Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 2748215, 22 pages
https://doi.org/10.1155/2022/2748215

Research Article

An Analysis of the Operation Factors of Three PSO-GA-ED Meta-Heuristic Search Methods for Solving a Single-Objective Optimization Problem

Ali Fozooni,[1] Osman Kamari,[2] Mostafa Pourtalebiyan,[3] Masoud Gorgich,[4] Mohammad Khalilzadeh,[5,6] and Amin Valizadeh [7]

[1]*Foster School of Business, University of Washington, Seattle, WA 98105, USA*
[2]*Department of Business Management, University of Human Development, Sulaymaniyah, Iraq*
[3]*Department of Industrial Engineering, University of Science and Culture, Tehran, Iran*
[4]*Department of Industrial Engineering, Velayat University, Iranshahr, Iran*
[5]*CENTRUM Católica Graduate Business School, Lima, Peru*
[6]*Pontificia Universidad Católica del Perú, Lima, Peru*
[7]*Department of Mechanical Engineering, Ferdowsi University of Mashhad, Mashhad, Iran*

Correspondence should be addressed to Amin Valizadeh; amin.valizadeh@mail.um.ac.ir

Received 29 May 2022; Revised 4 July 2022; Accepted 6 July 2022; Published 14 October 2022

Academic Editor: Dalin Zhang

In this study, we evaluate several nongradient (evolutionary) search strategies for minimizing mathematical function expressions. We developed and tested the genetic algorithms, particle swarm optimization, and differential evolution in order to assess their general efficacy in optimization of mathematical equations. A comparison is then made between the results and the efficiency, which is determined by the number of iterations, the observed accuracy, and the overall run time. Additionally, the optimization employs 12 functions from Easom, Holder table, Michalewicz, Ackley, Rastrigin, Rosen, Rosen Brock, Shubert, Sphere, Schaffer, Himmelblau's, and Spring Force Vanderplaats. Furthermore, the crossover rate, mutation rate, and scaling factor are evaluated to determine the effectiveness of the following algorithms. According to the results of the comparison of optimization algorithms, the DE algorithm has the lowest time complexity of the others. Furthermore, GA demonstrated the greatest degree of temporal complexity. As a result, using the PSO method produces different results when repeating the same algorithm with low reliability in terms of locating the optimal location.

1. Introduction

A nongradient optimization method is a stochastic method, which means that, unlike gradient optimization, the results are heavily randomized. A scenario similar to Darwinian evolution is simulated in which the closest point to a maximum or a minimum value is selected as the optimal point in a function [1–4]. Unlike gradient methods, evolutionary optimization does not heavily rely on mathematics, and the initial starting point does not have nearly as much impact. Because of the random nature of evolutionary optimization, it is mostly less efficient than gradient-based optimization since it does not even guarantee an optimal solution [5, 6]. However, the method is more aggressive and considers more solutions than gradient methods do, allowing it to find multiple local minima points, which give it some advantages. The way evolutionary optimization works is that first, one must generate a mathematical function to create a scenario with specific conditions and then various points will be randomly plotted throughout the function in ideal locations [7–10]. The results will be compared and then used to converge throughout the function. These results then adapt and converge toward the optimized points chaotically through trial and error. The step size for updating unknowns

is generally required when applying gradient-based optimization algorithms [11, 12]. To achieve better generalization and convergence, learning rate scheduling schemes have been used in addition to the fixed learning rate. Staircases [13] and exponential decay [40] are simple, but popular schemes for reducing stochastic noises. AdaGrad [14], AdaDelta [15, 16], RMSprop [17], and Adam [18] have also been developed for parameterwise adaptive learning rate scheduling. So while finding the optimal point is not guaranteed, it is at least possible to find these points' potential locations.

Since evolutionary optimization has a variety of starting points, it is not subject to the same weakness as gradient optimization. Gradient optimization accurately converges on the local minima. The function, however, does not know whether it has reached the global minima. As a result, less-optimal solutions are often reached than what is possible [18]. With evolutionary optimization, starting points are all across the function, which raises the probability of one starting near the global minima. They all converge toward their local minima, and the results are then compared. Based on these, we can more easily approximate the global minima within the bounds of our function. The best results, in general, can come from combining gradient and nongradient-based optimization to converge on the best solution, for this one would start with the broad function and implement evolutionary optimization [19, 20].Despite the fact that it is not very analytical, it would often instinctively converge near the global minima, providing an indication of the general location. Afterward, a gradient-based algorithm may be used with the determined area as a starting point. Using a mathematical function, it will converge toward the global minima and provide an accurate result. It is possible to find the global minima for any function by combining the two algorithm types accurately (see Table 1).

Multiobjective optimization has been applied in many fields of science, including engineering, economics, and logistics where optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives. There are many applications in computer science such as cloud computing [28–30], image processing, [31], medical science [32], robotics and mechanics [33], controller design [34], wireless sensor network [35–37], architectural design [38], and metaheuristic methods convergence [39, 40]. There are some other applications for prediction methods, Feynman's Path Integral [41], Semantic Segmentation [42], Internet of things [43, 44], Signal processing [45], distributed networks [46, 47], and Software Defect Prediction [48]. Some other optimization methods are adaptive regeneration framework [49], Mean Extra-Gradient [50], Bi-LSTMC [51], random key genetic algorithm [52], and Complementary-Label Source Domain [53]. Moreover, signal processing fields include Ultrawideband Rejection [53], GaAs technology [54], Visual question answering [55], Visual Reasoning [56, 57], Semantic Network [58, 59], attack detection [60], Smart Homes [61], Fog computing [62], Neural Tracking [63], light detection [64], Buffering Algorithm [65], decision making [66], classification [57, 67–69], Growth Cycles [70], Remote sensing [71], power generation [72–74], vehicle routing problem [75], and structure design [76, 77].

2. Methods and Materials

2.1. Genetic Algorithm. The genetic algorithm is a learning program that mimics natural evolution concepts such as reproduction crossover and mutation to produce what the program considers optimal offspring. It is the most general type of evolutionary optimization. It takes the general ideas behind it and puts them into action. It starts with various points spread randomly throughout the function, taking into account the various possible solutions within the problem's parameters. It allows the program to consider various possible solutions and focus on each of them to determine the best one. Once the algorithm has its values, it calculates each solution's fitness generated in the function. Then a pair of solutions can be selected so long as they increase the chances of generating offspring; each parent can be used more than once per iteration to generate offspring. Once the points are selected, cross over them to create two new potential solutions. Otherwise, plot the new points over the parent points. Finally, you mutate the new points and generate the resulting points.

The way that selection occurs is by comparing potential parents with potential partners in its local area. The values with a higher fitness value are more likely to produce offspring than those with lower fitness to better simulate evolution. Selection is often made by random chance, with the high fitness results being more likely to be picked. The probability of selection (pi) is represented by equation (1), with f_i being the fitness value of individual i and N being the local population relative to a parent:

$$p_i = \frac{f_i}{\sum_{j=i}^{N} f_j}. \tag{1}$$

The algorithm uses a crossover process to generate two new values to plot into the next iteration when selection is complete. These new values perturb old solutions as they try to steer away from the flaws. The general equation for the crossover stage is shown below for y_k and x_k, respectively, where α is the crossover blending factor and r_k is the uniformly distributed random number in the interval [0, 1]. However, some highly successful members of the next iteration are allowed to remain the same as they were beforehand:

$$\begin{aligned} y_k &= (1 + 2\alpha) r_k - \alpha, \\ x_k^{(i,t+1)} &= (1 - y_k) x_k^{(1,t)} + y_k x_k^{(2,t)}, k = 1, \ldots, n \text{ var}. \end{aligned} \tag{2}$$

To prevent the new iterations from becoming the same and promote more out-of-the-box solutions, a mutation factor is used to diversify the solutions and prevent the population from becoming stagnant. A mutation is a deviation from the crossover logic, which randomizes the solutions generated to hurl them closer or further from the end goal or toward another goal. The equations used to determine the mutation effect is shown below, with r being a

Table 1: Review of nongradient-based methods and their application.

Author	Year	Gradient-based method	Application	Results
Chen et al. [21]	2022	Evolutionary optimization	Learning variational quantum reinforcement	It provided a natural way to compress the input dimension efficiently, enabling further quantum RL uses on noisy intermediate-scale quantum devices
Zhang et al. [2]	2021	Nongradient topology optimization in acoustic meta-materials	Realizing a complete and directional bandgap design	The optimized designs converged to show the orderly material distribution and numerical validations to show expected propagation properties
Pazouki [22]	2021	Volume balance model and multiobjective evolutionary optimization algorithms.	Designing a practical surface irrigation system	The proposed model in most fields and indicators achieve better results, and the results are close together
Dhiman et al. [23]	2021	A new evolutionary multiobjective optimization algorithm for global optimization	To map out seagulls better than modern optimization algorithms	The empirical research indicates that the EMoSOA algorithm works better than other algorithms
Pan et al. [24]	2021	An efficient surrogate-assisted hybrid optimization algorithm	Solving expensive optimization problems	The hybrid algorithm works better than preexisting ones, able to solve problems that were previously unattainable
Abualigah et al. [25]	2021	A novel evolutionary arithmetic optimization algorithm	Multilevel thresholding segmentation of covid-19 ct images	The DAOA produces higher quality solutions than other similar approaches and is ranked the best for various test cases
Naeimi et al. [26]	2021	A nature-inspired algorithm for high-dimensional optimization problems	To develop an algorithm based on horses' herding behaviors for high-dimensional optimization	The proposed algorithm proved to be highly efficient for solving serious dimensional global optimization problems, outperforming the standard algorithms used today in terms of accuracy and efficiency
Meraihi et al. [27]	2021	Genetic algorithm optimization	To develop an algorithm based on the foraging and swarming behaviors of grasshoppers	The GOA algorithm gives superior results for most applications, having a high exploitation ability and convergence and excelling at preventing local minima stagnation.

uniformly distributed number in the interval [0, 1], x_k^l and x_k^u being the upper and lower bounds of x kT is the number of generations, T is the maximum number of generations, b is the strength of the mutation operator, and the function for y is given by $\Delta(t, y)$:

$$\begin{aligned} x_k^{(i,t+1)} &= x_k^{(1,t)} + \Delta\left(t, x_k^u - x_k^{(2,t)}\right) r \le 0.5, k = 1, \ldots, n, \\ x_k^{(i,t+1)} &= x_k^{(1,t)} - \Delta\left(t, x_k^{(2,t)} - x_k^{(2,t)}\right) r > 0.5, k = 1, \ldots, n, \\ \Delta(t, y) &= y\left(1 - r^{1-t/T}\right)^b. \end{aligned} \tag{3}$$

2.2. Particle Swarm Optimization. In 1995, electrical engineer Russel Eberhart and social psychologist James Kennedy developed this alternative to the genetic algorithm. This nongradient algorithm considers the individuality and sociability of the population members. Specifically, the idea came from watching birds look for a nesting place. Not enough individuality led to too many birds trying to nest in the same place. However, not enough sociability led to many birds unable to find suitable nesting places. In general, the program uses social rules and individual deviations to find the ideal locations. It is calculated by accounting for the velocity vector of each particle as they travel. The vector considers the pack movement and individual instinct that goes into its movement and adds it to the initial inertia of the iteration. The basic equation for particle swarm vector optimization is shown below, with α being the inertia factor, β_1 being the individuality factor, β_2 being the sociability factor, $r_1^{(i)}$ and $r_2^{(i)}$ being uniformly distributed numbers in the interval [0, 1], $X^{(i,t)}$ being the individual's vector, $P^{(i)}$ being the best individual value and $P^{(i)}$ being the best value in the population. Within the vector equation, $\alpha v^{(i,t)}$ represents the inertia, $\beta_1 r_1^{(i)}(P^{(i)} - X^{(i,t)})$ represents the individuality, and $\beta_2 r_2^{(i)}(P^{(g)} - X^{(i,t)})$ represents sociability:

$$\begin{aligned} v^{(i,t+1)} &= \alpha v^{(i,t)} + \beta_1 r_1^{(i)}\left(P^{(i)} - X^{(i,t)}\right) + \beta_2 r_2^{(i)}\left(P^{(g)} - X^{(i,t)}\right), \\ X^{(i,t+1)} &= X^{(i,t)} + v^{(i,t+1)}. \end{aligned} \tag{4}$$

Other than this, it functions like the genetic algorithm; it begins with many solutions on the field. Each solution is evaluated for fitness. The result is compared to their previous swarm fitness, and the previous individual fitness and its position are updated accordingly. Its best individual fitness and position are then used to calculate the next iteration.

All in all, particle swarm optimization edges out over the genetic algorithm, namely, because it does not need to sort fitness as the genetic algorithm does. It means that swarm optimization requires less-computational power. It tends to be cheaper to use than the genetic algorithm, especially with many values.

Table 2: The properties of optimization functions.

