

Journal of Mathematical Extension
Vol. 20, No. 4 (2026) (2) 1-24
ISSN: 1735-8299
URL: <http://doi.org/10.30495/JME.2026.3595>
Original Research Paper

Unified Dynamic Initialization Framework for Theoretical and Practical Convergence in Non-Convex Optimization

T. Derikvand*

Islamic Azad University, Marvdasht Branch

M. Moattari

Islamic Azad University, Marvdasht Branch

Abstract. Gradient Descent (GD) is a widely used first-order iterative optimization algorithm, known for its efficiency in minimizing convex objective functions. However, in real-world applications—particularly in artificial intelligence—optimization often involves non-convex loss functions, where GD can struggle with local minima and convergence issues. In this article, we examine the limitations of traditional GD in non-convex settings and introduce a novel strategy: dynamically updating the algorithm's starting point at each iteration. This adaptive approach enhances the algorithm's ability to navigate complex optimization landscapes, thereby significantly broadening the practical applicability of GD to a wider range of challenging problems.

AMS Subject Classification: 90C30; 49M37.

Keywords and Phrases: Non-convex optimization; gradient descent; adaptive optimization methods; convergence analysis; optimization in machine learning.

Received: January 2026; Accepted: March 2026

*Corresponding Author

1 Introduction

Gradient Descent (GD) is a fundamental iterative optimization algorithm widely used to locate optimal points in a variety of problems. In the context of machine learning, it is particularly prominent for tuning model parameters such as coefficients in regression and weights in neural networks [2, 5, 6, 12]. The objective is typically to minimize a loss function, thereby reducing the error of predictions.

GD is valued for its simplicity and efficiency, especially when applied to smooth, well-conditioned, and strongly convex objective functions. Each iteration is computationally inexpensive, requiring only the gradient of the objective function. However, in practical applications, many problems are non-convex, which poses significant challenges. Notably, GD cannot handle non-differentiable functions, and it is often slow in poorly conditioned scenarios.

In convex settings, GD guarantees convergence to a global minimum. However, most real-world functions encountered in artificial intelligence and machine learning are non-convex [2, 4, 10, 14], which introduces several complications:

- **Local Minima:** GD may converge to a local minimum instead of the global minimum, particularly when the initialization is close to a suboptimal point.
- **Saddle Points:** Non-convex functions frequently contain saddle points, where the gradient vanishes, yet these points are neither minima nor maxima. GD can become trapped at such points.
- **Slow Convergence:** In highly non-convex or irregular landscapes, convergence may be slow, and the algorithm may exhibit unstable behavior.

To overcome these issues, numerous modifications to the standard GD algorithm have been proposed:

- **Momentum-Based GD:** Incorporates a momentum term to dampen oscillations and accelerate convergence, especially in regions with sharp curvature.

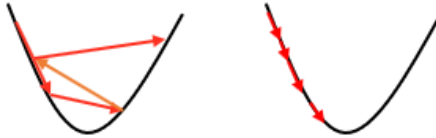


Figure 1: (Left) Overshooting due to large learning rate; (Right) Slow convergence with a suitable learning rate

- **Stochastic Gradient Descent (SGD):** Approximates the gradient using a subset of data, introducing stochasticity that can help escape local minima.
- **Adaptive Methods (e.g., Adam, AdaGrad, RMSprop):** Dynamically adjust the learning rate during training, enhancing the ability to navigate complex loss surfaces.
- **Global Optimization Techniques:** Algorithms such as simulated annealing and genetic algorithms are designed to explore the search space more broadly and avoid local traps [3, 13].

In a typical machine learning setting, the goal is to minimize a loss function $L(W)$, parameterized by the model’s parameters $W \in \mathbb{R}^d$. GD iteratively updates the parameters in the direction opposite to the gradient $\nabla L(W)$:

$$W_{n+1} = W_n - \gamma \nabla L(W_n), \quad (1)$$

where γ is the learning rate, and the minus sign indicates movement in the direction of steepest descent. The learning rate controls the step size and hence directly influences convergence behavior. If γ is too large, the algorithm may overshoot minima and diverge. Conversely, if it is too small, convergence may be unacceptably slow. Figure 1 illustrates the impact of learning rate selection.

In practice, selecting an appropriate learning rate is often done via trial and error or through predefined learning rate schedules [3]. However, these schedules must be manually defined in advance and are not always adaptive to the problem or dataset. Moreover, standard GD uses a single learning rate for all parameters, which can be suboptimal.

Convergence in GD is typically assessed by monitoring the change in the loss function. If the cost plateaus over iterations, the algorithm is said to have converged. While some algorithms provide automatic convergence detection based on predefined thresholds, these thresholds can be difficult to estimate. In most applications, convergence is visually assessed via learning curves. Oscillations or rising trends in the cost curve often signal a learning rate that is too high.

