Optimizing Complex Functions: A Numerical and ML Comparison

Xuechao Zhang
Mathematics Teaching and Research, Office， Zhengzhou, Professional Technical Institute of Electronic & Information, Zhengzhou, Henan, 451450, China
E-mail：zxfchao1982@126.com

Keywords: complex function optimization, convergence rate, benchmark functions, computational efficiency, machine learning algorithms, numerical methods

Received: February 20, 2025

Optimization plays a vital role across disciplines such as engineering, economics, and artificial intelligence. Complex functions, which map complex numbers to complex outputs, often cannot be solved analytically, necessitating numerical or machine learning-based approaches. This study presents a comparative analysis of numerical optimization methods—specifically Gradient Descent and Newton's Method—against machine learning-based techniques, including Genetic Algorithms, Particle Swarm Optimization, and Deep Q-Learning. These methods are evaluated using standard benchmark functions: Ackley, Rastrigin, and Rosenbrock. The comparison focuses on convergence rate and runtime performance. Results show that numerical methods offer faster runtimes but lower convergence rates, while machine learning approaches achieve higher convergence at the cost of increased computational time. This analysis underscores the trade-offs between efficiency and robustness in optimization techniques, offering practical insights for selecting appropriate methods based on specific application needs, especially in scenarios involving complex, non-linear, or high-dimensional functions.

Povzetek: Prispevek išče kompromis med hitrostjo in robustnostjo kompleksnih funkcij. Uporablja GD/Newton vs. GA/PSO/DQN na Ackley/Rastrigin/Rosenbrock, brez standardnih postopkov.

1 Introduction

Complex function optimization is a significant area of research within optimization problems. The methodologies for solving optimization issues can be categorized into analytical methods and numerical computations [1]. The analytical method addresses the problem by examining the correlation between the derivative of the objective function and its extreme values. This strategy is applicable solely to optimization problems characterized by relatively uncomplicated objective functions. According to the variation principle of the objective function value, one should proceed in suitable increments along the direction that maximizes the objective function value. A method for approximate calculation that incrementally approaches the optimal point of the objective function. This technique is effective for resolving continuous differentiable convex optimization problems. However, with the ongoing proliferation of engineering optimization challenges, the majority of objective functions are non-convex optimization problems. The advent of group intelligent optimization algorithms offers a constrained approach to difficult function optimization challenges [2].

Numerical Optimization is a recognized domain within Mathematical Sciences that seeks to determine the extreme values of a function, namely its maxima and minima. In the past two decades, optimization approaches have emerged as essential instruments for management, decision-making, technological enhancement, and development, conferring competitive benefits to diverse systems. Consequently, optimization models and algorithms have become prominent in various domains, including industry [3], disease diagnosis [4], scheduling and resource allocation [5], and finance [6].

A prominent domain dedicated to addressing issues through mathematical models and algorithms is Machine Learning. In certain real-world scenarios, a substantial amount of data must be handled. This volume of data typically necessitates computational support to convert the information into pertinent knowledge for problem-solving. In this situation, machine learning techniques are highly beneficial. These models and algorithms aim to provide a mathematical representation that characterizes the data collection and extrapolates insights to unfamiliar data samples [7]. Machine learning models and algorithms are applicable in various fields, including industry, healthcare, finance, and education.

Owing to the practical significance of both domains, numerous methods have been devised to address optimization and machine learning challenges. While the majority of algorithms are effective in addressing their respective challenges, none are entirely flawless [8]. A pure algorithm denotes a singular approach or method utilized to resolve an issue comprehensively from inception to completion. A hybrid algorithm amalgamates many algorithms or techniques from disparate disciplines to address a problem or enhance the efficacy of a singular solution. Hybrid methods can combine many methodologies to capitalize on the strengths of each algorithm while alleviating their weaknesses. The hybrid algorithm represents a synthesis of concepts and techniques aimed at investigating the capabilities of several approaches while mitigating their deficiencies. By integrating complementary algorithms, one can leverage their strengths and mitigate limits, resulting in enhanced overall performance.

A hybrid algorithm that integrates optimization and machine learning approaches is an effective approach that leverages the strengths of both disciplines to provide a robust framework for addressing complicated issues [9]. This integration enhances decision-making capacities by incorporating optimization approaches into the machine learning process and vice versa. A hybrid algorithm can utilize optimization techniques to direct the learning process, thereby improving the accuracy and effectiveness of decision-making. This integration allows the algorithm to utilize explicit mathematical optimization methods and data-driven learning capabilities, resulting in enhanced and more efficient decision-making [10].

This paper delineates and examines the primary characteristics of numerical optimization and machine learning techniques. By recognizing and examining the primary aspects of each methodology in conjunction, they can mitigate difficulties and improve one or both processes through hybrid approaches. The objective is to determine the optimal combinations that yield hybrid approaches, taking into account a machine learning algorithm influenced by optimization strategies or vice versa.

This paper compares numerical optimization methods, specifically gradient descent and Newton's method, with machine learning optimization algorithms, including genetic algorithms, particle swarm optimization, and deep Q-learning. The benchmark optimization test functions Ackley, Rastrigin, and Rosenbrock are employed to assess the optimization methods based on convergence rate and runtime. The machine learning algorithms demonstrate the capability to effectively address complex function optimization problems and enhance the convergence rate relative to numerical methods.

The remaining part of this research is organized as, section 2 provides the background details including complex function, numerical optimization, machine learning and related works summary. Section 3 explains the proposed methodology. Section 4 provides the experimental results and section 5 conclude the research paper.