Function	Variable	X domain	Y domain	Global minimum	Optimum value
Ackley	2	$-5 \le x \le 5$	$-5 \le y \le 5$	$x^* = (0, 0)$	$f(x^*) = 0$
Easom	2	$-10 \le x \le 10$	$-10 \le y \le 10$	$x^* = (\pi, \pi)$	$f(x^*) = -1$
Holder table	2	$-10 \le x \le 10$	$-10 \le y \le 10$	$x^* = (8.06, 9.66)$ $x^* = (-8.06, 9.66)$ $x^* = (8.06, -9.66)$ $x^* = (-8.06, -9.66)$	$f(x^*) = -19.21$
Michalewicz	2	$0 \le x \le \pi$	$0 \le y \le \pi$	$x^* = (2.20, 1.57)$	$f(x^*) = -1.80$
Rastrigin	2	$-5 \le x \le 5$	$-5 \le y \le 5$	$x^* = (0, 0)$	$f(x^*) = 0$
Rosen	2	$0 \le x \le 6$	$0 \le y \le 6$	$x^* = (5.33, 5.33)$	$f(x^*) = -18.57$
Rosenbrock	2	$-2 \le x \le 2$	$-5 \le y \le 4$	$x^* = (1, 1)$	$f(x^*) = 0$
Shubert	2	$-10 \le x \le 10$	$-10 \le y \le 10$	$x^* = (1.67, -2.01)$	$f(x^*) = -186.73$
Sphere	2	$-5 \le x \le 5$	$-5 \le y \le 5$	$x^* = (0, 0)$	$f(x^*) = 0$
Schaffer	2	$-10 \le x \le 10$	$-10 \le y \le 10$	$x^* = (0, 0)$	$f(x^*) = 0$
Himmelblau	2	$-6 \le x \le 6$	$-6 \le y \le 6$	$x^* = (3, 2)$	$f(x^*) = 0$
Spring force Vanderplaats	2	$-20 \le x \le 20$	$-20 \le y \le 20$	$x^* = (0, 0)$	$f(x^*) = 0$

2.3. Differential Evolution. Differential evolution was developed around 1955 and was made to try simulating Darwinian evolution. It combines the parents' features to form a child. However, unlike previous methods, the new value may inherit features from multiple parents. It is the closest to gradient optimization that evolution optimization can get in this assignment. It is used for multidimensional real-valued functions without needing it to be differentiable, making it a robust algorithm.

Using two different parent equation values ($P1$ and $P2$), the method produces a series of children ($C1,\ldots,C_n$). In these equations, α, β, and γ are random parent features, m is the mutation factor between 0.5 and 1, and δ_1 and δ_2 are binomial crosses over coefficients. CR is the crossover, while R represents a random number with distribution [0, 1]:

$$\begin{aligned} P_n &= \alpha + m(\beta - \gamma), \\ C_n &= P_2\delta_1 + P_1\delta_2, \\ X_i^{k+1} &= \delta_1 X_i^k + \delta_2(\alpha + m(\beta - \gamma)), \\ \delta_1 &= 0\,(\text{if } R < CR), \\ \delta_1 &= 1\,(\text{if } R > CR), \\ \delta_2 &= 0\,(\text{if } R > CR), \\ \delta_2 &= 1\,(\text{if } R < CR). \end{aligned} \tag{5}$$

It is an algorithm that only acts when the product of the two-parent points produces a child with better fitness. When weighing its options on its results, it always selects the offspring with the excellent fitness. It abandons the rest, increasing the efficiency of the evolution. Furthermore, any improvements found by the function will be immediately included. As a result, the general solution is often more accurate than in either the genetic algorithm or particle swarm optimization.

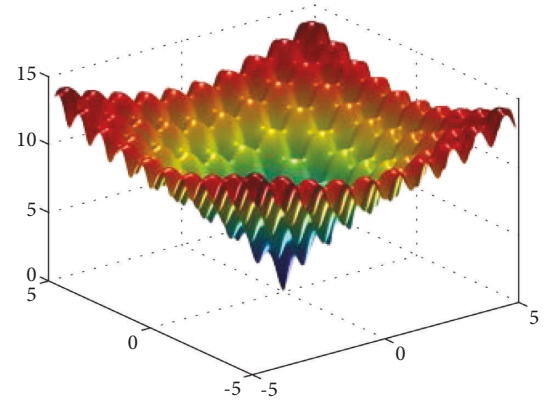

Figure 1: Ackley function.

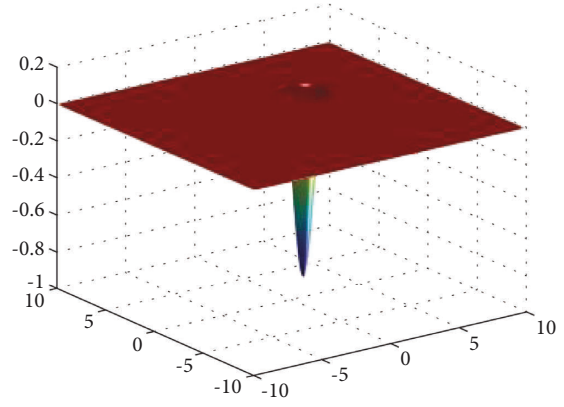

Figure 2: Easom function.

3. Results and Discussion

In this report, we used three meta-heuristic algorithms of genetic algorithm, particle swarm optimization, and differential evaluation as two nongradient-based methods for optimization of some mathematical surfaces. In this report,

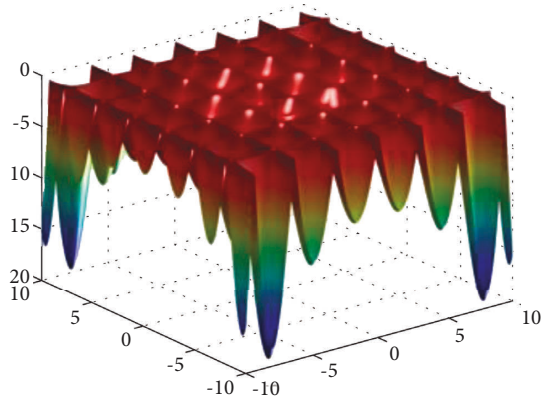

Figure 3: Holder table function.

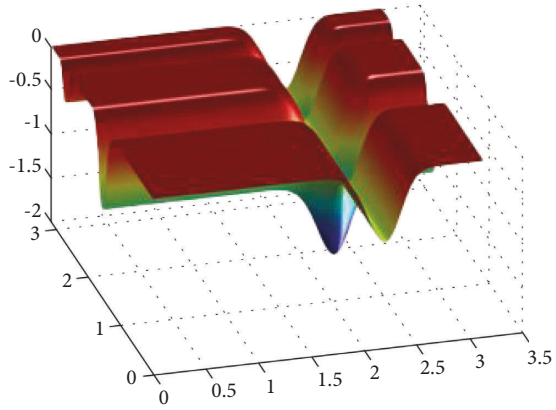

Figure 4: Michalewicz function.

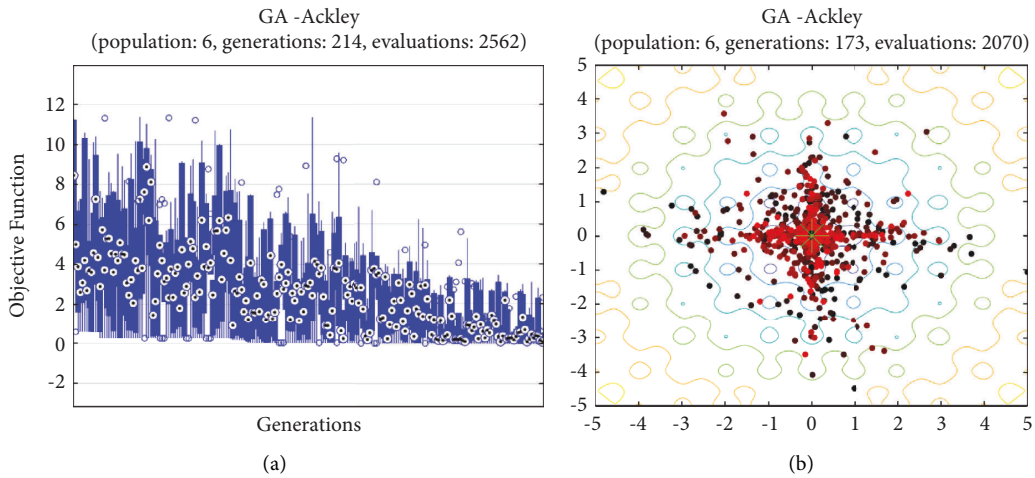

Figure 5: Genetic algorithm results for Ackley function: (a) objective function in each generation, (b) plot of populations accumulation.

12 functions of Easom, Holder table, Michalewicz, Ackley, Rastrigin, Rosen, Rosenbrock, Shubert, Sphere, Schaffer, Himmelblau's, and Spring Force Vanderplaats are employed for the optimization. The properties of these functions are as follows (see Table 2):

In this report, we used GA to optimize Easom, Holder table, Michalewicz, Ackley functions shown in Figures 1–12. Moreover, Figures 5, 7, 9, and 11 illustrate the objective function values in each generation of genetic algorithm and plot of populations accumulation to find the optimum value.

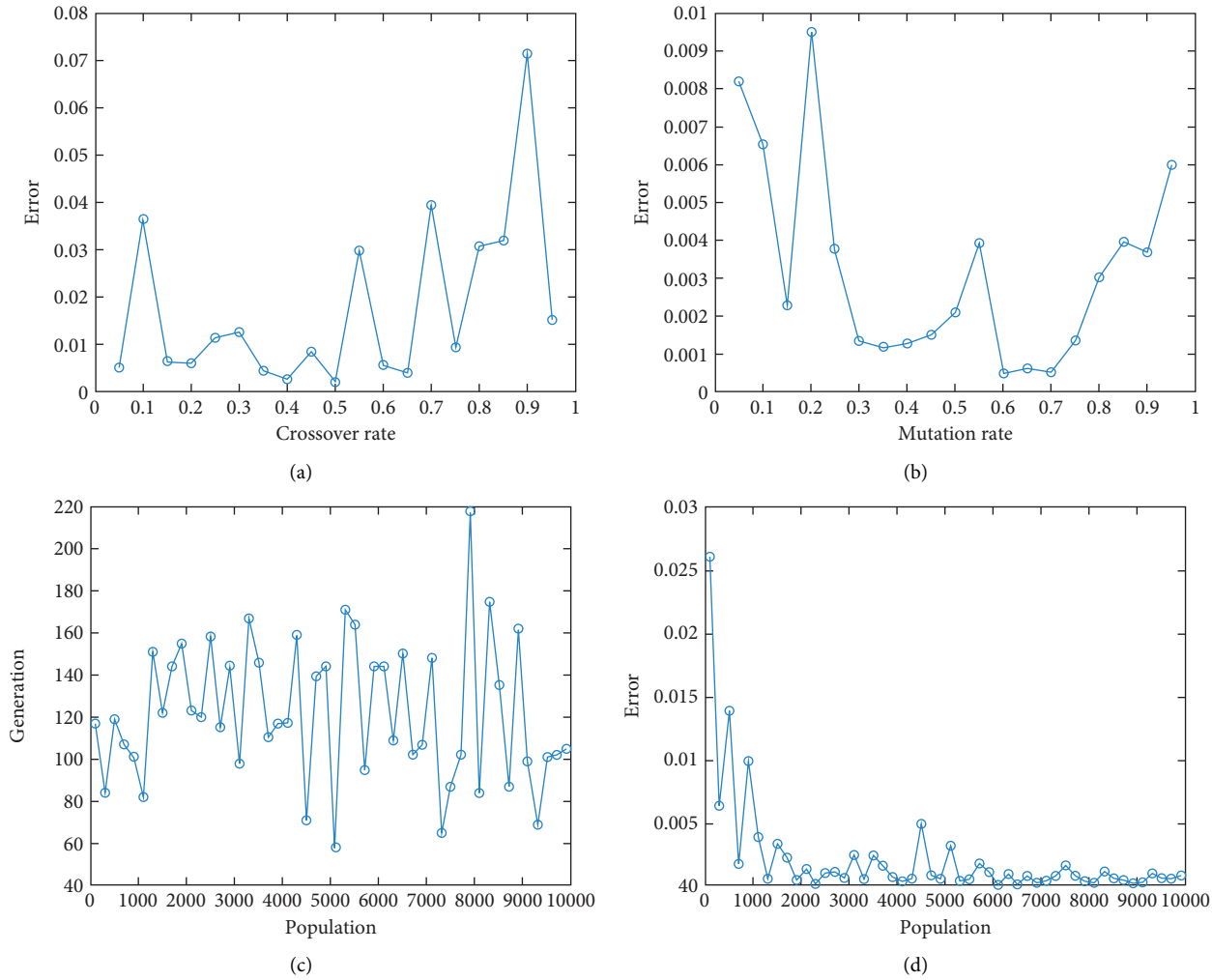

Figure 6: Genetic algorithm results for Ackley function: (a) error value based on increasing on crossover rate, (b) error value based on the mutation rate, (c) number of generation vs. population, (d) Error value with increase in population.

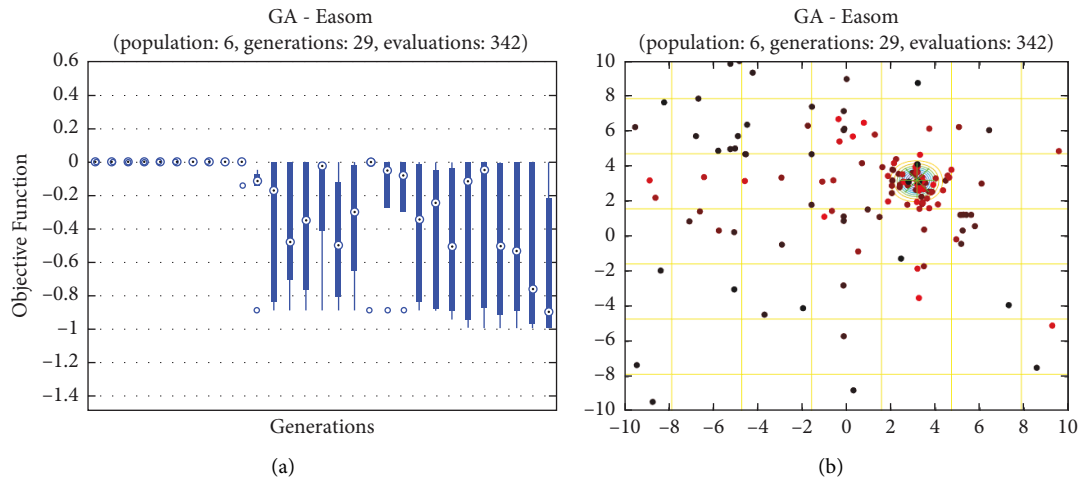

Figure 7: Genetic algorithm results for Easom function: (a) objective function in each generation, (b) plot of populations accumulation.