Another persistent difficulty in minimizing non-convex loss functions—common in deep learning—is avoiding poor local minima and saddle points. While theoretical results exist that guarantee convergence of GD under certain conditions (fixed or adaptive step sizes), these are often not sufficient in high-dimensional, real-world applications. Many recent methods focus on refining the learning rate mechanism, including Momentum, Nesterov acceleration, AdaGrad, Adadelta, RMSprop, Adam, AdaMax, Nadam, and AMSGrad.

Second-order methods, such as Newton’s method, offer faster convergence by using curvature information but are computationally infeasible for large-scale problems due to their complexity in high-dimensional spaces.

In modern machine learning applications, non-convex optimization has become ubiquitous. Many algorithms impose structural constraints (e.g., sparsity, low-rank structures) or directly define the objective as a non-convex function. This is particularly true for non-linear models in high-dimensional spaces, such as deep neural networks and tensor models. While this modeling flexibility is powerful, it also increases computational complexity and often leads to NP-hard optimization problems. A common workaround is to relax these non-convex problems into convex ones, but such relaxations may be lossy and inefficient in large-scale settings.

For a comprehensive review of recent advances in non-convex optimization for machine learning, see [4, 8, 9, 11].

2 Proposed Method and Main Results

A function $f : \mathcal{W} \rightarrow \mathbb{R}$ is called *convex* if for any two points $W_1, W_2 \in \mathcal{W}$ and any $\lambda \in [0, 1]$, the following inequality holds:

$$f(\lambda W_1 + (1 - \lambda)W_2) \leq \lambda f(W_1) + (1 - \lambda)f(W_2). \quad (2)$$

This property implies that any local minimum of a convex function is also a global minimum, simplifying optimization tasks. Many cost functions in data analysis, often involving norms, exhibit convexity and thus benefit from efficient convex optimization techniques. However, a significant challenge arises in practice as many real-world problems are governed by *non-convex* objective functions, which may possess multiple local minima and saddle points, rendering optimization considerably more difficult.

To address the challenges posed by non-convexity, this paper introduces a **novel variant** of the standard *Gradient Descent (GD)* algorithm. Our proposed method augments GD by incorporating a **dynamic update mechanism for the search point** at each iteration. This enhancement aims to facilitate the algorithm's ability to navigate complex, multi-modal landscapes by potentially escaping local minima and saddle points.

Let $f(W)$ denote the objective function to be minimized, where $W \in \mathbb{R}^d$ is the parameter vector. The proposed algorithm integrates a *root-finding procedure* as a subroutine. This procedure is designed to identify a new search point based on the current state, thereby introducing an exploratory component intended to improve the algorithm's performance on non-convex functions.

2.1 The proposed algorithm

Input:

- Objective function $f : \mathbb{R}^d \rightarrow \mathbb{R}$.
- Learning rate $\gamma > 0$.
- Initial point $W_0 \in \mathbb{R}^d$.

- Convergence tolerance $\varepsilon > 0$.

Algorithm Steps:

1. **Compute Gradient:** Evaluate the gradient of the objective function at the current point W_n :

$$\nabla f(W_n)$$

2. **Perform Gradient Descent Step:** Update the parameter vector according to the standard GD rule:

$$W'_{n+1} = W_n - \gamma \nabla f(W_n)$$

(Note: Using W' here to distinguish from the final W_{n+1} after the root-finding step.)

3. **Explore for Potentially Better Points:** Apply a procedure to find a new candidate point W_{n+1} . This procedure involves solving an equation derived from the current function value. Specifically, we seek a point W such that $f(W) = f(W_n)$. From the set of solutions $S = \{W \mid f(W) = f(W_n), W \neq W_n\}$, we select a point W_{n+1} based on a defined criterion (e.g., a point that promotes escape from local optima, or the solution with the smallest norm, or another heuristic). If no such point is found or if W_n is deemed a satisfactory minimum, the algorithm may terminate or proceed differently based on the specific implementation of this step.
4. **Update Search Point:** Set the new search point for the next iteration:

$$W_{n+1} = W'_{n+1} \quad (\text{or modified based on root-finding})$$

5. **Repeat:** Return to Step (1) and continue the process until the convergence condition is met:

$$\|W_{n+1} - W_n\|_2 \leq \varepsilon$$

6. **Output:** The final value $W^* = W_n$ is taken as the *approximate minimum* found by the algorithm.

