Optimization Algorithms in Machine Learning

Saurav Singh Chandel[†]

[†] *Department of Mathematics and Statistics, Memorial University of Newfoundland, St. John's (NL) A1C 5S7, Canada*

E-mail: sschandel@mun.ca

Abstract: In the ever-evolving landscape of machine learning, the optimization of model parameters is a fundamental pursuit for enhancing performance.
This research delves into the comprehensive exploration of optimization algorithms, with a particular focus on their design and empirical evaluation within the context of machine learning. Our research methodology unfolds in two distinct yet interconnected phases. Firstly, we embark on the development of Python-based optimization algorithms, tailored to single-parameter loss functions.
These algorithms undergo rigorous empirical testing, as we systematically introduce controlled variations to single-parameter loss functions and assess their performance under different learning rates. The second phase extends this investigation to two-parameter loss functions, with a specific emphasis on their capability to reach global minimaâ€"a critical milestone in the optimization journey.
The findings from this research contribute valuable insights into the intricacies of optimization and offer practical guidance for selecting effective strategies. This research not only equips machine learning practitioners and researchers with essential knowledge but also advances the discourse on optimization in machine learning, empowering the community to make informed decisions to enhance model performance.
Keywords: Machine learning, Optimizers

1 Introduction

Machine learning has become integral to solving complex problems in numerous domains. At the core of its success lies the optimization of model parameters, which depends on choosing and using the right optimization algorithms. This research investigates the critical role of optimization in machine learning, focusing on the design and empirical evaluation of diverse optimization functions.

The paper aims to address the significant implications of optimizing machine learning models. The choice of an optimization algorithm profoundly affects convergence, efficiency, and final performance. Our study investigates the performance of six key optimization algorithms: *Stochastic Gradient Descent (SGD), Vanilla Momentum, Nesterov Accelerated Gradient, AdaGrad, RMSProp, and Adam.* These algorithms represent the core methods underpinning model training and parameter tuning.

Our primary goals are to assess these algorithms across different learning rates and identify the factors contributing to their performance. We scrutinize their behavior, considering the number of steps required for convergence and the achieved final loss. These insights offer valuable guidance to machine learning practitioners seeking optimal algorithm selection and hyperparameter tuning.

2 Methods

In this section, we provide a comprehensive overview of the methodology employed in the research, highlighting the systematic approach taken to develop and evaluate the optimization algorithms. Our methodology is structured into two distinct phases:

2.1 Phase 1: Development of Optimization Algorithms

The initial phase of our research was dedicated to the design and creation of optimization algorithms tailored for machine learning tasks. We implemented these algorithms using the Python programming language to ensure flexibility and reproducibility. This phase comprised the following key steps:

2.1.1 Algorithm Design

We began by designing optimization algorithms suitable for optimizing single-parameter loss functions. We mainly focused on six key algorithms *Stochastic Gradient Descent (SGD), Vanilla Momentum, Nesterov Accelerated Gradient, AdaGrad, RMSProp, and Adam.*[3]

2.1.2 Implementation

Following the design phase, we proceeded to implement these algorithms in Python. Due to this research being mainly focused on one parameter loss function, these algorithms were specifically designed according to our needs.

2.2 Phase 2: Empirical Evaluation

The second phase of our research focused on empirically evaluating the performance of the developed optimization algorithms. This phase was vital in understanding how these algorithms would function in practical machine-learning scenarios.

2.2.1 Modification of Loss Function

Initially, we considered the following Loss function: $L(p) = sin(2p) + asin(4p)$
Where p is the parameter of the loss function and a is an arbitrary constant that we alter while experimenting.
We kept $p^{(0)} = 0.75$ and had two different values of $a = 0.499$ and $a = 0.501$

2.2.2 Learning Rate Experiments

To gain insights into the algorithms' robustness and convergence characteristics, we executed experiments using different learning rates. This investigation helped us determine the sensitivity of the algorithms to the learning rate parameter.
We chose Learning rate

$$\eta\varepsilon$$

{ 0.1, 0.01, 0.001}.