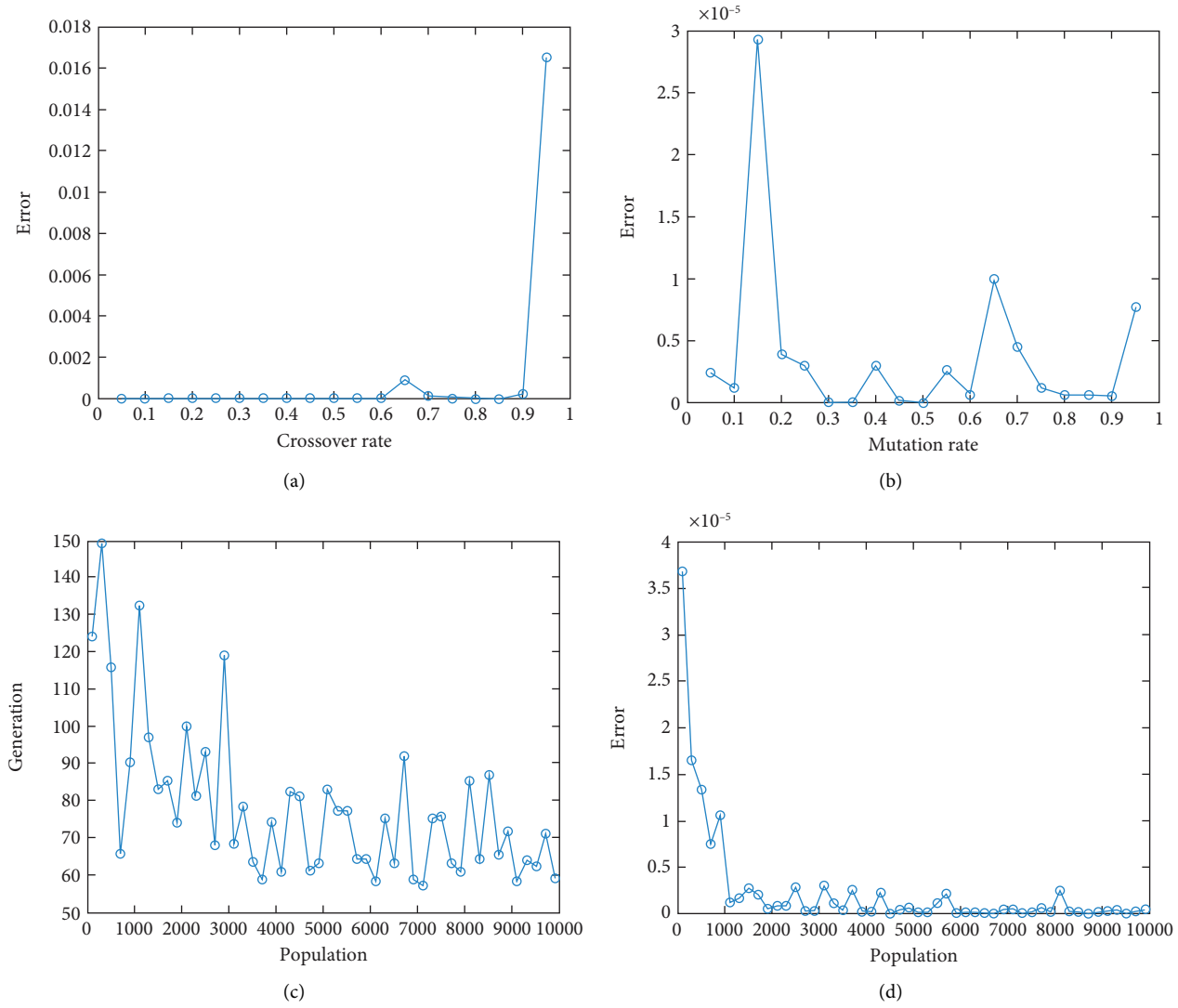

Figure 8: Genetic algorithm results for Easom function: (a) error value based on increasing on crossover rate, (b) error value based on the mutation rate, (c) number of generation vs. population, (d) error value with increase in population.

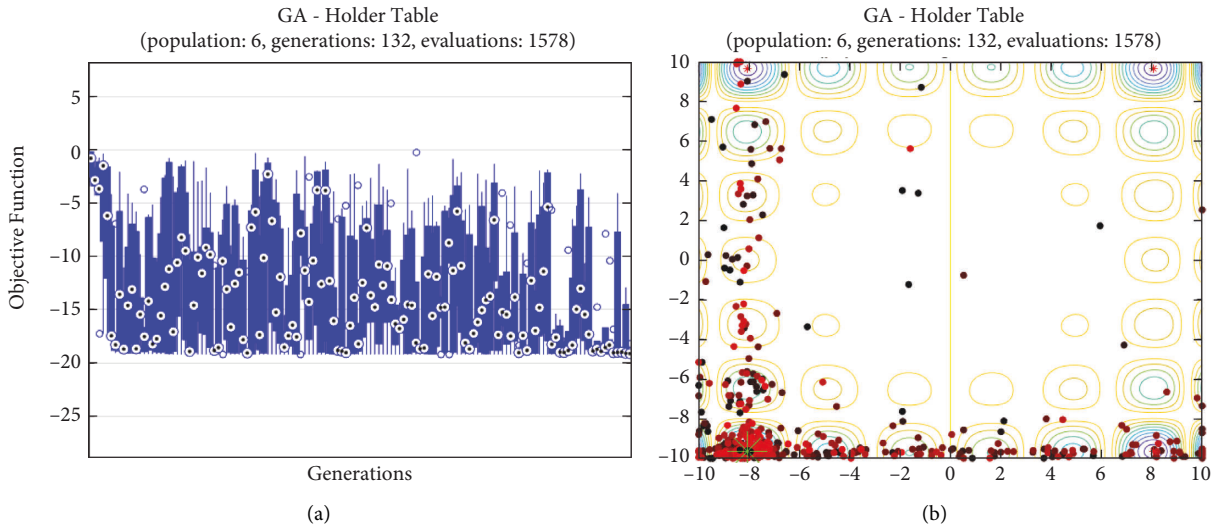

Figure 9: Genetic algorithm results for holder table function: (a) objective function in each generation, (b) plot of populations accumulation.

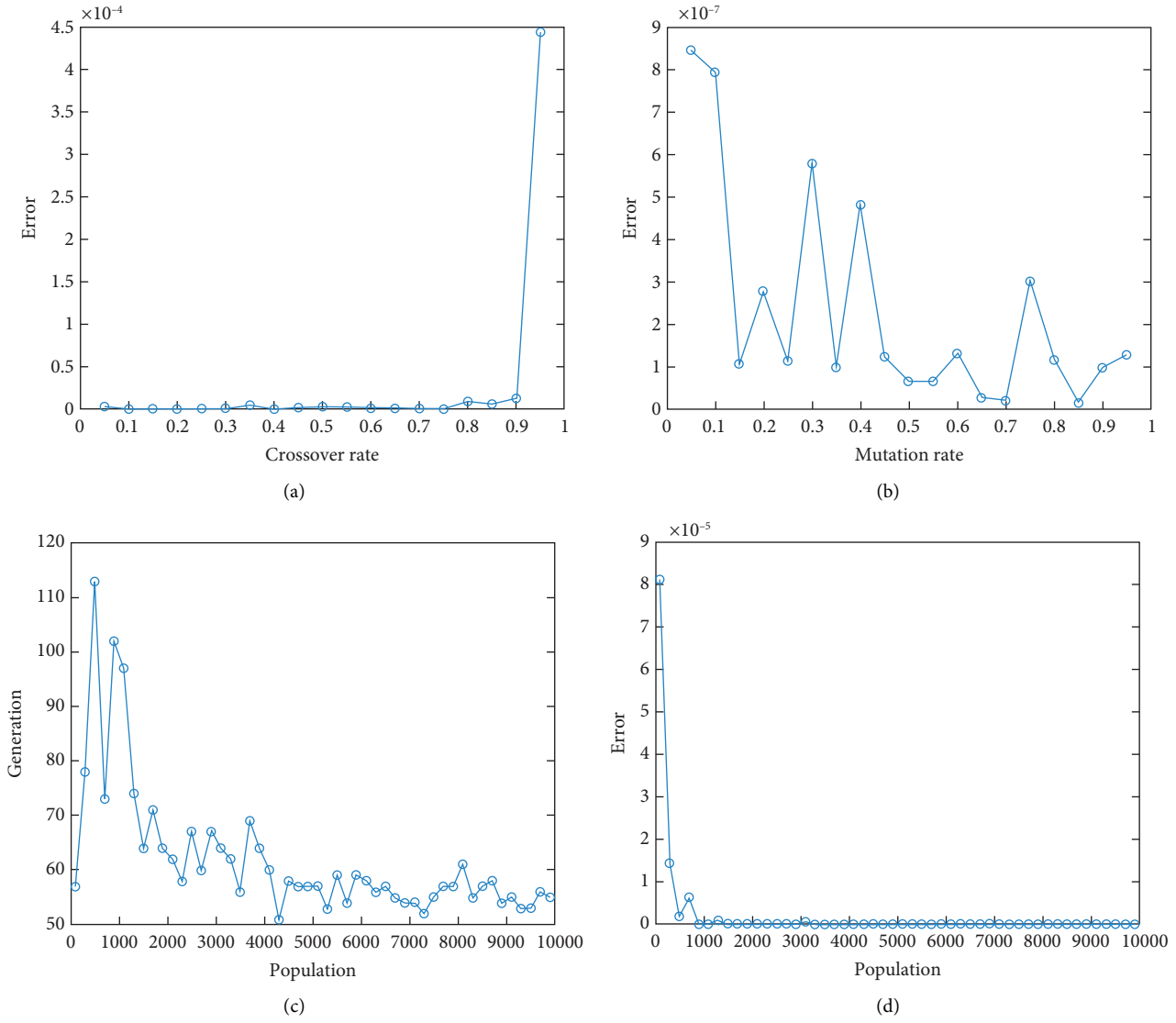

Figure 10: Genetic algorithm results for holder table function: (a) error value based on increasing on crossover rate, (b) error value based on the mutation rate, (c) number of generation vs. population, (d) error value with increase in population.

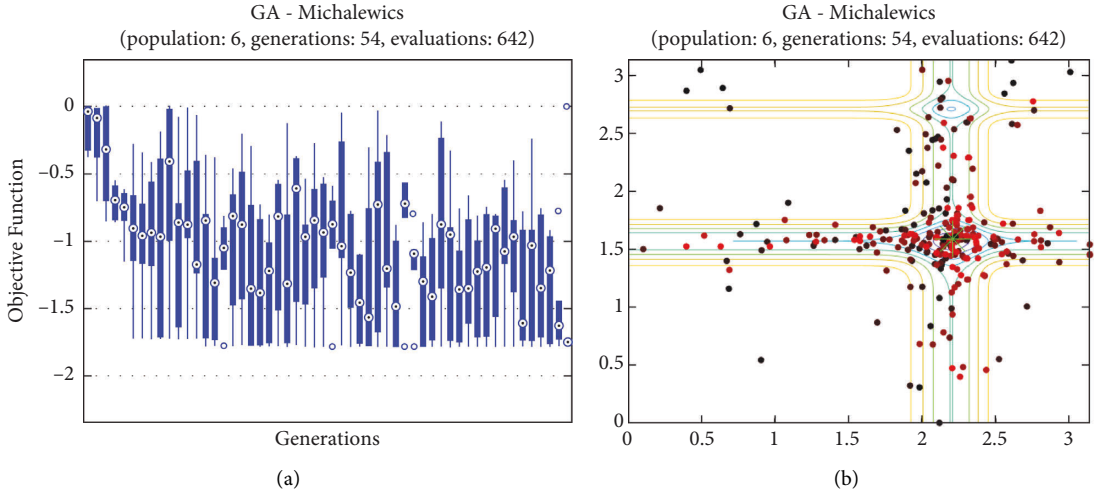

Figure 11: Genetic algorithm results for Michalewicz function: (a) objective function in each generation, (b) plot of population accumulation.

