Optimization via Greedy Oriented Progression

Xiaoli Li [1] **Xiaohong Chen** [2] **Wei Biao Wu** [1]

Abstract

In this paper, we propose a novel optimization framework termed Greedy Oriented Progression (GOP). Unlike varying step-size strategies in traditional gradient methods, GOP dynamically constructs updates by finding the intersection of the descent direction with the feasible region's boundary, followed by a linear combination step. This mechanism naturally enforces a "wide-to-narrow" search trajectory, enabling efficient global exploration in early phases and fine-grained exploitation in later stages. Theoretically, we prove that under a mild Local Radial Alignment assumption, GOP converges to the global minimizer when augmented with a growing set of starting points, and it achieves a convergence rate of $O(1/n)$. Empirically, we validate GOP and its adaptive variant (Adam-GOP) on complex multi-modal synthetic functions and large-scale deep learning tasks. Our experiments demonstrate that GOP consistently outperforms SGD, achieving lower training loss and faster convergence speed with comparable computational efficiency.

1. Introduction

Stochastic optimization lies at the heart of modern machine learning. The success of deep learning is largely attributed to the effectiveness of first-order gradient-based algorithms in navigating the high-dimensional, non-convex loss landscapes of neural networks. While Stochastic Gradient Descent (SGD), also known as the Robbins-Monro algorithm (1951), and its momentum-based variants (Polyak, 1964) act as the workhorses of the field, they fundamentally operate as local search heuristics. The objective functions of deep neural networks are replete with local minima, saddle points, and flat plateaus (Dauphin et al., 2014; Choromanska et al., 2015; Li et al., 2018). Consequently, standard SGD is prone to premature convergence to suboptimal solutions, limiting the potential performance of the model.

To accelerate convergence and reduce sensitivity to hyperparameter tuning, a rich family of adaptive gradient methods has emerged. Algorithms such as AdaGrad (Duchi et al., 2011), RMSProp (Tieleman & Hinton, 2012), and Adam (Kingma & Ba, 2014) adapt the step size for each parameter based on historical gradient moments. Despite their rapid initial progress, recent studies suggest that adaptive methods may converge to sharp local minima, leading to poorer generalization compared to SGD (Wilson et al., 2017; Reddi et al., 2018; Dereich et al., 2025). Furthermore, approaches like AdamW (Loshchilov & Hutter, 2019) attempt to fix weight decay issues but do not fundamentally alter the local nature of the search trajectory. On the contrary, the ability to find the global minima has been increasingly appealing to researchers, especially for those in the fields of economics and medicine, whose primary concern is inference (Arnoud et al., 2019; Alvarez & Calaminici, 2025).

Therefore, addressing the "local trap" issue has spurred significant research into algorithms capable of escaping saddle points and approximating global minima. One prominent direction involves noise injection, such as Stochastic Gradient Langevin Dynamics (SGLD) (Welling & Teh, 2011) and Perturbed SGD (Jin et al., 2017), which add isotropic noise to gradients to "jump" out of shallow traps. However, these methods often require carefully annealed noise schedules and can slow down convergence. Another line of work utilizes curvature information, such as Cubic Regularization (Nesterov & Polyak, 2006) or Trust Region methods. While theoretically powerful in escaping saddle points, their reliance on Hessian information (or Hessian-vector products) introduces a prohibitive computational burden—often $O(d^2)$ or $O(d^3)$—rendering them impractical for modern deep networks with millions of parameters. Finally, derivative-free global optimization techniques, such as Simulated Annealing (Kirkpatrick et al., 1983) or Particle Swarm Optimization, theoretically guarantee global convergence but succumb to the curse of dimensionality, failing to scale beyond small problems.

Our proposed approach shares conceptual roots with the Frank-Wolfe algorithm (Frank & Wolfe, 1956; Jaggi, 2013) and the SMCO algorithm (Chen et al., 2026). Frank-Wolfe

[1]Department of Statistics, the University of Chicago, Chicago, IL 60637, USA [2]Department of Economics, Yale University, New Haven, CT 06520, USA. Correspondence to: Wei Biao Wu <wbwu@uchicago.edu>.

Preprint. February 9, 2026.

methods avoid projection steps by updating towards a vertex of the constraint set (the Linear Minimization Oracle). While highly efficient for constrained convex optimization, standard Frank-Wolfe is not designed for non-convex exploration in the way global heuristics are. On the contrary, SMCO algorithm can efficiently solve non-convex optimization for fixed target function by exploiting the sign information of the population-level gradient, but how to extend it to the stochastic optimization case remains unclear due to the discrete nature. Our method combines and reshapes their update structure. Instead of solving a linear subproblem, we perform a greedy search towards the boundary of the feasible region defined by the gradient direction.

In this paper, we bridge the gap between efficient first-order training and global exploration by proposing Greedy Oriented Progression (GOP). Distinct from SGD, GOP does not rely on a fixed step size. Instead, it dynamically determines the step length by identifying the intersection of the current descent direction with the boundary of the search space. This strategy naturally induces a "wide-to-narrow" search pattern.

Exploration: In early stages, the distance to the boundary is large, allowing the algorithm to take aggressive steps that traverse the landscape and bypass local optima.

Exploitation: As the optimization proceeds, the step sizes naturally decay, allowing for fine-grained convergence to the minimizer.

The search strategy of GOP is significantly distinct from the variants of SGD, including decaying learning rate schedule, in the sense that it first geometrically searches over the boundary and then pushes to the interior. In this way, GOP potentially enjoys a much wider search region due to the high-dimensional structure.

We further propose Adam-GOP, a hybrid variant that incorporates the momentum and adaptive scaling of Adam, making the method robust for training deep neural networks such as Transformers.

1.1. Contributions

The specific contributions of this work are four-fold:

- **Algorithmic Innovation:** We introduce GOP, a first-order, gradient-based, efficient algorithm that utilizes boundary information to adaptively determine the step size and escape local minima without the computational cost of second-order methods. Similarly to SGD, our proposed GOP is friendly to parallel computing and batched processing. In particular, the computational complexity is comparable to SGD and other fast stochastic optimization algorithms.
- **Justified Novel Geometric Condition:** We propose a novel Local Radial Alignment assumption which has simple and explicit geometric interpretation. Even though this condition holds under general scenarios, we further justify it by deriving sharp Nagaev-style tail bounds that control the probability of large deviations.
- **Theoretical Rigor:** We establish the theoretical foundation of GOP under the novel Local Radial Alignment assumption. We prove that GOP converges to the global minimizer when augmented with a growing set of starting points and provide a convergence rate analysis of $O(1/n)$, offering stronger guarantees than typical non-convex convergence results.
- **Empirical Validation:** We validate GOP and Adam-GOP on diverse tasks, ranging from multi-modal synthetic functions (Rastrigin, Michalewicz) to high-dimensional deep learning benchmarks (DeepReLU MLP, Transformer). Our results demonstrate that GOP consistently achieves lower training loss and better convergence speed than SGD and Adam.

1.2. Related Literature

Stochastic Gradient Methods. Stochastic Gradient Descent (SGD) (Robbins & Monro, 1951) and its variants remain the dominant optimization paradigm in deep learning. Momentum-based methods (Polyak, 1964; Nesterov, 1983) accelerate training by accumulating gradients, while adaptive methods like Adam (Kingma & Ba, 2014) and RMSProp (Tieleman & Hinton, 2012) adjust learning rates per parameter. Other robust variants, including SignGD (Bernstein et al., 2018; Moulay et al., 2019; Safaryan & Richtarik, 2021) are also proposed to improve stability and convergence in some cases (Li et al., 2024b). Despite their success, their performances remain highly sensitive to the choices of step sizes and initial points. Consequently, these methods face challenges, particularly in high-dimensional settings, where step size determination is complex. Additionally, they are prone to gradient explosion in early training phases (Bengio et al., 1994; Pascanu et al., 2013), causing slow convergence or divergence. Our GOP may look like SGD since both use the gradient direction, but is quite different in that our updating mechanism automatically adapts step sizes with respect to both the iteration counter and the current optimizer position. In simulations, how our algorithms compares favorably with SGD and its variants in terms of optimization performances is clearly demonstrated, especially in optimization problems with challenging landscapes.

Nonconvex Optimization. While the theoretical analysis of SGD and its variants has been extensively studied for convex functions, including the convergence (Blum, 1954; Li et al., 2024a), asymptotic normality (Gitman et al., 2019) and the last-iterate behavior (Garrigos et al., 2025), the theoretical properties in non-convex settings has been largely unexplored, leaving a significant gap in the literature. Meanwhile, recent analyses (Reddi et al., 2018; Zhou et al., 2024; Dereich et al., 2025) have highlighted that adaptive methods can suffer from convergence issues in non-convex settings and may generalize worse than SGD. In contrast, our proposed GOP provides a mechanism to escape saddle points and local minima where standard gradient-based methods stagnate, and enjoys theoretical convergence guarantee under mild assumptions. Another research line develops various algorithms which have ability to escape saddle points. For example, Perturbed Gradient Descent (Jin et al., 2017) and Stochastic Gradient Langevin Dynamics (Welling & Teh, 2011) introduce noise to destabilize saddle points. While effective, these methods often require slow annealing schedules to ensure convergence. Second-order methods, such as Cubic Regularization (Nesterov & Polyak, 2006), utilize Hessian information to navigate curvature. However, computing the full Hessian or even Hessian-vector products is computationally prohibitive for high-dimensional neural networks. Unlike these approaches, GOP is a purely first-order method that achieves global exploration without explicit noise injection or expensive curvature computations.

2. Preliminaries

2.1. Problem Setting

To be specific, we consider the following general optimization problem

$$\theta^* \in \arg\min_{\theta\in\Gamma\subset\mathbb{R}^d} G(\theta),\ \text{where}\ G(\theta) = \mathbb{E}_{X\sim\Pi}[g(\theta, X)]. \quad (1)$$

In (1), $\Gamma \subset \mathbb{R}^d$ is a bounded convex set, $G : \Gamma \to \mathbb{R}$ is the differentiable objective function, and $g(\theta, X)$ is the noise-perturbed loss function with sequentially arriving i.i.d. random data $X_i \sim \Pi$ from some unknown distribution Π. We have access to $\nabla g(\theta, X) = \partial g(\theta, X)/\partial\theta = (\partial g(\theta, X)/\partial\theta_1, \ldots, \partial g(\theta, X)/\partial\theta_d)^\top$, the stochastic gradient with respect to θ, as claimed by various SGD algorithms.

