, x$$

$$\sum_{k=1}^{T} \mathbb{E}\left[|f_k(x) - f_{k-1}(x)|^2\right] \leqslant \tilde{W}_T, \ \forall T, x.$$

That bound implies that $R_g^T/T \to 0$ if $W_T = o\left(T^{1/2}\right)$ and $\tilde{W}_T = o(T)$. Authors also consider ([241][Section 5]) the stochastic online optimization case where $\hat{f}_t = F_t(x, \xi_t)$ s.t. $\mathbb{E}\left[F_t(x, \xi_t)\right] = f_t(x)$ and show that under the assumptions of the same form as above (with $W_{T,\xi}$ and $\tilde{W}_{T,\xi}$) similar regret bounds can be achieved.

In their numerical experiments, authors compare conventional one-point and two-point approaches with one-point residual feedback. Even though the latter works worse than the two-point variant, it has lower variance and achieves better results than the conventional one-point feedback, and can be used in practice, in contrast to two-point feedback.

8 Globalization Techniques

In the previous sections we mainly considered guarantees for the methods to converge to a stationary point or local extremum. Global performance guarantees are available only for some subclasses of non-convex minimization problems. Despite that there are several practical techniques for convergence globalization for the local methods, which we briefly describe next, following [242].

8.1 Multistart Technique

The first approach involves using an algorithm which converges to a local minimum and running it multiple times from different starting points. This may result in the algorithm for finding multiple local minima of the objective, some of which might in fact be global solutions.

To be more concrete, we consider the problem

$$\min_{x \in [0,1]^n} f(x).$$

Let the initial points be sampled from the uniform distribution on $[0,1]^n$. If the Lebesgue measure of the attraction basin (the set of points, initialized at which

the local algorithm converges to the global minimum) of the global minimum is $\mu > 0$, then the expected number of points required to find the global minimum is $m = \tilde{\mathrm{O}}(1/\mu)$. If the attraction basin is a ball of radius r, then $\mu \sim r^n$. Hence, it is reasonable to expect that the number of initial points required depends on n exponentially. For that reason, this approach to global optimization becomes impractical as n grows.

The effectiveness of this approach also depends on the chosen initial points. The quality of a family of initial points $\left\{x^{0,i}\right\}_{i=1}^{m}$ can be characterized by the quantity

$$d_n\left(\left\{x^{0,i}\right\}_{i=1}^{m}\right) = \max_{x\in[0,1]^n} \min_{i=1,\ldots,m} \left\|x - x^{0,i}\right\|_2.$$

One of the ways to iteratively generate the starting points $\left\{x^{0,k}\right\}_{k=1}^{m}$ is called the quasi Monte Carlo scheme using low-discrepancy sequences, for example, the Van der Corput sequence. Let $\{p_i\}_{i=1}^{n}$ be a sequence of distinct prime numbers, and let $\phi_i(k)$ be the k-th element of the Van der Corput sequence in base p_i. Explicitly, $\phi_i(k) = \sum\limits_{j=0}^{l_{k,i}} a_j p_i^{-j-1}$, where $l_{k,i}$ is the length of the representation of k in base p_i $k = \sum\limits_{j=0}^{l_{k,i}} a_j p_i^j$. Finally, set $x^{0,k} = (\phi_1(k), ..1., \phi_n(k))$, $k = 1, ..., m$. In this case $d_n\left(\left\{x^{0,i}\right\}_{i=1}^{m}\right) = \mathrm{O}\left(\sqrt{n} m^{-1/n} \ln m\right)$, while the optimal value, which is achieved at the uniform grid, is $\mathrm{O}\left(\sqrt{n} m^{-1/(2n)}\right)$.

8.2 Multidimensional Bisection

The main shortcoming of the approach described above is that the family $\left\{x^{0,k}\right\}_{k=1}^{m}$ is constructed without taking into account any properties of $f(x)$. Assume now that, for all $x, y \in [0,1]^n$, $|f(y) - f(x)| \leqslant M\|y - x\|$. Then, for any y, the function $f(y) - M\|x - y\|$ is a minorant of $f(x)$. Consequently, for any $\{y^k\}_{k=1}^{m}$ the function $\max\limits_{k=1,\ldots,m} f(y^k) - M\|x - y^k\|$ is also a minorant of $f(x)$. Then one may choose the next initial point to be the minimizer of the minorant constructed using the previous initial points:

$$x^{0,m+1} = \arg\min_x \max_{k=1,\ldots,m} \left\{f\left(x^{0,k}\right) - M\left\|x - x^{0,k}\right\|\right\}.$$

In the one-dimensional case, each minorant is just a piecewise linear function, and its minimum is easy to compute explicitly. In higher-dimensions, this idea is more difficult to implement, and the resulting algorithms also tend to become slower as n increases. This method also requires an estimate of the Lipschitz constant and is sensitive to the accuracy of this estimate.

8.3 Langevin Dynamics

The last but not least approach which we consider in this section is inspired by the Langevin dynamics, which is defined by the stochastic differential equation

$$dx(t) = -\nabla f(x(t))dt + \sqrt{2T}dW(t),$$

where $W(t)$ is a Wiener process (also known as Brownian motion) and T is the temperature parameter. It has been shown that the distribution of $x(t)$ converges to a distribution with density

$$\frac{\exp\left(-f(x)/T\right)}{\int \exp\left(-f(y)/T\right)dy}$$

as $t \to \infty$, and as $T \to 0+$ this distribution concentrates around the global minima. To apply this in practice, the continuous dynamics has to be discretized. One of the ways to do that is as follows:

$$x_{k+1} = x_k - h\nabla f(x_k) + \sqrt{2hT}\varepsilon_k,$$

where $h > 0$ is the step size and ε_k is standard gaussian random variable. Non-asymptotic results demonstrating the convergence of this method to an approximate global minimum were presented in the work [223]. In this paper, the temperature parameter T was assumed to be constant. However, other strategies are sometimes used in practice, for example,

$$T_k = \frac{c}{\ln(2+k)},$$

which ensures $T_k \to 0+$ as $k \to \infty$.