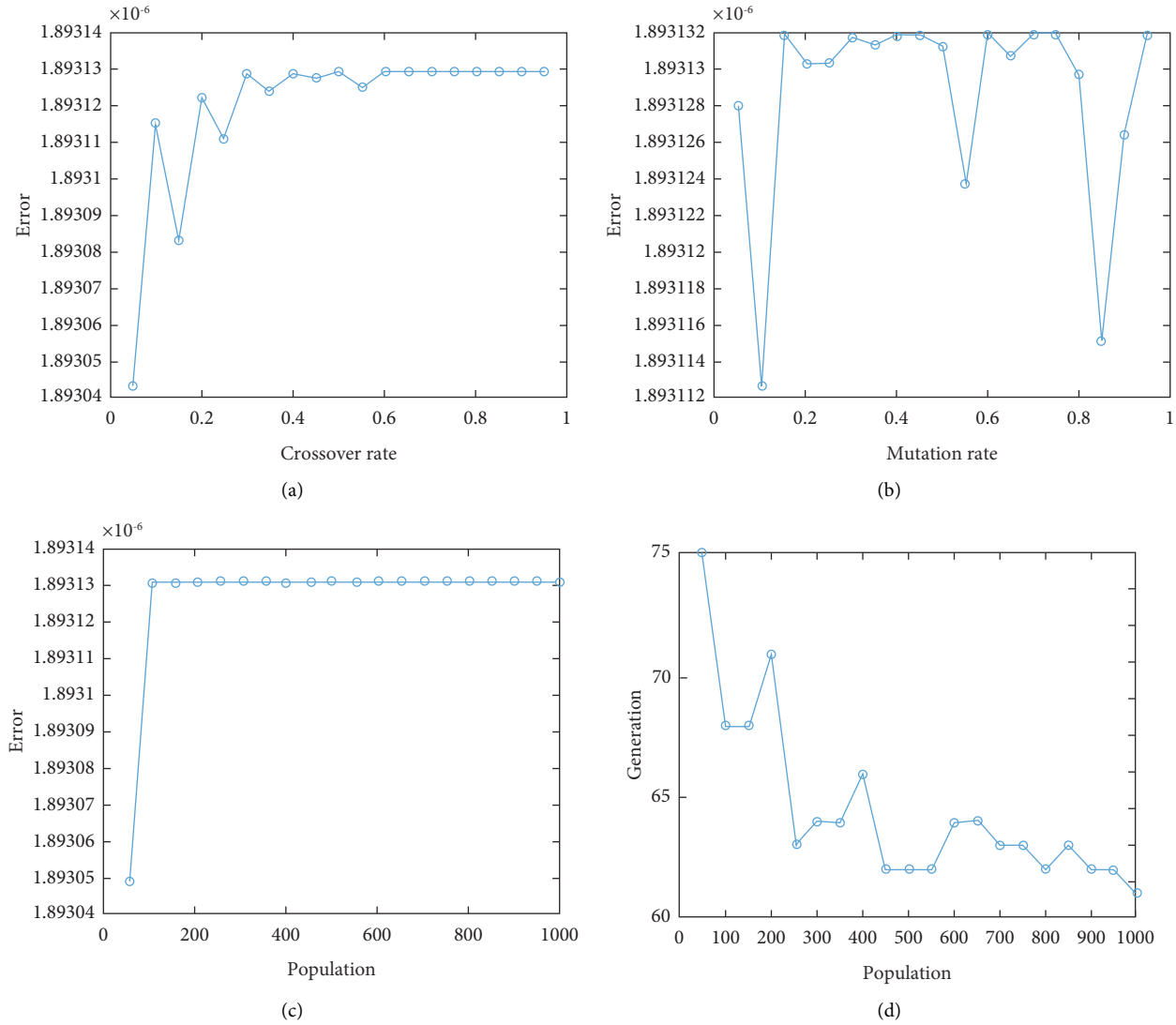

Figure 12: Genetic algorithm results for Michalewicz function: (a) error value based on increasing on crossover rate, (b) error value based on the mutation rate, (c) number of generation vs. population, (d) error value with increase in population.

Table 3: The expressions of optimization problems.

Function	Equations
Ackley	$f(x,y) = -20e^{-0.02\sqrt{x^2+y^2/2}} - e^{\cos(2\pi x)+\cos(2\pi y)/2} + e + 20$
Easom	$f(x,y) = -\cos x.\cos y.\exp(-(x-\pi)^2 - (y-\pi)^2)$
Holder table	$f(x,y) = -\lvert\sin x.\cos y.\exp(\lvert 1 - \sqrt{x^2} + y^2/\pi\rvert)\rvert$
Michalewicz	$f(x,y) = -\sin x.\sin^{20}(x^2/\pi) - \sin y.\sin^{20}(2y^2/\pi)$
Rastrigin	$f(x,y) = 20 + x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi x)$
Rosen	$f(x,y) = 0.25x^4 - 3x^3 + 11x^2 - 13x + 0.25y^4 - 3y^3 + 11y^2 - 13y$
Rosenbrock	$f(x,y) = (1-x)^2 + 100(y-x^2)^2$
Shubert	$f(x,y) = (\sum_{i=1}^{5}(i\cos(i+1)x+i))(\sum_{i=1}^{5}(i\cos(i+1)x+i))$
Sphere	$f(x,y) = x^2 + y^2$
Schaffer	$f(x,y) = 1/2 + \sin^2(x^2+y^2) + 0.5/(1+0.001(x^2+y^2))^2$
Himmelblau	$f(x,y) = f = (x^2+y-11)^2 + (x+y^2-7)^2$
Spring force Vanderplaats	$f(x,y) = 4(\sqrt{x^2+(10-y)^2} - 10)^2 + 1/2(\sqrt{x^2+(10-y)^2} - 10)^2 - 5x - 5y$

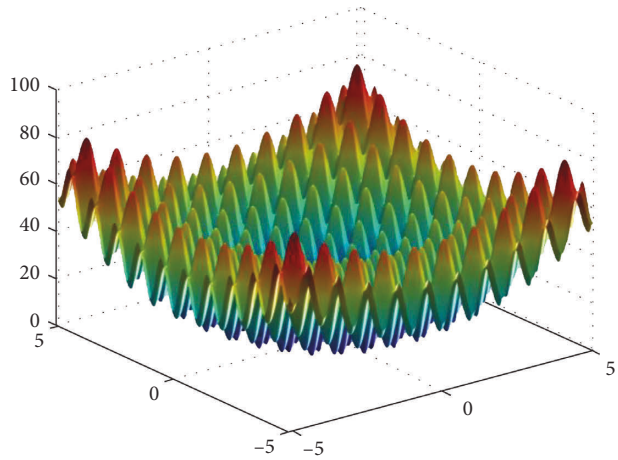

Figure 13: Rastrigin function.

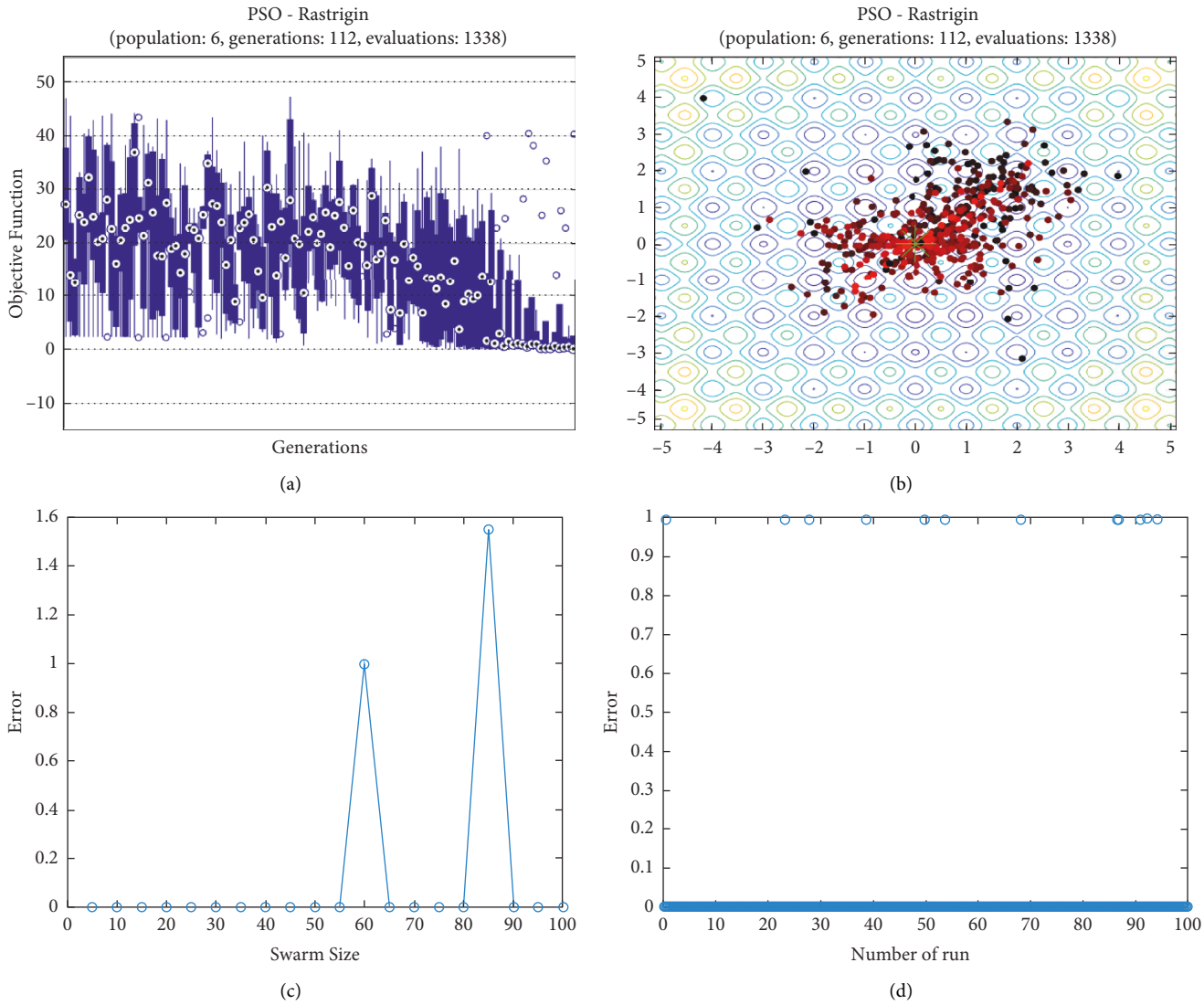

Figure 14: PSO results for Rastrigin function: (a) objective function in each generation, (b) plot of populations accumulation, (c) error value with variation f swarm size, (d) error value with repetition of a single run.

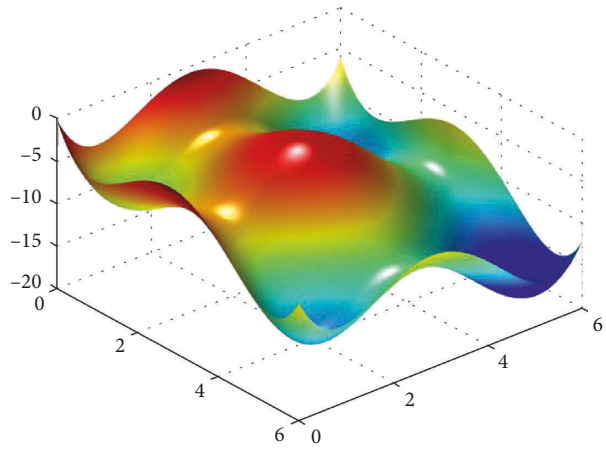

Figure 15: Rosen function.

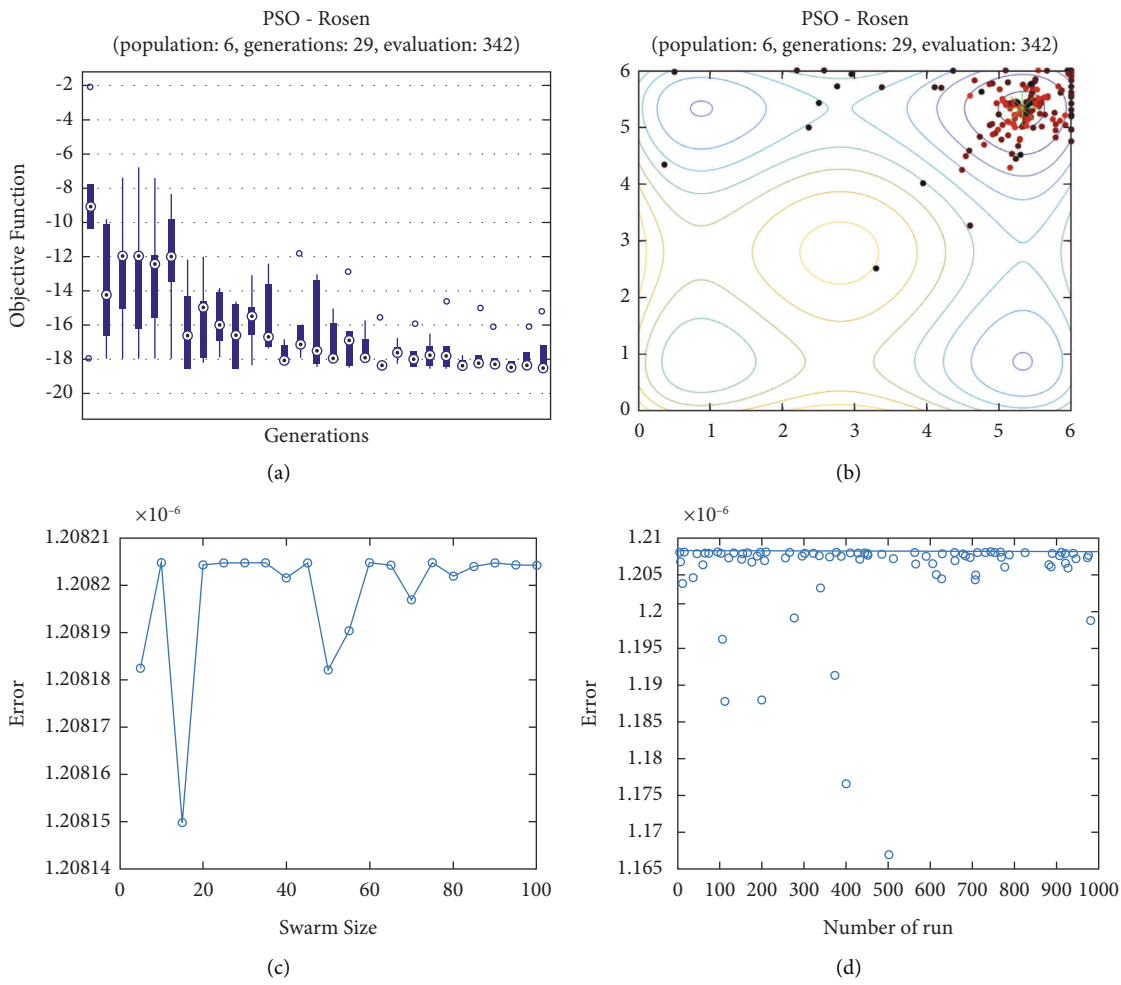

Figure 16: PSO results for Rosen function: (a) objective function in each generation, (b) plot of populations accumulation, (c) error value with variation f swarm size, (d) error value with repetition of a single run.