Problem (1) is general enough to represent most of modern optimization problems in the sense that Γ can be set artificially, and in that restricting the feasible optimization region is equivalent to adding some regularization term.

For example, G can be a multi-modal and high-dimension differential function, such as Rastrigin function, with L_p-regularization for some $p \geq 1$. In this case, it is equivalent to setting $\Gamma = \{\theta : \|\theta\|_p \leq \lambda\}$ for some $\lambda > 0$ and solving problem (1). It is well-known that developing a fast algorithm, whose convergence to a global maximum is mathematically proven, is extremely difficult without assuming G being a global convex function.

2.2. Notation

For a vector $a = (a_1, \ldots, a_d)^\top \in \mathbb{R}^d$ and $q > 0$, we denote $\|a\|_q = (\sum_{i=1}^d |a_i|^q)^{1/q}$, $\|a\|_\infty = \max_{1\leq i\leq d} |a_i|$ and $\|a\| = \|a\|_2$. For any $s > 0$ and a random vector X, we say $X \in L_s$ if $\mathbb{E}[\|X\|_2^s] < \infty$. The symbols ∇ and ∇^2 represent the gradient vector and Hessian matrix of a function $f : \mathbb{R}^d \to \mathbb{R}$, respectively. For an event A, the indicator function that the event A happens is represented by $I\{A\}$. We use $\partial\Gamma$ to denote the boundary of any convex region Γ and $\mathcal{B}(a, r)$ to denote the common L_2 ball with center a and radius r.

3. Theory and Algorithms

Having discussed the limitations of existing methods, we now formally present the GOP algorithm and its theoretical analysis.

3.1. Greedy Oriented Progression (GOP) Algorithm

We propose a novel gradient based algorithm for optimizing target function with stochastic gradient, which is named as Greedy Oriented Progression (GOP). Similarly to the stochastic gradient descent (SGD) method, it can be easily implemented under the parallel computing structure. The details of this method are shown in Algorithm 1.

Algorithm 1 Greedy Oriented Progression (GOP)

```
1: Input: Loss function $g(\theta; X)$, identically distributed independent data points $(X_1, \ldots, X_N)$, batch size $B$, convex feasible region $\Gamma$, maximum number of iterations $M$ and the initial point $\theta_0$.
2: for $n = 1$ to $\min(M, \lfloor N/B \rfloor)$ do
3:     Calculate the noisy gradient vector direction $d_n = \frac{\sum_{i=(n-1)B+1}^{nB} \nabla g(\theta_{n-1}, X_i)}{\|\sum_{i=(n-1)B+1}^{nB} \nabla g(\theta_{n-1}, X_i)\|}$.
4:     Get the intersection point $Z_n \in \{\theta_{n-1} - t d_n, t \geq 0\} \cap \partial\Gamma$.
5:     Update $\theta_n = \theta_{n-1} + \frac{1}{n+1}(Z_n - \theta_{n-1})$.
6: end for
```

Motivated by the similarity between stochastic optimization and statistical learning, the choice of $\frac{1}{n+1}$ is natural in that if we denote $S_n = \sum_{i=0}^n Z_i$, where $Z_0 = \theta_0$, then the update mechanism can be formulated as $\theta_n = S_n/(n + 1)$, which is the sample mean of random variables sampled near the boundary. In particular, it is justified by the Strategic Law of Large Numbers (StLLN) as well, see the discussions in (Chen et al., 2026).

Compared to SGD, this method adaptively chooses the step size based on the current iteration number and the location. In other words, instead of moving along the gradient direction with a pre-specified step size, in each step we greedily progress to the boundary, which enables our algorithm to search widely in the initial few updates. Our GOP algorithm can also be interpreted as gradient optimal projection (GOP) or gradient oriented projection (GOP).

Due to the benefits of using the mini-batch trick, GOP can easily handle the large scale optimization problems. In this efficient way that we input the stochastic gradient computed batch by batch, instead of the whole gradient, the computational cost is significantly reduced. Meanwhile, since we have many efficient algorithms, such as binary search, to compute the intersection point Z_n, our GOP algorithm can potentially be as fast as SGD.

3.2. Analysis on the Updating Mechanism

To effectively address the non-convexity of the problem, we introduce a set of general assumptions. Let $\mathcal{M} = \{\theta \in \Gamma : |\nabla G(\theta)| = 0\}$ represent the set of critical points, and $\mathcal{M}_0 \subset \mathcal{M}$ denote the set of local minima. To proceed, we impose the following standard regularity conditions on G and g, which are essential for establishing our results.

Assumption 3.1. The set $\mathcal{M}_0$ is non-empty and finite. Moreover, $\mathcal{M}_0 \cap \partial\Gamma = \emptyset$.

The assumption that $\mathcal{M}_0$ is non-empty is standard in the non-convex optimization literature, and appears in works such as (Gitman et al., 2019; Jin et al., 2022) in the context of almost sure convergence of SGD. Additionally, we assume that $\mathcal{M}_0$ is finite. While this may seem restrictive, many well-known non-convex problems, such as matrix sensing, phase retrieval, matrix completion, and regression with non-convex constraints, satisfy this condition; see (Ge et al., 2015; Tan & Vershynin, 2023). For neural networks, although saddle-point regions of positive measure may exist, it has been demonstrated that gradient based algorithms like SGD can escape these regions with arbitrarily high probability (Kleinberg et al., 2018; Wang et al., 2020). The interior condition of $\mathcal{M}_0$ will be automatically satisfied if we push the boundary a little and make a smooth flat extension of the function G (Chen et al., 2026). Therefore, Assumption 3.1 can be reasonably justified for most practical applications.

Furthermore, the optimization landscape of many non-convex statistical problems is often benign, allowing us to introduce additional assumptions regarding the behavior of G and g near the local minima. Specifically, we introduce the following condition.

Assumption 3.2. (Local Radial Alignment Condition). For each $\theta \in \mathcal{M}_0$, there exist $\delta(\theta) > 0$ and $c(\theta) > 0$ such that for all $\theta' \in \mathcal{B}(\theta, \delta(\theta))$,

$$\langle \theta' - \theta, \nabla G(\theta') \rangle \geq c\|\theta' - \theta\|\|\nabla G(\theta')\|. \tag{2}$$

Note that (2) is an acute angle condition. Geometrically, this assumption ensures that near the local minimum, the true gradient direction is strictly aligned with the position direction, i.e. the angle is strictly acute. Due to Assumption 3.1, we can assume $\delta(\theta) \equiv \delta$ and $c(\theta) \equiv c$ for all $\theta \in \mathcal{M}_0$.

Remark 3.3. Note that with local strong convexity and local Lipschitz continuity of $\nabla G(\theta)$ near $\mathcal{B}(\theta, \delta)$, we have

$$\nabla G(\theta')^\top (\theta' - \theta) \geq \mu\|\theta' - \theta\|^2 \geq \frac{\mu}{\mathcal{L}}\|\theta' - \theta\|\|\nabla G(\theta')\|, \tag{3}$$

where μ is the strong convexity constant and $\mathcal{L}$ is the Lipschitz constant. Thus, Assumption 3.2 holds for a broad class of optimization problems. Such localized assumptions are commonly employed in non-convex analysis, such as (Yu et al., 2021; Zhong et al., 2023).

Next we analyze the iterates of GOP. Specifically, let $R(\theta, X) = \nabla G(\theta) - \nabla g(\theta, X)$ denote the gradient noise and

$$R_B(\theta) = B^{-1} \sum_{i=1}^{B} R(\theta, X_i), \tag{4}$$

which is the approximation error of the noisy gradient computed in a mini-batch with batch size B. To ensure the estimation error of GOP exhibits a quantifiable asymptotic behavior, it is crucial to control the randomized gradient $\nabla g(\theta_{n-1}, X_n)$. Thus we assume the exchangeability of integration and differentiation in the analysis, and impose mild smoothness conditions on the gradient noise $R(\theta, X)$.

Assumption 3.4. (Regularity of Gradient Noise). The function $g(\theta, X)$ is assumed to be continuously differentiable with respect to θ for any fixed X, and $E[\|\nabla g(\theta, X)\|^2]$ is also continuously differentiable. Additionally, it is assumed that $E[R(\theta_{n-1}, X_n)|\mathcal{F}_{n-1}] = 0$ for $\mathcal{F}_{n-1} = \sigma(\{\theta_0, \ldots, \theta_{n-1}\})$.

Assumption 3.5. (Stochastic Local Hölder's Continuity). The gradient noise $R(\theta, X)$ satisfies

$$E[\sup_{\theta \in \Gamma} \|R(\theta, X_i)\|_q] < M, \; q > 2, \tag{5}$$

for some $M > 0$. Moreover, there exists constants $\gamma' > 0$ and $\alpha' > 0$ such that for any fixed X,

$$\begin{gathered} \|R(\theta, X) - R(\theta', X)\|_\infty \leq \mathcal{L}'(X)\|\theta - \theta'\|^{\alpha'}, \\ \text{for all } \theta, \theta' \in \Gamma \text{ with } \|\theta - \theta'\| \leq \gamma', \end{gathered} \tag{6}$$

where $\mathcal{L}'(X)$ is a non-negative random variable with finite q-th moment.

Assumptions 3.4 and 3.5 consist of uniformly bounded q-th moment and local Hölder continuity of the gradient noise, which are widely used in the literature and can be found in works such as (Zhu et al., 2023; Wei et al., 2025). Building on these assumptions, we can develop a Fuk-Nagaev type inequality to bound the error $R_B(\theta)$, which is shown in Theorem 3.6.

Theorem 3.6. *Suppose that the function g and G satisfy Assumptions 3.4 and 3.5. Then for any $y > 0$,*

$$\begin{aligned} &P\left[\sup_{\theta\in\Gamma} \|R_B(\theta)\|_\infty \geq c_1 d \log B/\sqrt{B} + y\right] \\ &\leq c_2 \frac{B^{1-q}}{y^q} + \exp(-c_3 y^2 B), \end{aligned} \tag{7}$$

where c_1, c_2 and c_3 are positive constants independent of d and B. Consequently, for any $\varepsilon > 0$ and

$$\arcsin\left(\frac{2c_1 d^{3/2}\log B}{\varepsilon\sqrt{B}} \wedge 1\right) \leq y \leq \pi/2,$$

we have

$$\begin{aligned} &P\left[\sup_{\theta\in\Gamma} \omega(\theta) I\{\|\nabla G(\theta)\| > \varepsilon\} > y\right] \\ &\leq c_2 \frac{d^{q/2} B^{1-q}}{(\varepsilon \sin y)^q} + \exp(-c_3 B(\varepsilon \sin y)^2/d), \end{aligned} \tag{8}$$

where

$$\omega(\theta) = \arccos \frac{\nabla G(\theta)^\top (\sum_{i=1}^B \nabla g(\theta, X_i))}{\|\nabla G(\theta)\| \|\sum_{i=1}^B \nabla g(\theta, X_i)\|}$$

is the angle between the true gradient $\nabla G(\theta)$ and the noisy gradient $\frac{1}{B}\sum_{i=1}^B \nabla g(\theta, X_i)$.