Acknowledgements

The authors are grateful to A. Gornov, A. Nazin, Yu. Nesterov, B. Polyak and K. Scheinberg for fruitful discussions and their suggestions which helped to improve the quality of the text.

The research was partially supported by by the Ministry of Science and Higher Education of the Russian Federation (Goszadaniye) no 075-00337-20-03.

References

1. Collection of optimizers for pytorch. `https://github.com/jettify/pytorch-optimizer`.

2. N. Agarwal, Z. Allen-Zhu, B. Bullins, E. Hazan, and T. Ma. Finding approximate local minima faster than gradient descent. In Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, pages 1195–1199, 2017.
3. K. Ahn, C. Yun, and S. Sra. Sgd with shuffling: optimal rates without component convexity and large epoch requirements. Advances in Neural Information Processing Systems, 33, 2020.
4. D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In Advances in Neural Information Processing Systems, pages 1709–1720, 2017.
5. Z. Allen-Zhu. Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. In International Conference on Machine Learning, pages 89–97, 2017.
6. Z. Allen-Zhu. How to make the gradients small stochastically: Even faster convex and non-convex sgd. In Advances in Neural Information Processing Systems, pages 1157–1167, 2018.
7. Z. Allen-Zhu. Katyusha x: Simple momentum method for stochastic sum-of-nonconvex optimization. In International Conference on Machine Learning, pages 179–185, 2018.
8. Z. Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. In Advances in Neural Information Processing Systems, pages 2675–2686, 2018.
9. Z. Allen-Zhu and Y. Li. Neon2: Finding local minima via first-order oracles. In Advances in Neural Information Processing Systems, pages 3716–3726, 2018.
10. Z. Allen-Zhu and Y. Li. Can sgd learn recurrent neural networks with provable generalization? In Advances in Neural Information Processing Systems, pages 10331–10341, 2019.
11. Z. Allen-Zhu, Y. Li, and Y. Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In Advances in neural information processing systems, pages 6158–6169, 2019.
12. Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. In International Conference on Machine Learning, pages 242–252. PMLR, 2019.
13. Z. Allen-Zhu, Y. Li, and Z. Song. On the convergence rate of training recurrent neural networks. In Advances in neural information processing systems, pages 6676–6688, 2019.
14. Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, A. Sekhari, and K. Sridharan. Second-order information in non-convex stochastic optimization: Power and limitations. In Conference on Learning Theory, pages 242–299, 2020.
15. Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, N. Srebro, and B. Woodworth. Lower bounds for non-convex stochastic optimization. arXiv preprint arXiv:1912.02365, 2019.
16. S. Arora, N. Cohen, N. Golowich, and W. Hu. A convergence analysis of gradient descent for deep linear neural networks. arXiv preprint arXiv:1810.02281, 2018.
17. F. Bach, R. Jenatton, J. Mairal, G. Obozinski, et al. Optimization with sparsity-inducing penalties. Foundations and Trends® in Machine Learning, 4(1):1–106, 2012.
18. R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. Constructive Approximation, 28(3):253–263, 2008.
19. A. S. Berahas, L. Cao, K. Choromanski, and K. Scheinberg. Linear interpolation gives better gradients than gaussian smoothing in derivative-free optimization, 2019.
20. A. S. Berahas, L. Cao, K. Choromanski, and K. Scheinberg. A theoretical and empirical comparison of gradient approximations in derivative-free optimization, 2020.
21. A. S. Berahas, L. Cao, and K. Scheinberg. Global convergence rate analysis of a generic line search algorithm with noise, 2019.
22. E. H. Bergou, E. Gorbunov, and P. Richtárik. Stochastic three points method for unconstrained smooth minimization, 2019.
23. A. Beznosikov, S. Horváth, P. Richtárik, and M. Safaryan. On biased compression for distributed learning. arXiv preprint arXiv:2002.12410, 2020.
24. S. Bhojanapalli, A. Kyrillidis, and S. Sanghavi. Dropping convexity for faster semi-definite optimization. In Conference on Learning Theory, pages 530–582, 2016.