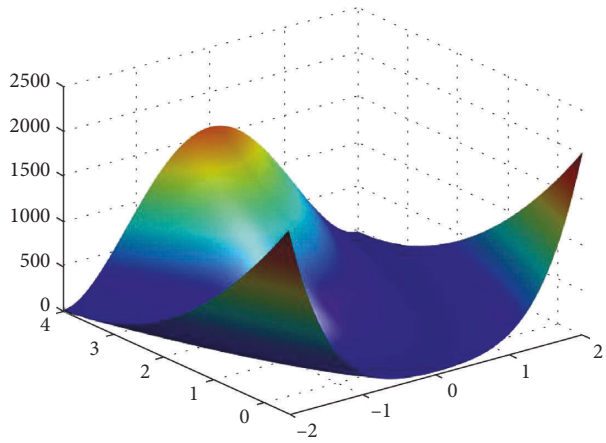

Figure 17: Rosenbrock function.

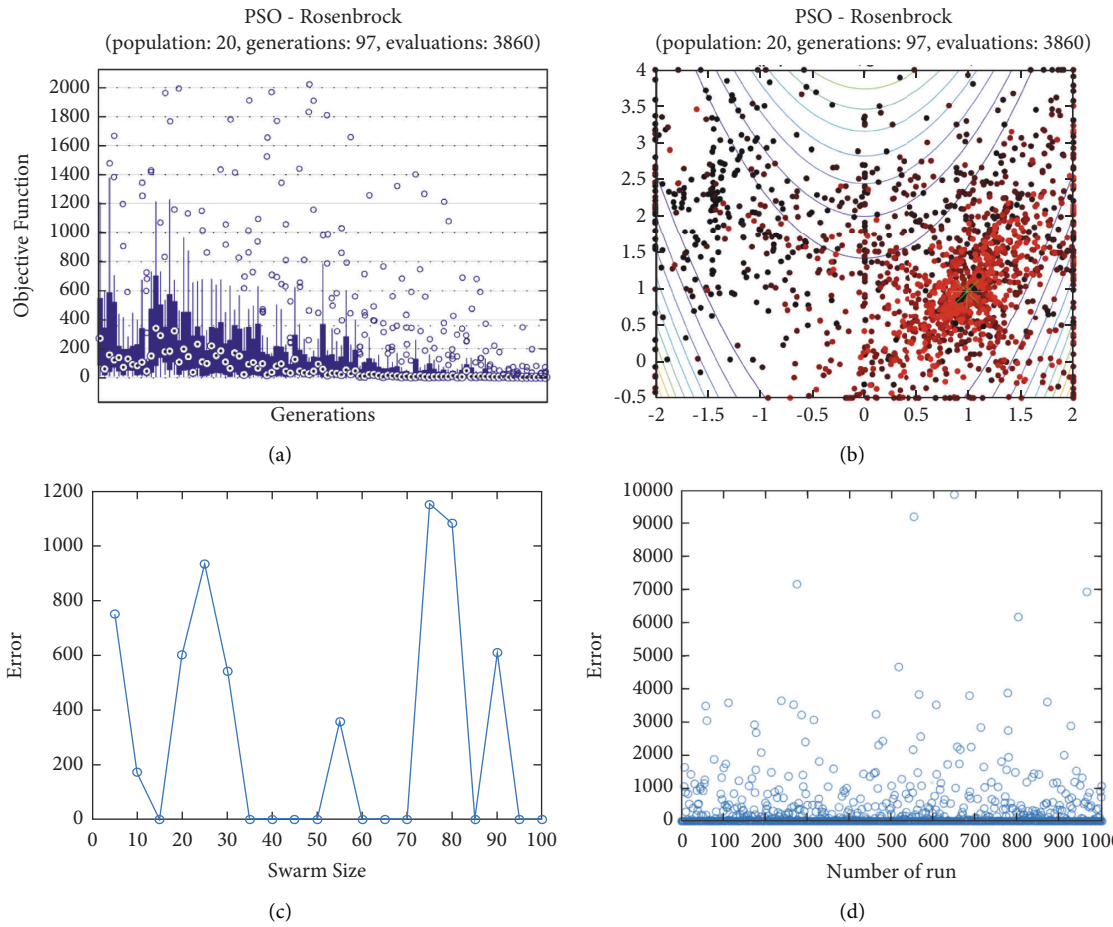

Figure 18: PSO results for Rosenbrock function: (a) objective function in each generation, (b) plot of populations accumulation, (c) error value with variation f swarm size, (d) error value with repetition of a single run.

Furthermore, Figures 6, 8, 10, and 12 show the genetic algorithm error value based on increasing on crossover rate mutation rate. In the number of generations versus population, error values increase with increase in population. Based on the analysis results, the best value of crossover rate for optimization of Ackley function is 0.4–0.5, and mutation rate is 0.6–0.7 (Table 3).

Moreover, based on Figure 6(c), it can be estimated that with the increase of the population to 10,000, there is no significant increase/decrease in the number of generations in

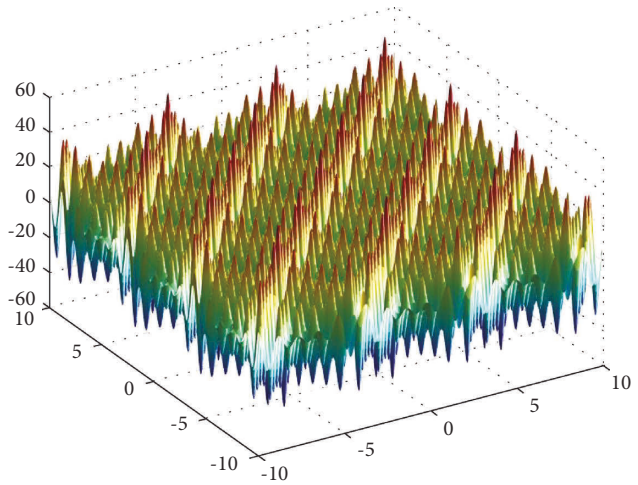

Figure 19: Shubert function.

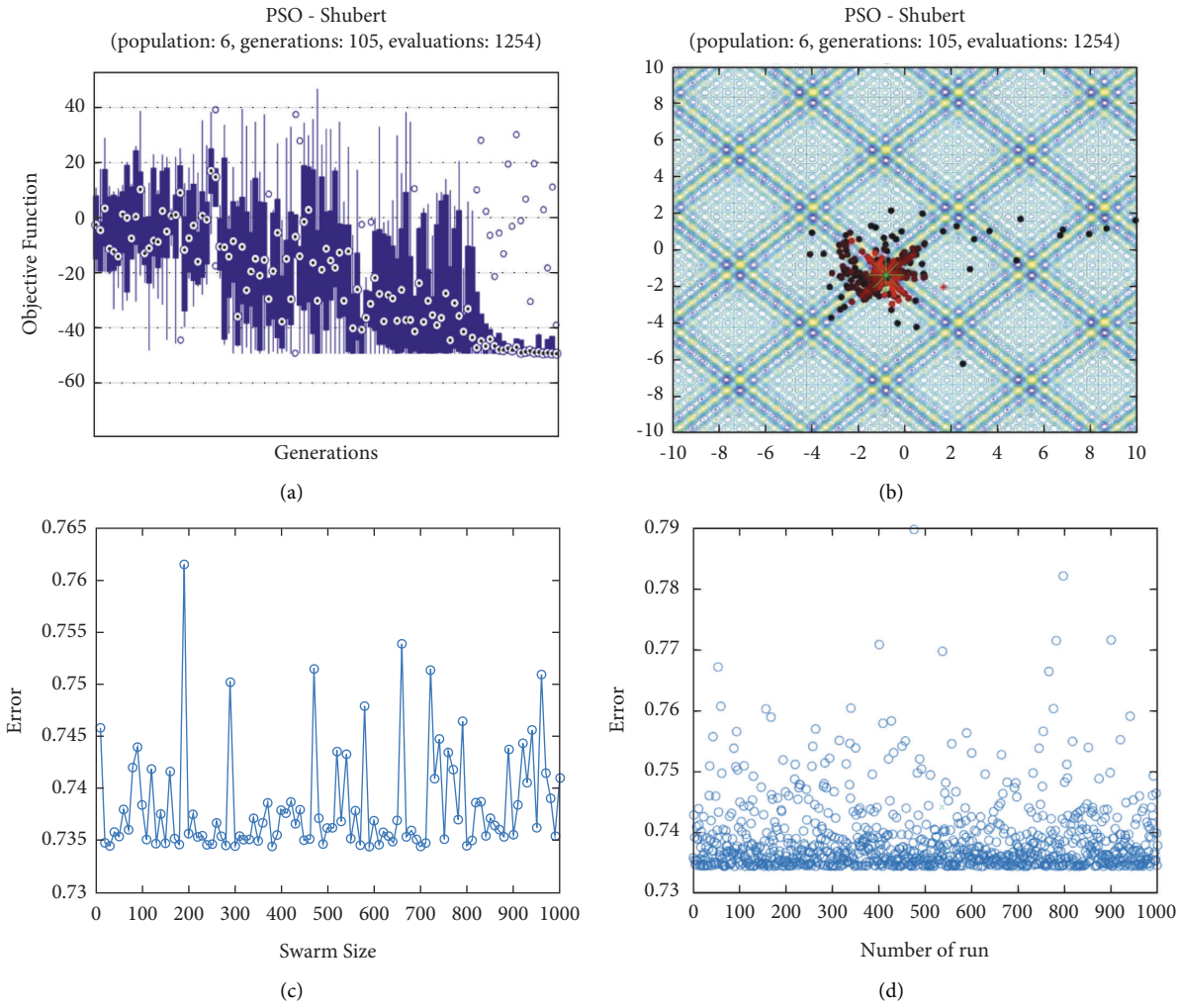

Figure 20: PSO results for Shubert function: (a) objective function in each generation, (b) plot of populations accumulation, (c) error value with variation *f* swarm size, (d) error value with repetition of a single run.

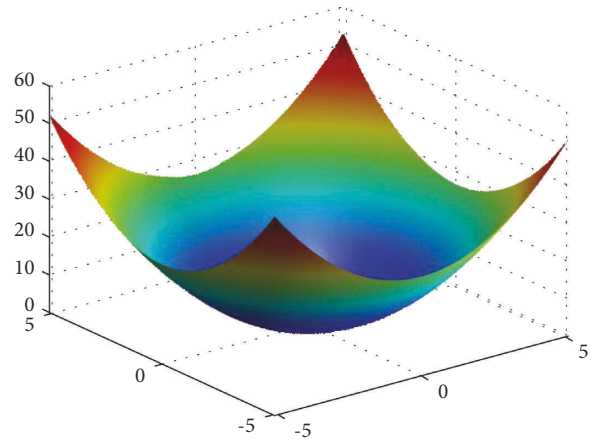

Figure 21: Sphere function.

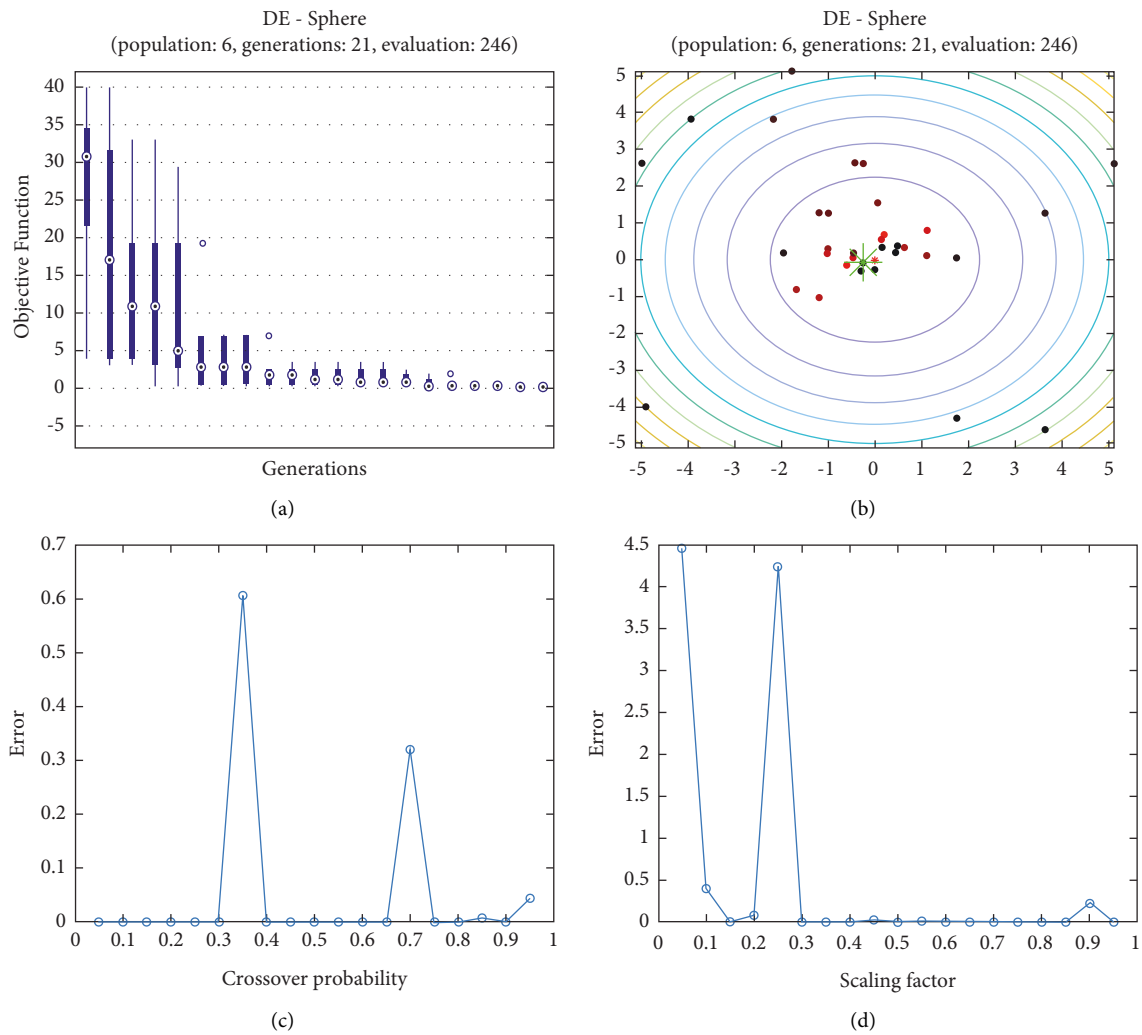

Figure 22: DE results for sphere function: (a) objective function in each generation, (b) plot of population accumulation, (c) error value with the variation of crossover probability, (d) error value changing of scaling function.

