Mathematical Biosciences and Engineering

MBE, 19(6): 5867–5904.
DOI: 10.3934/mbe.2022275
Received: 10 February 2022
Revised: 21 March 2022
Accepted: 31 March 2022
Published: 08 April 2022

http://www.aimspress.com/journal/MBE

Research article

The heterogeneous Aquila optimization algorithm

Juan ZHAO[1] and Zheng-Ming GAO[2,3,*]

[1] School of electronics and information engineering, Jingchu University of Technology, Jingmen 448000, China
[2] School of computer engineering, Jingchu University of Technology, Jingmen 448000, China
[3] Institute of intelligent information technology, Hubei Jingmen industrial technology research institute, Jingmen 448000, China

* **Correspondence:** Email: gaozming@jcut.edu.cn; Tel: +867242355622.

Abstract: A new swarm-based optimization algorithm called the Aquila optimizer (AO) was just proposed recently with promising better performance. However, as reported by the proposer, it almost remains unchanged for almost half of the convergence curves at the latter iterations. Considering the better performance and the lazy latter convergence rates of the AO algorithm in optimization, the multiple updating principle is introduced and the heterogeneous AO called HAO is proposed in this paper. Simulation experiments were carried out on both unimodal and multimodal benchmark functions, and comparison with other capable algorithms were also made, most of the results confirmed the better performance with better intensification and diversification capabilities, fast convergence rate, low residual errors, strong scalabilities, and convinced verification results. Further application in optimizing three benchmark real-world engineering problems were also carried out, the overall better performance in optimizing was confirmed without any other equations introduced for improvement.

Keywords: Aquila optimization algorithm; meta-heuristic algorithm; nature-inspired algorithm; multiple updating principle

1. Introduction

Along with the development of modern science and technology, the details of problems in nature are considered so much as to every influential factor being considered. These factors would play the roles influencing both the apparent showing to human and each other. As a result, analytical solutions could be no longer obtained whenever the problems are a little complicated. To conquer such difficulties, the nature-inspired algorithms had been proposed several decades of years ago.

Genetic algorithm (GA) [1] might be the first candidate in literature, it soon became a hot spot to solve most of the problems human met. Convinced by the better performance, other optimization algorithms were soon proposed, such as the ant colony optimization (ACO) [2] algorithm, the particle swarm optimization (PSO) [3] algorithm and so on.

Talking about the PSO algorithm, scientists and engineers around the world were soon addicted to the beautiful and simple structure. Thorough details were contributed, the reason why it converged so fast, its stability together with the constraints of variables c_1, c_2 were confirmed [4]. Researches applied the PSO algorithm to solve every optimization problem they found, and they were also not satisfied with its current performance. Finding ways to improve the performance, in either convergence ratio or residual errors, soon became a hot spot at those years. Inertia weights [5], local unimodal sampling [6], regrouping [7], guaranteed convergence [8], and other improvements were soon proposed and better performance confirmed. Literally speaking, the improvements could be classified as: 1) improvements of variables, such as the chaotic mapping [9], Levy flight [10]. 2) re-constructions of updating equations, such as the guaranteed convergence, regrouping methods. 3) improvements of controlling parameters, for instance, the inertia weights. 4) Hybridizations, such as the hybridization of PSO with GA [11], PSO with ACO [12].

Among all of the improved PSO algorithms, the heterogeneous PSO (HPSO) improvement [13] was a most shining one, it might be an inspiration of most of the modern optimization algorithms proposed recently.

In the early years of computational optimization, all of the individuals in swarms would following a same style to update their positions. For example, in the GA, they would crossover, mutate, while in the PSO algorithm, all of the individuals would update their positions according to their current velocities, the global best candidate, and their current best trajectories. Individuals in swarms of the bat algorithm (BA) [14] would also follow a same equation to update their current positions. While in the HPSO algorithm, individuals in swarms would select a way randomly among the cognitive-only, social-only, barebones, and modified barebones operations. The HPSO was also confirmed better in optimization.

Heterogeneous improvements would give individuals in swarm multiple ways [15] or tunnels [16] to update their positions. In such conditions, individuals could seek help for more variable operations. Some of them could take larger steps for a better exploration capability, some of them could take smaller steps to exploit the local optima and confirm whether it is the global one or not, and they would not follow a same way, which could increase their capability in both exploration and exploitation procedure. We can see that individuals in the sine cosine algorithm (SCA) [17] have two choices to update their positions, the arithmetic optimization algorithm (AOA) [18] and Harris hawk optimization (HHO) algorithm have four choices. More ways to update their positions might not always achieve in better performance, individuals in swarms with such operations could indeed afford variable choices and multiple characteristics. Henceforth, the heterogeneous improvements could be induced to be a multiple updating principle.

Aquila optimizer (AO) [19] was just proposed recently, it was claimed and verified with fast convergence rate and better performance than most of the other algorithms, it have been applied in intrusion detection [20], production forecasting [21].

The individuals in swarms of the AO algorithm also have four ways to update their positions, however, they can only choose two of them during the first 2/3 full procedure in exploration, and two of them at the rest exploitation procedure. Although the overall results were quite promising as reported by the proposer, individuals in swarms could not always take further operations to obtain better results in the exploitation procedure, which could be easily found in figure 9 in referenced paper [19]. This paper would report a heterogeneous improvement of the Aquila optimization algorithm with abbreviation HAO. The constant value 2/3 which constraint the individuals to follow two ways each during their exploration and exploitation procedure would be changed to a probability value, and consequently, individuals in the HAO swarms could have four ways to update their positions.

The main contribution of this paper would be:

1) The heterogeneous improvement of the AO algorithm was proposed in this paper, which did not introduce any further equations and improve the computer complexity. The proposed HAO algorithm only reconstructed the strategies which were proposed by the proposer.

2) The four strategies were classified by two groups with two proportional values p_1 and p_2, their influence of the convergence rate was simulated and a best set was chosen.

3) Qualitative analysis, intensification, diversification, and scalability capability of the proposed HAO algorithm were simulated and the acceleration rate were tested on either unimodal or multimodal benchmark functions.

4) Simulation experiments on unimodal, multimodal, together with three classical real-world engineering problems were carried out and comparison were made. Results confirmed the better performance of the propose HAO algorithm than the original AO algorithm together with some other well-known optimization algorithms.

The rest of this paper would be arranged as follows: In Section 2, the original AO algorithm was briefly reported, and the improved HAO was proposed. Simulation experiments would be carried out both on benchmark functions in Section 3 and real-world engineering problems in Section 4. Discussions would be made and conclusions would be drawn in Section 5.

2. The AO and proposed HAO algorithms

2.1. Strategies in the AO algorithm

There are four strategies for individuals in the AO algorithm:

Strategy 1: Expanded exploration.

$$X_i(t+1) = X_{best}(t) \times \left(1 - \frac{t}{T}\right) + X_M(t) - X_{best}(t) * r_1 \tag{1}$$

where $X_i(t+1)$, $X_{best}(t)$, $X_M(t)$ represent the position of i-th individuals at iteration $t+1$, the best location at the current iteration and the mean positions of all individuals at the current iteration respectively. $X_M(t)$ would be calculated as follows:

$$X_M(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t) \tag{2}$$

where $X_i(t)$ represents the position of i-th individuals at iteration t. N represents the number of individuals in swarms. r_1 is the random number in Gaussian distribution with the interval of 0 and 1.

Strategy 2: Narrowed exploration.

$$X_i(t+1) = X_{best}(t) \times Levy(D) + X_R(t) + (y-x) * r_2 \tag{3}$$

where $Levy(D)$ represents the Levy flights following equations:

$$\mathrm{Levy(D)} = \mathrm{s} \times \frac{\mu \times \sigma}{|\nu|^{\frac{1}{\beta}}} \tag{4}$$

where $s = 0.01$ is a constant parameter, r_2 is another random number. μ, ν are random numbers between 0 and 1. σ is calculated as follows:

$$\sigma = \frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\frac{\beta-1}{2}}} \tag{5}$$

where $\beta = 1.5$ is a constant value. $X_R(t)$ is a random selected candidate at the current iteration. y and x represent the spiral shape:

$$y = r \times \cos(\theta) \tag{6}$$

$$x = r \times \sin(\theta) \tag{7}$$

$$r = r_1 + U \times D_1 \tag{8}$$

$$\theta = -\omega \times D_1 + \theta_1 \tag{9}$$

$$\theta_1 = \frac{3\pi}{2} \tag{10}$$

where r_1 is a fixed number between 1 and 20. D_1 is integer numbers from 1 to the length of the problems. $\omega = 0.005$ is a fixed constant number.

Strategy 3: Expanded exploitation.

$$X_i(t+1) = \alpha \times [X_{best}(t) - X_M(t)] \\ +\delta \times [(UB - LB) \times r_3 + LB] \tag{11}$$

where [LB, UB] is the definitional domain of the given problem. α and δ are two fixed small numbers. r_3 is the third random number in Gaussian distribution.

Strategy 4: Narrowed exploitation.

$$X_i(t+1) = QF \times X_{best}(t) - G_1 \times X_i(t) \times r_4 \\ -G_2 \times Levy(D) + r_5 \times G1 \tag{12}$$

where QF denotes to a quality function used to equilibrium the search strategy, and calculated with the following equation:

$$QF(t) = t^{\frac{2 \times r_6 - 1}{(1-T)^2}} \tag{13}$$

$$G_1 = 2r_7 - 1 \tag{14}$$

$$G_2 = 2 \times \left(1 - \frac{t}{T}\right) \tag{15}$$

r_4, r_5, r_6, r_7 are the fourth to seventh random numbers involved in this algorithm.

2.2. Proposed HAO improved algorithm

Although individuals in the original AO swarm have four strategies and four ways to update their positions, however, according to the status of the prey and the Aquila, individuals could only choose the first two strategies in the early 2/3 iteration times called exploration procedure, while they would choose the latter two strategies at the rest of exploitation procedure. Apparently, the first two strategies would be more aggressive and could result in fast convergence, however, the latter two involve the re-initializing operations, which would give the individuals a chance to jump out of the local optima. Both of the first and latter two strategies have their own merits and result in better performance of the original AO algorithm.

Table 1. Pseudo code of HAO algorithm.

Stage	Pseudo-code
Initializing	Maximum iteration number M
	population size N
	dimensionality D
	constraint variables
	probabilities p_1, p_2, p_3
Exploration and Exploitation	While iteration <M
	calculate the fitness values
	calculate $\mathrm{X}_{\mathrm{best}}, X_M$
	for i=1 to N
	update parameters
	random number r_1, r_2, r_3
	if $\mathrm{r}_1 < p_1$
	if $r_2 < p_2$
	strategy 1
	else
	strategy 2
	end if
	else
	if $r_3 < p_2$
	strategy 3
	else
	strategy 4
	end if
	end if
	end for
	end while
output	the best candidate

However, we can easily find that the latter two strategies lack capability in approaching global optima, especially for the multimodal benchmark functions. Figure 9 in the referenced paper [19] showed that individuals would fail to obtain better positions any more in the later iterations for F5–F10, F12–

F15, F17–23, who were some unimodal benchmark functions and most of the multimodal benchmark functions. To eliminate such defects, the choice of strategies could be a random way with different probabilities, let alone the solemn separation of exploration and exploitation. The individuals are recommended to choose a way with different probabilities to explore or exploit the whole definitional domain with their own willing. Therefore, we introduce the heterogeneous improvement for the original AO algorithm, and proposed an improvement called the heterogeneous Aquila Optimizer (HAO). For simplicity, a pseudo code of the proposed HAO algorithm would be shown in Table 1.

The complexity would remain as $O(M \times N \times D)$ with a minimum change in proportional values.

3. Simulation experiments on benchmark functions

In this section, simulation experiments on benchmark functions would be carried out, simulation results would be discussed to find whether the proposed HAO algorithm would perform better than the original AO algorithm or not. And furthermore, considering the development of swarm intelligence and better performance, several other algorithms would also be involved in comparison, such as original AO, the antlion optimization (ALO) [22], African vultures optimization (AVOA) [23], equilibrium optimizer (EO) [24], grasshopper optimization algorithm (GOA) [25], the grey wolf optimizer (GWO) [26], Harris hawk optimization (HHO) [27], Moth-frame optimization (MFO) [28], mayfly optimization algorithm (MOA) [29], PSO, SCA and whale optimization algorithm (WOA) [30]. The parameters of all of the compared algorithms are shown in Table 2.

Table 2. Parameters setup.

Algorithm	values
AO	$U = 0.00565$; $r_1 = 10$; $\omega = 0.005$; $\alpha = 0.1$; $\delta = 0.1$; $G_1 \in [-1, 1]$; $G_2 \in [2, 0]$
ALO	$r_1 \in [0, 1]$; $I = 10^w$; $w = 2,3,4,5,6$
AVOA	$p_1 = 0.6$; $p_2 = 0.4$; $p_3 = 0.6$; $\alpha = 0.8$; $\beta = 0.2$; $\gamma = 2.5$
EO	$V = 1$; $a_1 = 2$; $a_2 = 1$; $GP = 0.5$
GOA	cMax = 1; cMin = 4e-5
GWO	$a_0 = 2$; $r_1, r_2 \in [0, 1]$
HHO	$q \in [0, 1]$; $r \in [0, 1]$; $E0 \in [-1, 1]$; $E1 \in [2, 0]$; $E \in [-2, 2]$;
MFO	$b = 1$
MOA	g = 0.8; gdamp = 0.8; a1 = 1.0; a2 = 1.5; a3 = 1.5; $\beta = 2$; dance = 5; fl = 1; dance_damp = 0.8; fl_damp = 0.99;
PSO	$c_1 = 1.5$; $c_2 = 1.5$;
SCA	$r_1 \in [1, 0]$; $r_2 \in [0, 2\pi]$; $r_3 \in [0, 2]$
WOA	$r_1, r_2 \in [0, 1]$; $a_0 = 2$; $b = 1$; $p \in [1, 0]$

3.1. Benchmark functions

Verifying the capability of an algorithm with benchmark functions is a classical way in optimization. In this paper, the improved HAO would be also applied in optimization benchmark functions. 5 unimodal, 5 unimodal with three-dimensional basin-like landscapes, and 11 multimodal benchmark functions would be involved [31], as shown in Table 3–5.

Table 3. unimodal scalable benchmark functions (D = 10).

No.	Names	Equations	domain[lb,ub]
F1	Ackley 1	$f(x) = -20e^{-0.02\sqrt{\frac{\sum_{i=1}^{d} x_i^2}{d}}} - e^{\frac{\sum_{i=1}^{d}\cos(2\pi x_i)}{d}} + 20 + e$	[−100, 100]
F2	Exponential	$f(x) = 1 - e^{-0.5\sum_{i=1}^{d} x_i^2}$	[−3, 3]
F3	Powell Sum	$f(x) = \sum_{i=1}^{d} \lvert x_i \rvert^{i+1}$	[−1, 1]
F4	Sargan	$f(x) = \sum_{i=1}^{d} d\left(x_i^2 + 0.4 \sum_{j=1, j\neq i}^{d} x_i x_j \right)$	[−100, 100]
F5	Sphere	$f(x) = \sum_{i=1}^{d} x_i^2$	[−100, 100]

Table 4. unimodal scalable benchmark functions with three-dimensional basin-like-landscapes (D = 10).