2 Background

2.1 Complex function

Let S denote a collection of complex numbers. A function f defined on S is a mapping that allocates to each z in the complex numbers a corresponding w. The quantity w is referred to as the value of f at z and is represented as f(z); so, w = f(z). The set S is referred to as the domain of definition of f.

If each value of z corresponds to only one value of w, we denote w as a single-valued function of z or f(z) that is single-valued. If many values of w correspond to each value of z, w is classified as a multiple-valued or many-valued function of z.

A multiple-valued function can be regarded as a compilation of single-valued functions, with each constituent referred to as a branch of the function. Typically, we designate one specific member as the principal branch of the multi-valued function, with the matching function value referred to as the principal value.

Polynomial function:

For $x_n, x_{n-1}, \ldots, x_0$ complex constants,

$$p(y) = x_n y^n + x_{n-1} y^{n-1} + \cdots + x_1 y + x_0 \quad (1)$$

Where $x_n \neq 0$ and n is a integer value.

Exponential function

If $z = a + ib$, the exponential function e^z is represented by,

$$e^z = e^a e^{ib} \quad (2)$$

Where $e^{ib} = \cos b + i \sin b$

$$e^z = e^z(\cos b + i \sin y) \quad (3)$$

Logarithmic Function

The complex logarithm is an extension of the standard real natural logarithm (base e) into the complex domain. In polar coordinates, the complex logarithm is expressed as,

$$\begin{aligned} \log z &= \log\left(re^{i\theta}\right) \\ &= \log r + \log e^{i\theta} \\ &= \log r + i\theta \end{aligned} \quad (4)$$

Trigonometric functions:

The sine and cosine of a complex variable are described as,

$$\sin z = \frac{e^{iz} - e^{-iz}}{2i} \quad (5)$$

$$\cos z = \frac{e^{iz} + e^{-iz}}{2} \quad (6)$$

In optimization, a complex function refers to one in which the input and/or output are complex numbers, with the objective of identifying maximum, minimum, or other key points, frequently subject to constraints or special conditions. These functions may be nonlinear, multivariable, and incorporate both real and imaginary components in their inputs and outputs, rendering them more complex to evaluate and optimize than real-valued functions.Optimization problems may involve complex functions when the objective function originates from physical systems represented by complex-valued variables, or in signal processing, control systems, and specific machine learning models where data or parameters are inherently articulated in the complex domain.

2.2 Numerical optimization

Although there are various strategies and algorithms available for addressing optimization difficulties, none are universally applicable or flawless in properly resolving all such concerns. Each algorithm possesses distinct properties, making it more suitable for addressing specific challenges based on these attributes. The formulation of the problem, selection of methods, and choice of algorithms are pivotal aspects in addressing an optimization issue, as certain methods and algorithms are more suitable than others based on the specific challenge at hand. The initial stage in an optimization problem is the formulation of the mathematical model. A mathematical model seeks to represent a real-world situation as a mathematical function applicable in optimization techniques. An optimization problem can be articulated in mathematical terms through a collection of variables and numerical relationships that encapsulate an abstraction of the issue, aiming to identify the optimal solution within Rn from a set of potential alternatives. To construct a mathematical model, four essential procedures must be performed [11].

- Specify the decision parameters. ($x_1,x_2,x_3,\ldots,x_n$)
- Construct the objective function f(x) or the set of objective functions $f_1(x)$, $f_2(x),f_3(x),\ldots,f_k(x)$ that rely on the choice variables and yield a real value.
- If required, provide a collection of equality and/or inequality constraints, $g_i(x) = 0$ and $h_j(x) \leq 0$ for i = 1, 2, ..., n_g and j = 1,2, ...n_h, that must be satisfied by the decision variables.
- Define the domain sets $D_1,D_2,D_3,\ldots,D_n$ corresponding to the decision parameters, $x_1,x_2,x_3,\ldots,x_n$ accordingly.

The optimization issue is defined by the objective function and the restrictions on the variables. The objective function delineates the goal of the problem, which may encompass any quantity or amalgamation of quantities represented by a singular numeral, such as personnel, time, materials, energy, etc.; conversely, the constraints directly influence the decision space and outcomes, imposing restrictions on the algorithmic options. The primary purpose of an optimization issue is to reduce or maximize the objective function while adhering to the restrictions. Furthermore, optimization models can be categorized into two principal classes based on the number of objectives: Single-objective and multi-objective optimization issues [12]. Single optimization problems typically consist of a singular target, with or without restrictions, while multi-objective optimization pertains to multiple-criteria decision-making, comprising the simultaneous optimization of numerous objective functions, with or without constraints [13]. Following the creation of the mathematical model, it is essential to ascertain the most suitable strategy for identifying the optimal solution.

A solution to an optimization issue can be characterized by local and global optimization, with corresponding techniques referred to as local search and global search. Local and global search optimization methods are employed in various contexts or to address distinct optimization inquiries [14]. Local optimization aims to identify a solution that reduces (or increases) the objective function within a defined region of the search space, hence locating a local solution among feasible points in the vicinity. This type of search does not ensure an objective value that is lower (or higher) than all other viable locations. Conversely, global optimization seeks to identify the point that reduces (or increases) the objective function [15].