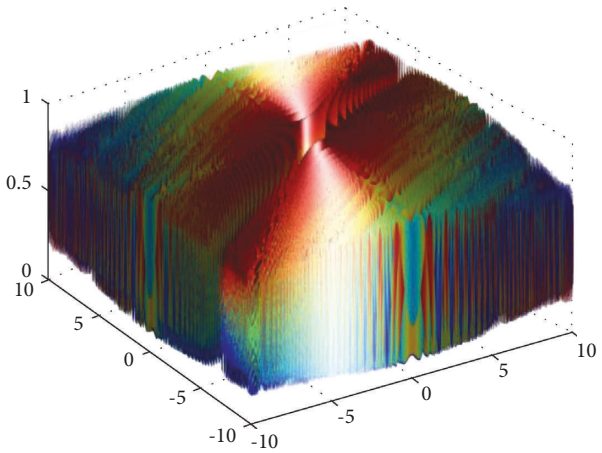

Figure 23: Schaffer function.

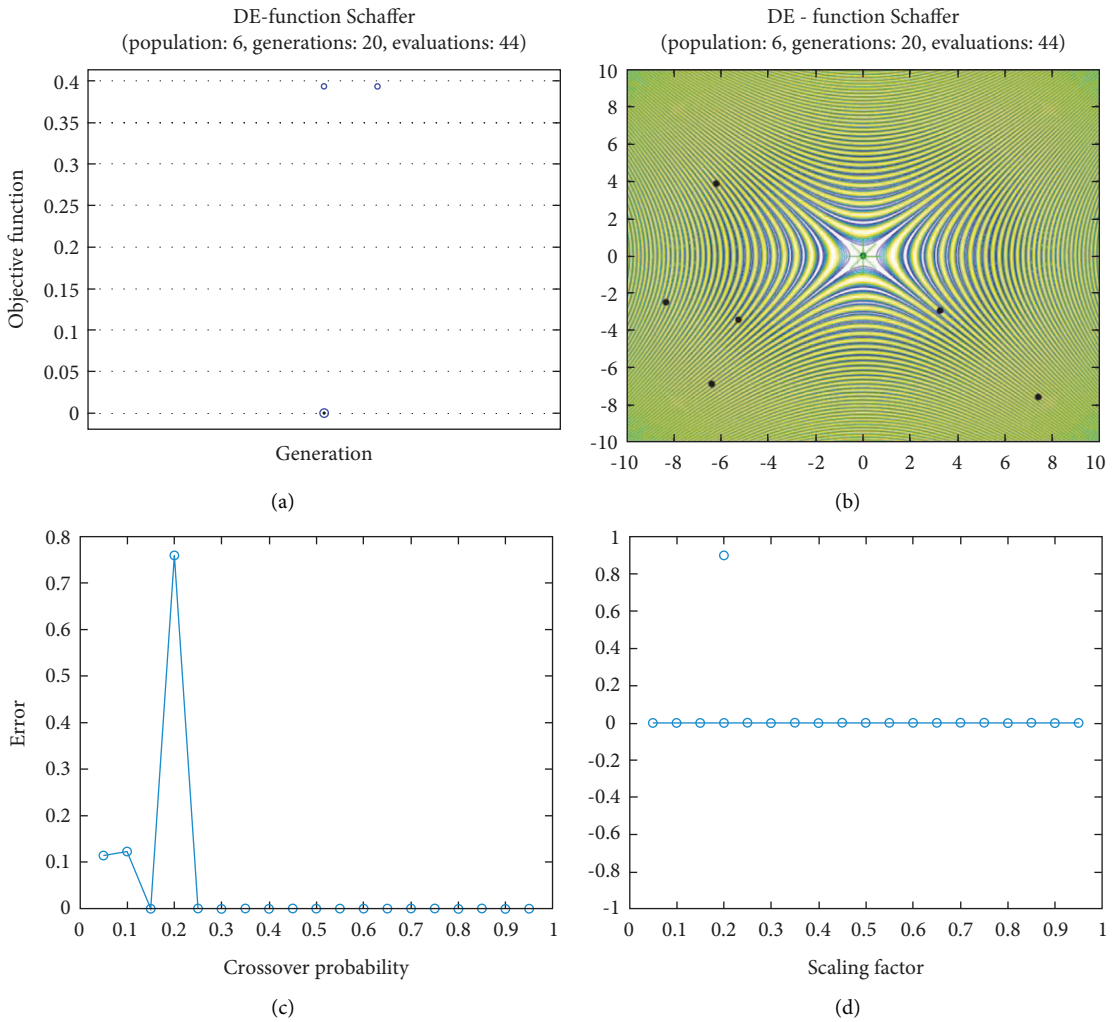

Figure 24: DE results for Schaffer function: (a) objective function in each generation, (b) plot of population accumulation, (c) error value with the variation of crossover probability, (d) error value changing of scaling function.

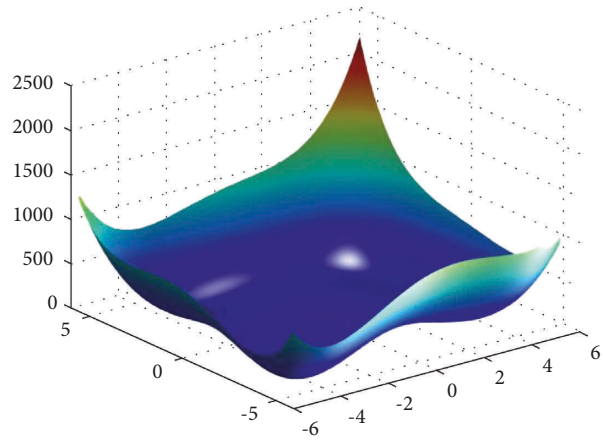

Figure 25: Himmelblau function.

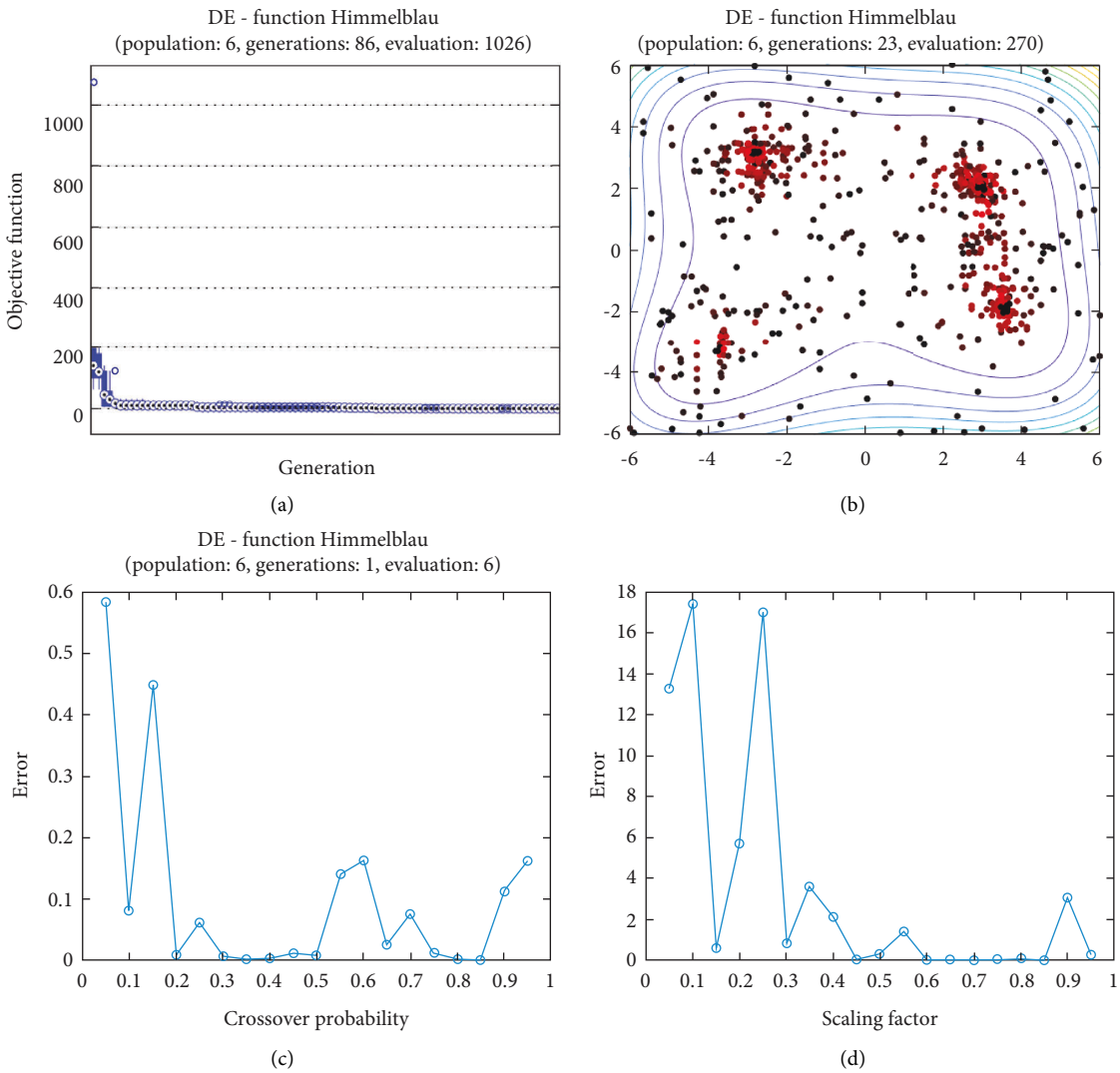

Figure 26: DE results for Himmelblau function: (a) objective function in each generation, (b) plot of population accumulation, (c) error value with the variation of crossover probability, (d) error value changing of scaling function.

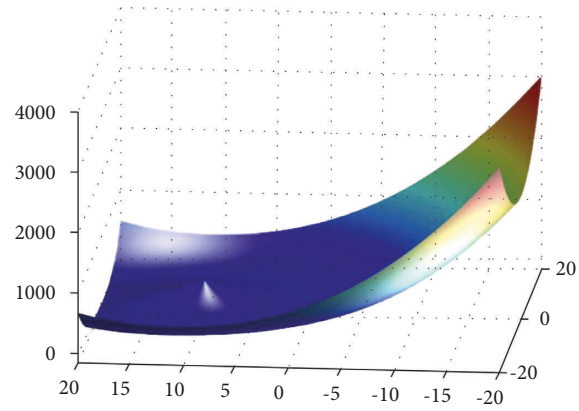

Figure 27: Spring force Vanderplaats function.

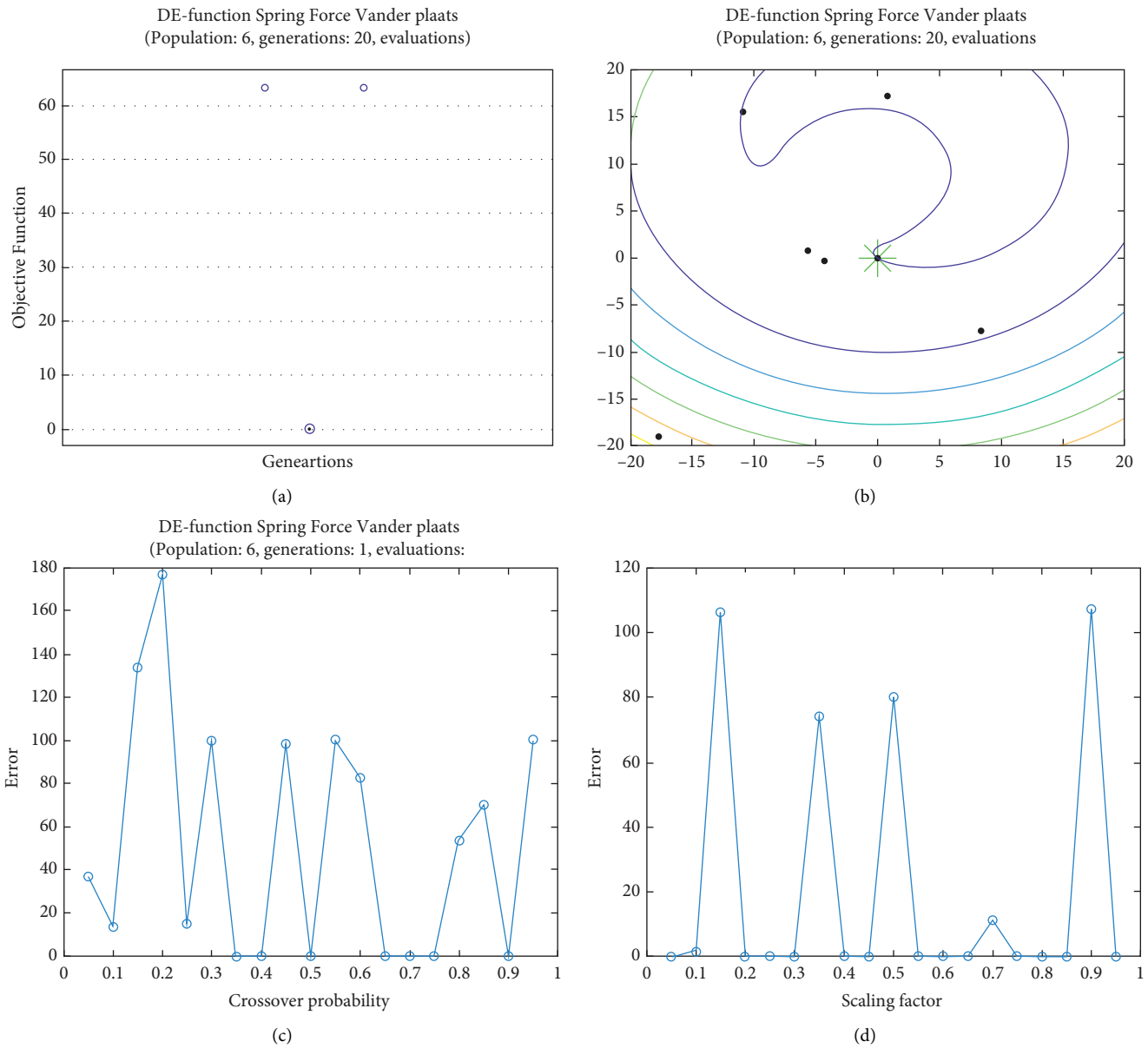

Figure 28: DE results for spring force Vanderplaats function: (a) objective function in each generation, (b) plot of populations accumulation, (c) error value with the variation of crossover probability, (d) error value changing of the scaling function.