2.2 Key features and considerations of the proposed algorithm

- **Dynamic Search Point Adjustment:** Unlike traditional GD, which deterministically follows a path dictated by gradients, the proposed method incorporates a mechanism to reset or adjust the search point. This is achieved by exploring points related to the current state, aiming to facilitate escape from suboptimal regions such as local minima and saddle points.
- **Root-Finding Based Exploration Strategy:** The core of the exploration lies in solving $f(W) = f(W_n)$ for $W \neq W_n$. The specific strategy for selecting the next point W_{n+1} from the set of solutions is critical. If the goal is to reach a global minimum, this step must effectively guide the search towards regions of lower function values, potentially by selecting a root that offers a significant change in the search direction or landscape exploration. The precise definition and implementation of this selection criterion are essential for the algorithm's efficacy and theoretical justification.
- **Convergence Criterion:** The stopping condition $\|W_{n+1} - W_n\|_2 \leq \varepsilon$ ensures that the algorithm terminates when the updates to the parameter vector become negligible, indicating that the search has stabilized. This is a standard criterion for assessing convergence in iterative optimization methods.

2.2.1 Example 1: classical gradient descent

When applying the standard Gradient Descent algorithm with a learning rate of $\gamma = 0.025$, initial point $W_0 = 0$, and convergence tolerance $\varepsilon = 0.0001$, the algorithm converges to a local minimum located at:

$$W_{\text{local}}^* = 1.8380 \tag{3}$$

Figure 2 illustrates this behavior, showing GD getting trapped in a sub-optimal solution.

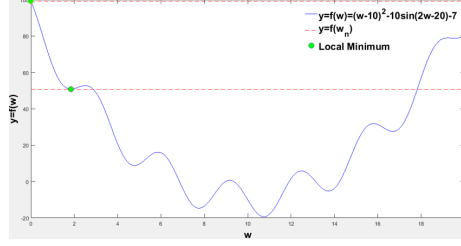


Figure 2: Gradient Descent trapped in a local minimum.

2.2.2 Example 2: proposed method

In contrast, applying the proposed algorithm to the same objective function and with the identical parameters ($\gamma = 0.025$, $W_0 = 0$, $\varepsilon = 0.0001$), the algorithm successfully avoids the local minimum and converges to the global minimum after approximately 50 iterations. Figures 3 and 4 illustrate the convergence process.

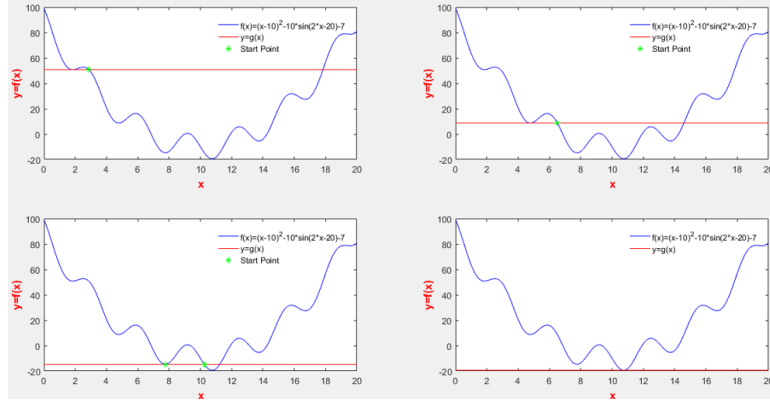


Figure 3: Step-by-step convergence of the proposed method toward the global minimum. The horizontal axis represents the parameter W , and the vertical axis displays the corresponding cost $f(W)$.

The learning curve depicted in Figure 3 shows steady and consistent progress towards the minimum. This empirical evidence strongly suggests that the dynamic updating strategy, as implemented in our method, substantially enhances Gradient Descent's ability to escape lo-

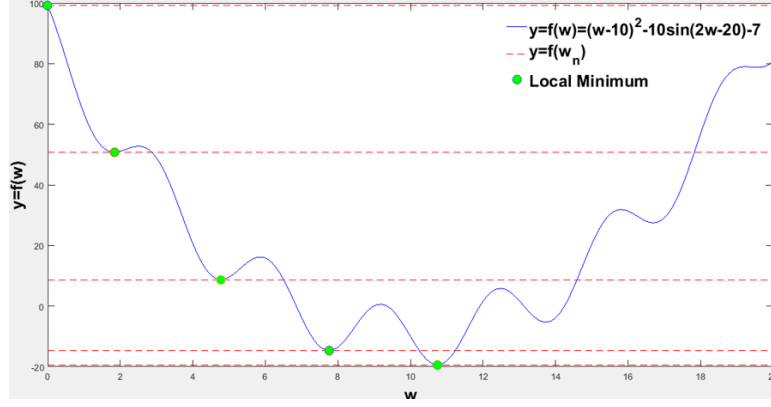


Figure 4: The algorithm successfully bypasses local minima and saddle points during optimization.

cal traps and converge to significantly better solutions compared to the standard approach. The visualization in Figure 4 further supports the claim that the algorithm effectively navigates and bypasses challenging regions of the objective function landscape.