No.	Names	Equations	domain[lb,ub]
F6	Chung Reynolds	$f(x) = \left(\sum_{i=1}^{d} x_i^2 \right)^2$	[−100, 100]
F7	Csendes	$f(x) = \sum_{i=1}^{d} x_i^6 \left(2 + \sin\frac{1}{x_i} \right)$	[−100, 100]
F8	Holzman’s 2	$f(x) = \sum_{i=1}^{d} i x_i^4$	[−100, 100]
F9	Hyper-Ellipsoid	$f(x) = \sum_{i=1}^{d} i^2 x_i^2$	[−100, 100]
F10	Schumer Steiglitz 2	$f(x) = \sum_{i=1}^{d} x_i^4$	[−100, 100]

Table 5. multimodal scalable benchmark functions (D = 10).

No.	Names	Equations	domain[lb,ub]
F11	Alpine 1	$f(x)=\sum_{i=1}^{d}\lvert x_i\sin(x_i)+0.1x_i\rvert$	[−100, 100]
F12	Cosine Mixture	$f(x)=\frac{d}{10}+\sum_{i=1}^{d}x_i^2-\frac{1}{10}\sum_{i=1}^{d}\cos(5\pi x_i)$	[−100, 100]
F13	Griewank	$f(x)=\sum_{i=1}^{d}\frac{x_i^2}{4000}--\cos\prod\left(\frac{x_i}{\sqrt{i}}\right)+1$	[−100, 100]
F14	Inverted Cosine-Wave	$f(x)=d-1-\sum_{i=1}^{d}\left\{e^{\left[\frac{-\left(x_i^2+x_{i+1}^2+0.5x_ix_{i+1}\right)}{8}\right]}cos\left(4\times\sqrt{x_i^2+x_{i+1}^2+0.5x_ix_{i+1}}\right)\right\}$	[−100, 100]
F15	Pathological	$f(x)=\sum_{i=1}^{d-1}\left(0.5+\frac{\sin^2\sqrt{100x_i^2+x_{i+1}^2}-0.5}{1+0.001\left(x_i^2-2x_ix_{i+1}+x_{i+1}^2\right)^2}\right)$	[−100, 100]
F16 .	Rastrigin	$f(x)=\sum_{i=1}^{d}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	[−100, 100]
F17	Salomon	$f(x)=1-\cos\left(2\pi\sqrt{\sum_{i=1}^{d}x_i^2}\right)+0.1\sqrt{\sum_{i=1}^{d}x_i^2}$	[−100, 100]
F18	Shubert 6	$f(x)=\sum_{i=1}^{d-1}0.5+\frac{\sin^2\sqrt{x_i^2+x_{i+1}^2}-0.5}{\left[1+0.001\left(x_i^2+x_{i+1}^2\right)\right]}$	[−100, 100]
F19	Streched V Sine Wave	$f(x)=\sum_{i=1}^{d-1}\left(x_i^2+x_{i+1}^2\right)^{0.25}\left\{\sin^2\left[50\left(x_i^2+x_{i+1}^2\right)^{0.1}\right]+0.1\right\}$	[−100, 100]
F20	Venter and Sobiezcczanski-Sobieski's	$f(x)=\sum_{i=1}^{d}\left[x_i^2-100\cos^2(x_i)-100\cos\left(\frac{x_i^2}{30}\right)\right]+400$	[−100, 100]
F21	Xin-She YANG 6	$f(x)=1+\left\{\left[\sum_{i=1}^{d}\sin^2(x_i)\right]-e^{\sum_{i=1}^{d}x_i^2}\right\}\cdot e^{-\sum_{i=1}^{d}\sin^2\left(\sqrt{\lvert x_i\rvert}\right)}$	[−100, 100]

3.2. Simulation platform

Simulation experiments would be carried out with a HP server platform whose assembly is shown in Table 6.

Table 6. Hardware for simulation experiments.

Items	Component
Server	HPE ProLiant DL380 Gen10
CPU	Intel Xeon Bronze 3106×2
RAM	Kingston 32GB

The source code was compiled with Matlab 2021b. And for simplicity, the maximum iteration number would be fixed 200, and the population size would be 40. In order to reduce the influence of random numbers involved in the algorithms, Monte Carlo simulation methods would be introduced and all of the results, if needed, would be the averaged over 30 separated runs.

For fair comparison, all the algorithms involved in this paper would be set the same with the above setup.

3.3. Probability parameters

For the proposed HAO algorithm, there would be three probability parameters p_1, p_2, p_3. According to the source code of the AO algorithm provided by the proposer, p_1 might be a constant number near 2/3, and $p_2 = p_3 = 0.5$. However, we do not know exactly whether it is suitable for the HAO algorithm. As a probability value, they all in a definitional domain fallen into [0, 1], and therefore, we simulate both $p_1 - p_2$ and $p_1 - p_3$ with the mean least iteration number (MLIN) when the best fitness values are smaller than 5e–4, which is 0.5‰, a constant number frequently used in engineering. Reducing the influence or randomness involved in the algorithm, Monte Carlo method is introduced and the final results are the average over 30 separated independent runs. Results were shown in Figures 1–6.

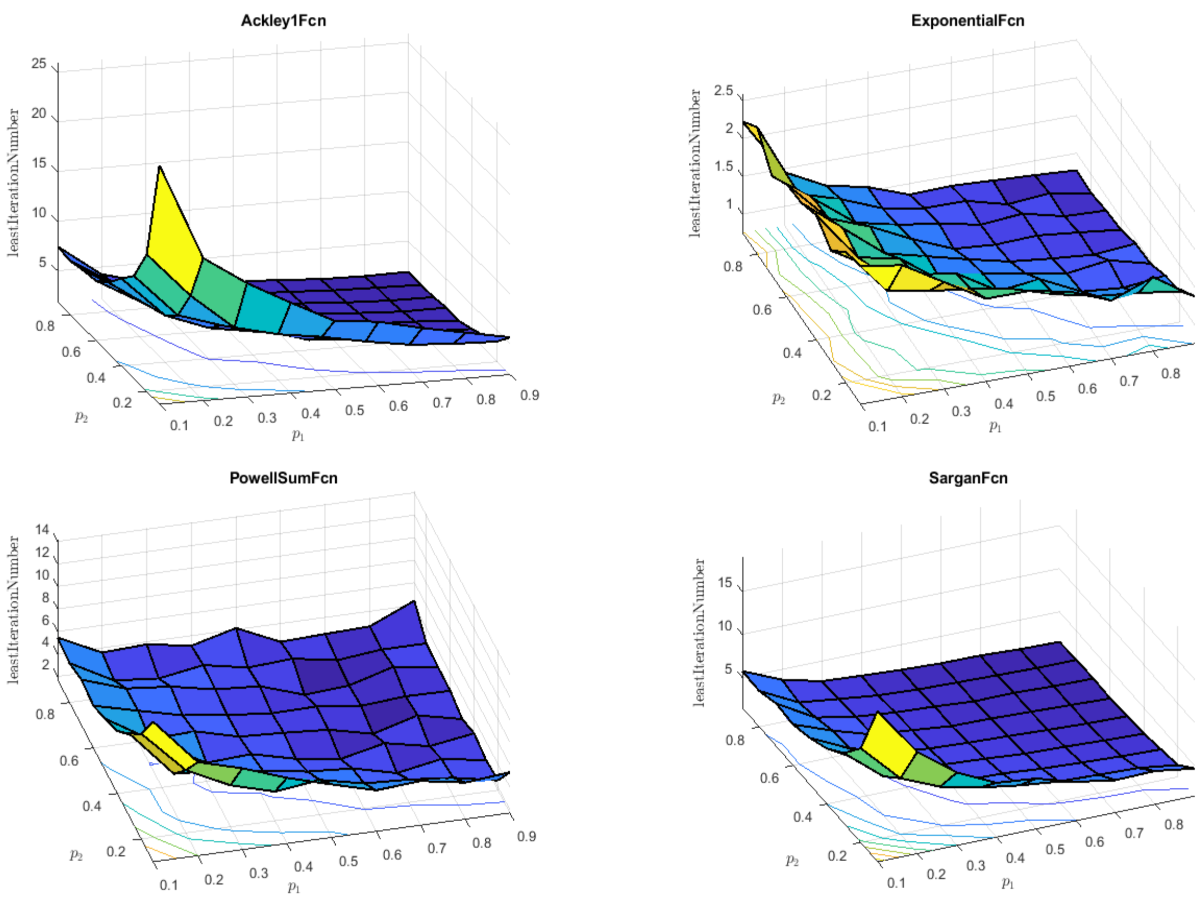

Figure 1. $p_1 - p_2$ relationship for different benchmark functions.

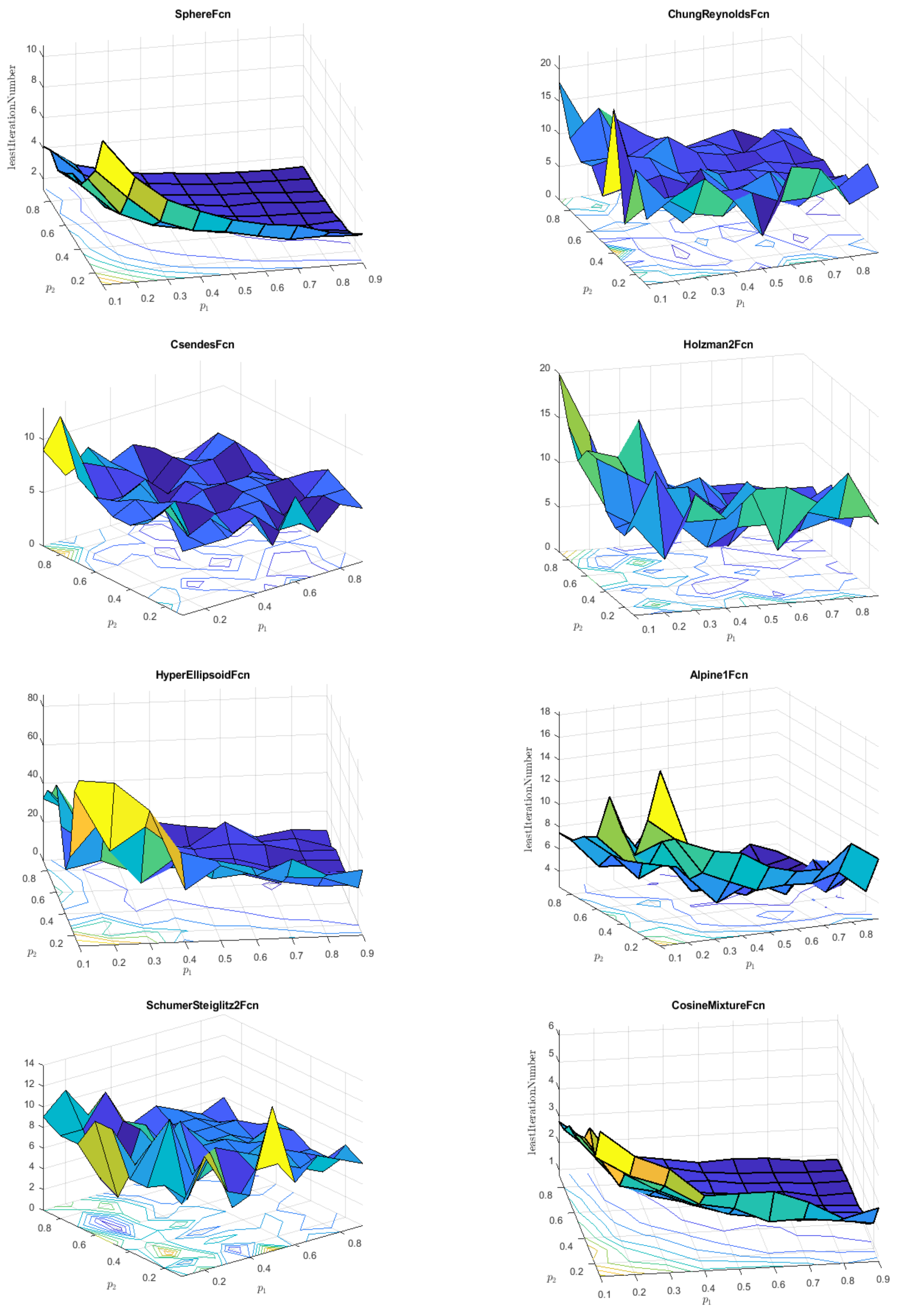

Figure 2. $p_1 - p_2$ relationship for different benchmark functions-continued.

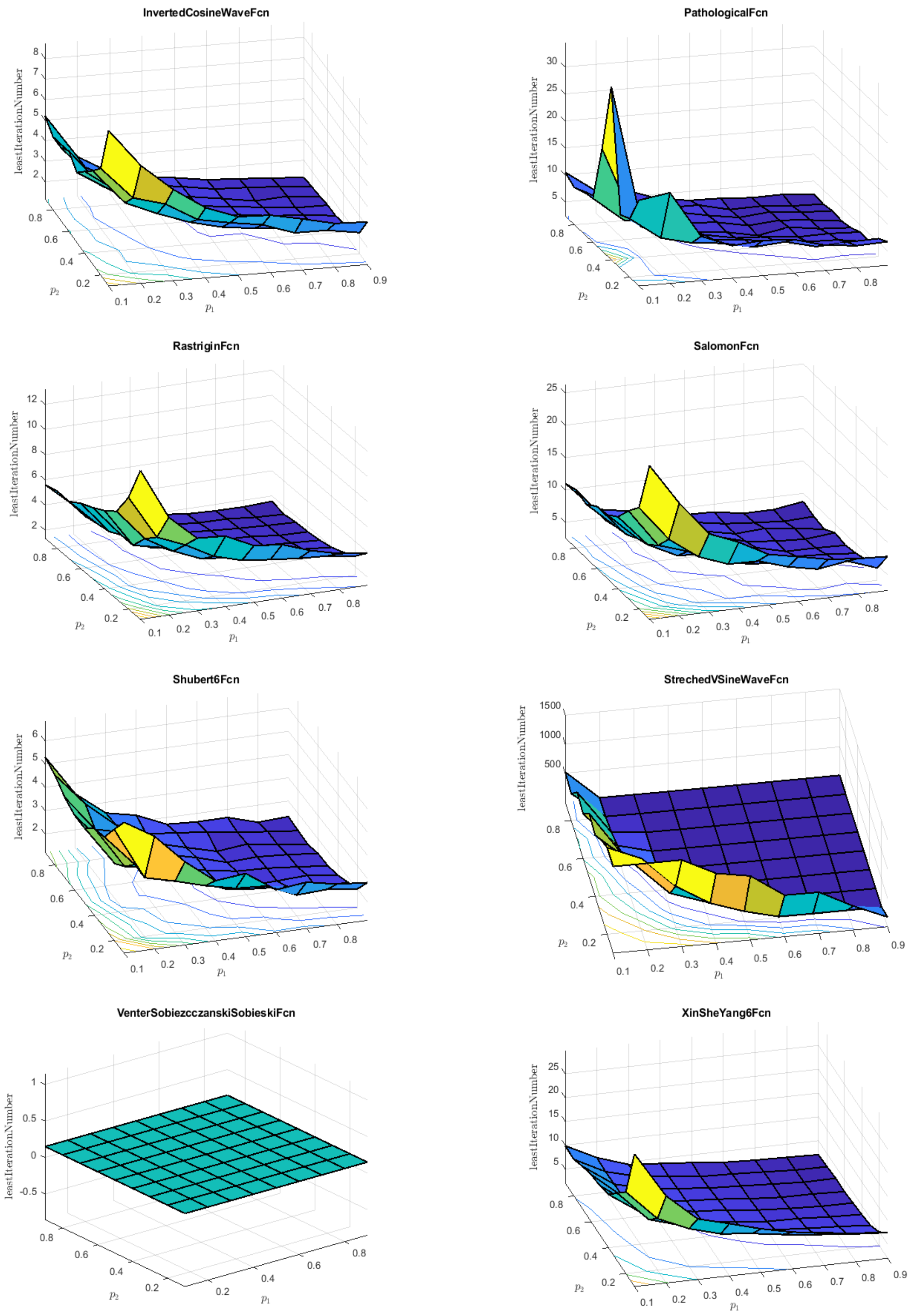

Figure 3. $p_1 - p_2$ relationship for different benchmark functions-continued.