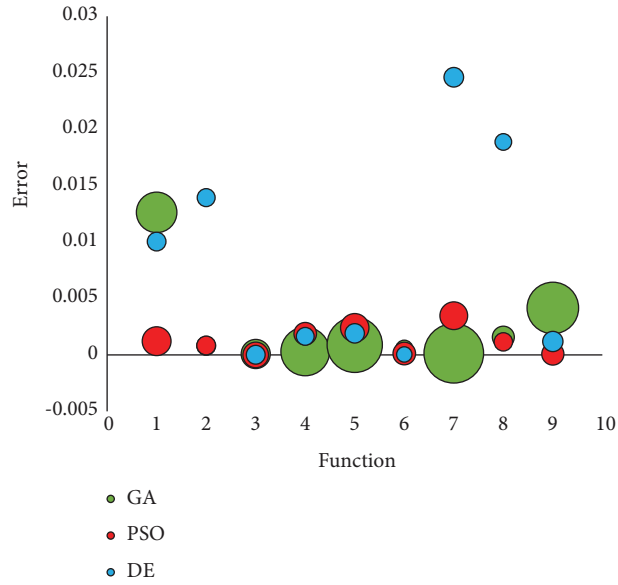

Figure 29: The comparison of GA, PSO, and DE.

Table 4: The comparison of GA, PSO, and DE using error and runtime metrics.

No	Function	GA		PSO		DE	
		Error	Runtime	Error	Runtime	Error	Runtime
1	Ackley	0.012625	21.00289	0.001218	10.63231	0.010042	4.467835
2	Easom	0.000807	4.485303	0.000836	4.488985	0.013936	4.121538
3	Holder table	0.000082	10.84023	0	8.384456	0.000012	4.529235
4	Michalewicz	0.000303	30.27151	0.00188	6.489204	0.001643	4.065657
5	Rastrigin	0.000894	39.14045	0.002408	10.31332	0.001909	4.802492
6	Rosen	0.000541	3.833337	0.000124	6.523122	0.000042	3.013651
7	Rosenbrock	0.00014	45.31977	0.003471	9.796138	0.024601	4.906589
8	Sphere	0.001576	6.141895	0.001154	4.310654	0.018867	3.522635
9	Himmelblau's	0.004158	33.9344	0.000077	6.249301	0.00119	5.255951

Ackley function. Therefore, GA can find the optimum value with the minimum population value. The GA method is low complexity in finding the global minimum of the Ackley function. Furthermore, based on Figure 6(d), the minimum population value for reaching the best complexity is 1000. Easom and Holder table functions results are shown in Figures 7–10. Based on the results, there are no significant effects between changing crossover, mutation rate, and error value because with the small population and 100 generations, GA can find the minimum value of the function. Regarding the results of Michalewicz function with the increase of the number of populations, generation is decremented. However, there is no optimum value of crossover mutation rate for this function because of less complexity of GA for optimization of these functions.

For testing the PSO, the effects of swarm size are compared for each of Rastrigin, Rosen, Rosen Brock, Shubert functions (Figures 13–17. Based on the results, two of 60 and 85 swarms have not accurate results. Therefore, we repeat the optimization 1000 times with a specific swarm size. It can be seen that 1% of evaluations cannot find the optimum value of Rastrigin function (seen Figure 18(d)). However, for the Rosen function, 100% of runs are accurate. One of the complicated formulas in optimization is the Rosenbrock function, based on the results, many runs are not accurate results regarding Figure 19(d). Moreover, there is no relationship between swarm size and optimization accuracy, because sometimes PSO cannot find the optimum value. These results are also repeated in the Shubert function in Figure 20 based on the results, PSO does not have higher robustness for finding the optimum value of these function types because it can no longer be reliable results at least these equations.

For analysis of DE algorithms, four Sphere, Schaffer, Himmelblau's, and Spring Force Vanderplaats are used. Figures 21–24 depict the 3D surface of the following equations, and Figures 25–28 illustrate the DE evaluation results. We tested the crossover rate and scaling factor in the accuracy of the DE method. Based on the results for optimization Sphere, the best scaling factor is 0.3. There is no relationship between error and crossover rate for crossover rate. Overly, one of the properties of DE is using a lower number of initial populations with lower time complexity to find the optimum value of the functions. However, it is sensitive in choosing the crossover rate. Based on

Figure 27, the optimum crossover value is 0.3, and the scaling factor is 0.45. Moreover, in DE, there is no relationship between the crossover and scaling factor rate on error for the spring force Vanderplaats function (see Figure 29).

In the next step all the nine (1) Ackley, (2) Easom, (3) Holder table, (4) Michalewicz, (5) Rastrigin, (6) Rosen, (7) Rosenbrock, (8) Sphere, and (9) Himmelblau's are tested using GA, PSO, and DE algorithm. For all the functions, number of the population is identical and 20 (see Table 4).

Based on the comparison results between the optimization methods, the DE algorithm has the lowest time complexity among other methods. Moreover, GA illustrated the highest time complexity. However, the PSO algorithm has lower reliability to find the optimum point.

4. Conclusion and Future Works

The objective of this report is to evaluate nongradient-based methods for optimizing some mathematical surfaces by applying three meta-heuristic algorithms, including genetic algorithms, particle swarm algorithms, and differential evaluation algorithms. In this report, 12 functions of Easom, Holder table, Michalewicz, Ackley, Rastrigin, Rosen, Rosen Brock, Shubert, Sphere, Schaffer, Himmelblau's, and Spring Force Vanderplaats are used for optimization. We utilized GA to optimize Easom, Holder tables, Michalewicz, and Ackley functions in this report. The number of generations versus the population, error value as the population increases. According to the results of the analysis, the best crossover rate for optimization of the Ackley function is 0.4–0.5, and the best mutation rate is 0.6–0.7. For GA, it is estimated that with the increase in population to 10,000, there is no significant increase or decrease in the number of generations in Ackley function. Consequently, GA is able to find the optimal value with a minimum population value. Using the GA method, the global minimum of the Ackley function can be determined with a low degree of complexity. Additionally, the minimum population value for the best degree of complexity is 1000. There are no significant effects of changing crossover, mutation rate, and error value for Easom and Holder table functions. Michaelewicz function shows that generation decreases with an increase in the number of populations. Due to the simplicity of GA in optimizing these functions, there is no optimal crossover mutation rate for this function.

In order to test the PSO, the effects of swarm size are compared for Rastrigin, Rosen, Rosenbrock, and Shubert functions. The optimization is repeated 1000 times with the same swarm sizes. It can be seen that 1% of evaluations are not able to determine the optimum value for the Rastrigin function. In contrast, 100% of evaluations are able to determine the Rosenbrock function. The Rosenbrock function is one of the most complex formulas in optimization. According to the results, there is no relationship between swarm size and optimization accuracy. These results indicate that PSO does not have higher robustness for finding optimum values of these function types since it is no longer able to produce reliable results, at least for these equations. An analysis of DE algorithms uses four Spheres, Schaffers, Himmelblaus, and Spring Force Vanderplaats. To test the accuracy of the DE method, we tested the crossover rate and scaling factor. According to the results for optimization Sphere, the best scaling factor is 0.30. In terms of the crossover rate, there is no relationship between error and crossover rate. In general, one of the characteristics of DE is that it uses fewer initial populations with a shorter time complexity to find the optimal values. It is sensitive to the crossover rate, however. Furthermore, there is no relationship between the crossover and the scaling factor rate on error for the spring force Vanderplaats function in DE. Comparing the results of the optimization methods, it appears that the DE algorithm has the lowest time complexity. The GA algorithm has the highest time complexity. In contrast, the PSO algorithm is less reliable for finding the optimum point.

The use of meta-heuristics has enabled engineers to solve several engineering problems that could not be solved with standard optimization approaches. Examples include the simplicity with which they can be combined in finite element software in any domain, where the combination/permutation of solutions available to each method enables the discovery of optimum projects without the need of explicit functions. Literature contains numerous examples of this phenomenon. Developing a meta-heuristic that can accomplish this with fewer populations and iterations (lower processing costs) and more accuracy is the point of contention in the literature between new algorithms attempting this goal. If the algorithm is evolutionary in nature, swarms, behaviors, and physical occurrences are all features that contribute to the primary purpose outlined above. I believe that the universal law of time will reveal those algorithms that are truly superior and distinguishable from the others. Additionally, as a reviewer, you may request tests such as Wilcoxon to determine whether the way each meta-heuristic operates has changed.

Nomenclature

p_i: Probability of selection
f: Objective function
N: Number of populations
α: Crossover blending factor
r_k: Random number
t: Generation
β_1: Individuality factor
β_2: Sociability factor
α: PSO inertia factor
$\delta 1$ and $\delta 2$: Random parent features.

Data Availability

Data are available and can be provided over the e-mails querying directly to the author at the corresponding author (amin.valizadeh@mail.um.ac.ir).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. Ogaltsov, A. Gasnikov, V. Spokoiny, D. Dvinskikh, and P. Dvurechensky, "On the line-search gradient methods for stochastic optimization," *IFAC-PapersOnLine*, vol. 53, pp. 1715–1720, 2020.

[2] X. Zhang, J. Xing, P. Liu, Y. Luo, and Z. Kang, "Realization of full and directional band gap design by non-gradient topology optimization in acoustic metamaterials," *Extreme Mechanics Letters*, vol. 42, Article ID 101126, 2021.

[3] A. G. Carlon, B. M. Dia, L. Espath, R. H. Lopez, and R. Tempone, "Nesterov-aided stochastic gradient methods using Laplace approximation for Bayesian design optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 363, Article ID 112909, 2020.

[4] I. M. Bomze, M. Gabl, F. Maggioni, and G. C. Pflug, "Two-stage stochastic standard quadratic optimization," *European Journal of Operational Research*, vol. 299, no. 1, pp. 21–34, 2022.

[5] H. R. Najafabadi, T. Goto, M. Falheiro, T. C. Martins, A. Barari, and M. S. G. Tsuzuki, "Post-processing of non gradient-based topology optimization with simulated annealing," *IFAC-PapersOnLine*, vol. 54, no. 1, pp. 755–760, 2021.

[6] M. T. Jensen, "Guiding single-objective optimization using multi-objective methods," in *Proceedings of the Workshops on Applications of Evolutionary Computation*, pp. 268–279, Springer, Berlin, Heidelberg, April 2003.

[7] T. T. Nguyen, "A high performance social spider optimization algorithm for optimal power flow solution with single objective optimization," *Energy*, vol. 171, pp. 218–240, 2019.

[8] S. Zapotecas Martínez, A. Carlos, and C. Coello, "A proposal to hybridize multi-objective evolutionary algorithms with non-gradient mathematical programming techniques," in *Proceedings of the International Conference on Parallel Problem Solving from Nature*, pp. 837–846, Springer, Berlin, Heidelberg, May 2008.

[9] K.-H. Liang, X. Yao, and C. Newton, "Evolutionary search of approximated n-dimensional landscapes," *International Journal of Knowledge-Based and Intelligent Engineering Systems*, vol. 4, no. 3, pp. 172–183, 2000.

[10] O. Sigmund, "On the usefulness of non-gradient approaches in topology optimization," *Structural and Multidisciplinary Optimization*, vol. 43, no. 5, pp. 589–596, 2011.

[11] Y. Luo, J. Xing, and Z. Kang, "Topology optimization using material-field series expansion and Kriging-based algorithm: an effective non-gradient method," *Computer Methods in Applied Mechanics and Engineering*, vol. 364, Article ID 112966, 2020.

[12] S. Z. Martínez and C. A. Coello Coello, "A memetic algorithm with non gradient-based local search assisted by a meta-model," in *Proceedings of the International Conference on Parallel Problem Solving from Nature*, pp. 576–585, Springer, Berlin, Heidelberg, August 2010.

[13] W. Hare, J. Nutini, and S. Tesfamariam, "A survey of non-gradient optimization methods in structural engineering," *Advances in Engineering Software*, vol. 59, pp. 19–28, 2013.

[14] I. Jha, K. K. Pathak, M. Jha, and A. Ranjan, "A comparative study of gradient descent method and a novel non-gradient method for structural shape optimization," *International Journal of Mathematical, Engineering and Management Sciences*, vol. 7, no. 2, pp. 258–271, 2022.

[15] K. Nakamura, S. Soatto, and B. W. Hong, "Stochastic batch size for adaptive regularization in deep network optimization," *Pattern Recognition*, vol. 129, Article ID 108776, 2022.

[16] P. Dvurechensky, E. Gorbunov, and A. Gasnikov, "An accelerated directional derivative method for smooth stochastic convex optimization," *European Journal of Operational Research*, vol. 290, no. 2, pp. 601–621, 2021.

[17] Z. Yang, "Fast automatic step size selection for zeroth-order nonconvex stochastic optimization," *Expert Systems with Applications*, vol. 174, Article ID 114749, 2021.

[18] K. Gao, D. M. Doc, S. Chu, G. Wu, H. Alicia Kim, and C. A. Featherston, "Robust topology optimization of structures under uncertain propagation of imprecise stochastic-based uncertain field," 2022, https://arxiv.org/abs/2201.11513.