Theorem 3.6 implies that when the batch size B is large enough, the approximation error can be arbitrarily small and thus, the noisy gradient direction is well-aligned with the true gradient direction. It shows that the uniform error $\sup_{\theta\in\Gamma}\|R_B(\theta)\|_\infty$ is of probabilistic order $\mathcal{O}_P(d\log B/\sqrt{B})$. Consequently, with Assumption 3.2 on the true gradient, we can derive the following Lemma for the alignment of the updating direction, which is crucial for our theoretical analysis.

Lemma 3.7. *Suppose that the function g and G satisfy Assumptions 3.1, 3.2, 3.4 and 3.5. Then for any $\theta \in \mathcal{M}_0$ and $\varepsilon > 0$, there exist a batch size $B > 0$ and a radius $r > 0$, such that for any $n \in \mathbb{N}^+$, whenever $\theta_{n-1} \in \mathcal{B}(\theta, r)$ and $\|\nabla G(\theta_{n-1})\| \geq \epsilon$, we have*

$$E[\langle \theta - \theta_{n-1}, \theta_n - \theta_{n-1}\rangle | \mathcal{F}_{n-1}] \geq \frac{c_4}{n+1}\|\theta_{n-1} - \theta\|, \tag{9}$$

where $c_4 > 0$ is a constant.

3.3. Theoretical Results on Convergence

As discussed in Theorem 3.6 and Lemma 3.7, under mild conditions, Eq. (9) holds, which implies that the updating direction $\theta_n - \theta_{n-1}$ will be well-aligned with the position vector $\theta - \theta_{n-1}$. Since Eq.(9) may also hold under other conditions, we state it as an intermediate assumption for generality and brevity.

Assumption 3.8. (Local Updating Alignment Condition) For each $\theta \in \mathcal{M}_0$, there exist $\delta'(\theta) > 0$ and $c'(\theta) > 0$ such that for any $n \in \mathbb{N}^+$, whenever $\theta_{n-1} \in \mathcal{B}(\theta, \delta'(\theta))$,

$$E[\langle \theta - \theta_{n-1}, \theta_n - \theta_{n-1}\rangle | \mathcal{F}_n] \geq \frac{c'}{n+1}\|\theta_{n-1} - \theta\|. \tag{10}$$

Next, based on the intermediate Assumption 3.8, we are ready to state our main theorems on the convergence of GOP.

Theorem 3.9. *Let G be a twice differentiable function on Γ and $L = \sup\{\|x - y\| : (x, y) \in \Gamma \otimes \Gamma\}$. Suppose that the functions g and G satisfy Assumptions 3.1 and 3.8. Then for any $\theta \in \mathcal{M}_0$, $\varepsilon > 0$ and $0 < r \leq \delta'(\theta)$, there exist $r_1 = \sqrt{\varepsilon}r/2$ and an integer $N_0 = \lceil \frac{2L^2}{r^2\varepsilon} \rceil$ such that for any $m \geq N_0$, whenever $\theta_m \in \mathcal{B}(\theta, r_1)$, the event $\Omega_{r,\theta} = \{\theta_n \in \mathcal{B}(\theta, r),\ \forall\ n \geq m\}$ occurs with probability at least $1 - \varepsilon$.*

Furthermore, given that $\Omega_{r,\theta}$ occurs, we have that

$$P(\lim_{n\to\infty} \theta_n = \theta \mid \Omega_{r,\theta}) = 1. \tag{11}$$

As a result, given a growing (deterministically, randomly, or quasi-randomly generated) set of starting points $\mathcal{X}_m$ such that $\#(\mathcal{X}_m) = m$ and $P(\mathcal{B}(\theta^*, r_1) \cap \mathcal{X}_m \neq \emptyset) \to 1$ as $m \to \infty$, where θ^* is a global minimizer, the GOP algorithm run from all points in $\mathcal{X}_m$ with a "boosted" starting iteration index $n_0 = N_0$ is guaranteed to attain a global minimizer θ^* asymptotically, as $n \to \infty$ and $m \to \infty$.

Finally, we present a rate of convergence of our GOP algorithm given that $\Omega_{r,\theta}$ occurs, which is stated in the next Theorem 3.10.

Theorem 3.10. *Let G be a twice differentiable function on Γ and $L = \sup\{\|x - y\| : (x, y) \in \Gamma \otimes \Gamma\}$. Suppose that the functions g and G satisfy Assumptions 3.1 and 3.8 with the alignment parameter $c' > 0$. For any $\theta \in \mathcal{M}_0$, given that $\Omega_{\tilde{r},\theta}$ occurs, where $\tilde{r} \leq \min(c', \delta')$, then we have for all $n \geq m$,*

$$E\left[\|\theta_n - \theta\|^2\right] \leq \frac{n-m}{(n+1)^2}L^2 + \frac{(m+1)^2}{(n+1)^2}\tilde{r}^2. \tag{12}$$

4. Simulation Study

In this section, we present the results of the numerical experiments to empirically demonstrate the validity of our theory and the superiority of the proposed algorithm GOP. For brevity, the implementation details of algorithms, including discussion on the choice of hyperparameters such as the "learning rate" of GOP, and extra simulation results are deferred to Appendices B and C.

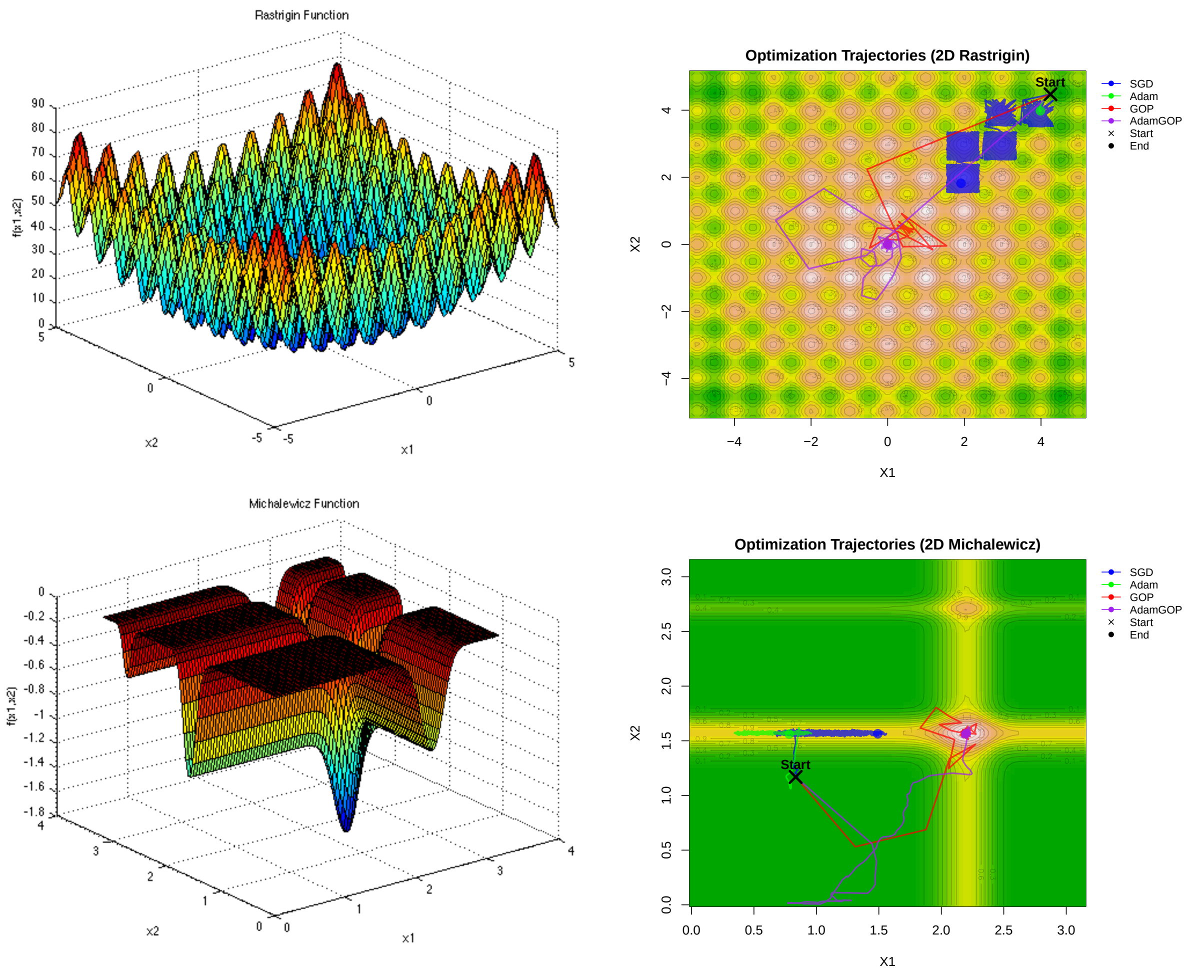

Figure 1. Left: Visualization of test functions retrieved from Virtual Library of Simulation Experiments: Test Functions and Datasets. Right: Trajectories of four optimization algorithms in a single run.

4.1. Random Optimization of Complicated Multimodal Functions

To investigate the global convergence properties of GOP in non-convex landscapes populated with numerous local optima, we benchmark its performance against standard stochastic optimizers and their variants (SGD and Adam). Specifically, we employ two classical non-convex benchmark functions known for their pathological landscapes: (1) The Rastrigin function on $[-5.12, 5.12]^d$. This function has a unique global minimum of value 0 at the origin, multiple global maxima at $\theta \in \{\pm 4.52299\}^d$, and many local optima. (2) The Michalewicz function on $[0, \pi]^d$, which features steep valleys and ridges, posing significant challenges for gradient-based methods to escape flat saddle points. This function has a unique global minimum, multiple global maxima, and many saddle points.