25. E. G. Birgin, J. Gardenghi, J. M. Martínez, S. A. Santos, and P. L. Toint. Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models. Mathematical Programming, 163(1-2):359–368, 2017.
26. A. Blum, J. Hopcroft, and R. Kannan. Foundations of data science. Cambridge University Press, 2016.
27. A. Blum and R. L. Rivest. Training a 3-node neural network is np-complete. In Advances in neural information processing systems, pages 494–501, 1989.
28. T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. Applied and computational harmonic analysis, 27(3):265–274, 2009.
29. L. Bogolubsky, P. Dvurechensky, A. Gasnikov, G. Gusev, Y. Nesterov, A. M. Raigorodskii, A. Tikhonov, and M. Zhukovskii. Learning supervised pagerank with gradient-based and gradient-free optimization methods. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems 29, pages 4914–4922. Curran Associates, Inc., 2016. arXiv:1603.00717.
30. L. Bottou. Curiously fast convergence of some stochastic gradient descent algorithms. In Proceedings of the symposium on learning and data science, Paris, 2009.
31. L. Bottou. Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT'2010, pages 177–186. Springer, 2010.
32. L. Bottou. Stochastic gradient descent tricks. In Neural networks: Tricks of the trade, pages 421–436. Springer, 2012.
33. L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. Siam Review, 60(2):223–311, 2018.
34. S. Boyd and L. Vandenberghe. Convex Optimization. NY Cambridge University Press, 2004.
35. J. Bu and M. Mesbahi. A note on Nesterov's accelerated method in nonconvex optimization: a weak estimate sequence approach. arXiv preprint arXiv:2006.08548, 2020.
36. S. Bubeck. Introduction to online optimization. 2011.
37. E. J. Candes, X. Li, and M. Soltanolkotabi. Phase retrieval via wirtinger flow: Theory and algorithms. IEEE Transactions on Information Theory, 61(4):1985–2007, 2015.
38. E. J. Candès and B. Recht. Exact matrix completion via convex optimization. Foundations of Computational mathematics, 9(6):717, 2009.
39. E. J. Candes and T. Tao. Decoding by linear programming. IEEE transactions on information theory, 51(12):4203–4215, 2005.
40. E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. IEEE Transactions on Information Theory, 56(5):2053–2080, 2010.
41. E. J. Candes, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. Journal of Fourier analysis and applications, 14(5-6):877–905, 2008.
42. Y. Carmon and J. C. Duchi. Gradient descent efficiently finds the cubic-regularized nonconvex newton step. arXiv preprint arXiv:1612.00547, 2016.
43. Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. "Convex until proven guilty": Dimension-free acceleration of gradient descent on non-convex functions. volume 70 of Proceedings of Machine Learning Research, pages 654–663, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.
44. Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Accelerated methods for nonconvex optimization. SIAM Journal on Optimization, 28(2):1751–1772, 2018.
45. Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points II: first-order methods. Mathematical Programming, Sep 2019.
46. Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points i. Mathematical Programming, 184(1):71–120, Nov 2020.
47. V. Charisopoulos, A. R. Benson, and A. Damle. Entrywise convergence of iterative methods for eigenproblems. arXiv preprint arXiv:2002.08491, 2020.
48. X. Chen, S. Liu, R. Sun, and M. Hong. On the convergence of a class of adam-type algorithms for non-convex optimization. arXiv preprint arXiv:1808.02941, 2018.
49. Y. Chen and Y. Chi. Harnessing structures in big data via guaranteed low-rank matrix estimation. arXiv preprint arXiv:1802.08397, 2018.

50. Y. Chen, Y. Chi, J. Fan, and C. Ma. Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval. Mathematical Programming, 176(1-2):5–37, 2019.
51. Z. Chen and T. Yang. A variance reduction method for non-convex optimization with improved convergence under large condition number. arXiv preprint arXiv:1809.06754, 2018.
52. Z. Chen and Y. Zhou. Momentum with variance reduction for nonconvex composition optimization. arXiv preprint arXiv:2005.07755, 2020.
53. Y. Chi, Y. M. Lu, and Y. Chen. Nonconvex optimization meets low-rank matrix factorization: An overview. arXiv preprint arXiv:1809.09573, 2018.
54. P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In Fixed-point algorithms for inverse problems in science and engineering, pages 185–212. Springer, 2011.
55. A. Conn, K. Scheinberg, and L. Vicente. Introduction to Derivative-Free Optimization. Society for Industrial and Applied Mathematics, 2009.
56. F. E. Curtis and K. Scheinberg. Optimization methods for supervised machine learning: From linear models to deep learning. arXiv preprint arXiv:1706.10207, 2017.
57. A. Cutkosky and F. Orabona. Momentum-based variance reduction in non-convex sgd. In Advances in Neural Information Processing Systems, pages 15236–15245, 2019.
58. C. D. Dang and G. Lan. Stochastic block mirror descent methods for nonsmooth and stochastic optimization. SIAM J. on Optimization, 25(2):856–881, Apr. 2015.
59. D. Davis and D. Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. SIAM Journal on Optimization, 29(1):207–239, 2019.
60. A. Defazio. Understanding the role of momentum in non-convex optimization: Practical insights from a lyapunov analysis. arXiv preprint arXiv:2010.00406, 2020.
61. A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In Proceedings of the 27th International Conference on Neural Information Processing Systems, NIPS'14, pages 1646–1654, Cambridge, MA, USA, 2014. MIT Press.
62. A. Defazio and L. Bottou. On the ineffectiveness of variance reduced optimization for deep learning. In Advances in Neural Information Processing Systems, pages 1753–1763, 2019.
63. A. Defazio, J. Domke, et al. Finito: A faster, permutable incremental gradient method for big data problems. In International Conference on Machine Learning, pages 1125–1133, 2014.
64. A. Défossez, L. Bottou, F. Bach, and N. Usunier. On the convergence of adam and adagrad. arXiv preprint arXiv:2003.02395, 2020.
65. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.
66. J. Diakonikolas and M. I. Jordan. Generalized momentum-based methods: A Hamiltonian perspective. arXiv preprint arXiv:1906.00436, 2019.
67. T. Ding, D. Li, and R. Sun. Spurious local minima exist for almost all over-parameterized neural networks. 2019.
68. J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12(Jul.):2121–2159, 2011.
69. J. Duchi, M. I. Jordan, and B. McMahan. Estimation, optimization, and parallelism when data is sparse. In Advances in Neural Information Processing Systems, pages 2832–2840, 2013.
70. D. Dvinskikh, A. Ogaltsov, A. Gasnikov, P. Dvurechensky, and V. Spokoiny. Adaptive gradient descent for convex and non-convex stochastic optimization. arXiv:1911.08380, 2019.
71. D. Dvinskikh, A. Ogaltsov, A. Gasnikov, P. Dvurechensky, and V. Spokoiny. On the line-search gradient methods for stochastic optimization. IFAC-PapersOnLine, 2020. 21th IFAC World Congress, accepted, arXiv:1911.08380.
72. P. Dvurechensky. Gradient method with inexact oracle for composite non-convex optimization. arXiv:1703.09180, 2017.
73. P. Dvurechensky, E. Gorbunov, and A. Gasnikov. An accelerated directional derivative method for smooth stochastic convex optimization. European Journal of Operational Research, 2020.