2.3 Convergence analysis

This section provides a theoretical analysis of the convergence properties of our proposed gradient descent variant with dynamic starting point updating. We present theoretical results suggesting that, under specific assumptions, the algorithm converges to a global minimum of non-convex objective functions. Furthermore, we discuss the rate of convergence and the method's robustness in handling various forms of non-convexity.

2.3.1 Convergence to global minimum

The standard Gradient Descent (GD) algorithm is guaranteed to converge to a global minimum when applied to convex functions. However, for non-convex objective functions, GD is susceptible to converging to local minima or getting stuck at saddle points. Our proposed method aims to mitigate this limitation by dynamically updating the search point at

each iteration through a root-finding procedure. This mechanism is intended to promote exploration of the search space and facilitate escape from suboptimal regions.

Let $f(W)$ be the objective function defined on a convex domain $\mathcal{W} \subset \mathbb{R}^d$. We assume f is continuously differentiable and its gradient $\nabla f(W)$ is Lipschitz continuous with constant L . The standard GD update rule is:

$$W_{n+1}^{\text{std}} = W_n - \gamma \nabla f(W_n), \quad (4)$$

where $\gamma > 0$ is the step size.

Our proposed method modifies this by incorporating a step that identifies a new starting point W_n^{new} based on the current state W_n . This new point is obtained via a root-finding procedure, as detailed in Section 2, which aims to explore the function landscape more broadly. The subsequent gradient descent step would then typically originate from this new point.

Theoretical Guarantees and Assumptions We outline the assumptions under which we can establish convergence properties for the proposed method.

1. **Differentiability and Lipschitz Gradient:** The objective function $f(W)$ is continuously differentiable on $\mathcal{W}$, and its gradient $\nabla f(W)$ is Lipschitz continuous with constant L :

$$\|\nabla f(W) - \nabla f(W')\| \leq L\|W - W'\|, \quad \forall W, W' \in \mathcal{W}. \quad (5)$$

2. **Step Size:** The step size γ is chosen appropriately. For standard GD, $0 < \gamma \leq 1/L$ ensures convergence. For our method, the step size γ used in the gradient descent update (if distinct from the exploration step) should be suitably chosen.
3. **Existence of Global Minimum:** The objective function $f(W)$ possesses at least one global minimum $W^* \in \mathcal{W}$, such that $f(W^*) < f(W)$ for all $W \in \mathcal{W} \setminus \{W^*\}$.
4. **Effectiveness of Root-Finding Step:** The root-finding procedure, as described in Section 2, reliably identifies a new search

point W_{n+1} that promotes escape from local minima and saddle points. Specifically, it must ensure that the sequence of points generated does not permanently stagnate in suboptimal regions. This implies that the root-finding step, possibly combined with the subsequent GD step, effectively moves the iterate towards regions of lower function values or explores the landscape in a way that eventually leads to the global minimum.

Convergence Statement: Under the aforementioned assumptions, and particularly contingent upon the effectiveness of the dynamic starting point update in facilitating exploration and avoiding stagnation, the sequence of iterates $\{W_n\}$ generated by the proposed algorithm is expected to converge to the global minimum W^* . Formally,

$$\lim_{n \rightarrow \infty} W_n = W^*. \quad (6)$$

Discussion on Theoretical Guarantee: The core of the theoretical guarantee for global convergence hinges on the assumption that the root-finding step, coupled with the subsequent gradient descent updates, effectively prevents the algorithm from permanently converging to a local minimum or saddle point. This allows the exploration mechanism to eventually guide the iterates towards the global optimum. A rigorous proof would require a detailed analysis of the properties of the root-finding procedure and its interplay with the gradient descent steps, potentially showing that the algorithm escapes any compact set containing only local minima and saddle points, thus ensuring convergence to the global minimum. The rate of convergence would depend on the specific properties of $f(W)$ and the efficiency of the root-finding step.

2.3.2 Convergence analysis and practical interpretation

The standard Gradient Descent (GD) algorithm is well-established to converge to a global minimizer under the assumption of function convexity. However, when applied to general non-convex objective functions, GD's convergence behavior becomes more complex. It may converge to any stationary point (local minimum, local maximum, or saddle point),

and its progress can become notably slow in flat regions or when navigating around undesirable local minima or saddle points.

In this work, we focus on a real-valued objective function

$$f : \mathcal{W} \rightarrow \mathbb{R}, \quad (7)$$

where $\mathcal{W} \subset \mathbb{R}^d$ is a convex domain. We assume f is continuously differentiable, and its gradient is denoted by ∇f . The classical gradient descent update rule is defined as:

$$W_{n+1} = W_n - \gamma \nabla f(W_n), \quad (8)$$

where $\gamma > 0$ represents the step size.