Since under these settings, the population level loss functions are known exactly, we inject isotropic Gaussian noise into the exact gradients to simulate the stochasticity inherent in mini-batch training. Specifically, in each step the stochastic optimization algorithms receive a corrupted gradient $\nabla g(\theta, \epsilon_n) = \nabla G(\theta) + \sigma\epsilon_n$, where $\epsilon_n \sim \mathcal{N}(0, I_d)$ represents the noise at step n and σ denotes the noise level. This setup allows us to rigorously test the robustness of the algorithms under controlled stochastic environments.

Figure 1 visualizes the optimization trajectories on 2D surfaces. A distinct contrast in behavior is observed: standard SGD and Adam rapidly converge to the nearest (suboptimal) local basin and stagnate. In contrast, GOP and Adam-GOP exhibit the "wide-to-narrow" search dynamic predicted by our theory. In the initial phase, the greedy boundary search mechanism drives the parameters to explore the feasible region broadly, effectively bypassing local traps. As iterations proceed, the update step naturally decays, allowing the algorithm to settle precisely into the global minimum.

Next, we assess the performance of these algorithms un-

Table 1. Comparison with Four Stochastic Optimization Algorithms.

		Function: Best Value	**Rastrigin:** 0			**Michalewicz:** -1.801		
		Algorithm	**Regret**	**Distance**	**Time**	**Regret**	**Distance**	**Time**
$d=2$	Min	GOP	0.159	0.112	0.101	0.086	0.110	0.108
		ADAM-GOP	0.805	0.508	0.113	0.150	0.165	0.118
		SGD	31.613	2.206	0.090	0.653	0.716	0.094
		ADAM	17.58	2.757	0.096	0.665	0.730	0.098
		Function: Best Value	**Rastrigin:** 80.707			**Michalewicz:** 0		
		Algorithm	**Regret**	**Distance**	**Time**	**Regret**	**Distance**	**Time**
	Max	GOP	2.894	–	0.101	0	–	0.091
		ADAM-GOP	1.909	–	0.108	7e-12	–	0.119
		SGD	41.157	–	0.083	3e-4	–	0.094
		ADAM	23.859	–	0.087	4e-4	–	0.095
		Function: Best Value	**Rastrigin:** 0			**Michalewicz:** -49.624		
		Algorithm	**Regret**	**Distance**	**Time**	**Regret**	**Distance**	**Time**
$d=50$	Min	GOP	3.224	0.235	1.595	1.595	–	3.779
		ADAM-GOP	26.288	0.717	1.604	1.526	–	3.681
		SGD	788.78	2.370	1.530	3.714	–	3.529
		ADAM	440.13	2.934	1.516	1.645	–	3.579
		Function: Best Value	**Rastrigin:** 2017.665			**Michalewicz:** 0		
		Algorithm	**Regret**	**Distance**	**Time**	**Regret**	**Distance**	**Time**
	Max	GOP	496.80	–	1.595	1e-12	–	3.612
		ADAM-GOP	490.11	–	1.604	8e-4	–	3.720
		SGD	1031.58	–	1.530	0.031	–	3.665
		ADAM	595.26	–	1.516	0.014	–	3.670

Note: Based on 100 repetitions. In each repetition, all algorithms run with a single starting point drawn from Sobol' sequence. Rastrigin function and Michalewicz function both have 2^d global maxima; thus we omit the measure "Distance". The exact global minimum of Michalewicz is unknown when $d = 50$.

der various settings. The quantitative results for different dimensions and starting points are summarized in Table 1. In conclusion, our proposed algorithms show state-of-the-art performance when optimizing these complex functions, whereas baseline methods incur significant optimization error.

4.2. Experiments of Large Machine Learning Models

In this section, we assess the scalability and generalization capability of GOP on high-dimensional deep learning tasks. We compare our proposed methods (GOP, Adam-GOP) with boosted starting iteration index against standard SGD and Adam.

We utilize two distinct neural network architectures trained on a synthetic regression task designed to mimic complex interactions: $y = \sin x_1 + \cos x_2 + x_3 x_4 + 0.1\mathcal{N}(0,1)$, where $x \in \mathbb{R}^{10}$. The models include **DeepReLU MLP**: A 4-layer fully connected network with 64 hidden units (approx. 1.3×10^4 parameters), and **Transformer**: An encoder-only Transformer with 2 layers, 4 heads, and model dimension 32 (approx. 1.7×10^4 parameters). We further scale this up to larger variants (up to 2×10^6 parameters) to test robustness.

Figures 2 illustrates the training dynamics in a single run. Across both architectures, GOP and Adam-GOP demonstrate superior convergence speeds, lower final Mean Squared Error (MSE) and similar computational cost compared to their baselines respectively. Specifically, in the Transformer task, Adam-GOP effectively combines the adaptive scaling of Adam with the global search of GOP, resulting in the most efficient optimization trajectory.

The plots for the normalized L_2-norm and L_∞-norm provide empirical validation of our theoretical claims. In particular, GOP exhibits a rapid increase in parameter norms during the initial iterations compared to the baselines. The ability to adaptively choose the "step-size" and the wide-search strategy in the initial optimization stages allow the model to escape poor local minima near the initialization point, facilitating convergence to a solution with better generalization properties.

In conclusion, GOP proves to be not only theoretically sound but also practically effective for training deep neural networks, offering a robust alternative to SGD and Adam in high-dimensional non-convex settings.

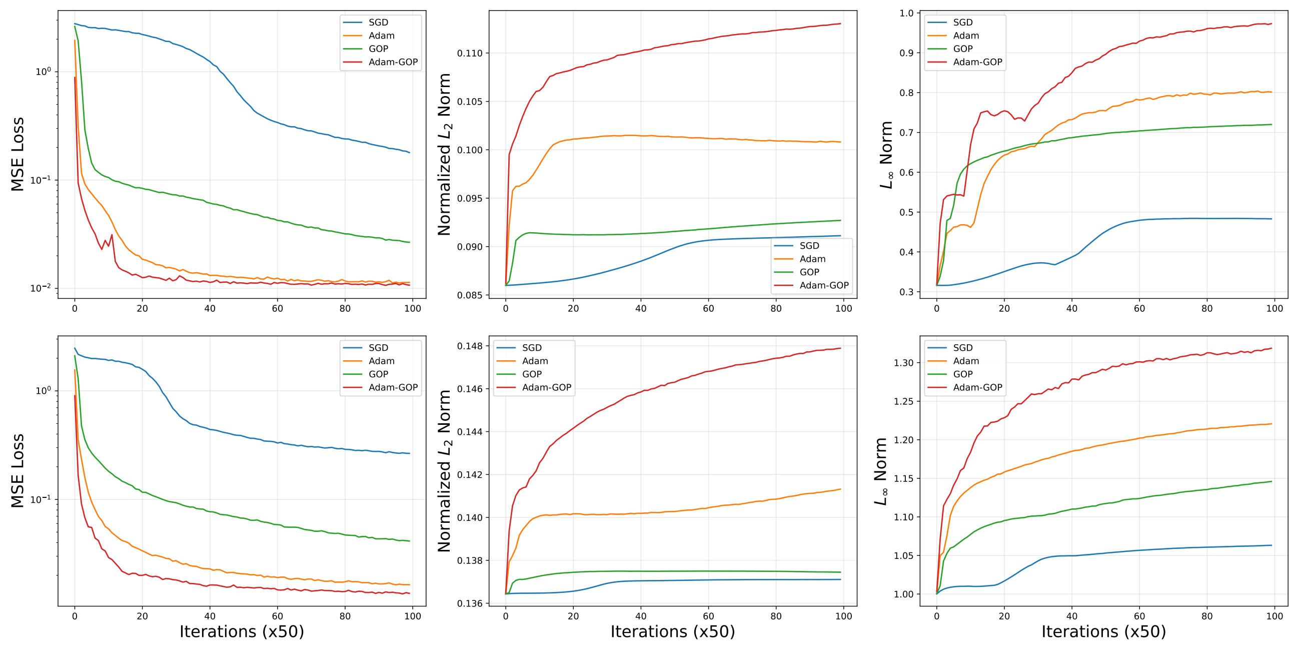

Figure 2. Performance comparison on DeepReLU MLP (first row) and Transformer (second row) in a single run. The first column shows the training loss; the second and third columns track the evolution of parameter norms. GOP variants consistently achieve lower loss. For DeepReLU MLP, the total dimension is $d \sim 1.3 \times 10^4$ and the training costs are $(45, 48, 50, 53)$ seconds respectively. For Transformer, the total dimension is $d \sim 1.7 \times 10^4$ and the training costs are $(124, 175, 175, 191)$ seconds respectively.

5. Conclusion and Discussion

5.1. Summary

In this paper, we introduced Greedy Oriented Progression (GOP), a novel first-order optimization framework designed to overcome the limitations of local search inherent in standard stochastic gradient methods. By leveraging the geometry of the feasible region to determine update steps, GOP naturally facilitates a ”wide-to-narrow” search trajectory. This mechanism allows the algorithm to explore the parameter space globally in early stages and exploit local structures efficiently in later stages.

We provided a rigorous theoretical analysis proving that, under the Local Radial Alignment assumption, GOP converges to the global minimizer with probability 1 as the number of starting points grows to infinity. This is a significant theoretical advancement over traditional SGD and adaptive methods, which typically only guarantee convergence to stationary points (local minima or saddle points). Furthermore, we established an $O(1/n)$ convergence rate, ensuring the method’s efficiency.

Our numerical experiments validated the practical efficacy of GOP. On challenging multi-modal synthetic functions, GOP successfully escaped local optima where baseline methods failed. In high-dimensional deep learning tasks, including DeepReLU MLPs and Transformers, Adam-GOP demonstrated superior performance, achieving faster convergence and lower generalization error compared to state-of-the-art optimizers like Adam and SGD.

5.2. Future Work

Several promising directions for future research remain. First, investigating the theoretical properties of GOP under relaxed assumptions is of interest, particularly in settings where the Local Radial Alignment condition is only partially met. Second, extending GOP to other non-convex domains, such as Reinforcement Learning or online optimization, presents an exciting avenue. Finally, given the method’s success on Transformers, scaling GOP to massive pre-trained language models (LLMs) to further test its scalability and stability is a priority for our future work.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Alvarez, J. A. S. and Calaminici, P. A review of global optimization methods for molecular structures: Algorithms, applications and perspectives. *Journal of Computational*

Chemistry, 46(28):e70243, 2025. doi: https://doi.org/10.1002/jcc.70243. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.70243.