74. C. Fang, C. J. Li, Z. Lin, and T. Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In Advances in Neural Information Processing Systems, pages 689–699, 2018.
75. C. Fang, Z. Lin, and T. Zhang. Sharp analysis for nonconvex sgd escaping from saddle points. In Conference on Learning Theory, pages 1192–1234, 2019.
76. I. Fatkhullin and B. Polyak. Optimizing static linear feedback: Gradient method. arXiv preprint arXiv:2004.09875, 2020.
77. M. Fazel, R. Ge, S. M. Kakade, and M. Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator, 2019.
78. S. Feizi, H. Javadi, J. Zhang, and D. Tse. Porcupine neural networks:(almost) all local optima are global. arXiv preprint arXiv:1710.02196, 2017.
79. A. D. Flaxman, A. T. Kalai, and H. B. McMahan. Online convex optimization in the bandit setting: Gradient descent without a gradient. In Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '05, pages 385–394, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.
80. C. A. Floudas and P. M. Pardalos. Encyclopedia of optimization. Springer Science & Business Media, 2008.
81. A. Gasnikov, P. Dvurechensky, M. Zhukovskii, S. Kim, S. Plaunov, D. Smirnov, and F. Noskov. About the power law of the pagerank vector component distribution. part 2. the buckley–osthus model, verification of the power law for this model, and setup of real search engines. Numerical Analysis and Applications, 11(1):16–32, 2018.
82. R. Ge, F. Huang, C. Jin, and Y. Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In Conference on Learning Theory, pages 797–842, 2015.
83. S. Ghadimi and G. Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. SIAM Journal on Optimization, 23(4):2341–2368, 2013. arXiv:1309.5549.
84. S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. Mathematical Programming, 156(1):59–99, 2016.
85. S. Ghadimi, G. Lan, and H. Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. Mathematical Programming, 155(1):267–305, 2016. arXiv:1308.6594.
86. S. Ghadimi, G. Lan, and H. Zhang. Generalized uniformly optimal methods for nonlinear programming. Journal of Scientific Computing, 79(3):1854–1881, Jun 2019.
87. M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM (JACM), 42(6):1115–1145, 1995.
88. I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning. MIT Press, 2016. `http://www.deeplearningbook.org`.
89. I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. Deep learning, volume 1. MIT press Cambridge, 2016.
90. E. Gorbunov, M. Danilova, and A. Gasnikov. Stochastic optimization with heavy-tailed noise via accelerated gradient clipping. arXiv preprint arXiv:2005.10785, 2020.
91. E. Gorbunov, F. Hanzely, and P. Richtárik. A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent. In International Conference on Artificial Intelligence and Statistics, pages 680–690, 2020.
92. E. A. Gorbunov, A. Bibi, O. Sener, E. H. Bergou, and P. Richtárik. A stochastic derivative free optimization method with momentum. In ICLR, 2020.
93. A. Gotmare, N. S. Keskar, C. Xiong, and R. Socher. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. arXiv preprint arXiv:1810.13243, 2018.
94. R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik. Sgd: General analysis and improved rates. In International Conference on Machine Learning, pages 5200–5209, 2019.
95. R. M. Gower, O. Sebbouh, and N. Loizou. Sgd for structured nonconvex functions: Learning rates, minibatching and interpolation. arXiv preprint arXiv:2006.10311, 2020.

96. P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677, 2017.
97. A. O. Griewank. Generalized descent for global optimization. Journal of optimization theory and applications, 34(1):11–39, 1981.
98. S. Guminov, P. Dvurechensky, N. Tupitsa, and A. Gasnikov. Accelerated alternating minimization, accelerated Sinkhorn's algorithm and accelerated Iterative Bregman Projections. arXiv:1906.03622, 2019. WIAS Preprint No. 2695.
99. S. Guminov and A. Gasnikov. Accelerated methods for alpha-weakly-quasi-convex problems. arXiv preprint arXiv:1710.00797, 2017.
100. S. V. Guminov, Y. E. Nesterov, P. E. Dvurechensky, and A. V. Gasnikov. Accelerated primal-dual gradient descent with linesearch for convex, nonconvex, and nonsmooth optimization problems. Doklady Mathematics, 99(2):125–128, Mar 2019.
101. B. D. Haeffele and R. Vidal. Global optimality in neural network training. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7331–7339, 2017.
102. J. Z. HaoChen and S. Sra. Random shuffling beats sgd after finite epochs. arXiv preprint arXiv:1806.10077, 2018.
103. E. Hazan, K. Levy, and S. Shalev-Shwartz. Beyond convexity: Stochastic quasi-convex optimization. In Advances in Neural Information Processing Systems, pages 1594–1602, 2015.
104. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.
105. O. Hinder, A. Sidford, and N. Sohoni. Near-optimal methods for minimizing star-convex functions and beyond. In Conference on Learning Theory, pages 1894–1938. PMLR, 2020.
106. T. Hofmann, A. Lucchi, S. Lacoste-Julien, and B. McWilliams. Variance reduced stochastic gradient descent with neighbors. In Advances in Neural Information Processing Systems, pages 2305–2313, 2015.
107. S. Horváth, D. Kovalev, K. Mishchenko, S. Stich, and P. Richtárik. Stochastic distributed learning with gradient quantization and variance reduction. arXiv preprint arXiv:1904.05115, 2019.
108. P. Jain and P. Kar. Non-convex optimization for machine learning. Found. Trends Mach. Learn., 10(3–4):142–336, Dec. 2017.
109. P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. In Proceedings of the forty-fifth annual ACM symposium on Theory of computing, pages 665–674, 2013.
110. K. Ji, Z. Wang, Y. Zhou, and Y. Liang. Improved zeroth-order variance reduced algorithms and analysis for nonconvex optimization, 2019.
111. Z. Ji and M. J. Telgarsky. Gradient descent aligns the layers of deep linear networks. In 7th International Conference on Learning Representations, ICLR 2019, 2019.
112. C. Jin, R. Ge, P. Netrapalli, S. M. Kakade, and M. I. Jordan. How to escape saddle points efficiently. volume 70 of Proceedings of Machine Learning Research, pages 1724–1732, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.
113. C. Jin, P. Netrapalli, R. Ge, S. M. Kakade, and M. I. Jordan. On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points. arXiv preprint arXiv:1902.04811, 2019.
114. C. Jin, P. Netrapalli, and M. I. Jordan. Accelerated gradient descent escapes saddle points faster than gradient descent. In Conference On Learning Theory, pages 1042–1085. PMLR, 2018.
115. R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In Advances in neural information processing systems, pages 315–323, 2013.
116. H. Karimi, J. Nutini, and M. Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 795–811. Springer, 2016.
117. A. Khaled and P. Richtárik. Better theory for sgd in the nonconvex world. arXiv preprint arXiv:2002.03329, 2020.