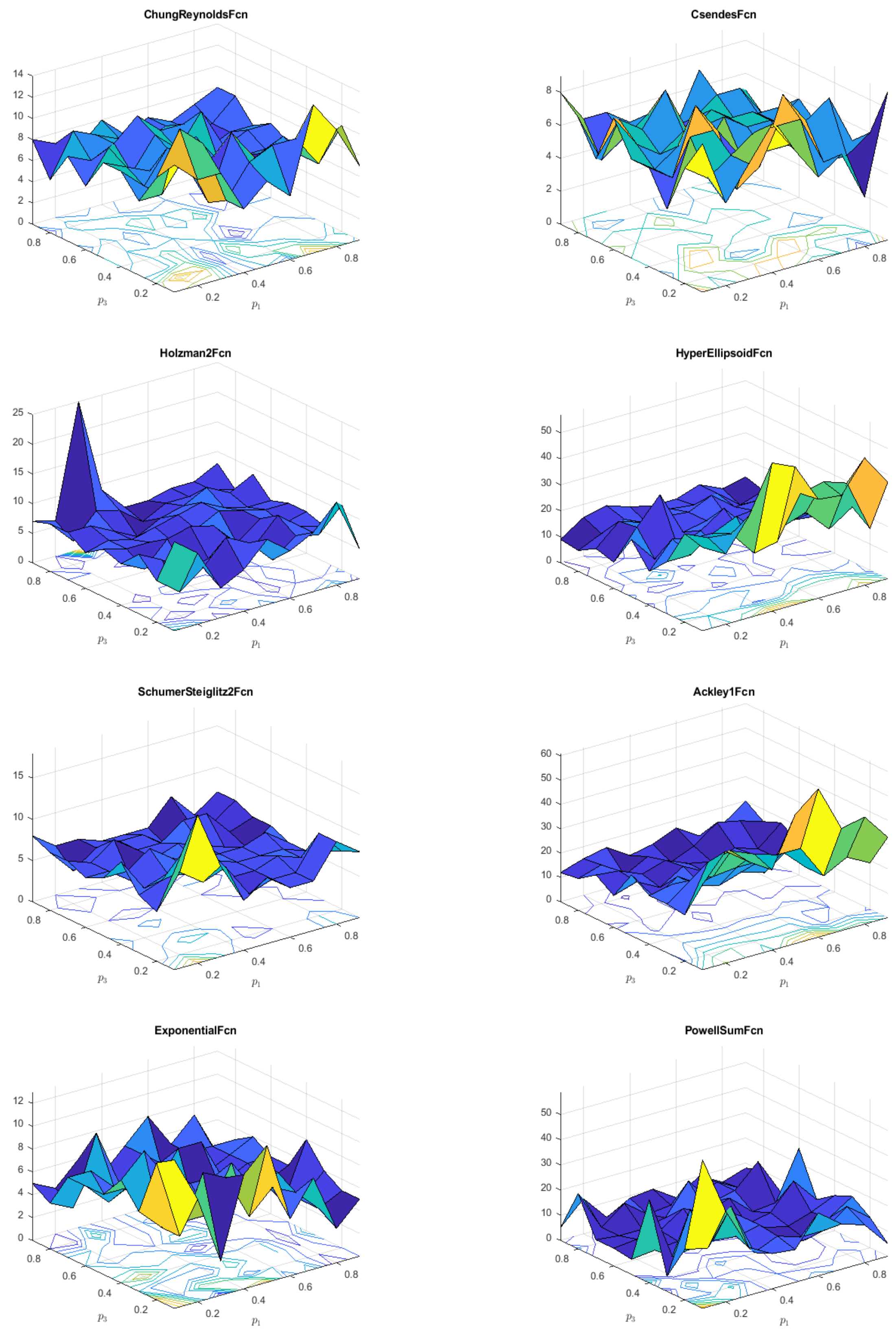

Figure 4. $p_1 - p_3$ relationship for different benchmark functions.

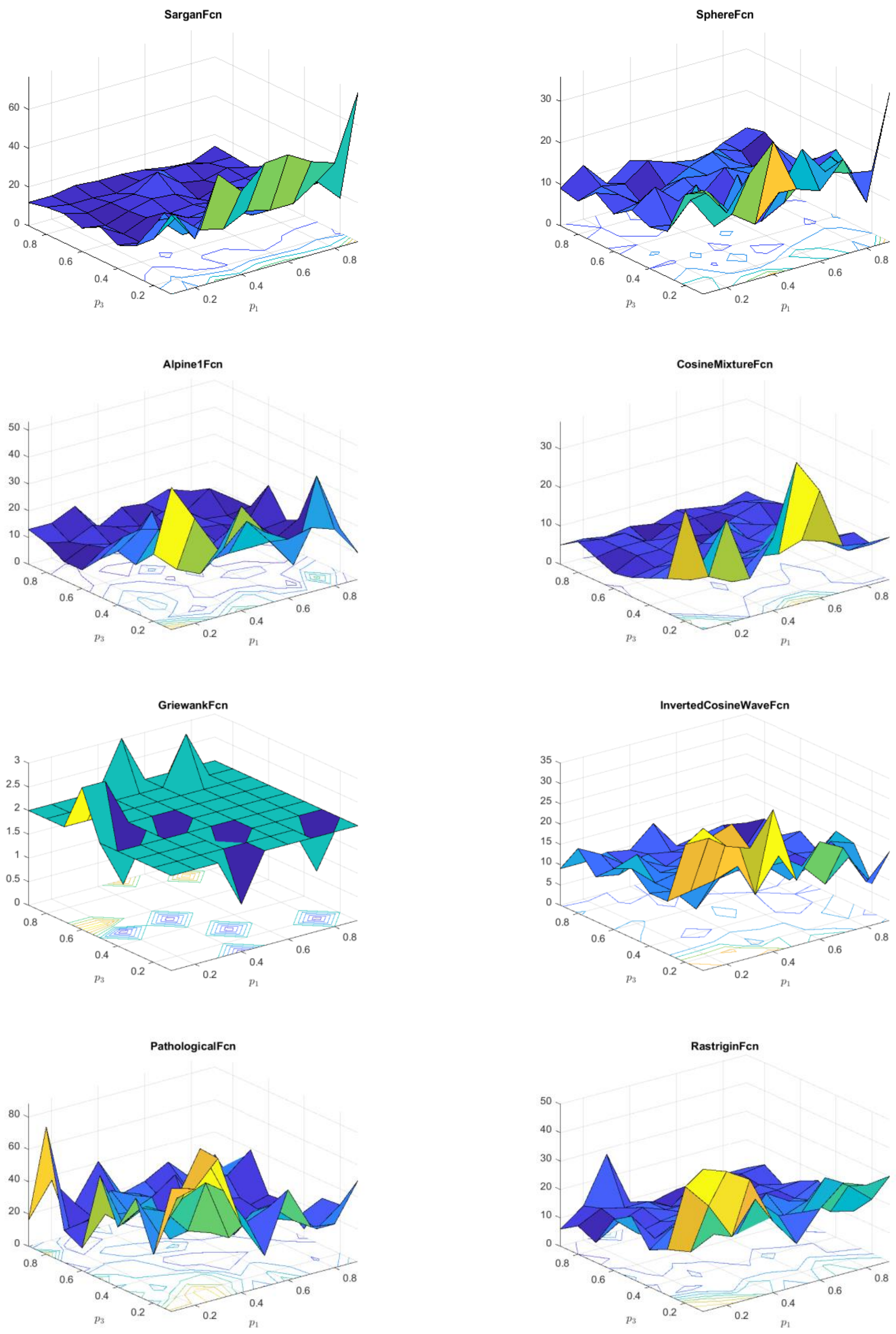

Figure 5. $p_1 - p_3$ relationship for different benchmark functions-continued.

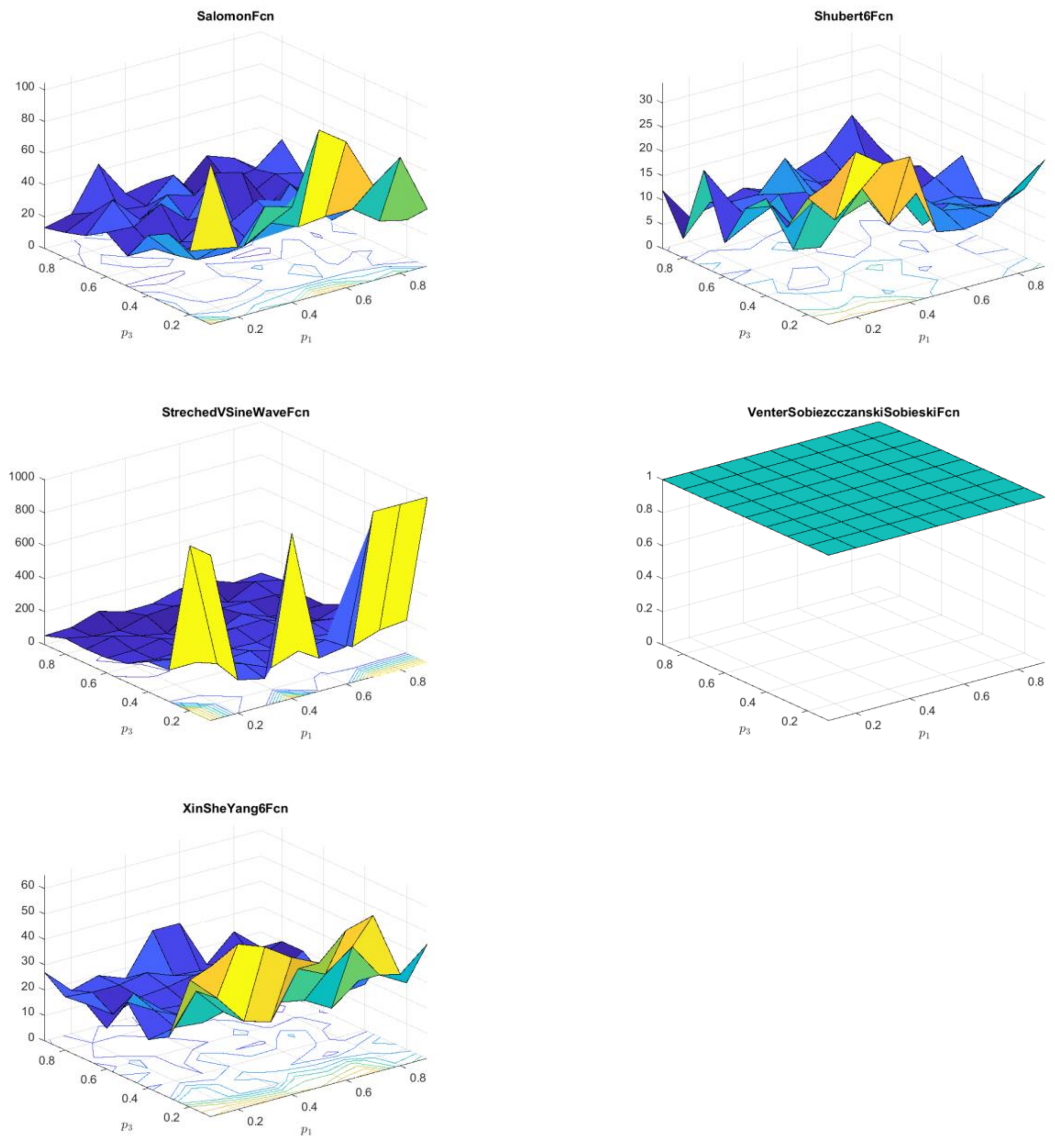

Figure 6. $p_1 - p_3$ relationship for different benchmark functions-continued.

We can conclude from Figure 1–3 that $p_1 = 0.7$, and $p_2 = 0.5$ might be a better value for most of benchmark functions, while some of the benchmark functions cannot tell a clear way, such as Csendes, Chung Renolds, Holzman's 2, Schumer Steiglitz 2 and Alpine 1 function. Specially, Venter Sobiezcczanski Sobieski function refused to be optimized with a minimum constraint of 1000 maximum iteration number, for saving time of running and computer complexity.

Considering the relationship between p_1 *and* p_3, no guaranteed relationship could be confirmed at the first glance of Figures 4–6. But the mean least iteration number for Hyper ellipsoid, Ackley 1, Sargan, Cosine Mixture and Stretched V Sine wave functions require non-smaller p_3 values. For simplicity, the values for p_3 might be also setup to 0.5 at the same value with p_2.

3.4. Qualitative experiments

Qualitative analysis is quite an import type of experiments, which would always give people a glance at the capability in optimization. Simulations would be carried out once for each of the benchmark functions involved in this paper, and results were shown in Figures 7–11.

For convenience, all of the dimensionality would be fixed to 10.