2.3 Visualization of 2D Functions

Previously, to better understand the workings of optimizers, we illustrated the optimizers with 1D datasets. Now that we have a basic understanding, we can start by going up in dimensions as that is a better representation of real-world scenarios where we have multiple parameters hence making it higher dimensional.
We start by plotting six functions

- Six-Hump Camel

$$f(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + \left(-4 + 4y^2\right)y^2$$

- Michalewicz

$$f(\mathbf{x}) = -\sum_{i=1}^{D} \sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$$

- Sphere

$$f(\mathbf{x}) = \sum_{i=1}^{D} x_i^2$$

- Ackley

$$f(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + \exp(1)$$

- Shubert

$$f(x,y) = \sum_{i=1}^{5} i\cos\left((i+1)x + i\right)\sum_{i=1}^{5} i\cos\left((i+1)y + i\right)$$

- Branin

$$f(x,y) = a(y - bx^2 + cx - r)^2 + s(1-t)\cos(x) + s$$

on a 3D Grid to see what we are dealing with.

2.4 Finding the Minima using Optimizers

Then we modify the previous optimizers we wrote for 1D datasets to process 2D datasets so that we can visualize the trajectory taken by the optimizers. Each of the optimizers is individually run for the previously mentioned 2D functions to illustrate the workings and interpret the results so we can conclude the use-case scenarios and discuss some of their characteristics. For testing purposes, we are going to select a relatively low learning rate of *0.001*.

3 Results

3.1 1D Functions

This section unveils the findings from our research on optimization algorithms in machine learning. We present the outcomes from the two main phases of our study: algorithm development and empirical evaluation. These results contribute to the discussion on optimization in machine learning and offer practical insights for optimizing strategies.

The following table shows the data from running several experiments with $p^{(0)} = 0.75$, $a = 0.499$, $a = 0.501$:

1	Name	Epochs	Learning Rate	Final Loss	Optimized Parameter
2	SGD	151	0.1	8.88178E-16	2.61780115
3	SGD	1574	0.01	8.43769E-14	2.617800892
4	SGD	15677	0.001	9.83658E-14	2.617800201
5	Vanilla momentum	272	0.1	5.81757E-14	2.617801424
6	Vanilla momentum	248	0.01	3.88578E-14	2.617798629
7	Vanilla momentum	1621	0.001	8.63754E-14	2.617801282
8	Nesterov Accelerated Gradient	16	0.1	7.77156E-15	2.617801158
9	Nesterov Accelerated Gradient	150	0.01	6.21725E-14	2.617801127
10	Nesterov Accelerated Gradient	1606	0.001	3.4861E-14	2.61780101
11	AdaGrad	1125	0.1	8.74856E-14	2.617800842
12	AdaGrad	92226	0.01	9.99201E-14	2.61779782
13	AdaGrad	100000	0.001	1.35738E-06	1.29186867
14	RMSProp	100000	0.1	0.001009424	2.666339446
15	RMSProp	262	0.01	1.75415E-14	2.617801129
16	RMSProp	1933	0.001	5.21805E-14	2.6178011
17	Adam	286	0.1	7.23865E-14	2.617801549
18	Adam	1150	0.01	8.21565E-14	2.617801286
19	Adam	6023	0.001	9.74776E-14	2.617800531