Arnoud, A., Guvenen, F., and Kleineberg, T. Benchmarking global optimizers. Working Paper 26340, National Bureau of Economic Research, October 2019. URL http://www.nber.org/papers/w26340.

Bengio, Y., Simard, P., and Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. doi: 10.1109/72.279181.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. signSGD: compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:7763588.

Blum, J. R. Approximation methods which converge with probability one. *The Annals of Mathematical Statistics*, 25(2):382–386, 1954.

Chen, X., Chen, Z., Gao, W. Y., Yan, X., and Zhang, G. Optimization via the strategic law of large numbers. *Proceedings of the National Academy of Sciences*, 123(4):e2519845123, 2026. doi: 10.1073/pnas.2519845123. URL https://www.pnas.org/doi/abs/10.1073/pnas.2519845123.

Chernozhukov, V., Chetverikov, D., and Kato, K. Inference on causal and structural parameters using many moment inequalities. *The Review of Economic Studies*, 86(5):1867–1900, 11 2018. ISSN 0034-6527. doi: 10.1093/restud/rdy065. URL https://doi.org/10.1093/restud/rdy065.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics (AISTATS)*, 2015.

Dauphin, Y., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization, 2014. URL https://arxiv.org/abs/1406.2572.

Dereich, S., Do, T., Jentzen, A., and von Wurstemberger, P. Adam symmetry theorem: characterization of the convergence of the stochastic adam optimizer, 2025. URL https://arxiv.org/abs/2511.06675.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Frank, M. and Wolfe, P. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

Garrigos, G., Cortild, D., Ketels, L., and Peypouquet, J. Last-iterate complexity of SGD for convex and smooth stochastic problems, 2025. URL https://arxiv.org/abs/2507.14122.

Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle points - online stochastic gradient for tensor decomposition. In *Annual Conference Computational Learning Theory*, 2015. URL https://api.semanticscholar.org/CorpusID:11513606.

Gitman, I., Lang, H., Zhang, P., and Xiao, L. Understanding the role of momentum in stochastic gradient methods. *Advances in Neural Information Processing Systems*, 32:9633–9643, 2019.

Jaggi, M. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning (ICML)*, 2013.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. In *International Conference on Machine Learning (ICML)*, 2017.

Jin, R., Xing, Y., and He, X. On the convergence of mSGD and AdaGrad for stochastic optimization, 2022. URL https://arxiv.org/abs/2201.11204.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

Kleinberg, B., Li, Y., and Yuan, Y. An alternative view: When does SGD escape local minima? In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2698–2707. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/kleinberg18a.html.

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. In *Advances in neural information processing systems (NeurIPS)*, 2018.

Li, J., Lou, Z., Richter, S., and Wu, W. B. Stochastic gradient descent: a nonlinear time series persective, 2024a. Manuscript.

Li, X., Lin, K.-Y., Li, L., Hong, Y., and Chen, J. On faster convergence of scaled sign gradient descent. *IEEE Transactions on Industrial Informatics*, 20(2):1732–1741, 2024b. doi: 10.1109/TII.2023.3280938.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Moulay, E., Léchappé, V., and Plestan, F. Properties of the sign gradient descent algorithms. *Information Sciences*, 492:29–39, 2019. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2019.04.012. URL https://www.sciencedirect.com/science/article/pii/S0020025519303135.

Nesterov, Y. A method of solving the convex programming problem with convergence rate O(1/k^2). *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

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

Pascanu, R., Mikolov, T., and Bengio, Y. On the difficulty of training recurrent neural networks. In Dasgupta, S. and McAllester, D. (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1310–1318, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/pascanu13.html.

Polyak, B. T. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

Reddi, S. J., Kale, S., and Kumar, S. On the convergence of Adam and beyond. In *International Conference on Learning Representations (ICLR)*, 2018.

Robbins, H. and Monro, S. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

Safaryan, M. and Richtarik, P. Stochastic sign descent methods: New algorithms and better theory. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9224–9234. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/safaryan21a.html.

Tan, Y. S. and Vershynin, R. Online stochastic gradient descent with arbitrary initialization solves non-smooth, non-convex phase retrieval. *Journal of Machine Learning Research*, 24(58):1–47, 2023. URL http://jmlr.org/papers/v24/20-902.html.

Tieleman, T. and Hinton, G. 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.

Wang, J.-K., Lin, C.-H., and Abernethy, J. D. Escaping saddle points faster with stochastic momentum. *ArXiv*, abs/2106.02985, 2020. URL https://api.semanticscholar.org/CorpusID:212802994.

Wei, Z., Zhu, W., and Wu, W. B. Weighted averaged stochastic gradient descent: Asymptotic normality and optimality, 2025. URL https://arxiv.org/abs/2307.06915.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *International Conference on Machine Learning (ICML)*, 2011.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. The marginal value of adaptive gradient methods in machine learning. In *Advances in neural information processing systems (NeurIPS)*, 2017.

Yu, L., Balasubramanian, K., Volgushev, S., and Erdogdu, M. A. An analysis of constant step size SGD in the non-convex regime: Asymptotic normality and bias. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 4234–4248. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/21ce689121e39821d07d04faab328370-Paper.pdf.

Zhong, Y., Kuffner, T., and Lahiri, S. Online bootstrap inference with nonconvex stochastic gradient descent estimator, 2023. URL https://arxiv.org/abs/2306.02205.

Zhou, D., Chen, J., Cao, Y., Yang, Z., and Gu, Q. On the convergence of adaptive gradient methods for nonconvex optimization, 2024. URL https://arxiv.org/abs/1808.05671.

Zhu, W., Chen, X., and Wu, W. B. Online covariance matrix estimation in stochastic gradient descent. *Journal of the American Statistical Association*, 118(541):393–404, 2023.

A. Theory and Proofs

A.1. Proof of Theorem 3.6

Proof. Let $\mathcal{N}_\epsilon$ be the smallest ϵ-net of Γ. Since Γ is compact, there exists some constant $L' > 0$ independent of d and ϵ, such that $|\mathcal{N}_\epsilon| \le (\frac{L'}{\epsilon})^d$. Here we choose ϵ small enough such that $\epsilon \le \gamma'$. Therefore, based on the Holder continuity of the gradient noise R, we have

$$\begin{aligned}\sup_{\theta\in\Gamma}\|BR_B(\theta)\|_\infty &\le \sup_{\theta\in\mathcal{N}_\epsilon}\|BR_B(\theta)\|_\infty + \sup_{\|u\|\le\epsilon}\sup_{\theta\in\mathcal{N}_\epsilon} B\|R_B(\theta+u)-R_B(\theta)\|_\infty \\ &\le \max_{1\le j\le p}|\sum_{i=1}^B Y_{ij}| + \sum_{i=1}^B \mathcal{L}'(X_i)\epsilon^{\alpha'},\end{aligned} \tag{13}$$

where $p = d|\mathcal{N}_\epsilon|$ and Y_i is the vectorized version of $(R(\theta, X_i))_{\theta\in\mathcal{N}_\epsilon}$. For the second term, since $\mathcal{L}'(X)$ is a non-negative integrable random variable, based on Markov's inequality, we have

$$P(\frac{1}{B}\sum_{i=1}^B \mathcal{L}'(X) > y) \le B^{1-q}E[|\mathcal{L}'(X)|^q]/y^q. \tag{14}$$

Next we deal with the first term. By using Lemma D.2 and Lemma D.3 in (Chernozhukov et al., 2018), we have that for any $t > 0$,

$$\begin{aligned}&P\left(\max_{1\le j\le p}|\sum_{i=1}^B Y_{ij}| \ge 2K\left(\sigma\sqrt{\log p} + \sqrt{E[Q^2]}\log p\right) + Bt\right) \\ &\le e^{-(Bt)^2/(3\sigma^2)} + K_q BM/(Bt)^q,\end{aligned} \tag{15}$$

where K is a universal constant, K_q is a constant that solely depends on q, $\sigma^2 = \max_j \sum_{i=1}^B E[Y_{ij}^2]$ and $Q = \max_{1\le i\le B}\max_{1\le j\le p}|Y_{ij}|$. Because Assumption 3.4 ensures that $E[\sup_{\theta\in\Gamma}\|R(\theta, X_i)\|_q] < M$, we have $\sigma^2 \le Bc_M$ and $E[Q^2] \le BE[\max_j |Y_{ij}|^2] \le Bc_M$. Here c_M is a constant that only depends on q and M. Therefore, we have

$$\begin{aligned}&P\left[\sup_{\theta\in\Gamma}\|R_B(\theta)\|_\infty \ge 2K\sqrt{Bc_M}\left(\sqrt{\log p} + \log p\right) + t\right] \\ &= P\left[\frac{1}{B}\left(\max_{1\le j\le p}|\sum_{i=1}^B Y_{ij}| + \sum_{i=1}^B \mathcal{L}'(X_i)\epsilon^{\alpha'}\right) \ge 2K\sqrt{Bc_M}\left(\sqrt{\log p} + \log p\right) + t\right] \\ &\le P\left(\max_{1\le j\le p}|\sum_{i=1}^B Y_{ij}| \ge 2K\left(\sigma\sqrt{\log p} + \sqrt{E[Q^2]}\log p\right) + Bt/2\right) + P(\frac{1}{B}\sum_{i=1}^B \mathcal{L}'(X_i)\epsilon^{\alpha'} \ge Bt/2) \\ &\le e^{-(Bt/2)^2/(3\sigma^2)} + 2K_q BM/(Bt/2)^q + BE[|\mathcal{L}'(X)|^q]/(B^2 t\epsilon^{-\alpha'}/2)^q \\ &\le \exp(-Bt^2/(12c_M)) + 2^{q+1}(K_q M + B^{-q}E[|\mathcal{L}'(X)|^q]\epsilon^{q\alpha'})\frac{B^{1-q}}{t^q}.\end{aligned} \tag{16}$$

Note that by choosing $\epsilon = B^{-1/\alpha'}$, we have that $2K\sqrt{B^{-1}c_M}\left(\sqrt{\log p} + \log p\right) \le 4K\sqrt{B^{-1}c_M}(\log d + d\log\mathcal{L}' + \frac{d}{\alpha'}\log B)$. Thus, by choosing $c_1 = 4K\sqrt{c_M}/\alpha'$, $c_2 = 2^{q+1}(K_q M + E[|\mathcal{L}'(X)]^q)$ and $c_3 = 1/(12c_M)$, we get the desired inequality.