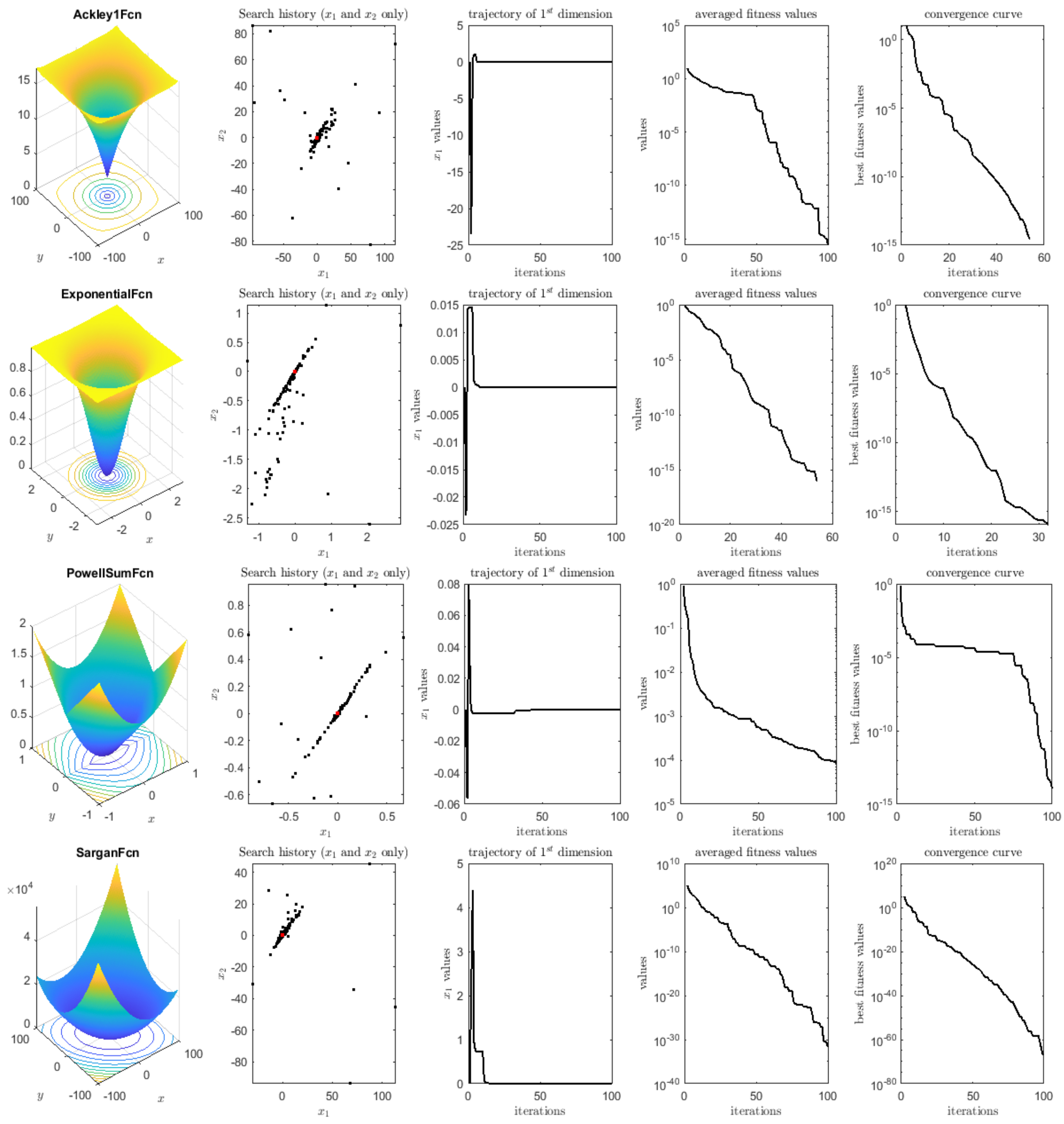

Figure 7. Qualitative results for the representatives.

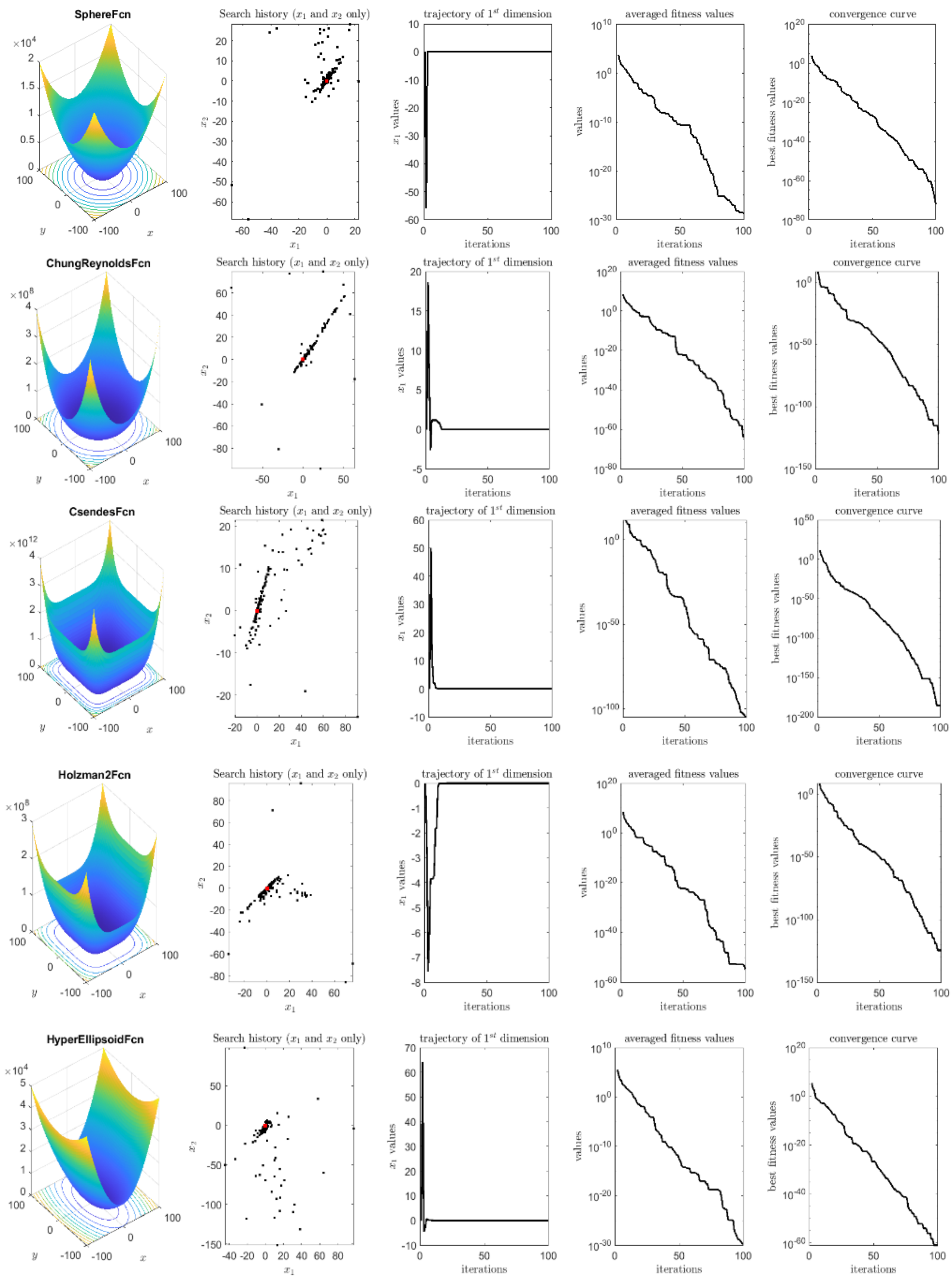

Figure 8. Qualitative results for the representatives -continued.

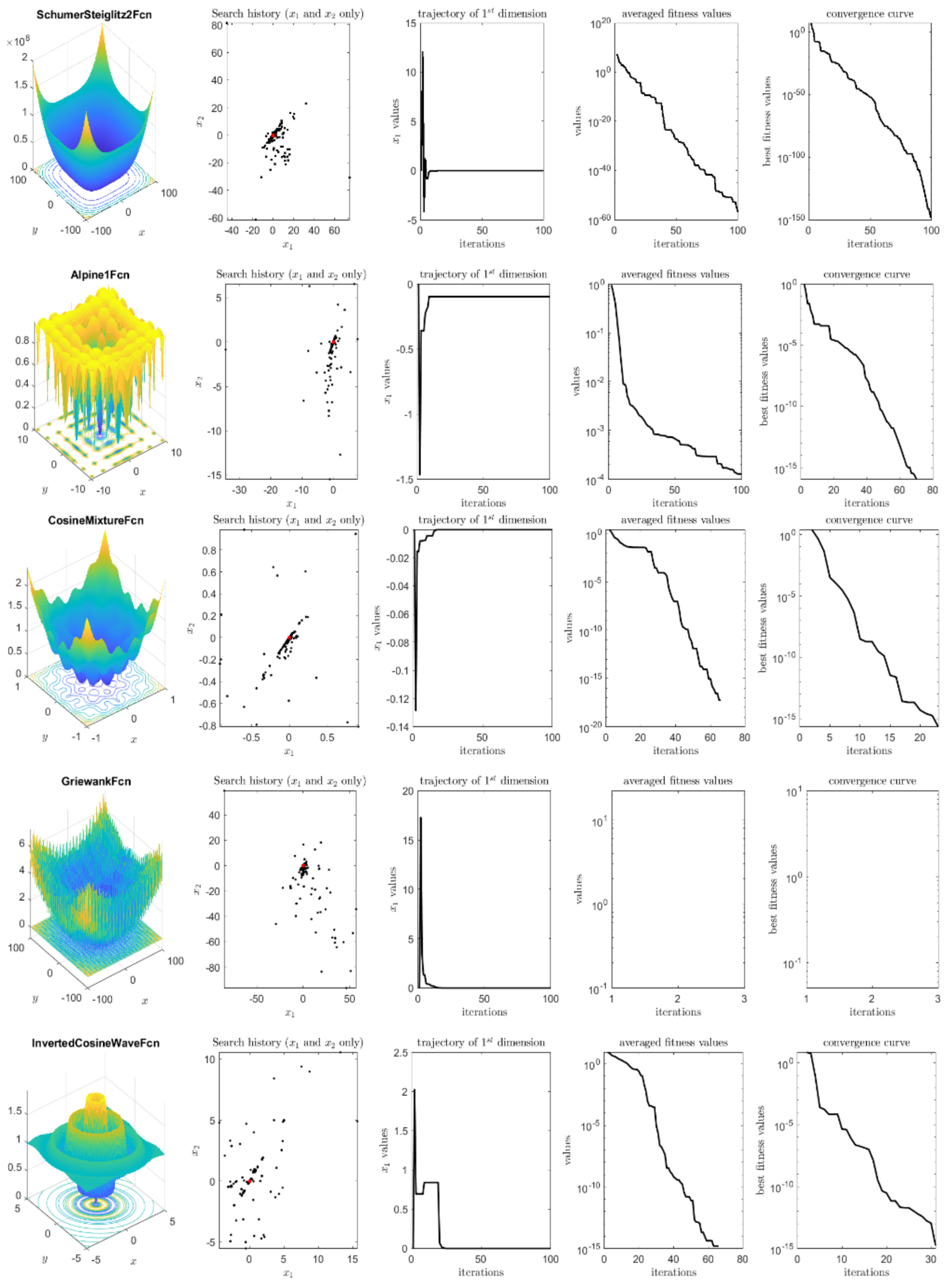

Figure 9. Qualitative results for the representatives -continued.

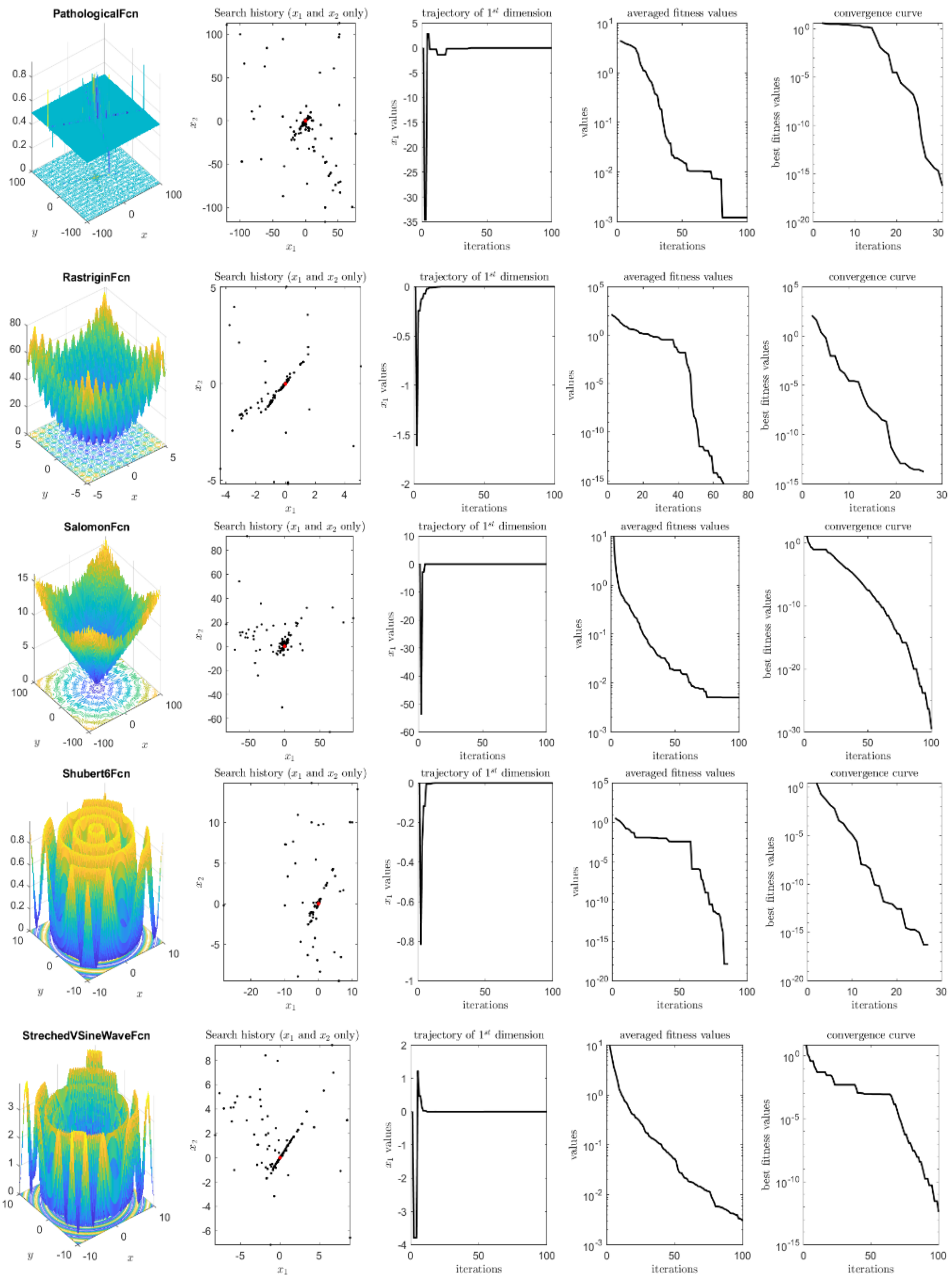

Figure 10. Qualitative results for the representatives -continued.