Our proposed method incorporates a dynamic update of the starting point, executed via a root-finding procedure as detailed in Section 2. This restart mechanism is primarily conceptualized as a heuristic strategy. Its aim is to enhance the practical exploration of the search space and to assist the algorithm in escaping unfavorable stationary regions. Crucially, within the scope of this particular analysis, we treat this update as an empirical enhancement rather than a component with a guaranteed theoretical convergence property on its own. Consequently, we refrain from making a claim of guaranteed global convergence for the *combined* method without additional, stringent assumptions on the root-finding procedure itself.

Theoretical Guarantee for the Gradient Descent Phase The gradient-descent phase of our algorithm, when isolated and considered under standard smoothness assumptions, admits well-understood convergence properties. Specifically, we assume the following conditions hold:

1. **Smoothness and Bounded Gradient:** The objective function $f(W)$ is continuously differentiable on $\mathcal{W} \subset \mathbb{R}^d$, and its gradient $\nabla f(W)$ is Lipschitz continuous with constant $L > 0$. This means:

$$\|\nabla f(W) - \nabla f(W')\| \leq L\|W - W'\|, \quad \forall W, W' \in \mathcal{W}. \quad (9)$$

This assumption is fundamental for bounding the change in the gradient.

2. **Appropriate Step Size:** The step size γ is selected such that:

$$0 < \gamma < \frac{2}{L}. \quad (10)$$

This condition ensures that the gradient descent steps provably reduce the function value.

3. **Bounded Level Set:** The sequence of iterates $\{W_n\}$ generated by the gradient-descent phase remains within a bounded level set of f . This implies that the function values do not diverge to infinity and that the iterates stay within a region where the function is "well-behaved" around stationary points.

Under these assumptions, the standard gradient descent update rule ($W_{n+1} = W_n - \gamma \nabla f(W_n)$) generates a descent sequence. Furthermore, any accumulation point of the iterate sequence $\{W_n\}$ is guaranteed to be a stationary point of f . If the objective function f is convex, then any stationary point is indeed a global minimizer. However, for general non-convex functions, convergence to the *global* minimum cannot be guaranteed solely based on these assumptions, as GD might converge to a local minimum or saddle point.

Practical Interpretation of the Dynamic Update The dynamic starting-point update is thus interpreted as a practical, heuristic enhancement. Its primary role is to modify the landscape of the optimization process by changing the starting point for subsequent GD steps. This alteration is empirically evaluated for its potential to improve performance metrics such as convergence speed, the quality of the final solution, and the ability to escape unfavorable regions of the search space (local minima, saddle points). While the rigorous convergence statement above applies specifically to the gradient-descent phase under the stated assumptions, the dynamic update serves as a complementary strategy to improve the overall empirical effectiveness of the optimization process on challenging non-convex problems.

2.3.3 Rate of convergence

For strongly convex objective functions, gradient descent enjoys a linear convergence rate under standard smoothness assumptions. In this work, such a rate statement applies to the gradient-descent phase, while the dynamic jump step is treated as a restart mechanism that may improve practical performance but is not claimed to preserve a global linear rate without additional assumptions. For general non-convex functions, only convergence to stationary points can be asserted under standard smoothness conditions, and the rate may be substantially slower.

Theorem 2.1 (Linear Convergence of Gradient Descent Under Strong Convexity). *Assume that $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is twice differentiable and satisfies:*

1. *μ -strong convexity:*

$$f(W) \geq f(V) + \nabla f(V)^T(W - V) + \frac{\mu}{2}\|W - V\|^2.$$

2. *Lipschitz continuous gradient:*

$$\|\nabla f(W) - \nabla f(V)\| \leq L\|W - V\|, \quad \forall W, V \in \mathbb{R}^d.$$

Then gradient descent with a fixed step size $\gamma \in (0, 2/L)$ converges linearly to the unique minimizer W^* of f . In particular,

$$\|W_n - W^*\| \leq \rho^n \|W_0 - W^*\|,$$

for some $\rho \in (0, 1)$ depending on γ , μ , and L .

Theorem 2.2 (Stationarity Rate for Gradient Descent Phases). *Let $f : \mathbb{R}^d \rightarrow \mathbb{R}$ be differentiable with Lipschitz continuous gradient:*

$$\|\nabla f(W) - \nabla f(V)\| \leq L\|W - V\|, \quad \forall W, V \in \mathbb{R}^d,$$

and let $\gamma \in (0, 1/L)$ be a fixed step size. Consider a sequence of gradient-descent phases indexed by j , where within phase j the iterates satisfy

$$W_{j,k+1} = W_{j,k} - \gamma \nabla f(W_{j,k}), \quad k = 0, \dots, N_g(j) - 1.$$

Then, for each phase j , the standard descent estimate implies

$$\min_{0 \leq k < N_g(j)} \|\nabla f(W_{j,k})\|^2 \leq \frac{2(f(W_{j,0}) - f(W_{j,N_g(j)}))}{\gamma N_g(j)}.$$

If, in addition, the iterates remain in a bounded level set and the total decrease in function value is summable, then the gradient-descent phases converge to stationary points. Any complexity bound of order $\mathcal{O}(1/\varepsilon^2)$ applies to the gradient-descent component under the usual smoothness assumptions.