Next, note that for any $\theta \in \Gamma$, $\arcsin\left(\frac{2c_1 d^{3/2}\log B}{\epsilon\sqrt{B}} \wedge 1\right) \le y \le \pi/2$ and $\epsilon > 0$, based on the geometry of the triangle

$$\begin{aligned}
&P\big(\sup_{\theta\in\Gamma} \arccos \frac{\nabla G(\theta)^\top (\sum_{i=1}^n \nabla g(\theta, X_i))}{\|\nabla G(\theta)\| \cdot \|\sum_{i=1}^n \nabla g(\theta, X_i)\|} \cdot I\{\|\nabla G(\theta)\| > \epsilon\} > y\big) \\
&\le P(\sup_{\theta\in\Gamma} \|R_B(\theta)\| > \epsilon \sin y) \\
&\le P(\sup_{\theta\in\Gamma} \|R_B(\theta)\|_\infty > \frac{\epsilon}{\sqrt{d}} \sin y) \\
&\le P(\sup_{\theta\in\Gamma} \|R_B(\theta)\|_\infty > c_1 d \log B/\sqrt{B} + \frac{\epsilon}{2\sqrt{d}} \sin y) \\
&\le c_2 \frac{d^{q/2} B^{1-q}}{(\epsilon \sin y)^q} + \exp(-c_3 \epsilon^2 \sin^2 y B/d).
\end{aligned} \tag{17}$$

□

A.2. Proof of Lemma 3.7

Proof. Without loss of generality, we let $\theta = 0$. Consequently, we have for any $n \ge 1$,

$$\langle \theta - \theta_{n-1}, \theta_n - \theta_{n-1}\rangle = \langle -\theta_{n-1}, -\eta_n d_n\rangle = \eta_n \langle \theta_{n-1}, d_n\rangle, \tag{18}$$

where d_n is the noisy gradient unit and η_n is the adaptive step size at iteration t. Since Γ is compact, the step size $\eta_n = \|Z_n - \theta_{n-1}\|/(n+1) \le L/(n+1)$, where L is the maximal diameter of Γ. Moreover, Assumption 3.1 ensures that θ is an interior point. Thus if $\theta_{n-1} \in \mathcal{B}(0, r)$, we also have

$$\eta_n = \|Z_n - \theta_{n-1}\|/(n+1) \ge l/(n+1), \tag{19}$$

where $l = \inf \operatorname{dist}(\partial\Gamma, \mathcal{B}(0,\delta)) > 0$ by choosing r small enough. Next, based on Assumption 3.2, we have if $\theta_{n-1} \in \mathcal{B}(0, \delta)$,

$$\langle \theta_{n-1}, d_n^*\rangle \ge c\|\theta_{n-1}\|, \tag{20}$$

where d_n^* is the true gradient unit at iteration n. Then with Cauchy's inequality, we have

$$\begin{aligned}
E[\eta_n\langle \theta_{n-1}, d_n\rangle | \mathcal{F}_{n-1}] &= E[\eta_n \theta_{n-1}^\top d_n^* | \mathcal{F}_{n-1}] + E[\eta_n \theta_{n-1}^\top (d_n - d_n^*) \mathcal{F}_{n-1}] \\
&\ge \frac{cl}{n+1}\|\theta_{n-1}\| - \sqrt{E[\eta_n^2|\mathcal{F}_{n-1}]}\sqrt{E(\|d_n - d_n^*\|^2 \mid \mathcal{F}_{n-1})}\|\theta_{n-1}\| \\
&\ge \frac{\tilde{c}}{n+1}\|\theta_{n-1}\|,
\end{aligned} \tag{21}$$

where $\tilde{c} = cl - L\sqrt{\sup_{\theta\in\mathcal{B}(0,r), \|\nabla G(\theta)\| \ge \varepsilon} E[\|\frac{\nabla G(\theta)}{\|\nabla G(\theta)\|} - \frac{\sum_{i=1}^B \nabla g(\theta, X_i)}{\|\sum_{i=1}^B \nabla g(\theta, X_i)\|}\|^2]} > 0$ since according to Theorem 3.6, the second term is negligible when the batch size B is large enough and $\|\nabla G(\theta)\| \ge \varepsilon$. Specifically, note that

$$\begin{aligned}
&\sup_{\theta\in\mathcal{B}(0,r), \|\nabla G(\theta)\| \ge \varepsilon} E\left[\|\frac{\nabla G(\theta)}{\|\nabla G(\theta)\|} - \frac{\sum_{i=1}^B \nabla g(\theta, X_i)}{\|\sum_{i=1}^B \nabla g(\theta, X_i)\|}\|^2\right] \\
&\le E[\sup_{\theta\in\Gamma} \|\frac{\nabla G(\theta)}{\|\nabla G(\theta)\|} - \frac{\sum_{i=1}^B \nabla g(\theta, X_i)}{\|\sum_{i=1}^B \nabla g(\theta, X_i)\|}\|^2 I\{\|\nabla G(\theta)\| \ge \epsilon\}] \\
&\le E[\sup_{\theta\in\Gamma} |\omega(\theta) I\{\|\nabla G(\theta)\| \ge \epsilon\}|^2].
\end{aligned} \tag{22}$$

Since the inner part is upper bounded by 2π and $\lim_{B\to\infty} \sup_{\theta\in\Gamma} |\omega(\theta)| I\{\|\nabla G(\theta)\| \ge \epsilon\} = 0$ according to (17), based on Lebesgue's dominant convergence theorem, we have

$$\lim_{B\to\infty} \sup_{\theta\in\mathcal{B}(0,r), \|\nabla G(\theta)\| \ge \varepsilon} E\left[\|\frac{\nabla G(\theta)}{\|\nabla G(\theta)\|} - \frac{\sum_{i=1}^B \nabla g(\theta, X_i)}{\|\sum_{i=1}^B \nabla g(\theta, X_i)\|}\|^2\right] = 0. \tag{23}$$

□

A.3. Proof of Theorem 3.9

Proof. Without loss of generality, we let $\theta = 0$. Let $V_n = \|\theta_n\|^2$, the squared L_2 distance, as the Lyapunov function. Consequently, we have for any $n \geq 1$,

$$V_n = \|\theta_n\|^2 = V_{n-1} - 2\eta_n \langle \theta_{n-1}, d_n \rangle + \eta_n^2, \tag{24}$$

where d_n is the noisy gradient unit and η_n is the adaptive step size at iteration n. Since θ is an interior point and Γ is compact, the step size $\eta_n = \|Z_n - \theta_{n-1}\|/(n+1) \leq L/(n+1)$, where L is the maximal diameter of Γ.

Next, we define the stopping time $\tau = \inf\{n > m : \|\theta_n\| > r\}$ and let $Y_n = \|\theta_{n\wedge\tau}\|^2 + \sum_{k=n\wedge\tau}^{\infty} L^2/(k+2)^2$. Then

$$\begin{aligned} E[Y_{n+1}|\mathcal{F}_n] &= E[\|\theta_{(n+1)\wedge\tau}\|^2|\mathcal{F}_n] + \sum_{k=(n+1)\wedge\tau}^{\infty} \frac{L^2}{(k+2)^2} \\ &\leq Y_n I\{n \geq \tau\} + \left(\|\theta_n\|^2 + \eta_{n+1}^2 + \sum_{k=n+1}^{\infty} \frac{L^2}{(k+2)^2} \right) I\{n < \tau\} \\ &\leq Y_n, \end{aligned} \tag{25}$$

where the first inequality holds since if $n < \tau$, $\theta \in \mathcal{B}(0, r)$ and thus

$$E[\eta_{n+1}\langle \theta_n, d_{n+1}\rangle|\mathcal{F}_n] \geq \frac{c'}{n+2}\|\theta_n\|, \tag{26}$$

as implied by Assumption 3.8. Therefore, Y_n is a non-negative supermartingale with respect to $\mathcal{F}_n$. Then by applying Doob's maximal inequality, we have

$$P(\tau < \infty) = P(\sup_{n>m} \|\theta_{n\wedge\tau}\| > r) \leq P(\sup_{n>m} Y_n > r^2) \leq \frac{E[Y_m]}{r^2} \leq \frac{r_1^2 + \sum_{k=N_0}^{\infty} \frac{L^2}{(k+2)^2}}{r^2} \leq \varepsilon, \tag{27}$$

when $r_1 = \sqrt{\varepsilon} r/2$ and $N_0 = \lceil \frac{2L^2}{r^2\varepsilon} \rceil$. Therefore, the event $\Omega_{r,0}$ occurs with probability at least $1 - \varepsilon$.

Next, given that the event $\Omega_{r,0}$ occurs, we have that Eq. (26) holds according to Assumption 3.8. Thus,

$$E[V_{n+1}|\mathcal{F}_n] \leq V_n - \frac{2c'}{n+2}\|\theta_n\| + \frac{L^2}{(n+2)^2}. \tag{28}$$

Since $\sum_n L^2/(n+2)^2 \leq \infty$, by the Robbins-Siegmund Theorem, we have that $V_n \to V_\infty$ and $\sum_n 2c'\|\theta_n\|/(n+2) < \infty$ almost surely, where V_∞ is a finite random variable. Note that $\sum_n 2c'\|\theta_n\|/(n+2) < \infty$ implies that $\liminf_{n\to\infty} \|\theta_n\| = 0$ since $c' > 0$. Thus, we must have $V_\infty = 0$ almost surely, which implies that $\lim_{n\to\infty} \theta_n = 0$, i.e. the almost surely convergence. □

A.4. Proof of Theorem 3.10

Proof. Without loss of generality, we will take $\theta = 0$. Given $\Omega_{\tilde{r},\theta}$, define $V(\theta) = \|\theta\|^2$. Denote $S_k = \sum_{t=0}^{k} Z_k$, where $Z_0 = \theta_0$. Therefore we have $V(0) = 0$, $S_k = (k+1)\theta_k$. By Taylor's formula,

$$\begin{aligned} E[V(\theta_n)] &= \sum_{k=m+1}^{n} E\left[V(\frac{S_k}{n+1}) - V(\frac{S_{k-1}}{n+1}) \right] + E\left[V(\frac{(m+1)\theta_m}{n+1}) \right] \\ &= \sum_{k=m+1}^{n} E\left[\frac{Z_k}{n+1} \cdot D_\theta V(\frac{S_{k-1}}{n+1}) \right] + \frac{1}{(n+1)^2} \sum_{k=m+1}^{n} E[\|Z_k\|^2] + \frac{(m+1)^2}{(n+1)^2} E[\|\theta_m\|^2] \\ &= \sum_{k=m+1}^{n} \frac{2k}{(n+1)^2} E[\theta_{k-1} \cdot Z_k] + \frac{1}{(n+1)^2} \sum_{k=m+1}^{n} E[\|Z_k\|^2] + \frac{(m+1)^2}{(n+1)^2} E[\|\theta_m\|^2] \\ &\leq \frac{n-m}{(n+1)^2} L^2 + \frac{(m+1)^2}{(n+1)^2} \tilde{r}^2 = O(\frac{1}{n}). \end{aligned} \tag{29}$$

The last inequality holds because given $\Omega_{\tilde{r},\theta}$, $\|Z_k\| \leq L$, $\|\theta_m\| \leq \tilde{r}$ and for all $k \geq m+1$

$$E[\theta_{k-1}^\top Z_k] = E[\theta_{k-1}^\top(\theta_{k-1} - (k+1)\eta_k d_k)|\mathcal{F}_{k-1}] \leq \|\theta_{k-1}\|^2 - c'\|\theta_{k-1}\| \leq 0, \tag{30}$$

due to Assumption 3.8 and that $\|\theta_{k-1}\| \leq \tilde{r} \leq c'$.