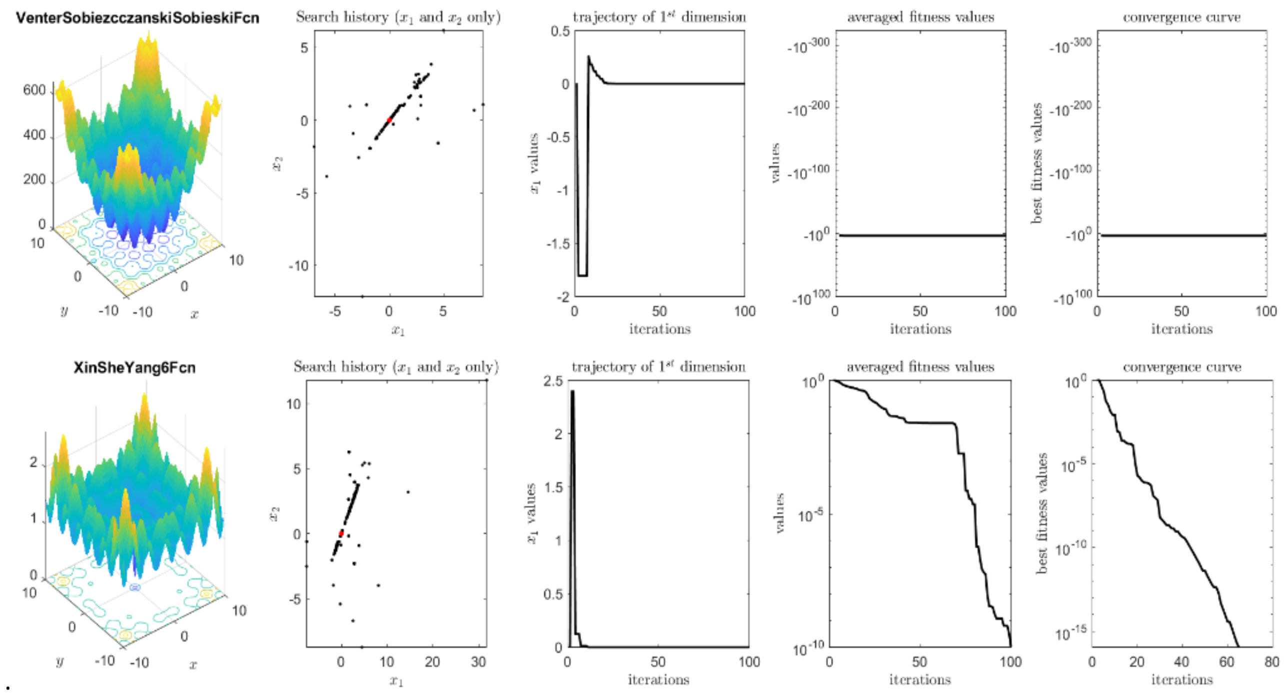

Figure 11. Qualitative results for the representatives -continued.

We can see from Figures 7–11 that almost all of the benchmark functions, either unimodal or multimodal, would be quickly optimized with the improved HAO algorithm, except Venter and Sobiezcczanski-Sobieski's, which should be so complicated as to fail to optimize by many algorithms. Individuals in swarms would quickly find the global optima, with fast convergence, lower residual errors. The search history for the former two dimensionality showed their better capability. Most of the trajectories of the first dimensionality confirmed the fast convergence.

3.5. *Intensification capability experiments*

Specifically speaking about the unimodal benchmark functions, they are easy to optimize at most times. Due to the only one global optima existed in their whole domain, individuals in swarms would approach the global optima without interference and disturbance. However, there is indeed a parameter balancing their capabilities. We called intensification experiments.

To reduce the influence of random numbers involved in the algorithms, Monte Carlo method would be also used and the best, worst, median, mean, standard derivation of the best results after 200 iterations would be calculated for 30 separated independent. Results were shown in Table7.

We can see from Table7 that the proposed HAO is definitely better, it performs 8 best of 10, among which 7 best, one equally best with the original AO algorithm, and failed to gain the best position compared to the GWO algorithm. Note that only the HAO, AO, GWO, HHO algorithms have gained the best positions in these experiments. Their better performance would be mainly relevant to their inherit mechanisms and the GWO algorithm performed better for F3 function, which proved that in some cases, multiple top candidates could result in better performance, although the EO algorithm, also being involved multiple top candidates, performed worse all the time.

Table 7. Intensification experiments results (D = 10).

Fcn	Items	HAO	AO	ALO	EO	GWO	HHO	PSO	SCA	WOA
F1	best	−8.881E−16	−8.881E−16	0	0	0	−8.881E−16	0	0	0
	worst	**0**	**0**	4.9594886	1.09246E−13	0.5672173	**0**	0.6567802	0.6151628	2.1770439
	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**−1.33227E-16**	**−1.33227E-16**	0.349809815	3.77476E−15	0.002836087	**−1.33227E-16**	0.05238038	0.004490143	0.016635884
	std	**3.17939E−16**	**3.17939E−16**	0.930088506	1.14419E−14	0.04010832	**3.17939E−16**	0.145110562	0.045383772	0.173740625
F2	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**0**	**0**	2.41267E−08	1.11022E−16	2.22045E−16	**0**	1.07827E−08	2.57222E−06	1.11022E−16
	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**0**	**0**	6.71821E−10	5.55112E−19	1.72085E−17	**0**	2.07952E−10	8.93685E−08	2.77556E−18
	std	**0**	**0**	2.55674E−09	7.85046E−18	4.32465E−17	**0**	1.00964E−09	3.84951E−07	1.73768E−17
F3	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	1.71345E−07	1.3404E−06	0.002712803	5.09874E−41	**4.89302E−38**	5.38875E−29	1.99386E−09	1.27563E−07	2.83673E−32
	median	0	0	0	0	**0**	0	0	0	0
	mean	8.59349E−10	1.32862E−08	0.000188208	6.06182E−43	**7.02152E−40**	2.76018E−31	5.25788E−11	8.6424E−10	4.08456E−34
	std	1.21158E−08	1.1398E−07	0.000528951	4.92415E−42	**4.29465E−39**	3.81024E−30	2.43252E−10	9.32274E−09	2.80826E−33
F4	best	**0**	0	0	0	0	0	0	0	0
	worst	**1.6596E−123**	2.68981E−59	0.008148981	8.83296E−23	1.47557E−19	2.22569E−39	5.58975E−06	0.94986434	4.73958E−08
	median	**0**	0	0	0	0	0	0	0	0
	mean	**8.5616E−126**	1.51816E−61	0.000121248	6.57998E−25	3.60452E−21	2.39605E−41	1.39242E−07	0.014161434	2.64588E−10
	std	**1.1736E−124**	1.91108E−60	0.0006726	6.47525E−24	1.7605E−20	2.19965E−40	5.45541E−07	0.092392958	3.35656E−09
F5	best	**0**	0	0	0	0	0	0	0	0
	worst	**1.5031E−130**	7.21475E−47	1.93144E−05	1.46308E−27	7.13202E−23	1.88131E−43	4.29984E−08	0.006006323	3.6791E−31
	median	**0**	0	0	0	0	0	0	0	0
	mean	**1.6376E−132**	3.60737E−49	6.89004E−07	2.55882E−29	7.41054E−25	9.67108E−46	7.31406E−10	0.000112339	2.52022E−33
	std	**1.4821E−131**	5.1016E−48	2.31881E−06	1.52145E−28	5.24025E−24	1.33026E−44	3.82728E−09	0.000660868	2.62669E−32
F6	best	**0**	0	0	0	0	0	0	0	0
	worst	**4.6398E−240**	4.6215E−103	5.46088E−11	7.90852E−56	8.05404E−45	1.92346E−82	6.57684E−16	0.000149337	2.58423E−59
	median	**0**	0	0	0	0	0	0	0	0
	mean	**2.3199E−242**	2.3108E−105	5.83399E−13	1.15805E−57	5.01269E−47	9.61742E−85	5.3483E−18	1.64084E−06	1.29331E−61
	std	**0**	3.2679E−104	4.19661E−12	7.23204E−57	5.72004E−46	1.36009E−83	4.83976E−17	1.30259E−05	1.82732E−60
F7	best	**0**	0	0	0	0	0	0	0	0
	worst	**0**	2.4556E−128	6.67468E−14	8.57806E−67	1.61691E−49	6.2895E−127	7.41427E−15	0.178898289	1.39788E−45
	median	**0**	0	0	0	0	0	0	0	0
	mean	**0**	1.2278E−130	6.32863E−16	8.40608E−69	9.51101E−52	3.1748E−129	4.02523E−17	0.000992967	8.0714E−48
	std	**0**	1.7364E−129	5.20692E−15	7.45379E−68	1.14851E−50	4.4473E−128	5.24803E−16	0.012706253	9.97234E−47
F8	best	**0**	0	0	0	0	0	0	0	0
	worst	**1.7671E−253**	6.1768E−100	8.61175E−10	1.23148E−46	3.58526E−38	2.11815E−84	5.42758E−12	0.014168893	1.41375E−46
	median	**0**	0	0	0	0	0	0	0	0
	mean	**9.0354E−256**	3.0884E−102	1.17105E−11	8.13848E−49	4.38731E−40	1.07815E−86	4.08357E−14	0.000180906	1.02521E−48
	std	**0**	4.3677E−101	7.01814E−11	8.95168E−48	3.35327E−39	1.4978E−85	4.06497E−13	0.001412517	1.06371E−47

Continued on next page

Fcn	Items	HAO	AO	ALO	EO	GWO	HHO	PSO	SCA	WOA
	best	**0**	0	0	0	0	0	0	0	0
	worst	**6.5633E−126**	1.55672E−42	1902.684424	3.14469E−26	9.69724E−22	7.89724E−42	3.18004E−07	0.460748677	2.51842E−29
F9	median	**0**	0	0	0	0	0	0	0	0
	mean	**3.2924E−128**	7.78367E−45	85.65268113	3.44888E−28	1.70161E−23	6.59982E−44	9.55701E−09	0.005258053	2.9233E−31
	std	**4.6409E−127**	1.10077E−43	285.1771895	2.38927E−27	9.31255E−23	6.54555E−43	3.77459E−08	0.03811698	2.5359E−30
	best	**0**	0	0	0	0	0	0	0	0
	worst	**5.3821E−243**	2.1822E−116	5.07135E−10	1.35718E−47	2.7511E−38	1.39152E−81	1.69102E−13	0.029115168	2.47058E−46
F10	median	**0**	0	0	0	0	0	0	0	0
	mean	**2.6911E−245**	1.0939E−118	3.41545E−12	7.03595E−50	1.61158E−40	6.95764E−84	2.66075E−15	0.00027155	1.28746E−48
	std	**0**	1.543E−117	3.63283E−11	9.59781E−49	1.95278E−39	9.83952E−83	1.6196E−14	0.002562231	1.74734E−47

3.6. Diversification capability experiments

As for multimodal benchmark functions, they have more than one global optima, and consequently, individuals in swarms would be trapped in local optima. There is a need for individuals to be capable to run out of the local optima and approach the global one. This capability is called the diversification capability. To find out whether the proposed HAO algorithm has better diversification capability or not, similar experiments would be carried out with the intensification experiments, whereas this time the simulation would be carried out on multimodal benchmark functions. The results would be in a same situation with intensification experiments and shown in Table 8.

Table 8. Diversification experiments results (D = 10).

Fcn	Items	HAO	AO	ALO	EO	GWO	HHO	PSO	SCA	WOA
	best	0	0	0	0	0	**0**	0	0	0
	worst	0.000102444	0.000212601	0.686498568	4.1191E−08	0.000466289	**0**	0.000971432	0.06394849	0.940399825
F11	median	0	0	0	0	0	**0**	0	0	0
	mean	5.12222E−07	1.40426E−06	0.035056268	2.34813E−10	2.64455E−05	**0**	1.42278E−05	0.000620885	0.03160676
	std	7.24392E−06	1.5517E−05	0.117127269	2.92632E−09	9.19686E−05	**0**	7.93889E−05	0.004755222	0.149338695
	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**0**	**0**	0.591137009	0	3.33067E−16	**0**	0.147784252	8.23936E−06	3.33067E−16
F12	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**0**	**0**	0.039901749	0	1.05471E−17	**0**	0.000738941	8.4255E−08	4.996E−18
	std	**0**	**0**	0.117323503	0	4.92719E−17	**0**	0.010449923	6.21508E−07	3.57161E−17
	best	**−9**	**−9**	−8.271743663	−9	−9	**−9**	−8.763739656	−8.999963768	−9
	worst	**0**	**0**	0	0	0	**0**	0	0	0
F13	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**−1.35**	**−1.35**	−1.018132954	−1.332689027	−1.293365437	**−1.35**	−1.213239065	−1.24248688	−1.274377431
	std	**3.22**	**3.22**	2.465359483	3.180884043	3.091366535	**3.22**	2.898767185	2.991706983	3.072957522
	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**0**	**0**	6.553034763	3.031861151	3.258463936	**0**	3.095329066	3.857462194	5.252117521
F14	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**0**	**0**	0.619407638	0.234314999	0.234699015	**0**	0.315975027	0.135451624	0.098063259
	std	**0**	**0**	1.549603278	0.669642237	0.672165615	**0**	0.777237848	0.606556869	0.628334407

Continued on next page

Fcn	Items	HAO	AO	ALO	EO	GWO	HHO	PSO	SCA	WOA
	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**0**	**0**	3.999948399	2.552321449	2.647696583	**0**	3.501379209	3.745287122	3.999426852
F15	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**0**	**0**	0.504369	0.282061586	0.30369492	**0**	0.401829418	0.376056186	0.217401183
	std	**0**	**0**	1.214378287	0.699697313	0.740864965	**0**	0.973607626	0.916560706	0.705258618
	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**0**	**0**	55.71738174	4.974825684	8.97399354	**0**	19.36820549	35.47861084	36.42076237
F16	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**0**	**0**	3.517170973	0.034827673	0.413835634	**0**	1.334837167	0.758800881	1.081133301
	std	**0**	**0**	9.2891052	0.36483464	1.481162818	**0**	3.446218849	3.868735337	5.511231645
	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**9.33509E−56**	5.88658E−25	1.399873346	0.09987335	0.199873353	1.92798E−21	0.199874464	0.300008748	0.299873346
F17	median	**0**	0	0	0	0	0	0	0	0
	mean	**4.67551E−58**	3.3774E−27	0.059481002	0.014981002	0.016981002	1.79513E−23	0.021590163	0.020570275	0.018484168
	std	**6.60086E−57**	4.17777E−26	0.16893679	0.035751408	0.042619526	1.75014E−22	0.054933573	0.055290725	0.053082389
	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**0**	**0**	0.881248897	0.451552733	1.110889767	**0**	0.475219164	1.343309798	1.14382478
F18	median	**0**	**0**	0	0	0	**0**	0	0	0
	mean	**0**	**0**	0.044907776	0.029224494	0.038940139	**0**	0.034195028	0.133216919	0.093117857
	std	**0**	**0**	0.128685138	0.079145301	0.134079509	**0**	0.092296381	0.333790945	0.236320831
	best	**0**	**0**	0	0	0	**0**	0	0	0
	worst	**3.56539E−24**	0.00097255	2.147029601	1.08178633	2.709605815	1.08775E−07	2.820199371	4.749850325	1.300346311
F19	median	**0**	0	0	0	0	0	0	0	0
	mean	**2.84441E−26**	1.71583E−05	0.072212978	0.02114418	0.120502209	7.05637E−10	0.267364684	0.552169666	0.034804147
	std	**2.81437E−25**	0.000115889	0.266270392	0.087866749	0.389600108	7.94822E−09	0.66608576	1.323435619	0.153033519
	best	–1600	–1600	–1555.530553	–1600	–1600	–1600	–1599.983929	–1600	–1600
	worst	0	0	0	0	0	0	0	0	0
F20	median	0	0	0	0	0	0	0	0	0
	mean	–240	–240	–216.6310383	–240	–240	–240	–235.0797331	–239.6309232	–240
	std	572.7	572.7	518.2645127	572.7	572.7	572.7	561.0596732	571.884629	572.7
	best	**0**	0	0	0	0	**0**	0	0	0
	worst	**0**	0.00258651	1	1	1.00001814	**0**	1	1.000471085	1.000228419
F21	median	**0**	0	0	0	0	**0**	0	0	0
	mean	**0**	1.29326E−05	0.15	0.15	0.150000142	**0**	0.15	0.150043402	0.135006671
	std	**0**	0.000182894	0.36	0.36	0.3579678	**0**	0.36	0.35807104	0.342598028

Table 8 showed that both the proposed HAO, AO and the HHO algorithm could perform well in optimizing multimodal benchmark functions. Comparatively speaking, the proposed HAO succeeded for 9 among 11 experiments, while the HHO succeed 9, AO succeeded 8 times. We can see that the proposed heterogeneous improvements not only increase the diversification capability of the original AO algorithm, it further succeeded more than the HHO, it means that the HAO is definitely better than before.