2.3 Machine learning

The essence of human intelligence is on experiential learning and the transmission of personal knowledge between generations. Machine learning pertains to the development of computer programs that enhance their performance through experience. The experience is derived by a data analysis procedure executed by a customized algorithm. Consequently, the machine learning approach employs algorithms to identify patterns within a dataset through computational, analytical, optimization, and information discovery techniques [16].

There are three types of machine learning: Supervised Learning, Unsupervised Learning, and Reinforcement Learning. Within each category, methods can be differentiated based on their knowledge acquisition techniques, including classification, regression, clustering, association learning, relational analysis, differential equations, and others. Machine learning seeks to construct a hypothesis (model) capable of extracting information from training data and generalizing the learned knowledge to unfamiliar samples. This model must exhibit simplicity regarding complexity and demonstrate efficacy in minimizing empirical mistakes within the data.

Supervised learning techniques seek to identify the correlation between input attributes (independent

variables) and a target attribute (dependent variable). Supervised learning is employed for two categories of tasks: pattern categorization and regression. Occasionally, there is an absence of knowledge regarding the correlation between input and output attributes in machine learning challenges [17]. Consequently, the algorithms must identify similarities or differences within the data collection. The method necessitates greater human comprehension than supervised procedures, as a decision-maker, whether an individual or a collective, is accountable for the final decision-making.

While supervised and unsupervised approaches engage with data and necessitate exploration and comprehension of the data in relation to the application domain, significant distinctions exist between the two methodologies. The primary distinction is the lack of an output vector for the target variable, as observed in supervised approaches. Moreover, unsupervised learning is frequently linked to creative pursuits—exploration, comprehension, and enhancement—that do not conform to predefined protocols, unlike supervised methods [18]. Unsupervised learning is often categorized into clustering methods and dimensionality reduction methods.

Reinforcement learning is a machine learning technique predicated on a reward mechanism, predominantly utilized in dynamic control systems; however it is also applicable to optimization issues [19]. Reinforcement learning addresses the challenge of instructing an autonomous agent, which interacts with and perceives its environment, to select optimal actions for attaining its objectives. The agent acquires information regarding the present environmental conditions and must use its existing knowledge through a greedy approach to optimize rewards, while simultaneously engaging in exploration to identify superior actions for the future.

2.4 Related work summary

This section highlights the existing numerical and machine learning-based optimization strategies for complex function optimization problems. Table 1 provides the summary of existing optimization approach with limitation or research gap.

Table 1: Related work summary

Ref	Methods	Data set and Metrics	Limitations / Research Gap
[9]	Hybrid Bee algorithm for solving continues complex function	Benchmark function; Accuracy and mean value.	It does not handle numerical methods. Testing the enhanced algorithms on practical optimization challenges
[20]	Deep learning proxy model	Rosenbrock, Rastrigin and Ackley functions; Mean Euclidean distance and the standard deviation	Sparse sampling is ineffective, and the optimizers exhibit heightened sensitivity to the landscape derived from this kind of data.
[21]	First order optimization methods: Standard gradient descent, momentum, heavy ball, nesterov	Benchmark functions; Iteration, time, function value and weight	It only compares first order optimization methods and does not consider other optimization approaches
[22]	Improved Hypercube optimization algorithm	Benchmark functions; Mean, Standard deviation, Convergence rate and Time.	It offers optimal value and accelerates convergence processes; however, it does not address practical optimization concerns.
[23]	Genetic algorithm for high dimensional optimization problem.	Benchmark functions; Gain and convergence rate.	A robust criterion is required to dynamically determine the number of active subspace dimensions.
[24]	Local and global numerical optimization; Focusing robust optimization with uncertainty-based sampling approach	Benchmark functions; CPU Time, mean, median and success rate	It does not converge effectively to the global minimum for various contrived test functions, rendering its broad application inadvisable.

3 Methodology

Optimizing complex functions involves identifying the extrema of functions that utilize complex numbers as both inputs and outputs. These functions may encompass both real and imaginary components, and addressing optimization problems involving such functions is a prevalent endeavor in disciplines such as signal processing, control theory, physics, and machine learning. A complex function f(z) can be written as,

$$f(z) = f(x + iy) = u(x, y) + iv(x, y) \quad (7)$$

where, $u(x,y)$ denotes the real component and $v(x,y)$ signifies the imaginary component, with x and y representing real variables corresponding to the real and imaginary parts of the complex input z. The objective of optimization frequently involves minimizing (or maximizing) a complex-valued function, specifically identifying the point z in the complex plane that yields the optimal value of (z).

This paper compares numerical optimization methods, specifically gradient descent and Newton's method, with machine learning optimization algorithms, including genetic algorithms, particle swarm optimization, and deep Q-learning. Figure 1 shows the work flow of the proposed approach.