118. S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? SIAM Journal on Computing, 37(1):319–357, 2007.
119. R. Kidambi, P. Netrapalli, P. Jain, and S. Kakade. On the insufficiency of existing momentum schemes for stochastic optimization. In 2018 Information Theory and Applications Workshop (ITA), pages 1–9. IEEE, 2018.
120. L. Kiefer, M. Storath, and A. Weinmann. Iterative potts minimization for the recovery of signals with discontinuities from indirect measurements: The multivariate case. Foundations of Computational Mathematics, pages 1–46, 2020.
121. D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
122. J. M. Kohler and A. Lucchi. Sub-sampled cubic regularization for non-convex optimization. In International Conference on Machine Learning, pages 1895–1904, 2017.
123. D. Kovalev, S. Horváth, and P. Richtárik. Don't jump through hoops and remove those loops: Svrg and katyusha are better without the outer loop. In Algorithmic Learning Theory, pages 451–467, 2020.
124. A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.
125. T. Lacroix, N. Usunier, and G. Obozinski. Canonical tensor decomposition for knowledge base completion. In International Conference on Machine Learning, pages 2863–2872, 2018.
126. G. Lan. First-order and Stochastic Optimization Methods for Machine Learning. Springer, 2020.
127. G. Lan and Y. Yang. Accelerated stochastic algorithms for nonconvex finite-sum and multiblock optimization. SIAM Journal on Optimization, 29(4):2753–2784, 2019.
128. J. Larson, M. Menickelly, and S. M. Wild. Derivative-free optimization methods. Acta Numerica, 28:287–404, May 2019.
129. J. C. H. Lee and P. Valiant. Optimizing star-convex functions. In 2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS), pages 603–614, 2016.
130. Y. Lei, T. Hu, G. Li, and K. Tang. Stochastic gradient descent for nonconvex learning without bounded gradient assumptions. IEEE Transactions on Neural Networks and Learning Systems, 2019.
131. K. Y. Levy. The power of normalization: Faster evasion of saddle points. arXiv preprint arXiv:1611.04831, 2016.
132. D. Li, T. Ding, and R. Sun. Over-parameterized deep neural networks have no strict local minima for any continuous activations. arXiv preprint arXiv:1812.11039, 2018.
133. Y. Li, K. Lee, and Y. Bresler. Identifiability in blind deconvolution with subspace or sparsity constraints. IEEE Transactions on information Theory, 62(7):4266–4275, 2016.
134. Z. Li, H. Bao, X. Zhang, and P. Richtárik. Page: A simple and optimal probabilistic gradient estimator for nonconvex optimization. arXiv preprint arXiv:2008.10898, 2020.
135. Z. Li and P. Richtárik. A unified analysis of stochastic gradient methods for nonconvex federated optimization. arXiv preprint arXiv:2006.07013, 2020.
136. S. LIANG, R. Sun, Y. Li, and R. Srikant. Understanding the loss surface of neural networks for binary classification. In International Conference on Machine Learning, pages 2835–2843, 2018.
137. S. Liu, B. Kailkhura, P.-Y. Chen, P. Ting, S. Chang, and L. Amini. Zeroth-order stochastic variance reduction for nonconvex optimization, 2018.
138. R. Livni, S. Shalev-Shwartz, and O. Shamir. On the computational efficiency of training neural networks. In Advances in neural information processing systems, pages 855–863, 2014.
139. N. Loizou, S. Vaswani, I. Laradji, and S. Lacoste-Julien. Stochastic polyak step-size for sgd: An adaptive learning rate for fast convergence. arXiv preprint arXiv:2002.10542, 2020.
140. S. Lojasiewicz. A topological property of real analytic subsets. Coll. du CNRS, Les équations aux dérivées partielles, 117:87–89, 1963.
141. I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983, 2016.
142. A. Lucchi and J. Kohler. A stochastic tensor method for non-convex optimization. arXiv preprint arXiv:1911.10367, 2019.