3.7. Acceleration convergence analysis

Both the qualitative, intensification, and diversification experiments verified the better performance of the proposed HAO algorithm. In this section, acceleration convergence analysis would be carried out, the best fitness values versus iterations would show more apparent results, as shown in figures from Figures 12–16.

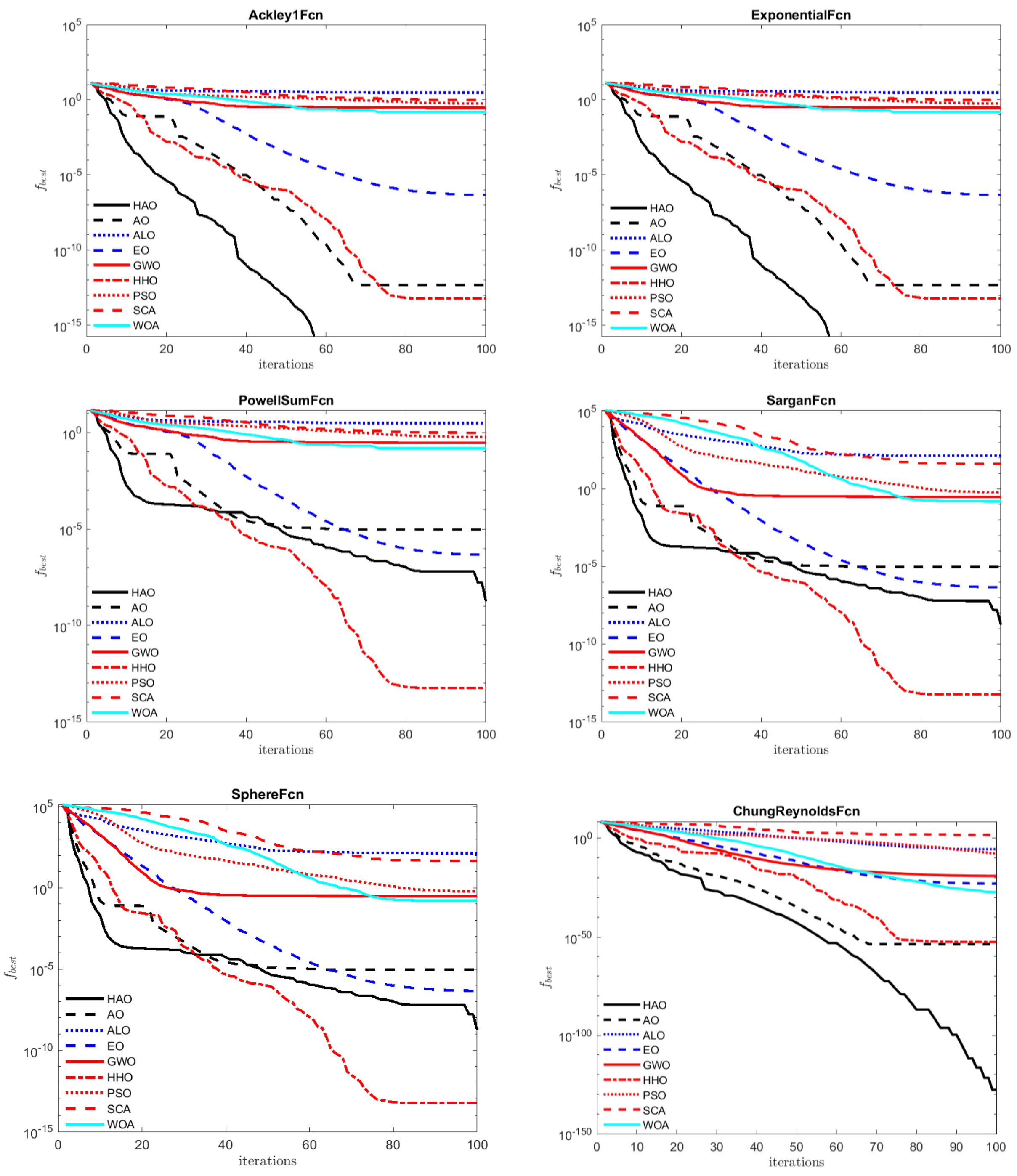

Figure 12. Acceleration convergence experiments on unimodal scalabe benchmark functions.

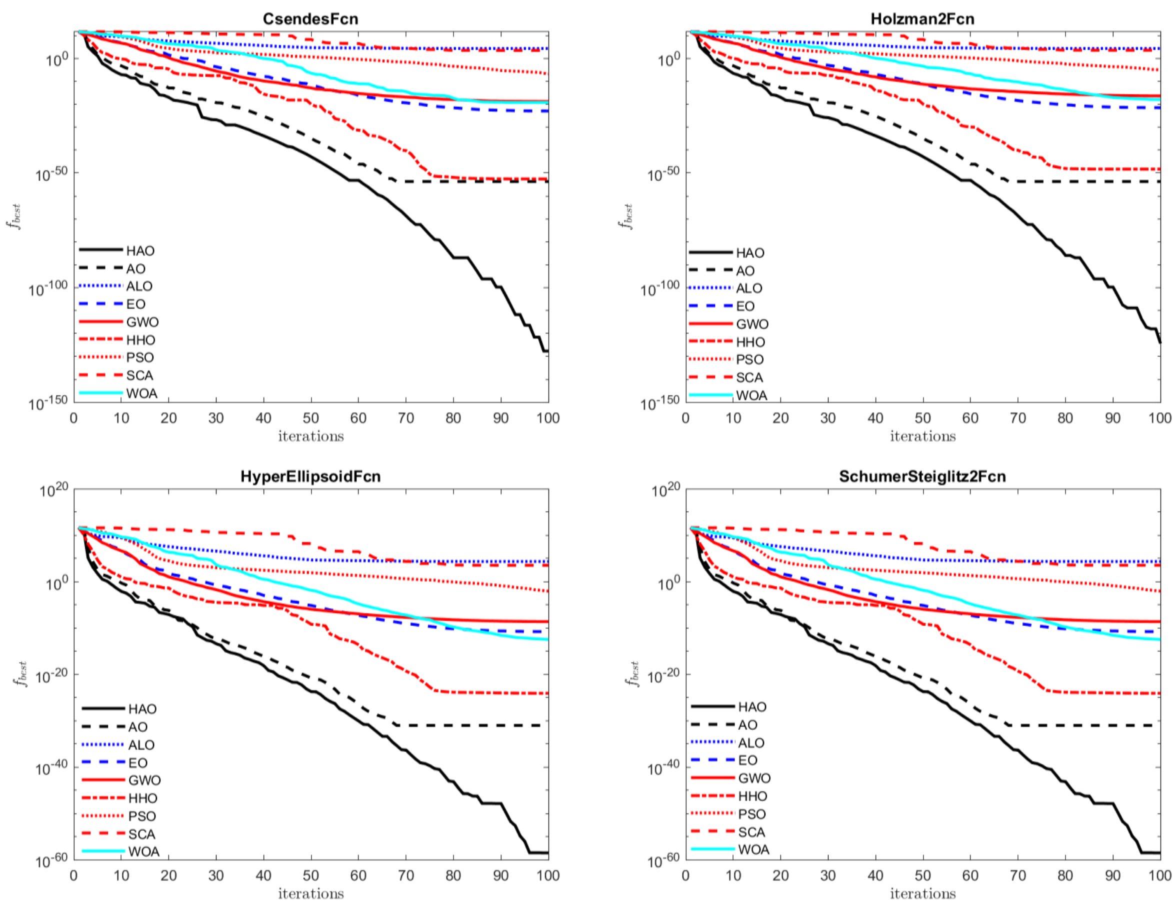

Figure 13. Acceleration convergence experiments on unimodal scalabe benchmark functions-continued.

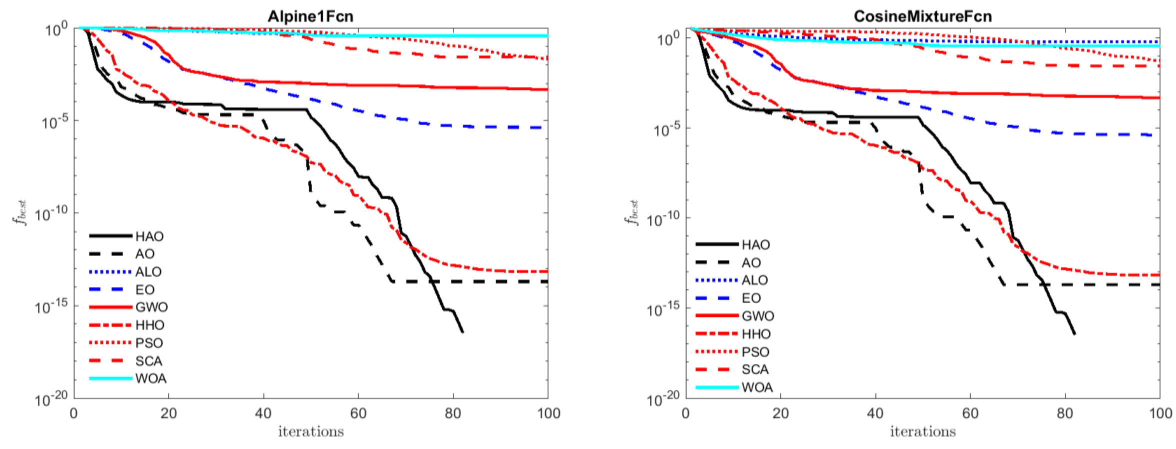

Figure 14. Acceleration convergence experiments on multimodal scalabe benchmark functions.

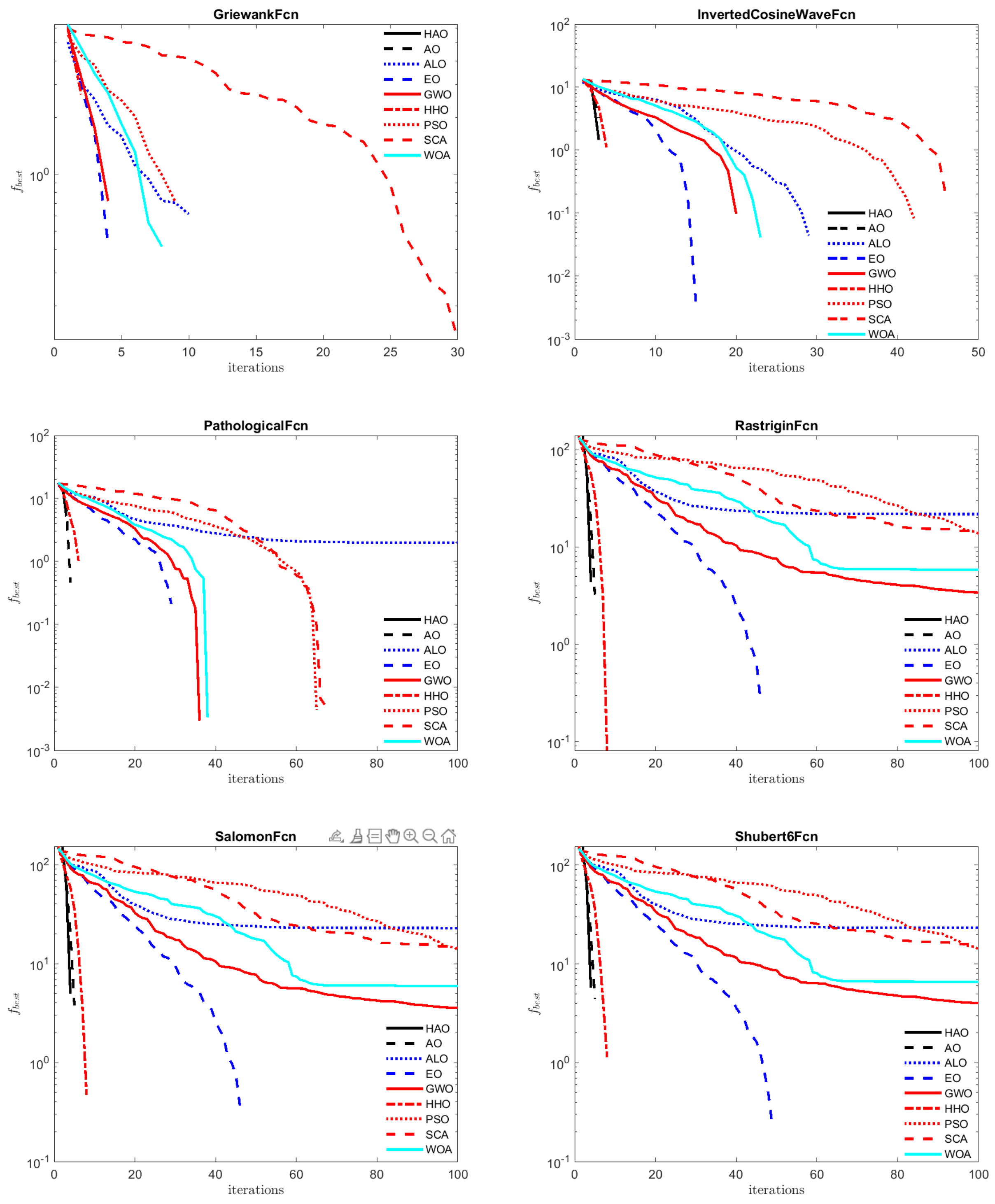

Figure 15. Acceleration convergence experiments on multimodal scalabe benchmark functions-continued.