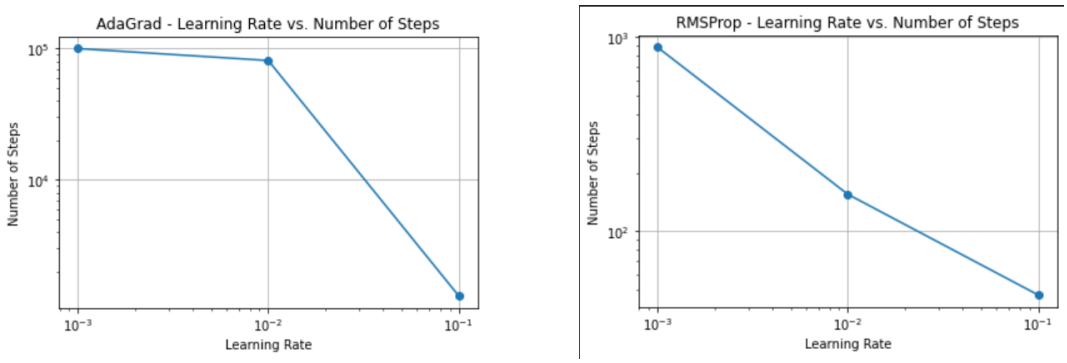
Figure 1. a = 0.499

1	Name	Epochs	Learning Rate	Final Loss	Optimized Parameter
2	SGD	215	0.1	9.79217E-14	1.552555968
3	SGD	1946	0.01	9.95384E-14	1.552550979
4	SGD	16885	0.001	9.9934E-14	1.552535369
5	Vanilla momentum	270	0.1	5.04041E-14	2.618185569
6	Vanilla momentum	272	0.01	3.21965E-14	2.618186157
7	Vanilla momentum	1781	0.001	9.98437E-14	1.552551119
8	Nesterov Accelerated Gradient	16	0.1	1.90958E-14	2.618186064
9	Nesterov Accelerated Gradient	150	0.01	2.90878E-14	2.618186007
10	Nesterov Accelerated Gradient	1808	0.001	9.93164E-14	1.552551122
11	AdaGrad	1312	0.1	9.89278E-14	1.552552339
12	AdaGrad	81134	0.01	9.99964E-14	1.552504844
13	AdaGrad	100000	0.001	1.35136E-06	1.291396571
14	RMSProp	47	0.1	7.53148E-14	1.552557842
15	RMSProp	155	0.01	8.30586E-14	1.552557792
16	RMSProp	886	0.001	9.15865E-14	1.552557737
17	Adam	286	0.1	7.30527E-14	2.618186451
18	Adam	1042	0.01	9.91013E-14	1.552554314
19	Adam	5730	0.001	9.99201E-14	1.552551256

Figure 2. a = 0.501

These tables consist of all the experimental data for the six algorithms used in this research for 3 distinct values of learning rates at the exact value of a.
We can interpret different patterns and behaviors within different algorithms and under other conditions. But before we make an inference, we should also see the graph for the particular set of experiments with $a = 0.501$

We can infer the convergence characteristics concerning the number of steps, as discerned from the graph, reveal crucial insights into the optimization algorithms' behavior. A steep decline in the loss function within the initial steps underscores the algorithms' rapid adaptability. Subsequently, a gradual leveling-off suggests that the optimization process approaches a stable state, indicating the algorithms' ability to converge efficiently. The specific rate of convergence becomes apparent from the slope of this plateau, offering valuable information for practitioners fine-tuning the algorithms for various applications. In analyzing this convergence trend, it becomes evident that optimization algorithms exhibit distinct dynamics in reaching the optimal solutions, facilitating informed decisions on their deployment and parameter configuration.



(a) AdaGrad (b) RMSProp

Figure 3. Learning Rate vs Steps

3.2 2D Functions

In the following section, we present the results of our optimization experiments on a set of diverse functions, including the *Six-Hump Camel, Michalewicz, Sphere, Ackley, Shubert, and Branin functions*. These functions have been carefully chosen for their distinct characteristics, such as multiple local minima, flat regions, and varying degrees of complexity. We utilize a range of optimization algorithms to explore and visualize the trajectory of each optimizer in search of the global minimum. The resulting plots provide valuable insights into the performance of different optimizers on these challenging functions.