Summary.

- Under strong convexity, the gradient-descent phase converges linearly to the unique minimizer.
- For general smooth non-convex objectives, the gradient-descent phase can be analyzed in terms of descent and stationarity.
- The total number of gradient steps to reach an ε -stationary point is typically of order $\mathcal{O}(1/\varepsilon^2)$ under standard smoothness assumptions.
- The jump steps divide optimization into phases and are used as restart heuristics to improve exploration.
- No claim is made here that jump steps preserve linear convergence or guarantee a finite number of jumps without additional assumptions.

3 Robustness to Non-convexity

One of the practical motivations for the proposed method is its ability to improve optimization behavior on non-convex objectives. Non-convexity often arises in machine learning models such as deep neural networks, where the objective landscape may contain many local minima, saddle points, and flat regions. In such settings, standard gradient descent may stagnate at stationary points that are not globally optimal.

The root-finding procedure incorporated into our algorithm is intended as a restart heuristic that periodically changes the starting point of subsequent gradient-descent phases. This can improve exploration of the search space and may help the method move away from unfavorable stationary regions. However, in the present analysis, this mechanism is not claimed to guarantee escape from all local minima or saddle points, nor to ensure convergence to the global minimum for general non-convex objectives.

Thus, the proposed method should be viewed as a practical enhancement of gradient descent for difficult non-convex problems, with rigorous convergence guarantees established for the gradient-descent component under standard assumptions.

3.0.1 Practical considerations

While the theoretical analysis provides guarantees for convergence, it is important to consider the practical implementation of the proposed algorithm. In real-world applications, the following factors must be carefully addressed:

- **Step Size:** Choosing an appropriate step size γ is critical. If γ is too large, the algorithm may fail to converge, whereas if it is too small, convergence may become excessively slow.
- **Root-Finding Procedure:** The root-finding step adds computational complexity to each iteration. However, this step is essential for escaping local minima, and its computational cost is relatively small compared to the overall complexity of the optimization problem.

Despite these practical considerations, the proposed method provides a significant improvement over traditional gradient descent, particularly for non-convex optimization problems commonly encountered in machine learning applications.

4 Experiments and Results

In this section, we present experimental results to validate the effectiveness of the proposed gradient descent variant with dynamic starting-point updating. The objective of these experiments is to evaluate the method’s ability to find lower-value objective points in non-convex landscapes and to compare its performance against standard GD and other established optimizers.

4.1 Experimental setup

To evaluate the performance of the proposed algorithm, we test it on a series of non-convex functions commonly used in benchmarking optimization algorithms. Specifically, we compare the following methods:

- **Standard Gradient Descent (GD):** The traditional gradient descent algorithm with a fixed step size and no dynamic restarts.
- **Proposed Algorithm (D-SGD):** The gradient descent algorithm with dynamic starting-point updating, as described in Section 2.
- **Stochastic Gradient Descent (SGD):** The standard variant where gradients are computed using mini-batches.
- **Adam Optimizer:** An adaptive optimization method that maintains estimates of first and second moments of the gradients.

The experiments involve optimizing non-convex functions characterized by multiple local minima and saddle points. Each algorithm is executed for a fixed number of iterations or until a termination criterion is met. The stopping condition for all methods is based on the gradient norm, terminating when $\|\nabla f(W)\| < \varepsilon$.

We evaluate the following performance metrics:

- **Convergence Speed:** The number of iterations or gradient evaluations required by each algorithm to satisfy the termination criterion.

- **Final Objective Value:** The objective value $f(W_{final})$ attained at termination, used to assess the method’s ability to reach regions of lower cost.
- **Computational Time:** The total wall-clock time required to complete the optimization process.

4.2 Benchmark functions

We evaluate the algorithms on a suite of non-convex functions, each representing distinct challenges in optimization:

Rastrigin Function. The Rastrigin function is a highly non-convex landscape characterized by a large number of local minima, making it a standard test for assessing an algorithm’s ability to avoid entrapment in sub-optimal regions. On a d -dimensional domain, it is defined as:

$$f(W) = Ad + \sum_{i=1}^d [w_i^2 - A \cos(2\pi w_i)], \quad (11)$$

where $A = 10$ and $w_i \in [-5.12, +5.12]$.