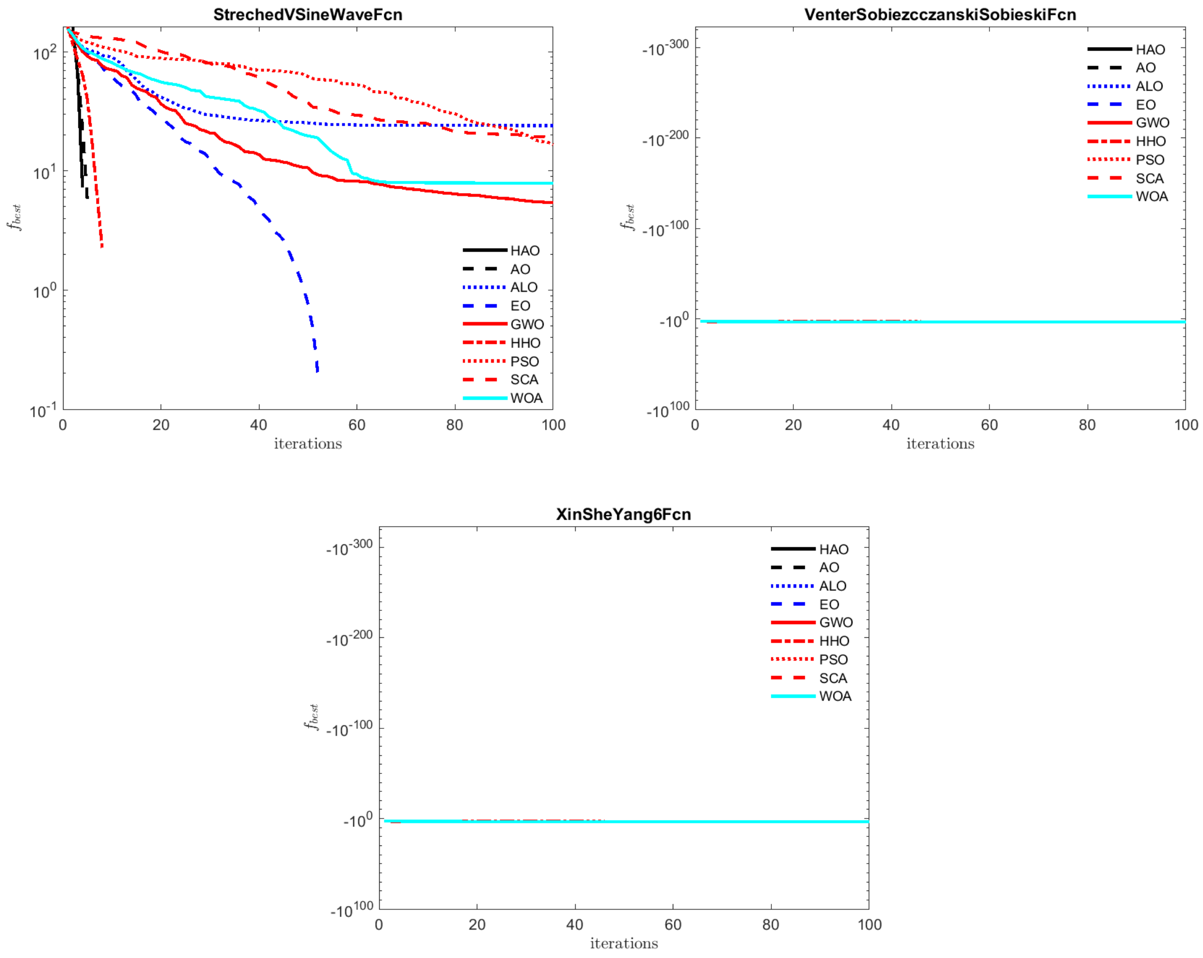

Figure 16. Acceleration convergence experiments on multimodal scalabe benchmark functions-continued.

Based on the averaged results in figures from Figures 12–16, we can see that the proposed improvement increases the convergence rate a lot, however, HHO algorithm became 3 bests of ten optimizing unimodal benchmark functions, while 7 bests of 10 for HAO algorithm, which remains most top lists. With Figure , we can find that the proposed HAO algorithm would perform the best for 8 from 11, while 3 multimodal benchmark functions, Griewank, Venter and Sobiezcczanski-Sobieski's, and Xin-She Yang 6 functions remain difficult to optimize. All of the involved nine algorithms could not optimize them at all.

Regarding the executive time of runs, the less time exhausted, the faster convergence rate. Under the same conditions that all results would be averaged over 30 independent runs, the executive time of the algorithms involved would be evaluated and compared, as shown in Figure 17.

We can see that for the unimodal Exponential function, the heterogeneous improvement took almost the same time as the original version. While for the multimodal Cosine Mixture function, it would take more time to finish the job than the original one. Meanwhile, a controversial conclusion might be drawn with simulation experiments on unimodal or multimodal benchmark functions. The executive time the algorithms take would be possibly based on the functions, other than the algorithms themselves.

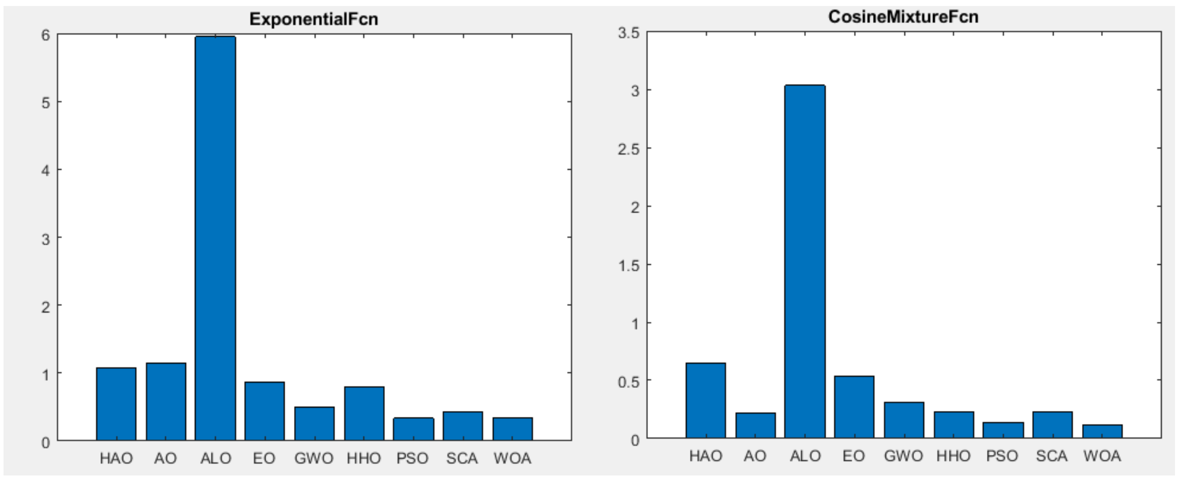

Figure 17. Executive time of parallel computing with workers = 14 for representative benchmark functions.

3.8. Scalability experiments

Although almost all of the above experiments verified the best performance of the proposed HAO algorithm in this paper, they were carried out with a fixed dimensionality, therefore, scalability experiments should also be carried out to confirm whether it would also perform the best when the dimensionality changed.

In this section, the dimensionality would be changed from 2, 10 and up to 100, with an interval of 10. The population size remains the same, and the overall results would remain an average over 30 separated independent runs with Monte Carlo method. Results were shown in figures from Figures 18–23.

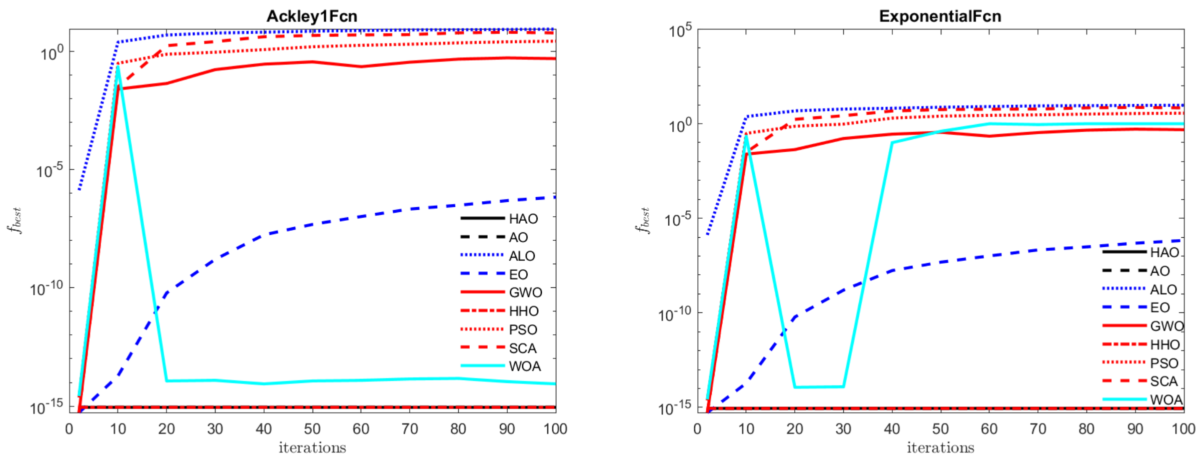

Figure 18. Scalability experiments on unimodal benchmark functions.

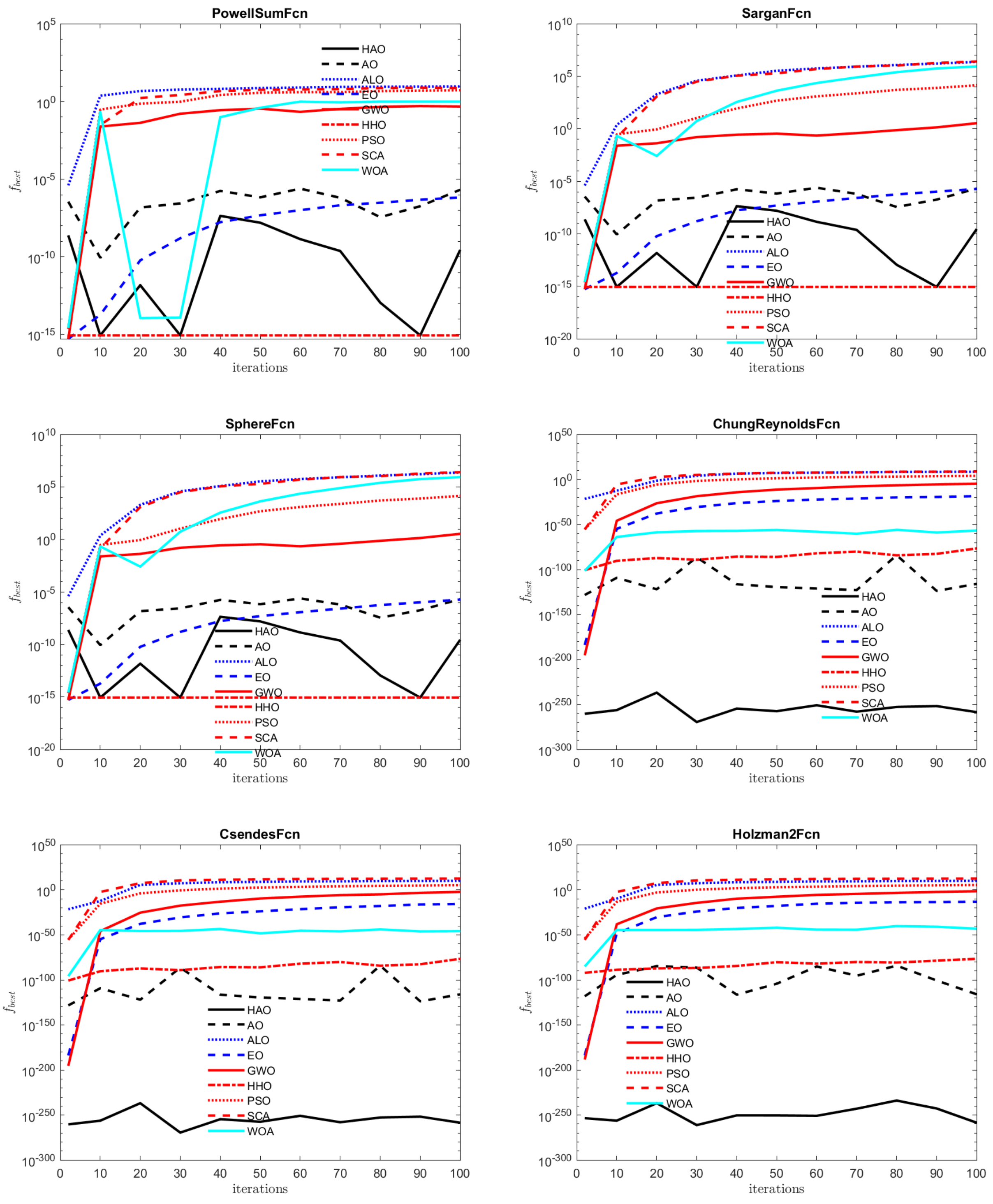

Figure 19. Scalability experiments on unimodal benchmark functions-continued.

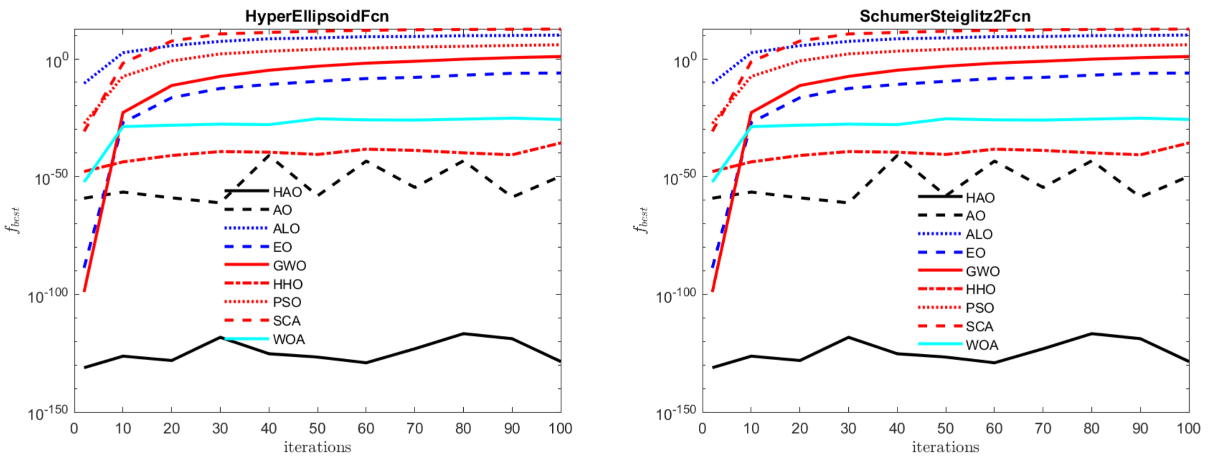

Figure 20. Scalability experiments on unimodal benchmark functions-continued.