143. C. Ma, K. Wang, Y. Chi, and Y. Chen. Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval and matrix completion. In International Conference on Machine Learning, pages 3345–3354. PMLR, 2018.
144. S. Ma, R. Bassily, and M. Belkin. The power of interpolation: Understanding the effectiveness of sgd in modern over-parametrized learning. In International Conference on Machine Learning, pages 3325–3334. PMLR, 2018.
145. J. Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. SIAM Journal on Optimization, 25(2):829–855, 2015.
146. D. Malik, A. Pananjady, K. Bhatia, K. Khamaru, P. L. Bartlett, and M. J. Wainwright. Derivative-free methods for policy optimization: Guarantees for linear quadratic systems, 2020.
147. J. Martens. Deep learning via hessian-free optimization. In ICML, volume 27, pages 735–742, 2010.
148. T. Mikolov. Statistical language models based on neural networks. Presentation at Google, Mountain View, 2nd April, 80, 2012.
149. K. Mishchenko, E. Gorbunov, M. Takáč, and P. Richtárik. Distributed learning with compressed gradient differences. arXiv preprint arXiv:1901.09269, 2019.
150. K. Mishchenko, A. Khaled, and P. Richtárik. Random reshuffling: Simple analysis with vast improvements. arXiv preprint arXiv:2006.05988, 2020.
151. K. G. Murty and S. N. Kabadi. Some np-complete problems in quadratic and nonlinear programming. Mathematical Programming, 39(2):117–129, Jun 1987.
152. J. A. Nelder and R. Mead. A simplex method for function minimization. The computer journal, 7(4):308–313, 1965.
153. A. Nemirovski. Orth-method for smooth convex optimization. Izvestia AN SSSR, Transl.: Eng. Cybern. Soviet J. Comput. Syst. Sci, 2:937–947, 1982.
154. Y. Nesterov. A method of solving a convex programming problem with convergence rate $o(1/k^2)$. Soviet Mathematics Doklady, 27(2):372–376, 1983.
155. Y. Nesterov. Introductory Lectures on Convex Optimization: a basic course. Kluwer Academic Publishers, Massachusetts, 2004.
156. Y. Nesterov. How to make the gradients small. Optima, 88:10–11, 2012.
157. Y. Nesterov. Lectures on convex optimization, volume 137. Springer, 2018.
158. Y. Nesterov, A. Gasnikov, S. Guminov, and P. Dvurechensky. Primal-dual accelerated gradient methods with small-dimensional relaxation oracle. Optimization Methods and Software, pages 1–28, 2020. arXiv:1809.05895.
159. Y. Nesterov and B. Polyak. Cubic regularization of newton method and its global performance. Mathematical Programming, 108(1):177–205, 2006.
160. Y. Nesterov and V. Spokoiny. Random gradient-free minimization of convex functions. Found. Comput. Math., 17(2):527–566, Apr. 2017. First appeared in 2011 as CORE discussion paper 2011/16.
161. B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring generalization in deep learning. In Advances in neural information processing systems, pages 5947–5956, 2017.
162. L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning, pages 2613–2621, 2017.
163. L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč. Stochastic recursive gradient algorithm for nonconvex optimization. arXiv preprint arXiv:1705.07261, 2017.
164. L. M. Nguyen, Q. Tran-Dinh, D. T. Phan, P. H. Nguyen, and M. van Dijk. A unified convergence analysis for shuffling-type gradient methods. arXiv preprint arXiv:2002.08246, 2020.
165. Q. Nguyen, M. C. Mukkamala, and M. Hein. On the loss landscape of a class of deep neural networks with no bad local valleys. arXiv preprint arXiv:1809.10749, 2018.
166. J. Nocedal and S. Wright. Numerical optimization. Springer Science & Business Media, 2006.

167. K. Osawa, Y. Tsuji, Y. Ueno, A. Naruse, R. Yokota, and S. Matsuoka. Second-order optimization method for large mini-batch: Training resnet-50 on imagenet in 35 epochs. arXiv preprint arXiv:1811.12019, 1:2, 2018.
168. N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning, 2017.
169. V. Papyan, Y. Romano, J. Sulam, and M. Elad. Convolutional dictionary learning via local processing. In Proceedings of the IEEE International Conference on Computer Vision, pages 5296–5304, 2017.
170. R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In International conference on machine learning, pages 1310–1318, 2013.
171. B. Polyak. Gradient methods for the minimisation of functionals. USSR Computational Mathematics and Mathematical Physics, 3(4):864 – 878, 1963.
172. B. Polyak. Introduction to Optimization. New York, Optimization Software, 1987.
173. B. T. Polyak. Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics, 4(5):1–17, 1964.
174. Q. Qu, X. Li, and Z. Zhu. A nonconvex approach for exact and efficient multichannel sparse blind deconvolution. In Advances in Neural Information Processing Systems, pages 4015–4026, 2019.
175. S. Rajput, A. Gupta, and D. Papailiopoulos. Closing the convergence gap of sgd without replacement. arXiv preprint arXiv:2002.10400, 2020.
176. S. J. Reddi, A. Hefny, S. Sra, B. Poczos, and A. Smola. Stochastic variance reduction for nonconvex optimization. In International conference on machine learning, pages 314–323, 2016.
177. S. J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. arXiv preprint arXiv:1904.09237, 2019.
178. S. J. Reddi, S. Sra, B. Poczos, and A. J. Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In Advances in Neural Information Processing Systems, pages 1145–1153, 2016.
179. A. Roy, K. Balasubramanian, S. Ghadimi, and P. Mohapatra. Escaping saddle-point faster under interpolation-like conditions. Advances in Neural Information Processing Systems, 33, 2020.
180. C. W. Royer and S. J. Wright. Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. SIAM Journal on Optimization, 28(2):1448–1477, 2018.
181. I. Safran and O. Shamir. Spurious local minima are common in two-layer relu neural networks. In International Conference on Machine Learning, pages 4433–4441. PMLR, 2018.
182. K. A. Sankararaman, S. De, Z. Xu, W. R. Huang, and T. Goldstein. The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. arXiv preprint arXiv:1904.06963, 2019.
183. M. Schmidt, N. Le Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. Mathematical Programming, 162(1-2):83–112, 2017.
184. M. Schmidt and N. L. Roux. Fast convergence of stochastic gradient descent under a strong growth condition. arXiv preprint arXiv:1308.6370, 2013.
185. M. Schumer and K. Steiglitz. Adaptive step size random search. IEEE Transactions on Automatic Control, 13(3):270–276, June 1968.
186. O. Sebbouh, R. M. Gower, and A. Defazio. On the convergence of the stochastic heavy ball method. arXiv preprint arXiv:2006.07867, 2020.
187. O. Sener and V. Koltun. Learning to guide random search. In International Conference on Learning Representations, 2020.
188. S. Shalev-Shwartz. Sdca without duality, regularization, and individual convexity. In International Conference on Machine Learning, pages 747–754, 2016.
189. Y. Shechtman, Y. C. Eldar, O. Cohen, H. N. Chapman, J. Miao, and M. Segev. Phase retrieval with application to optical imaging: a contemporary overview. IEEE signal processing magazine, 32(3):87–109, 2015.
190. Z. Shen, P. Zhou, C. Fang, and A. Ribeiro. A stochastic trust region method for non-convex minimization. arXiv preprint arXiv:1903.01540, 2019.