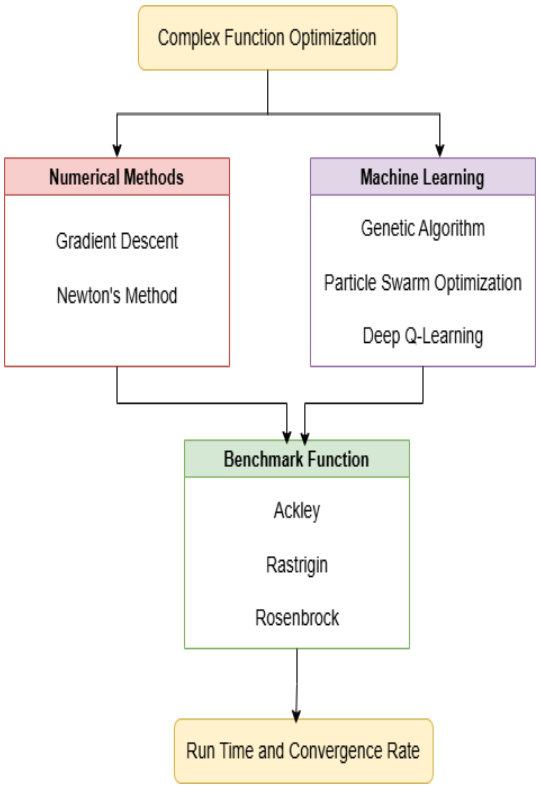

Figure 1: Proposed work flow

In all real-valued single-objective unconstrained optimization algorithms, the objective is to identify the minimum (or, equivalently, the maximum) of a scalar objective function (x), represented by the vector of free parameters $X=(x_1, x_2, x_3,...,x_m)$, where m denotes the problem's dimensionality. Consequently, f is a mapping from R^m to R. Assume the following hypothesis:

- Function f is accessible solely as a black box; thus, we possess no understanding or ability to manipulate its internal mechanisms. We access f solely through input-output mechanisms.
- The function f possesses a continuous domain inside the specified constraints; thus, every point within these bounds is mapped by f.
- f is well-defined in the domain, at least numerically; it is continuous and exhibits a degree of smoothness. This restricts very noisy functions, where spatial correlation is absent. However, there is also an implicit assumption of some degree of noise, wherein finite differences in the vicinity of a point do not resemble the derivatives of the noiseless function.

Numerical optimization techniques are employed to identify the most effective for intricate functions when statistical techniques are impractical or excessively challenging. The primary classifications of numerical optimization methods are gradient-based and gradient-free

techniques. In this paper two numerical optimization methods are considered: Gradient descent and Newton's method. The previous works only consider the first-order optimization but this research considers both first-order and second-order optimization. Due to the simplicity, the first order optimization i.e gradient descent is selected which reduces the training time.

Gradient Descent:

Standard gradient descent is among the most fundamental and prevalent optimization techniques. The fundamental concept involves incrementally adjusting the model parameters in accordance with the gradient of the loss function. The gradient signifies the direction in which the value of the loss function diminishes most rapidly. The objective is to minimize the function by modifying the model parameters accordingly.

The primary phases of Standard Gradient Descent:

Computation of gradients: At each stage of the procedure, the gradient of the loss function is computed for all model parameters. The gradient is a vector comprising the partial derivatives of the loss function with respect to each parameter, signifying the direction of the steepest ascent in the function.

Update of parameters: Subsequent to computing the gradient, the model parameters are revised in accordance with the formula:

$$\delta_{new} = \delta_{old} - \alpha \times \nabla\tau(\delta) \quad (8)$$

Where δ represents the model parameter, α indicates the learning rate and $\nabla\tau(\delta)$ is the gradient loss function.

***Rate of learning*:** This is a crucial parameter of the procedure. If the step size is very tiny, the optimization process will be sluggish; conversely, if it is overly high, it may bypass the minimum and induce instability in the process. Consequently, the appropriate selection of alpha is essential.

Repetitions: The parameter update process is iterated multiple times until a termination point is achieved, either by convergence criteria or after a predetermined number of iterations.

Gradient Descent is straightforward to implement, as it requires only the calculation of the gradient and the subsequent update of the model parameters for each iteration. Should the loss function exhibit smoothness and convexity, the approach can effectively identify the global minimum.

Newton's Method

Newton's approach is a second-order optimization approach that can be used to locate the roots of complex functions or optimize them. This method is based on the second derivative and often converges quicker than gradient descent for smooth functions.

Given a complex function $f(z) = z^2 - 2 + i$

1. Select an initial guess, z_0= 1+i
2. Compute the next approximation using the formula:

$$z_{n+1} = z_n - \frac{f(z_n)}{f'(z_n)} \quad (9)$$

Where $f(z) = z^2 - 2 + i$ and f'(z)=2z

$f(1+i)=(1+i)^2 - 2+i=1+2i-1+i=3i$

f'(1+i)=2(1+i)=2+2i

therefore, $z_1 = (1+i) - \frac{3i}{2+2i}$

3. Repeat this process iteratively until convergence.

Genetic Algorithm

Genetic Algorithm (GA) is an optimization technique based on population dynamics. These heuristic search methods, inspired by natural evolution, can optimize complex functions, particularly when the functions are noisy or non-differentiable. Genetic algorithms employ a population of potential solutions, which are iteratively evolved throughout multiple generations using selection, crossover, and mutation to identify the optimal answer. Algorithm-1 shows the general steps of genetic algorithm.

Algorithm-1: Genetic Algorithm

Input: Population Size (pop), Maximum iteration (MaxIter)
Output: Global best solution
Step1: Randomly generate initial population of n chromosomes
Step2: Initialize iteration count iter = 0
Step3: Compute the fitness value of each chromosomes
Step4: while (iter <MaxIter)
Step5: Select a pair of chromosomes from initial population based on fitness
Step6: Apply crossover operation on selected pair with crossover probability
Step7: Apply mutation on the offspring with mutation probability
Step8: Replace old population with newly generated population
Step9: Increment the iteration iter by 1
Step10: EndWhile
Step11: Return the best solution

The genetic algorithm (GA) employed a population size of between 20 or 200, contingent upon the number of variables. The crossover rate was 0.8, and the maximum number of iterations was 1000. The algorithm terminated when the average relative change in the optimal fitness function value over 100 iterations was less than or equal to 1e-6.