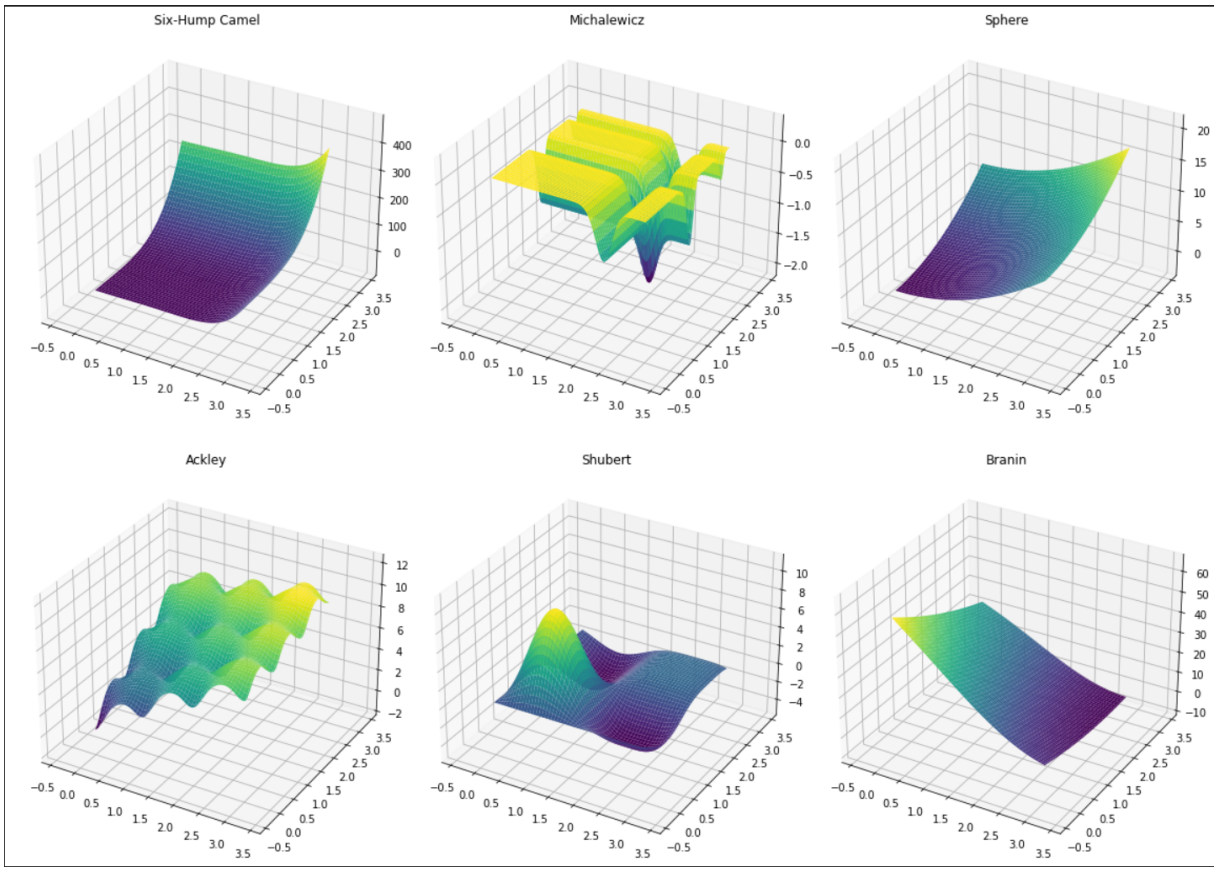


Figure 4. 2D Functions

In the following visualizations, we showcase the intricate landscapes of the *Six-Hump Camel, Michalewicz, Sphere, Ackley, Shubert, and Branin functions*, along with the trajectories mapped by various optimization algorithms. Each function represents a unique optimization challenge, with complex surfaces featuring multiple local minima. Our objective is to illustrate how different optimization techniques navigate these intricate landscapes, providing valuable insights into their ability to converge to the global minimum. These plots vividly display the journeys of optimizers as they search for optimal solutions, shedding light on the effectiveness of each algorithm.