191. B. Shi, W. J. Su, and M. I. Jordan. On learning rates and schr\" odinger operators. arXiv preprint arXiv:2004.06977, 2020.
192. L. Shi and Y. Chi. Manifold gradient descent solves multi-channel sparse blind deconvolution provably and efficiently. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5730–5734. IEEE, 2020.
193. I. Shibaev, P. Dvurechensky, and A. Gasnikov. Zeroth-order methods for noisy hölder-gradient functions, 2020.
194. Y. Shin. Effects of depth, width, and initialization: A convergence analysis of layer-wise training for deep linear neural networks. arXiv preprint arXiv:1910.05874, 2019.
195. N. Z. Shor. Generalized gradient descent with application to block programming. Kibernetika, 3(3):53–55, 1967.
196. L. N. Smith. Cyclical learning rates for training neural networks. In 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 464–472. IEEE, 2017.
197. M. V. Solodov. Incremental gradient algorithms with stepsizes bounded away from zero. Computational Optimization and Applications, 11(1):23–35, 1998.
198. M. Soltanolkotabi, A. Javanmard, and J. D. Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. IEEE Transactions on Information Theory, 65(2):742–769, 2018.
199. V. Spokoiny et al. Parametric estimation. finite sample theory. The Annals of Statistics, 40(6):2877–2909, 2012.
200. R. Sun. Optimization for deep learning: theory and algorithms. arXiv preprint arXiv:1912.08957, 2019.
201. I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In International conference on machine learning, pages 1139–1147, 2013.
202. G. Swirszcz, W. M. Czarnecki, and R. Pascanu. Local minima in training of deep networks. 2016.
203. Y. S. Tan and R. Vershynin. Online stochastic gradient descent with arbitrary initialization solves non-smooth, non-convex phase retrieval. arXiv preprint arXiv:1910.12837, 2019.
204. W. Tao, Z. Pan, G. Wu, and Q. Tao. Primal averaging: A new gradient evaluation step to attain the optimal individual convergence. IEEE transactions on cybernetics, 50(2):835–845, 2018.
205. A. Taylor and F. Bach. Stochastic first-order methods: non-asymptotic and computer-aided analyses via potential functions. In Conference on Learning Theory, pages 2934–2992, 2019.
206. T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 4(2):26–31, 2012.
207. N. Tripuraneni, M. Stern, C. Jin, J. Regier, and M. I. Jordan. Stochastic cubic regularization for fast nonconvex optimization. In Advances in neural information processing systems, pages 2899–2908, 2018.
208. P. Tseng. An incremental gradient (-projection) method with momentum term and adaptive stepsize rule. SIAM Journal on Optimization, 8(2):506–531, 1998.
209. I. Usmanova. Robust solutions to stochastic optimization problems. Master Thesis (MSIAM); Institut Polytechnique de Grenoble ENSIMAG, Laboratoire Jean Kuntzmann, 2017.
210. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017.
211. S. Vaswani, F. Bach, and M. Schmidt. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 1195–1204. PMLR, 2019.
212. S. Vaswani, A. Mishkin, I. Laradji, M. Schmidt, G. Gidel, and S. Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. In Advances in Neural Information Processing Systems, pages 3732–3745, 2019.