Particle swarm Optimization

The PSO algorithm is a gradient-free technique founded on the notion of swarm dynamics. In Particle Swarm Optimization (PSO), each sample is designated as a particle, and a collection of particles is termed a swarm. In the initial iteration of PSO, each particle commences at rest with a velocity vector of zero. The objective function's value is calculated for the initial iteration, and the particles' velocity and position are revised. The velocity of each particle is determined by three components: inertia, cognitive, and social components, along with certain random coefficients. The inertia component signifies a particle's velocity at the present iteration, inhibiting significant positional alterations. The cognitive component assesses a particle's performance in relation to its prior best, and the social component evaluates a particle's performance in comparison to the swarm's optimal location. The optimal positions are ascertained from the fitness function assessment in the preceding iteration, and the updates for velocity and position persist until the termination criteria are met. Algorithm-2 shows the general steps of PSO algorithm.

Algorithm-2: Particle swarm optimization
Input: Particle Size , Maximum iteration (MaxIter)
Output:best solution
Step1: Initialize particles, velocity, iter=0, pbest, gbest
Step2: Randomly generate initial particles
Step3: For each particle
Step4: Compute fitness function
Step5: Update pbest, gbest
Step6: End For
Step7: While (iter <MaxIter)
Step8: For each particle
Step9: Update velocity and particle
Step10: Compute fitness value
Step11: Update pbest, gbest
Step12: EndFor
Step13: EndWhile

The PSO employed a swarm size of between 20 or 200, contingent upon the number of variables. The weight was 0.1, and the maximum number of iterations was 1000.

Machine learning is applicable for optimizing complex functions, especially in the context of high-dimensional, non-convex, or noisy functions that are challenging for conventional approaches to manage.A neural network can be taught to approximate a complicated function by utilizing the real and imaginary components of the function as input and output. This can be particularly useful if the optimization process entails investigating complex spaces with numerous variables. To optimize the network, it can be trained using a loss function, such as Mean Squared Error (MSE), to reduce the disparity between the expected output of the complex function and the actual function. The network acquires the ability to forecast ideal input parameter values (complex values) that reduce the output of the complex function. The backpropagation algorithm updates the network's weights according to the prediction error.

Reinforcement learning (RL) can optimize intricate functions, particularly in scenarios where the function is costly to assess or exhibits noise. In reinforcement learning, the agent engages with the environment (a complicated function) by executing actions (selecting intricate input values) and obtaining rewards (ideal outputs).Methods such as Q-learning or Deep Q Networks (DQN) can be utilized to progressively enhance the answer through environmental feedback. Algorithm-3 shows the general steps of DQN.

Algorithm-3: DQN
Step1: Initialize parameters
Step2: For each episode
Step3: Observe initial state
Step4: For each step of episode
Step5: Select action
Step6: Execute action
Step7: Observe reward and new state
Step8: Store transition
Step9: Sample mini-batch of transitions
Step10: Compute target for each transition
Step11: Perform descent step with respect to the quantum circuit parameters
Step12: Every steps reset Q’ = Q
Step13: End For
Step14: EndFor

A hybrid approach integrates the advantages of numerical methods and machine learning approaches to enhance the efficiency of complex function optimization. For example, one might employ a genetic algorithm to investigate the search space of intricate functions, subsequently utilizing a neural network to refine the answers near the optimum. The hybrid approach is not scope of the current work. This may be considered as a future work.

4 Experimental results and analysis

4.1 Benchmark dataset

This research uses benchmark dataset (Optimization Test Functions [24]) for evaluate the performance of the optimization problem. This work uses three optimization test functions Ackley, Rastrigin, and Rosenbrock.

Ackley function is continuous, scalable, non-separable, and a substantially multimodal test function. This test function is defined as follows:

$$f(x) = -20\, exp\left(-0.2 \times \sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - exp\left(\frac{1}{d}\sum_{i=1}^{d} \cos(2\pi x_i)\right) + 20 + e \quad (10)$$

where d represents the number of dimensions and $x_i = (x_1, x_2, ..., x_d)$ is a d-dimensional row vector. The test region is typically assessed within the range of [-32.768, 32.768]^d. The global minimum $(x) = 0$ is achievable at $xi = (0,0)$. Figure 2 shows the surface plot of 2D representation of Ackley function.

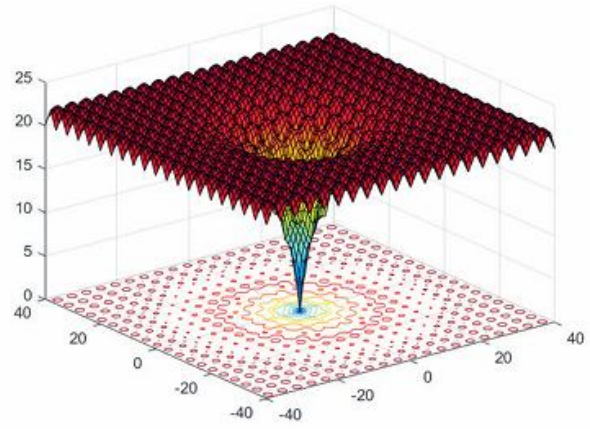

Figure 2: Ackley function - surface plot

Rastrigin function is continuous, scalable, separable, and a highly multimodal global optimization function. This test function is defined as follows:

$$f(x) = 10d + \sum_{i=1}^{d} (x_i^2 - 10\cos(2\pi x_i)) \quad (11)$$

where d represents the number of dimensions and $x_i = (x_1, x_2, ..., x_d)$ is a d-dimensional row vector. The test region is typically assessed within the range of [−5.12, 5.12]^d. The global minimum $(x) = 0$ is achievable at $x_i = (0,0)$. This test function poses significant challenges for numerous global optimization techniques. Figure 3 shows the surface plot of Rastrigin function.