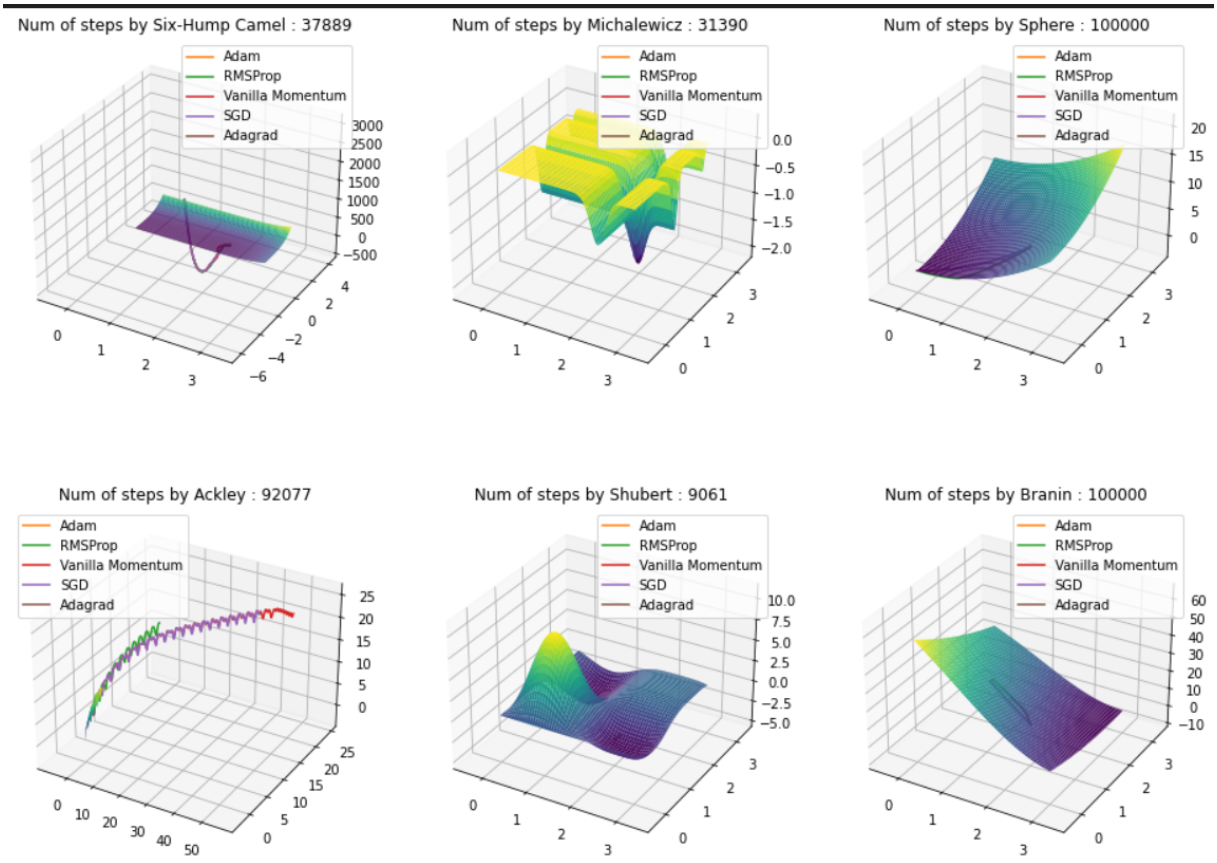


Figure 5. 2D Functions

To offer a more intuitive perspective, we present 2D projections of the *Six-Hump Camel, Michalewicz, Sphere, Ackley, Shubert, and Branin functions*, alongside the paths taken by diverse optimization algorithms. These projections onto the X-Y plane distill the complexity of the 3D landscapes, enabling a clearer visualization of how optimizers traverse these intricate terrains. By examining the trajectory of each optimizer on these 2D representations, we gain a deeper understanding of their ability to navigate and pinpoint global minima within the original 3D functions.