[19] D. Guirguis, W. W. Melek, and M. F. Aly, "High-resolution non-gradient topology optimization," *Journal of Computational Physics*, vol. 372, pp. 107–125, 2018.

[20] S. Afandizadeh, M. Ameri, and M. H. M. Moghaddam, "Introducing a modified gradient vector method for optimization of accident prediction non-linear functions," *Applied Mathematical Modelling*, vol. 35, no. 12, pp. 5500–5506, 2011.

[21] S. Y.-C. Chen, C.-M. Huang, C.-W. Hsing, H.-S. Goan, and Y.-J. Kao, "Variational quantum reinforcement learning via evolutionary optimization," *Mach. Learn. Sci. Technol.*, vol. 3, no. 1, Article ID 015025, 2022.

[22] E. Pazouki, "A practical surface irrigation system design based on volume balance model and multi-objective evolutionary optimization algorithms," *Agricultural Water Management*, vol. 248, Article ID 106755, 2021.

[23] G. Dhiman, K. K. Singh, A. Slowik et al., "EMoSOA: a new evolutionary multi-objective seagull optimization algorithm for global optimization," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 2, pp. 571–596, 2021.

[24] J. S. Pan, N. Liu, S. C. Chu, and T. Lai, "An efficient surrogate-assisted hybrid optimization algorithm for expensive optimization problems," *Information Sciences*, vol. 561, pp. 304–325, 2021.

[25] L. Abualigah, A. Diabat, P. Sumari, and A. H. Gandomi, "A novel evolutionary arithmetic optimization algorithm for multilevel thresholding segmentation of covid-19 ct images," *Processes*, vol. 9, no. 7, p. 1155, 2021.

[26] F. MiarNaeimi, G. Azizyan, and M. Rashki, "Horse herd optimization algorithm: a nature-inspired algorithm for high-dimensional optimization problems," *Knowledge-Based Systems*, vol. 213, Article ID 106711, 2021.

[27] Y. Meraihi, A. B. Gabis, S. Mirjalili, and A. Ramdane-Cherif, "Grasshopper optimization algorithm: theory, variants, and applications," *IEEE Access*, vol. 9, pp. 50001–50024, 2021.

[28] B. Cao, S. Fan, J. Zhao et al., "Large-scale many-objective deployment optimization of edge servers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3841–3849, 2021.

[29] B. Cao, M. Li, X. Liu, J. Zhao, W. Cao, and Z. Lv, "Many-objective deployment optimization for a drone-assisted camera network," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 4, pp. 2756–2764, 2021.

[30] X. Xu, D. Niu, L. Peng, S. Zheng, and J. Qiu, "Hierarchical multi-objective optimal planning model of active distribution network considering distributed generation and demand-side response," *Sustainable Energy Technologies and Assessments*, vol. 53, Article ID 102438, 2022.

[31] G. Zhou, X. Bao, S. Ye, H. Wang, and H. Yan, "Selection of optimal building facade texture images from UAV-based

multiple oblique image flows," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 2, pp. 1534–1552, 2021.

[32] M. Zhang, Y. Chen, and J. Lin, "A privacy-preserving optimization of neighborhood-based recommendation for medical-aided diagnosis and treatment," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10830–10842, 2021.

[33] P. Kumar, X. Changqing, A. Mehbodniya et al., "Hybrid optimization approach for energy control in electric vehicle controller for regulation of three-phase induction motors," *Hindawi Mathematical Problems in Engineering*, 2022.

[34] L. Yan, J. Webber, A. Mehbodniya, B. Moorthy, S. Sivamani, and M. Shabaz, "Distributed Optimization of Heterogeneous UAV cluster PID Controller based on machine learning," *Elsevier Computers & Electrical Engineering*, 2022.

[35] A. M. Kumar, J. Webber, M. Haq, K. Gola, P. Singh, and S. Karupusamy, "Optimal cluster head selection for energy efficient wireless sensor network using hybrid competitive swarm optimization and harmony search algorithm," *Elsevier Sustainable Energy Technologies and Assessments*, 2022.

[36] J. Webber, A. Mehbodniya, K. Yano, and Y. Suzuki, "Optimized WLAN Channel Allocation based on Gibbs Sampling with Busy Prediction using a Probabilistic Neural Network," in *Proceedings of the IEEE International Conference on Communications, Signal Processing, and their Applications (ICCSPA'19)*, Sharjah, UAE, March 2019.

[37] G. Javidannia, M. Bemanian, M. Mahdavinejad, S. Nejat, and L. Javidannia, "Generative design workflow for seismic-efficient architectural design of tall buildings; a multi-object optimization approach," *ACM Digital Library*, 2021.

[38] Y. Xi, W. Jiang, K. Wei, T. Hong, T. Cheng, and S. Gong, "Wideband RCS reduction of microstrip antenna array using coding metasurface with low q resonators and fast optimization method," *IEEE Antennas and Wireless Propagation Letters*, vol. 21, no. 4, pp. 656–660, 2022.

[39] B. Cao, J. Zhao, X. Liu et al., "Multiobjective evolution of the explainable fuzzy rough neural network with gene expression programming," *IEEE Transactions on Fuzzy Systems*, vol. 1, 2022.

[40] L. Zhang, T. Gao, G. Cai, and K. L. Hai, "Research on electric vehicle charging safety warning model based on back propagation neural network optimized by improved gray wolf algorithm," *Journal of Energy Storage*, vol. 49, 2022.

[41] X. Liu, G. Zhang, J. Li et al., "Deep learning for feynman's path integral in strong-field time-dependent dynamics," *Physical Review Letters*, vol. 124, no. 11, Article ID 113202, 2020.

[42] H. Sheng, R. Cong, D. Yang, R. Chen, S. Wang, and Z. Cui, "UrbanLF: a comprehensive light field dataset for semantic segmentation of urban scenes," *IEEE Transactions on Circuits and Systems for Video Technology*, 2022.

[43] B. Cao, J. Zhang, X. Liu et al., "Edge-Cloud Resource Scheduling in Space-Air-Ground Integrated Networks for Internet of Vehicles," *IEEE Internet of Things Journal*, vol. 1, 2021.

[44] L. Zhao and L. Wang, "A new lightweight network based on MobileNetV3," *KSII Transactions On Internet And Information Systems*, 2022.

[45] L. Zhang, H. Zhang, and G. Cai, "The multi-class fault diagnosis of wind turbine bearing based on multi-source signal fusion and deep learning generative model," *IEEE Transactions on Instrumentation and Measurement*, vol. 1, 2022.

[46] J. Mou, P. Duan, L. Gao, X. Liu, and J. Li, "An effective hybrid collaborative algorithm for energy-efficient distributed permutation flow-shop inverse scheduling," *Future Generation Computer Systems*, vol. 128, pp. 521–537, 2022.

[47] C. Lu, Q. Liu, B. Zhang, and L. Yin, "A Pareto-based hybrid iterated greedy algorithm for energy-efficient scheduling of distributed hybrid flowshop," *Expert Systems with Applications*, vol. 204, 2022.

[48] F. Meng, W. Cheng, and J. Wang, "Semi-supervised software defect prediction model based on tri-training," *KSII Transactions on Internet and Information Systems*, vol. 15, no. 11, pp. 4028–4042, 2021.

[49] G. Sun, C. Li, and L. Deng, "An adaptive regeneration framework based on search space adjustment for differential evolution," *Neural Computing and Applications*, 2021.

[50] T. Cai, D. Yu, H. Liu, and F. Gao, "Computational analysis of variational inequalities using mean extra-gradient approach," *Mathematics*, vol. 10, no. 13, Article ID 2318, 2022.

[51] Y. Wang, H. Wang, B. Zhou, and H. Fu, "Multi-dimensional prediction method based on Bi-LSTMC for ship roll," *Ocean Engineering*, vol. 242, Article ID 110106, 2021.

[52] Y. Xie, Y. Sheng, M. Qiu, and F. Gui, "An adaptive decoding biased random key genetic algorithm for cloud workflow scheduling," *Engineering Applications of Artificial Intelligence*, vol. 112, 2022.

[53] Y. Zhang, F. Liu, Z. Fang, B. Yuan, G. Zhang, and J. Lu, "Learning from a complementary-label source domain: theory and algorithms," *IEEE Transaction on Neural Networks and Learning Systems*, pp. 1–15, 2021.

[54] T. Hong, S. Guo, W. Jiang, and S. Gong, "Highly selective frequency selective surface with ultrawideband rejection," *IEEE Transactions on Antennas and Propagation*, vol. 70, no. 5, pp. 3459–3468, 2022.

[55] K. Xu, X. Weng, J. Li et al., "60-GHz third-order on-chip bandpass filter using GaAs pHEMT technology," *Semiconductor Science and Technology*, vol. 37, no. 5, 2022.

[56] W. Zheng, L. Yin, X. Chen, Z. Ma, S. Liu, and B. Yang, "Knowledge base graph embedding module design for Visual question answering model," *Pattern Recognition*, vol. 120, Article ID 108153, 2021.

[57] H. Zhu, M. Xue, Y. Wang, G. Yuan, and X. Li, "Fast visual tracking with siamese oriented region proposal network," *IEEE Signal Processing Letters*, vol. 29, Article ID 1437, 2022.

[58] W. Zheng, X. Liu, X. Ni, L. Yin, and B. Yang, "Improving visual reasoning through semantic representation," *IEEE access*, vol. 9, pp. 91476–91486, 2021.

[59] W. Zheng, X. Liu, and L. Yin, "Sentence representation method based on multi-layer semantic network," *Applied Sciences*, vol. 11, no. 3, p. 1316, 2021.

[60] J. Yu, L. Lu, Y. Chen, Y. Zhu, and L. Kong, "An indirect eavesdropping attack of keystrokes on touch screen through acoustic sensing," *IEEE Transactions on Mobile Computing*, vol. 20, no. 2, pp. 337–351, 2021.

[61] H. Kong, L. Lu, J. Yu, Y. Chen, and F. Tang, "Continuous authentication through finger gesture interaction for smart homes using WiFi," *IEEE Transactions on Mobile Computing*, vol. 20, no. 11, pp. 3148–3162, 2021.

[62] S. Zhao, F. Li, H. Li et al., "Smart and practical privacy-preserving data aggregation for fog-based smart grids," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 521–536, 2021.

[63] Q. Meng, X. Lai, Z. Yan, C. Su, and M. Wu, "Motion planning and adaptive neural tracking control of an uncertain two-link rigid-flexible manipulator with vibration amplitude constraint," *IEEE Transaction on Neural Networks and Learning Systems*, pp. 1–15, 2021.

[64] G. Zhou, X. Zhou, Y. Song et al., "Design of supercontinuum laser hyperspectral light detection and ranging (LiDAR) (SCLaHS LiDAR)," *International Journal of Remote Sensing*, vol. 42, no. 10, pp. 3731–3755, 2021.

[65] G. Zhou, R. Zhang, and S. Huang, "Generalized buffering algorithm," *IEEE access*, vol. 9, pp. 27140–27157, 2021.

[66] X. Liang, L. Luo, S. Hu, and Y. Li, "Mapping the knowledge frontiers and evolution of decision making based on agent-based modeling," *Knowledge-Based Systems*, vol. 250, 2022.

[67] J. Chen, L. Du, and Y. Guo, "Label constrained convolutional factor analysis for classification with limited training samples," *Information Sciences*, vol. 544, pp. 372–394, 2021.

[68] L. Liao, L. Du, and Y. Guo, "Semi-supervised SAR target detection based on an improved faster R-CNN," *Remote Sensing (Basel, Switzerland)*, vol. 14, no. 1, 143 pages, 2021.

[69] J. Gao, H. Sun, J. Han, Q. Sun, and T. Zhong, "Research on recognition method of electrical components based on FEYOLOv4-tiny," *Journal of Electrical Engineering & Technology*, 2022.

[70] H. Tian, Y. Qin, Z. Niu, L. Wang, and S. Ge, "Summer maize mapping by compositing time series sentinel-1A imagery based on crop growth cycles," *Journal of the Indian Society of Remote Sensing*, vol. 49, no. 11, pp. 2863–2874, 2021.

[71] H. Tian, Y. Wang, T. Chen, L. Zhang, and Y. Qin, "Early-Season Mapping of Winter Crops Using Sentinel-2 Optical Imagery," *Remote Sensing (Basel, Switzerland)*, vol. 13, no. 19, Article ID 3822, 2021.

[72] X. Xu, D. Niu, B. Xiao, X. Guo, L. Zhang, and K. Wang, "Policy analysis for grid parity of wind power generation in China," *Energy Policy*, vol. 138, Article ID 111225, 2020.

[73] Y. Xiao, X. Zuo, J. Huang, A. Konak, and Y. Xu, "The continuous pollution routing problem," *Applied Mathematics and Computation*, vol. 387, Article ID 125072, 2020.

[74] P. Chaurasiya, J. Webber, A. Mehbodniya, M. Haq, A. Kumar, and P. Paliwal, "Multi Agent based approach for generation expansion planning in isolated micro-grid with renewable energy sources and battery storage," *Journal of Supercomputing*, May 2022.

[75] Y. Xiao, Y. Zhang, I. Kaku, R. Kang, and X. Pan, "Electric vehicle routing problem: A systematic review and a new comprehensive model with nonlinear energy recharging and consumption," *Renewable & Sustainable Energy Reviews*, vol. 151, Article ID 111567, 2021.

[76] G. Javidannia, M. Bemanian, and M. Mahdavinejad, "Performance oriented design framework for early tall building form development," *Seismic Architecture View*, 2020.

[77] M. Mahdavinejad, M. Bemanian, and G. Abolvardi, "Analyzing the state of seismic consideration of architectural non-structural components (ANSCs) in design process (based on IBC)," *International Journal of Disaster Resilience in the Built Environment*, vol. 16, pp. 133–147, 2012.