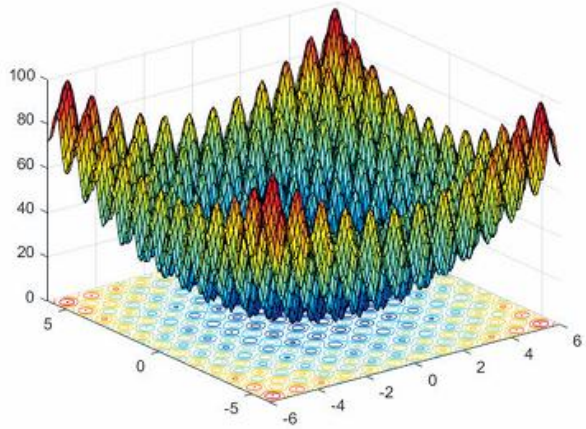

Figure 3: Rastrigin function -surface plot

Rosenbrock is a unimodal, valley-shaped function defined for dimensions d ≥ 2. This test function is continuous, scalable, inherently nonseparable, nonconvex, and unimodal. This test function is defined as follows:

$$f(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - (x_i)^2)^2 + (x_i - 1)^2] \quad (12)$$

where d > 2 represents the number of dimensions and $x_i = (x_1, x_2, ..., x_d)$ denotes a d-dimensional row vector. The test region is typically assessed within the range of [−2.048, 2.048]^d, where $i = (1,...,d)$. The global minimum $(x) = 0$ is achievable at $xi = (1,1)$. The function is mostly recognized for its exceedingly slow convergence at the minimum point. Figure 4 shows the surface plot of Rosenbrock function.

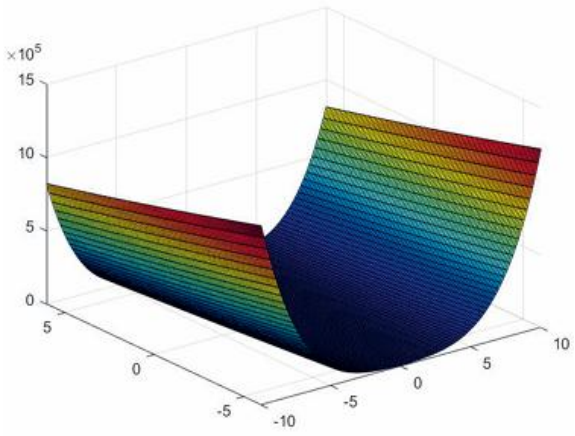

Figure 4: Rosenbrock function - surface plot

4.2 Results

This section compares the performance of the optimization approaches described. In each instance, function reduction was conducted commencing from 50 randomly chosen initial points within the search space of the test function. The comparisons were conducted throughout time, assessing the success rate of the optimizations. An optimization initiated from a specified set of beginning conditions is deemed successfully convergent to the global optimum if $|f(x) - f(x^*)| \leq 10^{-4}$, where $f(x*)$ and $f(x)$ represent the known accurate global minimum and the lowest obtained value of the objective function, respectively. Conversely, the optimization is deemed unsuccessful.

Table 2 presents the findings derived from the Ackley function. The GA and DQN approaches effectively converged to the global optimum, but the gradient descent and Newton methods failed to converge in all instances. The PSO approach achieved convergence in 98% of the total runs. The time for numerical methods are low compared to machine learning methods.

Table 2: Result of Ackley function for different methods

Method	Time (Sec)	Convergence Rate (%)
Gradient Descent	0.037	0
Newton's	0.0085	0
GA	0.17	100
PSO	0.051	98
DQN	0.55	100

Table 3, presents the findings derived from the Rastrigin function. The GA and DQN approaches effectively converged to the global optimum, but the gradient descent and Newton methods failed to converge in all instances. The PSO approach achieved convergence in 70% of the total runs. The time for numerical methods are low compared to machine learning methods.

Table 3: Result of Rastrigin Function for Different Methods

Method	Time (Sec)	Convergence Rate (%)
Gradient Descent	0.065	0
Newton's	0.010	0
GA	0.15	100
PSO	0.075	70
DQN	0.98	100

Table 4 presents the findings derived from the Rosenbrock function. In this case, the GA and DQN approaches has high convergence rate (98% and 90%), but the other approaches has lowered convergence rate. The time for numerical methods are low compared to machine learning methods.

Table 4: Result of Rosenbrock function for different methods

Method	Time (Sec)	Convergence Rate (%)
Gradient Descent	0.081	50
Newton's	0.0230	76
GA	2.5	98
PSO	1.9	65
DQN	3.2	90

4.3 Discussion

When comparing Numerical Optimization and Machine Learning Optimization methods for complicated functions, it is essential to assess their performance based on many factors, including accuracy, efficiency, scalability, robustness, and application. Each optimization technique presents unique advantages and disadvantages contingent upon the characteristics of the problem (e.g., differentiability, dimensionality, noise, etc.). Table 5 shows the comparison of numerical and machine learning optimization.