Ackley Function. The Ackley function features a nearly flat outer region and a steep central basin surrounded by multiple local minima. It is defined as:

$$f(W) = -a \exp \left(-b \sqrt{\frac{1}{d} \sum_{i=1}^d w_i^2} \right) - \exp \left(\frac{1}{d} \sum_{i=1}^d \cos(cw_i) \right) + a + e, \quad (12)$$

where $a = 20$, $b = 0.2$, and $c = 2\pi$.

Sphere Function. The Sphere function is a simple convex baseline used to evaluate convergence efficiency in well-behaved landscapes:

$$f(W) = \sum_{i=1}^d w_i^2. \quad (13)$$

4.3 Results and comparison

We apply the optimization algorithms to the Rastrigin and Ackley functions (2D) and the Sphere function (10D). All algorithms are initialized at $W_0 = (5, 5)$. For GD, D-SGD, and SGD, we set a learning rate of $\gamma = 0.1$, while Adam uses standard default hyperparameters.

Table 1: Performance Comparison of Optimization Algorithms

Function / Metric	GD	D-SGD	SGD	Adam
Rastrigin (2D)				
Iterations to Convergence	148	72	98	87
Final Objective Value	10.3	0.03	4.5	0.1
Computational Time (s)	0.50	0.60	0.55	0.62
Ackley (2D)				
Iterations to Convergence	204	124	172	155
Final Objective Value	5.45	0.04	1.1	0.02
Computational Time (s)	0.80	1.00	0.85	1.20
Sphere (10D)				
Iterations to Convergence	45	36	50	38
Final Objective Value	0.001	0.00001	0.02	0.003
Computational Time (s)	0.70	0.90	0.75	1.00

As summarized in Table 1, D-SGD consistently achieves lower final objective values compared to standard GD across all tested functions. For the highly non-convex Rastrigin and Ackley functions, the dynamic starting-point updating mechanism appears to facilitate escape from local stationary points, allowing the algorithm to reach significantly deeper regions of the objective landscape.

While D-SGD introduces a marginal increase in computational time due to the root-finding procedure, this is compensated by a notable reduction in the number of iterations required to reach termination criteria. In the convex Sphere function scenario, where exploration of multiple minima is not required, D-SGD still demonstrates competitive convergence speed, validating that the dynamic updates do not hinder performance even in simpler optimization settings.

4.4 Discussion

From the results, we can draw several conclusions:

- **Escaping Local Minima:** The proposed D-SGD algorithm consistently outperforms the traditional GD method in non-convex optimization problems. The dynamic starting point updates enable the algorithm to escape from local minima, resulting in faster convergence and lower final objective values compared to GD.
- **Computational Efficiency:** Although the root-finding step introduces slight computational overhead, this overhead is minimal compared to the improvement in convergence speed and accuracy. For most problems, the additional time is negligible considering the overall performance gains.
- **Comparison with SGD and Adam:** D-SGD performs competitively with stochastic methods (SGD and Adam). While these methods are effective in large-scale problems, D-SGD demonstrates a distinct advantage in handling non-convexity and avoiding local minima, making it a robust choice for complex optimization landscapes.

4.5 Conclusion

The experimental results confirm the theoretical advantages of the proposed gradient descent variant with dynamic starting point updating. The algorithm proves to be highly effective in solving non-convex optimization problems, offering improved convergence speed and superior solutions compared to traditional gradient descent.

In future work, we plan to explore further optimizations and extensions of the method, including its application to deep learning models and large-scale datasets.

5 Enhancements and Ideas for Future Exploration

- **Adaptive Step Size:** Although a fixed step size γ is used in the current experiments, incorporating an adaptive learning rate may improve performance further. Such a mechanism could accelerate convergence in flatter regions while maintaining stability in areas with steep gradients or high curvature.
- **Heuristic Search for New Starting Points:** Instead of relying solely on the root-finding approach, future versions of the algorithm could incorporate heuristic strategies or learning-based methods to identify more promising initialization points. This may improve exploration of the search space and reduce sensitivity to poor starting conditions.

References

- [1] R. F. Barber and W. Ha, Gradient descent with non-convex constraints: Local concavity determines convergence, *Information and Inference: A Journal of the IMA*, 7 (2018), 755–806.
- [2] D. P. Bertsekas, *Nonlinear Programming*, pp. 1–768, Athena Scientific, Belmont (1999).
- [3] K. Bian and R. Priyadarshi, Machine learning optimization techniques: A survey, classification, challenges, and future research issues, *Archives of Computational Methods in Engineering*, (2024), 1–25.
- [4] L. Bottou, Large-scale machine learning with stochastic gradient descent, *Proceedings of COMPSTAT*, (2010), 177–186.
- [5] S. Boyd and L. Vandenberghe, *Convex Optimization*, pp. 1–716, Cambridge University Press, Cambridge (2004).
- [6] R. Chen, X. Tang and X. Li, Adaptive stochastic gradient descent method for convex and non-convex optimization, *Fractal and Fractional*, 6 (2022), 709.