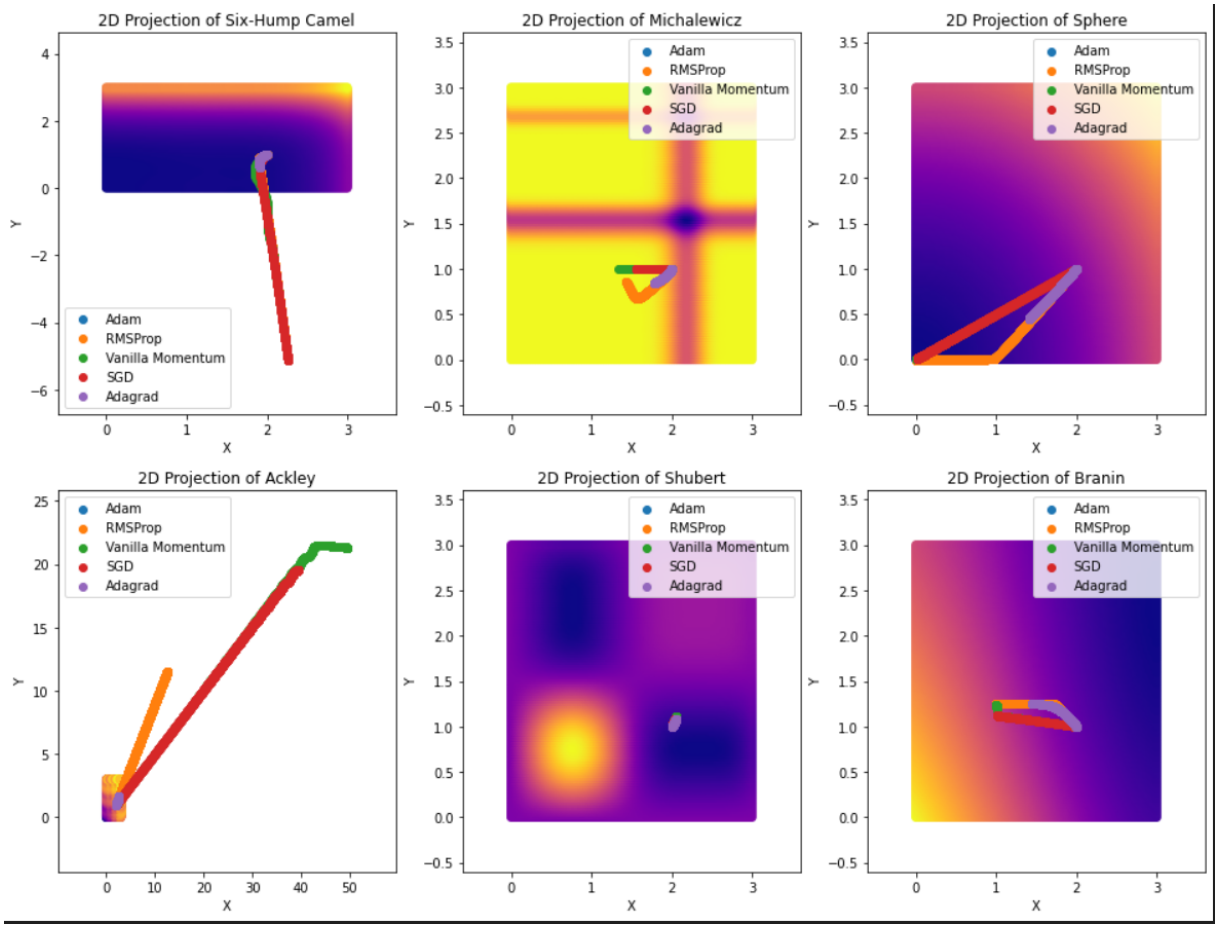


Figure 6. 2D Functions

3.2.1 Challenges with Ackley Function

The Ackley function is a bit tricky for optimizers. It has many small low points scattered around and one deep low point that we want to find. The problem is, that optimizers might get stuck in the small low points instead of reaching the deep one. Also, the Ackley function doesn't provide clear directions on which way to go, making it even more challenging for some optimizers. It's like searching for hidden treasure in a confusing maze.

Because of these difficulties, the Ackley function is often used to test how good optimizers are at finding the best solution in tough situations. It's like a tough obstacle course for optimization algorithms.[1]

3.2.2 Challenges with Sphere Function

A low learning rate has a significant impact on optimization algorithms, particularly in functions with minimal gradient information like the Sphere function. It leads to slower convergence, as algorithms take cautious, small steps during each iteration to avoid overshooting the minimum. However, this cautious approach also makes algorithms more susceptible to getting stuck in local minima, especially if those minima are shallow or wide. Moreover, low learning rates increase sensitivity to noise, raising computational costs and potentially causing premature convergence to suboptimal solutions. Striking the right balance between stability and convergence speed is crucial, as the ideal learning rate varies depending on the specific optimization problem.[2]

4 Conclusion

This research has embraced a comprehensive approach to design and empirically evaluate these algorithms, offering valuable insights for both practitioners and researchers.

The systematic development of optimization algorithms tailored for single-parameter loss functions, along with rigorous empirical assessments, has illuminated their adaptability and robustness. Through controlled modifications to loss functions and learning rates, we have gained a nuanced understanding of algorithm performance.

Our research has systematically designed and evaluated optimization algorithms for one-variable loss functions. By carefully adjusting loss functions and learning rates, we've gained valuable insights into algorithm adaptability and performance. We've also visualized how these algorithms behave in two-dimensional spaces, highlighting the challenges posed by complex functions like the Ackley problem. Moreover, our study of the sphere optimization problem revealed that low learning rates can lead to more steps for convergence.

In conclusion, this research contributes to the ongoing discourse surrounding optimization in machine learning. It equips the community with practical knowledge for selecting strategies that enhance model performance. As machine learning continues its rapid evolution, our findings provide a foundation for further advancements and refinements in the optimization field, offering a pathway toward more effective model training and ultimately, more powerful machine learning applications.

Acknowledgements

We would like to extend our sincere appreciation to ChatGPT, an AI language model, for its valuable guidance and support throughout this research. ChatGPT has played an instrumental role in offering insights, assisting with code, and providing explanations that enriched our understanding of various optimization algorithms. We are grateful for the assistance and contribution of ChatGPT to this project.

References

[1] Ackley D., *A connectionist machine for genetic hillclimbing*, vol. 28, Springer science & business media, 2012.

[2] Ferreira O., Iusem A. and NÃ©meth S., Sphere optimization, *Top* **accepted** (2014).

[3] Ruder S., An overview of gradient descent optimization algorithms, *arXiv preprint arXiv:1609.04747* (2016).