Table 5: Comparison of numerical Vs machine learning optimization

Criterion	Numerical Optimization	Machine Learning Optimization
Objective	Optimize a designated objective function (e.g., cost function).	Optimize loss or error functions utilizing models (e.g., neural networks, regression models).

Type of Functions Handled	Functions exhibiting explicit, deterministic characteristics (frequently differentiable)	Intricate, non-differentiable, stochastic, or opaque functions
Optimization Algorithms	Gradient-dependent methods (e.g., steepest descent, Newton's method) or gradient-independent methods (e.g., genetic algorithms, simulated annealing).	Stochastic algorithms (e.g., gradient descent, Adam, genetic algorithms), evolutionary strategies.
Computational Complexity	The cost is contingent upon the employed method; it may be computationally intensive for high-dimensional or non-linear functions.	May incur significant computing costs, particularly in high-dimensional environments (e.g., deep learning architectures).
Convergence Speed	Typically more efficient for smooth, convex, or differentiable functions.	Convergence may require an extended duration, particularly when addressing high-dimensional and noisy objective functions.
Scalability	May encounter difficulties in high-dimensional spaces, particularly with gradient-based techniques.	Capable of scaling to high-dimensional areas, particularly in deep learning or evolutionary algorithms.
Use of Derivatives	Heavily depends on gradients or Hessians (second-order derivatives) for convergence.	Frequently operates with or without derivatives (e.g., gradient-free techniques or models like as neural networks).
Ability to Handle Noise	Susceptible to noise and necessitates smooth functions or noise mitigation techniques such as regularization.	Resilient to noisy data and frequently capable of identifying optimal solutions in circumstances characterized by noise or insufficient data.
Global vs. Local Optima	Susceptible to entrapment in local minima for non-convex functions.	Local minima can be circumvented using methods such as stochastic gradient descent or evolutionary algorithms.
Flexibility	Most appropriate for issues with a well delineated objective function and restrictions.	Extremely adaptable and applicable to various issues, including regression, classification, and black-box optimization.
Learning and Adaptability	Generally does not get knowledge from the optimization process.	Models can "adapt" over time (e.g., neural networks modifying weights).
Examples of Use	Engineering design, optimal resource distribution, and resolution of physical models.	Machine learning activities (e.g., neural network training, hyperparameter tuning, reinforcement learning).
Robustness	Sensitive to starting conditions (e.g., first guess for gradient algorithms).	Resilient to initialization and noise in extensive datasets, particularly with ensemble techniques.
Interpretability	Generally more straightforward to interpret solutions, particularly for less complex models (e.g., linear programming).	Frequently less interpretable, especially in deep learning, where models may function as "black boxes".

Numerical optimization is particularly efficient for well-defined, differentiable, and smooth objective functions. This approach is typically more efficient for such issues but may encounter difficulties with non-linearity, large dimensionality, and noisy data. Machine Learning Optimization is particularly effective for high-dimensional, non-differentiable, or noisy functions. Machine learning techniques can exhibit greater adaptability and scalability; yet, they may require extended convergence periods and might occasionally lack interpretability compared to numerical methods.

5 Conclusion

Optimizing complex functions presents a formidable challenge that necessitates meticulous evaluation of both numerical techniques and machine learning methodologies. Conventional optimization techniques such as gradient descent, Newton's method, and evolutionary algorithms are well-recognized. Machine learning optimization techniques provide an array of instruments for addressing intricate function optimization, particularly when conventional numerical optimization approaches are inadequate due to non-linearity, non-differentiability, or high dimensionality of the issue. Utilizing techniques such as Bayesian Optimization and Deep Reinforcement Learning, machine learning offers enhanced flexibility and robustness, facilitating solutions for intricate real-world challenges across several domains. The benchmark optimization test functions Ackley, Rastrigin, and Rosenbrock are employed to assess the optimization methods for convergence rate and execution time. The results demonstrate that numerical optimization approaches exhibit a low convergence rate and diminished runtime, but machine learning optimization displays a high convergence rate and prolonged runtime. In the future, the hybrid optimization methods are considered to solve complex function. The hybrid approach can

combine numerical and machine learning algorithm to obtain the best efficient results.

References

[1] Liu, C., Niu, P., Li, G., Ma, Y., Zhang, W., & Chen, K. (2018). Enhanced shuffled frog-leaping algorithm for solving numerical function optimization problems. *Journal of Intelligent Manufacturing*, *29*, 1133-1153. https://doi.org/10.1007/s10845-015-1164-z

[2] Li, G., Cui, L., Fu, X., Wen, Z., Lu, N., & Lu, J. (2017). Artificial bee colony algorithm with gene recombination for numerical function optimization. *Applied Soft Computing*, *52*, 146-159. https://doi.org/10.1016/j.asoc.2016.12.017

[3] Fera, M., Fruggiero, F., Lambiase, A., Macchiaroli, R., & Todisco, V. (2018). A modified genetic algorithm for time and cost optimization of an additive manufacturing single-machine scheduling. *International Journal of Industrial Engineering Computations*, *9*(4), 423-438. https://doi.org/10.5267/j.ijiec.2018.1.001

[4] Nazir, A., Akhyar, A., Yusra, Y., & Budianita, E. (2022). Toddler nutritional status classification using C4. 5 and particle swarm optimization. *Sci. J. Informatics*, *9*(1), 32-41. https://doi.org/10.15294/sji.v9i1.33158