□

B. Details of Simulation Study

B.1. Practical Implementation of the GOP Algorithm

In this subsection, we describe in more details how we practically implement Algorithm 1 in R and Python, and introduce its variant, namely ADAM-GOP, which we use for the numerical experiments in Section 4. Algorithm 2 details the practical implementation of GOP. ADAM-GOP has the same updating mechanism as GOP, except for the progression direction, which is the same as ADAM instead of the vanilla gradient.

Algorithm 2 Practical Greedy Oriented Progression

```
1: Input: Loss function $g(\theta; X)$, identically distributed independent data points $(X_1, \dots, X_N)$, batch size $B$, convex
   feasible region $\Gamma$, maximum number of iterations $M$, initial point $\theta_0$, starting index $m$ and buffer parameter $\delta$.
2: for $n = 1$ to $K = \min(\lfloor cM \rfloor, \lfloor cN/B \rfloor)$ do
3:     Calculate the noisy gradient vector direction $d_n = \sum_{i=(n-1)B+1}^{nB} \nabla g(\theta_{n-1}, X_i)/\|\sum_{i=(n-1)B+1}^{nB} \nabla g(\theta_{n-1}, X_i)\|$.
4:     Calculate the intersection time $t_n = \inf\{t : \theta_{n-1} - td_n \in \partial\Gamma, t \geq 0\}$.
5:     Draw the perturbed intersection point $Z_n = \theta_{n-1} - (1+\xi_n)t_n d_n, t \geq 0$, where $P(|\xi_n| \leq \delta) = 1$.
6:     Update $\theta_n = \theta_{n-1} + \frac{1}{n+m}(Z_n - \theta_{n-1})$.
7:     if $\theta_n \notin \Gamma$ then
8:         Update $\theta_n \in \{\theta_{n-1} + t(Z_n - \theta_{n-1}), t > 0\} \cap \partial\Gamma$.
9:     end if
10: end for
```

Motivated by Theorem 3.9, we introduce the starting index m for a more local search. Specifically, given any starting point θ_0, it initializes the running sum as $S_0 = m\theta_0$, computes the first update as $(S_0 + Z_1)/(m+1)$, and so on and so forth. We also introduce extra randomness, represented by ξ_n to enlarge the original region so that the algorithm does not get stuck if the minimum is on the boundary. The computation of the intersection time t_n is simply a line search, whose computational cost is negligible as long as the region Γ is convex.

It is worth noting that the parameter m naturally enables us to fairly compare our GOP with baselines. For example, with a small m, the "step size" of GOP behaves similarly to a large fixed initial learning rate with polynomial decay, while a large m makes it almost constant. Moreover, with minor modifications, all of our theoretical results in Section 3.2 and Section 3.3 will also hold for this practical GOP, since our analysis is robust to the initial point and enlargement of the boundary.

Moreover, the GOP algorithm can be further improved if we use a refined version called GOP-R. First, GOP-R records and exploits the running maximum of all test function evaluations: This is purely bookkeeping and does not add any additional evaluations of the test functions beyond those for finite-difference calculations and convergence checks. From a programming perspective, the running maximum generated in past iterations is "free information" that requires essentially no additional computation and memory storage. Second, GOP-R adopts a two-stage procedure: it starts by running GOP for at most 50% iterations of the designated maximum number of iterations, and then runs the rest 50% iterations with a more local search (and smaller step sizes). Other hyperparameters are set to be the same as in GOP. The popular Cosine Annealing learning rate scheduler for SGD is a natural baseline for GOP-R. We leave the discussion and experiments about GOP-R for future work.

B.2. Discussion on Choice of Hyperparameters

We now describe the default choice of hyperparameters for our GOP algorithms. The batch size B is expected to be chose reasonably large, as suggested in most modern stochastic optimization settings. For the random variables ξ_n and the buffer parameter δ introduced in Algorithm 2, we set $\xi_n \sim \text{Unif}(-\delta^*, \delta^*)$ with default $\delta^* = 0.05$, i.e., 5% of the length of the domain.

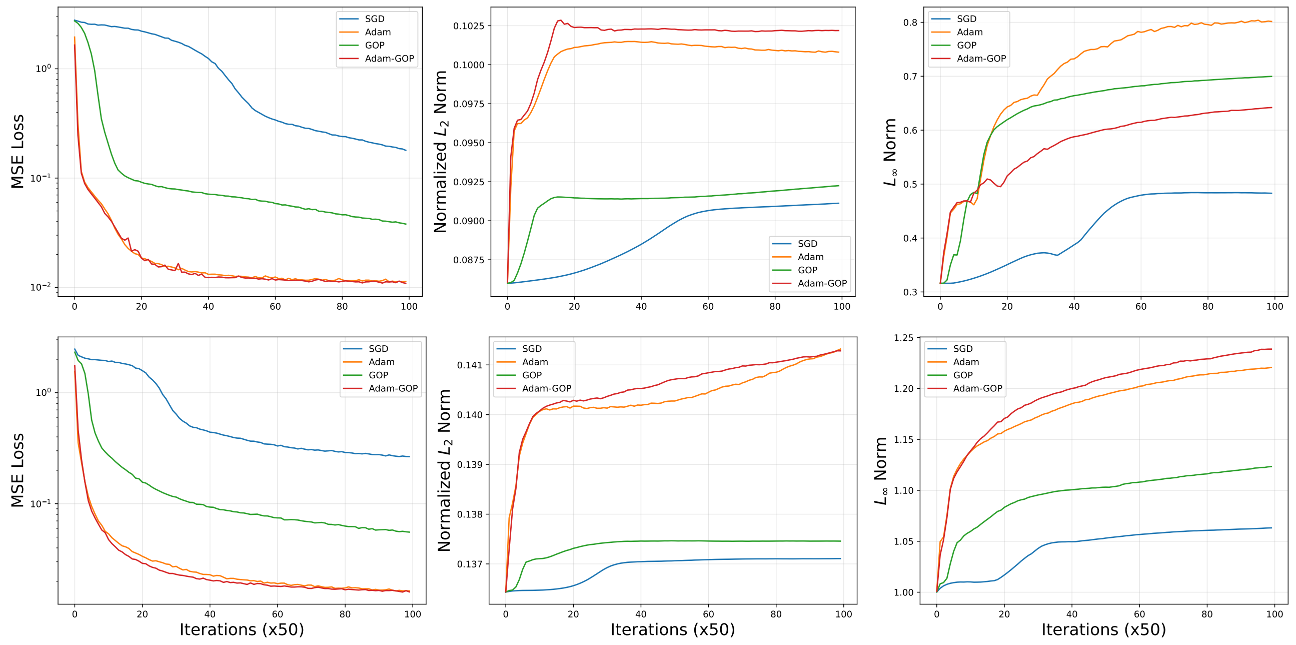

Figure 3. Performance comparison on DeepReLU MLP (first row) and Transformer (second row) in a single run. The first column shows the training loss; the second and third columns track the evolution of parameter norms. GOP variants consistently achieve lower loss. For DeepReLU MLP, the total dimension is $d \sim 1.3 \times 10^4$ and the training costs are $(46, 49, 50, 52)$ seconds respectively. For Transformer, the total dimension is $d \sim 1.7 \times 10^4$ and the training costs are $(103, 119, 117, 125)$ seconds respectively.

For the choice of region Γ, if the original problem is a convexly-constrained optimization problem, the region defined by the constraints is a natural choice. For general unconstrained optimization problems, hyper-cube centered at the initial point is a recommended option. In particular, if the user has prior information or belief about the upper bound of L_∞ norm of the target minimum, i.e. possible largest absolute value of the coordinates, then Γ can be set slightly larger than that domain.

For example, if the initial point is a warm start, a natural choice of Γ is a small hyper-cube around the warm start. This is a very common case in Reinforcement Learning, where the user is often asked to fine-tune a pre-trained model. A more practical but less efficient way is to first run the initial few steps of standard unconstrained optimizers like SGD and track the evolution of L_∞ norm in the early stage. In this case, Γ can be chosen as a hyper-cube which contains the trajectory. Moreover, our simulation results show that the performance of GOP is robust to the choice of the edge length of hyper-cube Γ.

B.3. Details on the Simulation in Section 4.1

As discussed in Section 4.1, in each step the stochastic optimization algorithms receive a corrupted gradient $\nabla g(\theta, \epsilon_n) = \nabla G(\theta) + \sigma\epsilon_n$, where $\epsilon_n \sim \mathcal{N}(0, I_d)$ represents the noise at step n and σ denotes the noise level. In particular, we choose $\sigma = 1$ to model the moderately strong noise case. We repeat 100 times with varying starting points drawn from Sobol' sequence. The random seed used is 123. For SGD and ADAM, they solve an equivalent Lagrangian problem $\min_\theta G(\theta) + \lambda^\top h(\theta)$, where the regularizer λ is chosen as 100 and $h(\theta)$ is the function of constraints. This penalty method does not unfairly disadvantage the baselines since the constraints are always inactive during the whole experiments.

We compare the following 4 algorithms.