213. S. A. Vavasis. Black-box complexity of local minimization. SIAM Journal on Optimization, 3(1):60–80, 1993.
214. R. Vidal, J. Bruna, R. Giryes, and S. Soatto. Mathematics of deep learning. arXiv preprint arXiv:1712.04741, 2017.
215. Z. Wang, K. Ji, Y. Zhou, Y. Liang, and V. Tarokh. Spiderboost: A class of faster variance-reduced algorithms for nonconvex optimization. arXiv preprint arXiv:1810.10690, 2018.
216. Z. Wang, K. Ji, Y. Zhou, Y. Liang, and V. Tarokh. Spiderboost and momentum: Faster variance reduction algorithms. In Advances in Neural Information Processing Systems, pages 2403–2413, 2019.
217. Z. Wang, Y. Zhou, Y. Liang, and G. Lan. Stochastic variance-reduced cubic regularization for nonconvex optimization. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 2731–2740. PMLR, 2019.
218. Z. Wang, Y. Zhou, Y. Liang, and G. Lan. Cubic regularization with momentum for nonconvex optimization. In Uncertainty in Artificial Intelligence, pages 313–322. PMLR, 2020.
219. R. Ward, X. Wu, and L. Bottou. Adagrad stepsizes: Sharp convergence over nonconvex landscapes. In International Conference on Machine Learning, pages 6677–6686. PMLR, 2019.
220. A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive gradient methods in machine learning. In Advances in neural information processing systems, pages 4148–4158, 2017.
221. S. J. Wright. Optimization algorithms for data analysis. The Mathematics of Data, 25:49, 2018.
222. F. Wu and P. Rebeschini. Hadamard wirtinger flow for sparse phase retrieval. arXiv preprint arXiv:2006.01065, 2020.
223. P. Xu, J. Chen, D. Zou, and Q. Gu. Global convergence of langevin dynamics based algorithms for nonconvex optimization. In Advances in Neural Information Processing Systems, pages 3122–3133, 2018.
224. P. Xu, F. Roosta, and M. W. Mahoney. Newton-type methods for non-convex optimization under inexact hessian information. Mathematical Programming, 184(1):35–70, 2020.
225. P. Xu, F. Roosta, and M. W. Mahoney. Second-order optimization for non-convex machine learning: An empirical study. In Proceedings of the 2020 SIAM International Conference on Data Mining, pages 199–207. SIAM, 2020.
226. Y. Xu. Momentum-based variance-reduced proximal stochastic gradient method for composite nonconvex stochastic optimization. arXiv preprint arXiv:2006.00425, 2020.
227. Y. Xu, R. Jin, and T. Yang. First-order stochastic algorithms for escaping from saddle points in almost linear time. In Advances in Neural Information Processing Systems, pages 5530–5540, 2018.
228. Y. Yan, T. Yang, Z. Li, Q. Lin, and Y. Yang. A unified analysis of stochastic momentum methods for deep learning. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, pages 2955–2961, 2018.
229. Z. Yang, L. F. Yang, E. X. Fang, T. Zhao, Z. Wang, and M. Neykov. Misspecified nonconvex statistical optimization for sparse phase retrieval. Mathematical Programming, 176(1-2):545–571, 2019.
230. C. Yun, S. Sra, and A. Jadbabaie. Small nonlinearities in activation functions create bad local minima in neural networks. arXiv preprint arXiv:1802.03487, 2018.
231. J. Yun, A. C. Lozano, and E. Yang. A general family of stochastic proximal gradient methods for deep learning. arXiv preprint arXiv:2007.07484, 2020.
232. M. Zaheer, S. Reddi, D. Sachan, S. Kale, and S. Kumar. Adaptive methods for nonconvex optimization. In Advances in neural information processing systems, pages 9793–9803, 2018.
233. B. Zhang, J. Jin, C. Fang, and L. Wang. Improved analysis of clipping algorithms for nonconvex optimization. Advances in Neural Information Processing Systems, 33, 2020.
234. C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530, 2016.

235. J. Zhang, T. He, S. Sra, and A. Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In International Conference on Learning Representations, 2020.
236. J. Zhang, S. P. Karimireddy, A. Veit, S. Kim, S. Reddi, S. Kumar, and S. Sra. Why are adaptive methods good for attention models? Advances in Neural Information Processing Systems, 33, 2020.
237. J. Zhang and L. Xiao. Stochastic variance-reduced prox-linear algorithms for nonconvex composite optimization. arXiv preprint arXiv:2004.04357, 2020.
238. J. Zhang, L. Xiao, and S. Zhang. Adaptive stochastic variance reduction for subsampled newton method with cubic regularization. arXiv preprint arXiv:1811.11637, 2018.
239. Y. Zhang, Q. Qu, and J. Wright. From symmetry to geometry: Tractable nonconvex problems. arXiv preprint arXiv:2007.06753, 2020.
240. Y. Zhang, Q. Qu, and J. Wright. From symmetry to geometry: Tractable nonconvex problems. arXiv preprint arXiv:2007.06753, 2020.
241. Y. Zhang, Y. Zhou, K. Ji, and M. M. Zavlanos. Boosting one-point derivative-free online optimization via residual feedback, 2020.
242. A. Zhigljavsky and A. Zilinskas. Stochastic global optimization, volume 9. Springer Science & Business Media, 2007.
243. D. Zhou and Q. Gu. Lower bounds for smooth nonconvex finite-sum optimization. In International Conference on Machine Learning, pages 7574–7583, 2019.
244. D. Zhou and Q. Gu. Stochastic recursive variance-reduced cubic regularization methods. In International Conference on Artificial Intelligence and Statistics, pages 3980–3990. PMLR, 2020.
245. D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu. On the convergence of adaptive gradient methods for nonconvex optimization. arXiv preprint arXiv:1808.05671, 2018.
246. D. Zhou, P. Xu, and Q. Gu. Stochastic nested variance reduced gradient descent for nonconvex optimization. Advances in neural information processing systems, 2018.
247. D. Zhou, P. Xu, and Q. Gu. Stochastic variance-reduced cubic regularization methods. Journal of Machine Learning Research, 20(134):1–47, 2019.
248. D. Zhou, P. Xu, and Q. Gu. Stochastic variance-reduced cubic regularization methods. Journal of Machine Learning Research, 20(134):1–47, 2019.
249. X. Zhu, J. Han, and B. Jiang. An adaptive high order method for finding third-order critical points of nonconvex optimization. arXiv preprint arXiv:2008.04191, 2020.