[5] Azevedo, B. F., Brito, T., Lima, J., & Pereira, A. I. (2021). Optimum sensors allocation for a forest fires monitoring system. *Forests*, *12*(4), 453. https://doi.org/10.3390/f12040453

[6] Zhao, R., & Tang, D. (2024). The integration of financial business and the transformation of financial management functions based on internal control optimization algorithm. *Informatica*, *48*(10). https://doi.org/10.31449/inf.v48i10.5672

[7] Azevedo, B., Bressan, G., Agulhari, C., Santos, H., & Endo, W. (2019). Three-phase induction motors faults classification using audio signals and decision trees. *Applied Mathematics and Information Sciences*, *13*(5), 847-858. http://dx.doi.org/10.18576/amis/130519

[8] Telikani, A., Tahmassebi, A., Banzhaf, W., & Gandomi, A. H. (2021). Evolutionary machine learning: A survey. *ACM Computing Surveys (CSUR)*, *54*(8), 1-35. https://doi.org/10.1145/3467477

[9] Nemmich, M. A., Debbat, F., & Slimane, M. (2020). Hybrid bees approach based on improved search sites selection by firefly algorithm for solving complex continuous functions. *Informatica*, *44*(2). https://doi.org/10.31449/inf.v44i2.2385

[10] Pan, J. S., Liu, N., Chu, S. C., & Lai, T. (2021). An efficient surrogate-assisted hybrid optimization algorithm for expensive optimization problems. *Information Sciences*, *561*, 304-325. https://doi.org/10.1016/j.ins.2020.11.056

[11] Sohrabi, M. K., & Azgomi, H. (2020). A survey on the combined use of optimization methods and game theory. *Archives of Computational Methods in Engineering*, *27*(1), 59-80. https://doi.org/10.1007/s11831-018-9300-5

[12] Lu, Y., Wang, S., Zhao, Y., & Yan, C. (2015). Renewable energy system optimization of low/zero energy buildings using single-objective and multi-objective optimization methods. *Energy and Buildings*, *89*, 61-75. https://doi.org/10.1016/j.enbuild.2014.12.032

[13] Pereira, J. L. J., Oliver, G. A., Francisco, M. B., Cunha Jr, S. S., & Gomes, G. F. (2022). A review of multi-objective optimization: methods and algorithms in mechanical engineering problems. *Archives of Computational Methods in Engineering*, *29*(4), 2285-2308. https://doi.org/10.1007/s11831-021-09663-x

[14] Rosso, M. M., Cucuzza, R., Aloisio, A., & Marano, G. C. (2022). Enhanced multi-strategy particle swarm optimization for constrained problems with an evolutionary-strategies-based unfeasible local search operator. *Applied Sciences*, *12*(5), 2285. https://doi.org/10.3390/app12052285

[15] Abdollahzadeh, B., Gharehchopogh, F. S., Khodadadi, N., & Mirjalili, S. (2022). Mountain gazelle optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. *Advances in Engineering Software*, *174*, 103282. https://doi.org/10.1016/j.advengsoft.2022.103282

[16] Gambella, C., Ghaddar, B., & Naoum-Sawaya, J. (2021). Optimization problems for machine learning: A survey. *European Journal of Operational Research*, *290*(3), 807-828. https://doi.org/10.1016/j.ejor.2020.08.045

[17] Krishnan, R., Rajpurkar, P., & Topol, E. J. (2022). Self-supervised learning in medicine and healthcare. *Nature Biomedical Engineering*, *6*(12), 1346-1352. https://doi.org/10.1038/s41551-022-00914-1

[18] Naeem, S., Ali, A., Anam, S., & Ahmed, M. M. (2023). An unsupervised machine learning algorithm: Comprehensive review. *International Journal of Computing and Digital Systems*. https://doi.org/10.12785/ijcds/130172

[19] Matsuo, Y., LeCun, Y., Sahani, M., Precup, D., Silver, D., Sugiyama, M., ... & Morimoto, J. (2022). Deep learning, reinforcement learning, and world models. *Neural Networks*, *152*, 267-275. https://doi.org/10.1016/j.neunet.2022.03.037

[20] Albreiki, F., Belayouni, N., & Gupta, D. K. (2023). On Using Deep Learning Proxies as Forward Models in Deep Learning Problems. *arXiv preprint arXiv:2301.07102*. https://doi.org/10.48550/arXiv.2301.07102

[21] Tunay, M., & Abiyev, R. (2022). Improved hypercube optimisation search algorithm for optimisation of high dimensional functions. *Mathematical Problems in Engineering*, *2022*(1), 6872162. https://doi.org/10.1155/2022/6872162

[22] Demo, N., Tezzele, M., & Rozza, G. (2021). A supervised learning approach involving active subspaces for an efficient genetic algorithm in high-dimensional optimization problems. *SIAM Journal on Scientific Computing*, *43*(3), B831-B853. https://doi.org/10.1137/20m1345219

[23] Goitom, S., Nagy, T., & Turányi, T. (2022, January). TESTING VARIOUS NUMERICAL OPTIMIZATION METHODS ON A SERIES OF ARTIFICIAL TEST FUNCTIONS. In *Annales Universitatis Scientiarum Budapestinensis de Rolando Eötvös Nominatae. Sectio Computatorica* (Vol. 53). https://doi.org/10.71352/ac.53.175

[24] Optimization Test Functions. https://www.sfu.ca/~ssurjano/ackley.html