- SGD: Stochastic Gradient Descent by (Robbins & Monro, 1951). We implement this algorithm in R with learning rate 0.01 and max number of iterations $5,000$.
- ADAM: Adaptive Moment Estimation method by (Kingma & Ba, 2014). We implement this algorithm in R with learning rate 0.01, max number of iterations $5,000$ and ADAM parameters $(\beta_1, \beta_2, \varepsilon) = (0.9, 0.999, 10^{-8})$.
- GOP: Greedy Oriented Progression as in Algorithm 2. The tuning parameters are set as $m = 0$ and $\delta = 0.05$ with max

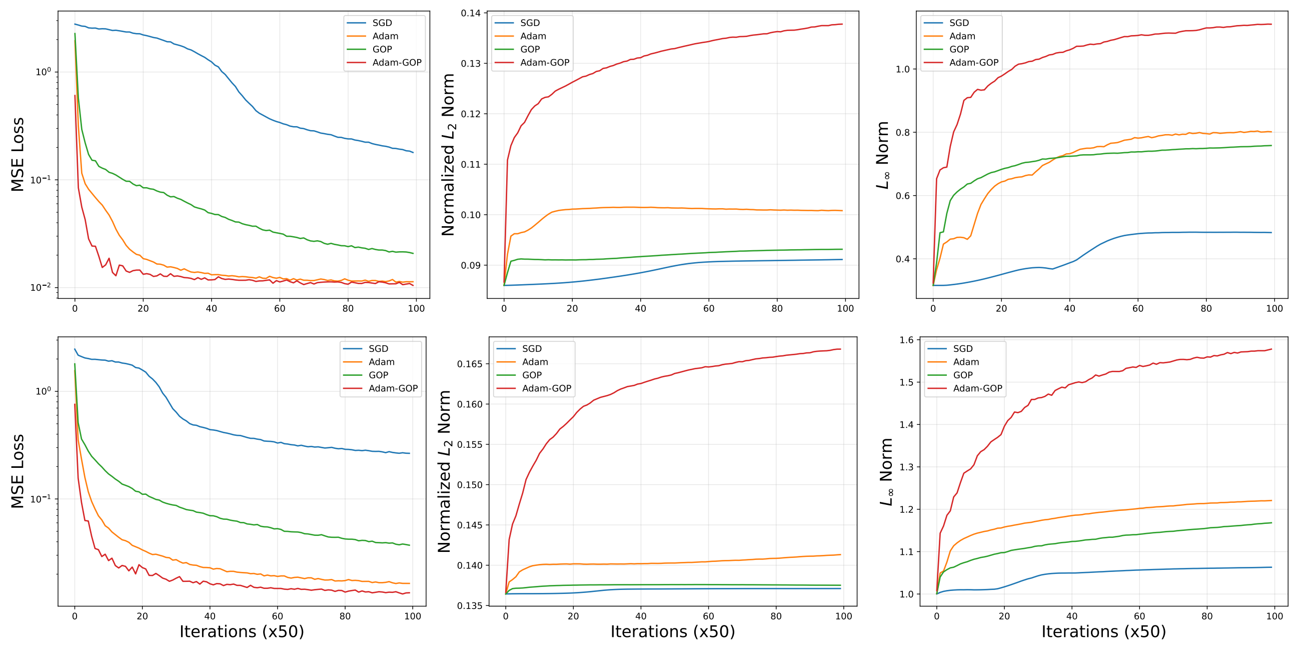

Figure 4. Performance comparison on DeepReLU MLP (first row) and Transformer (second row) in a single run. The first column shows the training loss; the second and third columns track the evolution of parameter norms. GOP variants consistently achieve lower loss. For DeepReLU MLP, the total dimension is $d \sim 1.3 \times 10^4$ and the training costs are $(46, 49, 51, 55)$ seconds respectively. For Transformer, the total dimension is $d \sim 1.7 \times 10^4$ and the training costs are $(109, 182, 178, 189)$ seconds respectively.

number of iterations $5,000$.

- ADAM-GOP: Greedy Oriented Progression with ADAM direction. The parameters are chosen the same as GOP with ADAM parameters $(\beta_1, \beta_2, \varepsilon) = (0.9, 0.999, 10^{-8})$.

The regret is defined as the average of absolute differences between global minimum/maximum value and the output given by the algorithm. The distance is defined as the average of normalized Euclidean distance between the global minimizer/maximizer and the point found by the algorithm.

B.4. Details on the Simulation in Section 4.2

The synthetic data are generated as $y = \sin x_1 + \cos x_2 + x_3 x_4 + 0.1\mathcal{N}(0, 1)$ with $x \sim 4\mathcal{N}(0, I_{10}) - 2$ The models include **DeepReLU MLP**: A n-layer fully connected network with d hidden units, and **Transformer**: An encoder-only Transformer with n layers, m heads, and model dimension d. We let $n \in \{2, 4, 8\}$, $m = 4$ and $d \in \{32, 64, 128, 256, 512\}$. Due to limit of computational resources, larger models are not considered.

All models are trained for 5 epochs with default initial point, full sample size $N = 10^6$, batch size $B = 1,000$ and the same random seed 123. The boundaries are set as $\{\theta : \|\theta\|_\infty \leq \lambda\}$, where $\lambda \in \{2, 5, 10\}$, since the initial point of the Transformer model satisfies $\|\theta_0\|_\infty = 1$. Because the constraints are always inactive during the whole experiments, SGD and ADAM can be used for unconstrained problems without sacrifice of fairness.

We compare the following 4 algorithms.

- SGD: Stochastic Gradient Descent by (Robbins & Monro, 1951). We implement this algorithm in Python with learning rate 0.001.
- ADAM: Adaptive Moment Estimation method by (Kingma & Ba, 2014). We implement this algorithm in Python with learning rate 0.001 and ADAM parameters $(\beta_1, \beta_2, \varepsilon) = (0.9, 0.999, 10^{-8})$.
- GOP: Greedy Oriented Progression as in Algorithm 2. The tuning parameters are set as $m = 1,000$ and $\delta = 0.05$.

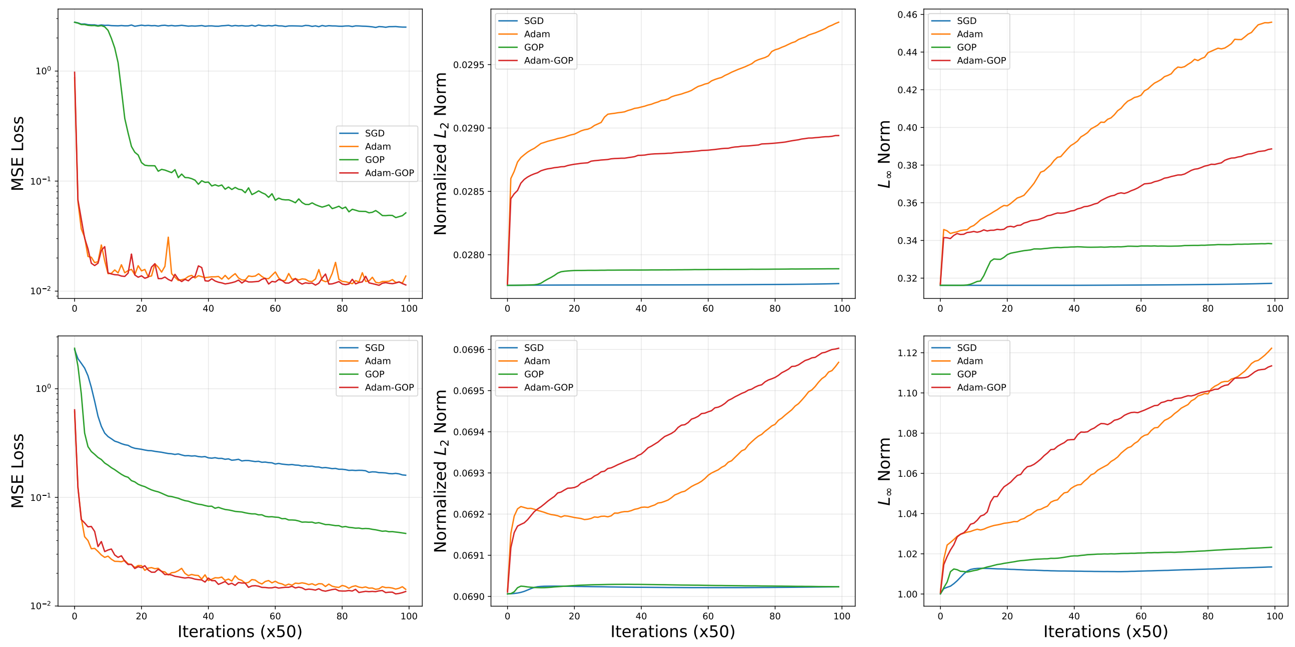

Figure 5. Performance comparison on DeepReLU MLP (first row) and Transformer (second row) in a single run. The first column shows the training loss; the second and third columns track the evolution of parameter norms. GOP variants consistently achieve lower loss. For DeepReLU MLP, the total dimension is $d \sim 2 \times 10^6$ and the training costs are $(277, 379, 410, 519)$ seconds respectively. For Transformer, the total dimension is $d \sim 5 \times 10^5$ and the training costs are $(483, 532, 550, 613)$ seconds respectively.

- ADAM-GOP: Greedy Oriented Progression with ADAM direction. The parameters are chosen the same as GOP with ADAM parameters $(\beta_1, \beta_2, \varepsilon) = (0.9, 0.999, 10^{-8})$.

C. Supplemental Numerical Results

In this section, we include supplemental numerical results for the settings described in Section B. Specifically, Figure 3 and Figure 4 show the experimental results for $\lambda = 2$ and $\lambda = 10$ respectively. The models include **DeepReLU MLP**: A 4-layer fully connected network with 64 hidden units (approx. 1.3×10^4 parameters), and **Transformer**: An encoder-only Transformer with 2 layers, 4 heads, and model dimension 32 (approx. 1.7×10^4 parameters). Clearly, in all settings, our proposed algorithms GOP and ADAM-GOP beat their baselines respectively. Based on these results, we empirically show that the performance of our algorithm is robust to the choice of the edge length of hyper-cube Γ.

Next, Figure 5 shows the performance for optimizing higher dimension models. The models include **DeepReLU MLP**: A 8-layer fully connected network with 512 hidden units (approx. 2×10^6 parameters), and **Transformer**: An encoder-only Transformer with 4 layers, 4 heads, and model dimension 128 (approx. 5×10^5 parameters). In this high-dimensional setting, ADAM-GOP slightly beats ADAM in terms of the final loss, while SGD loses to GOP by a large margin. This result shows that even in the case where the dimension is higher than the sample size, our proposed algorithms still win.