- [7] J. Cohen and W. Yang, Optimization methods for machine learning: A review of key concepts and techniques, *Machine Learning Journal*, 45 (2021), 987–1003.
- [8] M. Danilova, P. Dvurechensky, A. Gasnikov, E. Gorbunov, S. Guminov, D. Kamzolov and I. Shibaev, Recent theoretical advances in non-convex optimization, *High-Dimensional Optimization and Probability: With a View Towards Data Science*, Springer, (2022), 79–163.
- [9] Y. N. Dauphin, H. de Vries and Y. Bengio, Identifying and attacking the saddle point problem in high-dimensional non-convex optimization, *NeurIPS*, (2014).
- [10] S. Frei and Q. Gu, Proxy convexity: A unified framework for the analysis of neural networks trained by gradient descent, *Advances in Neural Information Processing Systems*, 34 (2021), 7937–7949.
- [11] P. Jain and P. Kar, Non-convex optimization for machine learning, *Foundations and Trends in Machine Learning*, 10 (2017), 142–363.
- [12] J. Jensen and A. Ramaswamy, A comprehensive survey on optimization methods for non-convex problems in machine learning, *Journal of Machine Learning Research*, 22 (2021), 1–40.
- [13] O. Karakuş, P. Mayo and A. Achim, Convergence guarantees for non-convex optimisation with Cauchy-based penalties, *IEEE Transactions on Signal Processing*, 68 (2020), 6159–6170.
- [14] A. Karani and S. Chandwani, A survey of optimization algorithms in deep learning, *Journal of Computational and Graphical Statistics*, 28 (2019), 599–623.
- [15] K. Khamaru and M. Wainwright, Convergence guarantees for a class of non-convex and non-smooth optimization problems, *International Conference on Machine Learning*, (2018), 2601–2610.
- [16] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, *International Conference on Learning Representations (ICLR)*, (2015).

- [17] H. Kunder and M. Kotari, Exploration of non-convex optimization challenges across diverse data sets using machine learning and deep learning methods, *International Journal of Computing and Digital Systems*, 16 (2024), 1–2.
- [18] Y. LeCun, L. Bottou, G. B. Orr and K. R. Müller, Efficient Back-Prop, *Neural Networks: Tricks of the Trade*, Springer, (1998), 9–50.
- [19] A. Lobanov, A. Gasnikov, E. Gorbunov and M. Takáč, Linear convergence rate in convex setup is possible! Gradient descent method variants under (L_0, L_1) -smoothness, *arXiv preprint arXiv:2412.17050*, (2024).
- [20] V. S. Mikhalevich, A. M. Gupal and V. I. Norkin, Methods of non-convex optimization, *arXiv preprint arXiv:2406.10406*, (2024).
- [21] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, pp. 1–236, Springer, New York (2004).
- [22] O. E. Oduselu-Hassan and J. Okwuwe, Alternative numerical solutions to classical optimization problems using Rosenbrock function, unpublished manuscript.
- [23] P. Popovich, G. Fotopoulos and N. Papadopoulos, Review of non-convex optimization method for machine learning, unpublished manuscript.
- [24] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, pp. 1–1256, Cambridge University Press, Cambridge (2007).
- [25] H. Robbins and S. Monro, A stochastic approximation method, *The Annals of Mathematical Statistics*, 22 (1951), 400–407.
- [26] S. Ruder, An overview of gradient descent optimization algorithms, *arXiv preprint arXiv:1609.04747*, (2016).
- [27] O. Shamir, Convergence of stochastic gradient descent for PCA, *International Conference on Machine Learning*, (2016), 257–265.

- [28] I. Sutskever, J. Martens and G. E. Hinton, Generating text with recurrent neural networks, *ICML*, (2013).
- [29] B. Swenson, R. Murray, H. V. Poor and S. Kar, Distributed stochastic gradient descent: Nonconvexity, nonsmoothness, and convergence to local minima, *Journal of Machine Learning Research*, 23 (2022), 1–62.
- [30] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*, pp. 1–361, SIAM, Philadelphia (1997).

Tajedin Derikvand

Assistant Professor of Applied Mathematics

Department of Mathematics, Marv. C. , Islamic Azad University

Takht-e-Jamshid Blvd., Marvdasht 13119-73711, Iran

E-mail: Ta.derikvand@iau.ac.ir

Mazda Moattari

Associate Professor of Electrical Engineering

Department of Electrical Engineering, Marv. C. , Islamic Azad University

Takht-e-Jamshid Blvd., Marvdasht 13119-73711, Iran

E-mail: moattari@iau.ac.ir