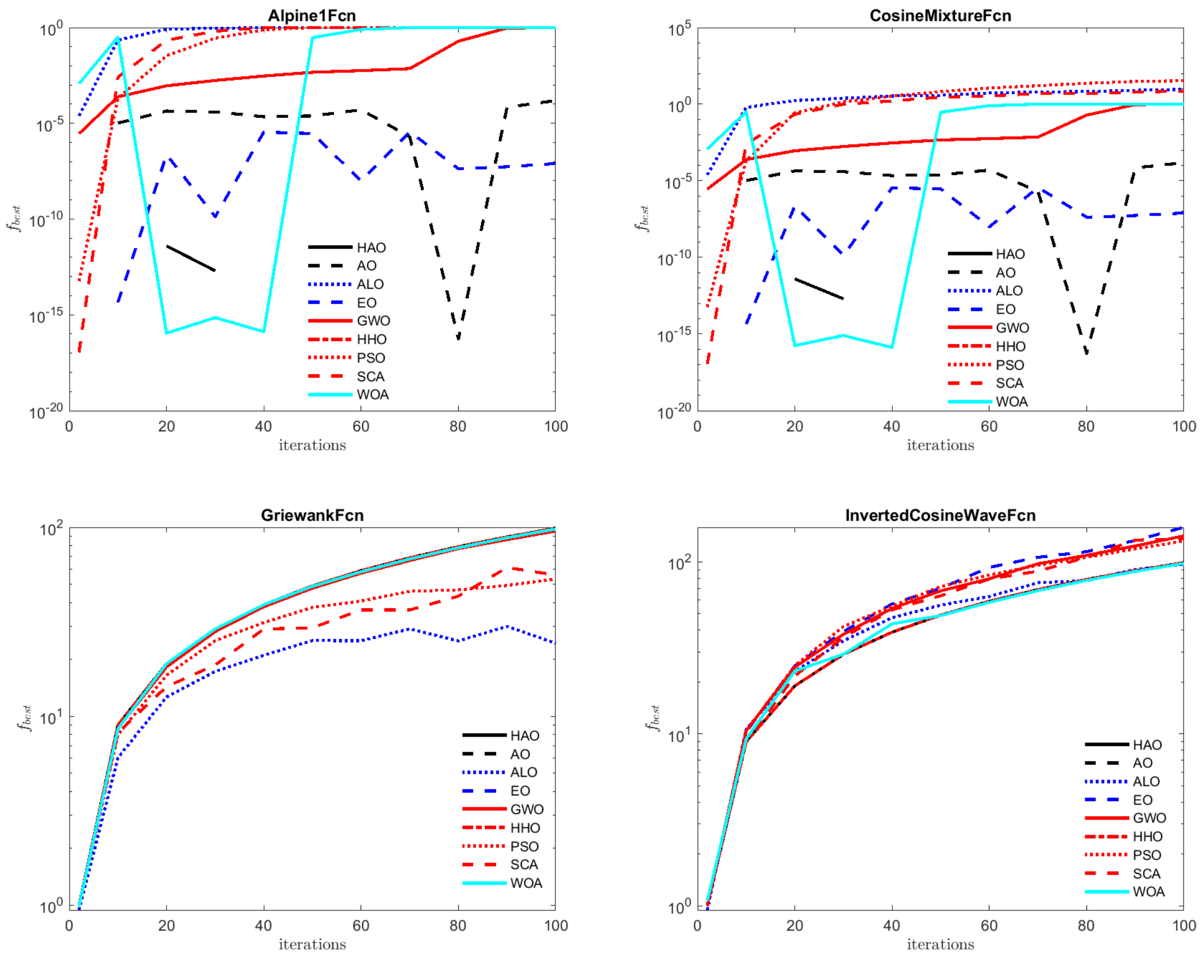

Figure 21. Scalability experiments on multimodal benchmark functions.

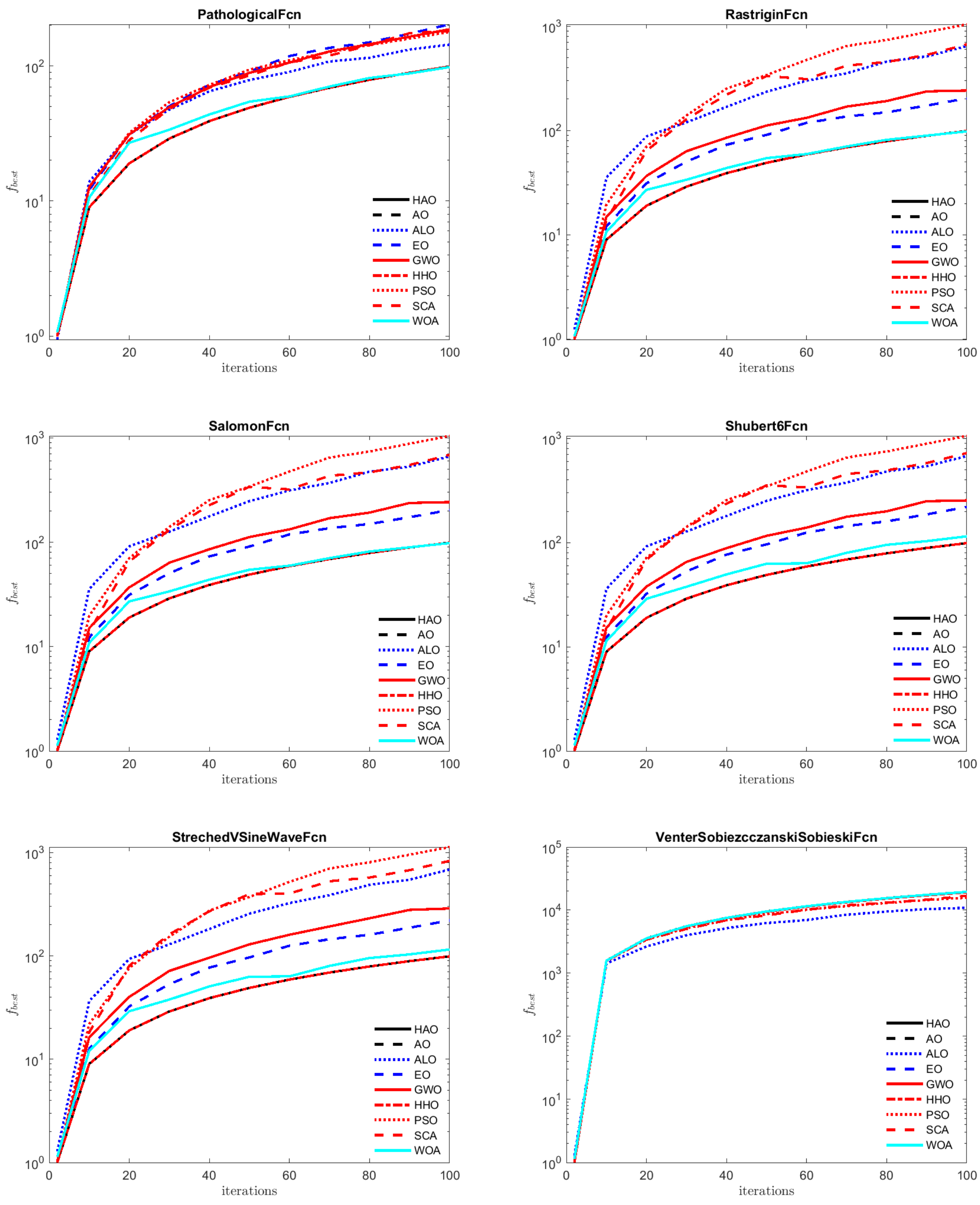

Figure 22. Scalability experiments on multimodal benchmark functions-continued.

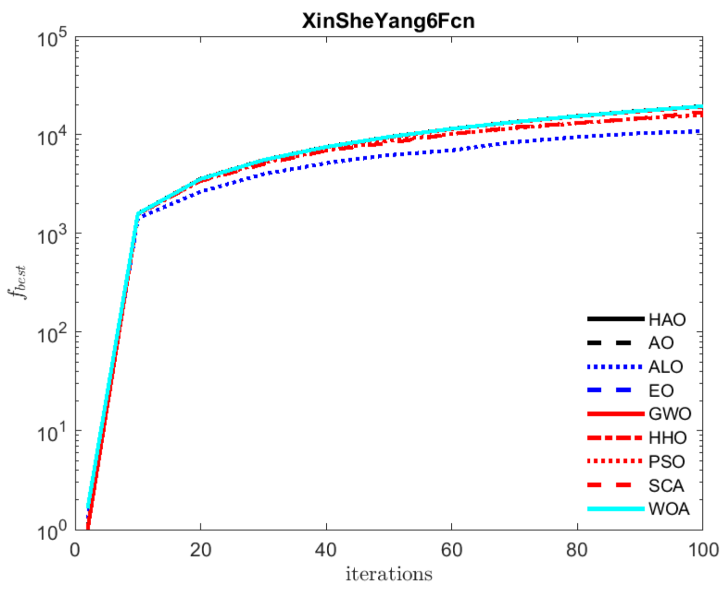

Figure 23. Scalability experiments on multimodal benchmark functions-continued.

We can see figures from Figures 18–20 that the proposed HAO disappeared in the results of Ackley 1 and Exponential benchmark function. Detailed study confirmed the zero values for AO and HAO. And results on unimodal benchmark functions verified the better performance, specifically, 8/10 bests.

Results on multimodal, as shown in figures from Figures 21–23, however, did not result in a same conclusion. The results of Alpine 1 and Cosine Mixture benchmark functions would follow a same style. However, all of the rest benchmark functions did not support the former conclusion. Although at most times, the proposed HHO algorithm perform better with HHO, AO algorithms than others. That is to say, for the multimodal benchmark functions, a fixed population size might be unable to be suitable the increasing dimensionality.

3.9. *Wilcoxon rank sum test*

Most of the conclusions demonstrated that the proposed HAO algorithm could perform better in optimization. Verification should be made furthermore. In this section, the Wilcoxon rank sum test would be carried out to confirm whether the better results are fallen in a same distribution with results obtained from other compared algorithms. The normal value $p = 0.05$ is adopted and verified, if $p \leq 0.05$, acceptance of the basic hypothesis would be made and consequently, the proposed HHO algorithm would perform better, on the contrary, if $p > 0.05$, rejection might be taken, and the datum would be derived from a same situation, and consequently, the proposed HHO algorithm could not be confirmed to perform better even though its mean, median, mean, worst, or standard derivation values are smaller. Results were shown in Table 9.

We can see from Table 9 that the proposed HHO could be verified at most times, only a few functions and algorithms against the hypothesis with bigger values.

Table 9. Intensification experiments results (D = 10).

Fcn	HAO	AO	ALO	EO	GWO	HHO	PSO	SCA	WOA
F1	NA	NA	6.38644E−05	6.24874E−05	6.38644E−05	NA	6.38644E−05	6.38644E−05	5.93632E−05
F2	NA	NA	6.38644E−05	**0.368120251**	1.59379E−05	NA	6.38644E−05	6.38644E−05	**0.368120251**
F3	NA	0.001706249	0.000182672	**0.212293836**	**0.472675594**	**0.427355314**	0.000182672	0.00058284	**0.427355314**
F4	NA	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F5	NA	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F6	NA	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F7	NA	6.38644E−05	6.38644E−05	6.38644E−05	6.38644E−05	6.38644E−05	6.38644E−05	6.38644E−05	6.38644E−05
F8	NA	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F9	NA	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F10	NA	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F11	NA	0.584248553	8.74499E−05	0.001699468	0.00016688	0.368120251	0.000565815	8.74499E−05	0.000533651
F12	NA	NA	6.38644E−05	NA	NA	NA	6.38644E−05	6.38644E−05	**0.168078319**
F13	NA	NA	6.38644E−05	0.002212542	6.38644E−05	NA	6.38644E−05	6.38644E−05	0.000751179
F14	NA	NA	6.38644E−05	0.000231246	6.38644E−05	NA	6.38644E−05	6.38644E−05	0.03484304
F15	NA	NA	6.38644E−05	6.38644E−05	6.38644E−05	NA	6.38644E−05	6.38644E−05	0.002212542
F16	NA	NA	6.38644E−05	**0.168078319**	6.34029E−05	NA	6.38644E−05	6.38644E−05	0.005858055
F17	NA	0.000439639	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F18	NA	NA	6.38644E−05	6.38644E−05	6.38644E−05	NA	6.38644E−05	6.38644E−05	6.38644E−05
F19	NA	0.000182672	0.025692445	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672
F20	NA	NA	6.38644E−05	NA	0.368120251	NA	6.38644E−05	6.38644E−05	NA
F21	NA	**0.368120251**	6.38644E−05	6.38644E−05	6.38644E−05	NA	6.38644E−05	6.38644E−05	6.38644E−05

4. Simulation experiments on real-world engineering problems

Based on the simulation experiments results on benchmark functions, we have found that the proposed HAO algorithm is quite promising in optimization. What about its capability in handling the real-world engineering problems? In this section, we would introduce the proposed HAO algorithm to find the best solution for some benchmark engineering problems.

Literally speaking, the difference between the real-world engineering problems and the benchmark functions might be the constraints. That is to say, there is no constraints for individuals in swarms when optimizing the benchmark functions, however, some definitional domain could not be searched or exploited when optimizing the real-world engineering problems.

For a given constraint problem:

$$\begin{gathered} \min\ f(x), x = \{x_1, x_2, \cdots, x_n\} \\ s.t.\ \ g_i(x) \le 0, i = 1,2,\cdots,m \\ h_i(x) = 0, \mathrm{i} = 1,2,\cdots,n \end{gathered}$$

where, $x_i \in [LB_i, UB_i]$ is the definitional domain for the i-th parameter. For simplicity, we introduce the penalty parameters to construct a new fitness function as follows:

$$F(x) = f(x) + P_{ie} \sum_{i=1}^{n} \max\{g_i(x), 0\}$$
$$+P_e \sum_{i=1}^{m} \max\{|h_i(x)|, \varepsilon\} \tag{16}$$

where m and n are the number of equal and unequal constraint equations. P_{ie}, P_e represent the penalty factors, which should be fixed numbers for simplicity.

In order to find the best solution for the real-world engineering problems, 10,000 separated independent runs would be involved in this section, and the final results would be the best one of them. The population size would be fixed to 40, and the maximum allowed iteration number is set with 1000.

4.1. Pressure vessel design

The pressure vessel design problem is four-dimensional structural design problem. It contains four non-equal and four equal constraints. Applying the proposed HAO algorithm, we got the results and compared with other results in literature, as shown in Table 10.

The proposed HAO algorithm found the best design option among the compared algorithms reported in literature.

Table 10. Results applied in pressure vessel design problem.

Algorithm	$T_s(x_1)$	$T_h(x_2)$	$R(x_3)$	$L(x_4)$	$f(x)$
GSA [32]	1.125	0.625	55.9886598	84.4542025	8538.8359
MVO [32]	0.8125	0.4375	42.090738	176.73869	6060.8066
WOA [30]	0.812500	0.437500	42.0982699	176.638998	6059.7410
BA [14]	0.812500	0.437500	42.098445	176.636595	6059.7143
GWO [26]	0.8125	0.4345	42.089181	176.758731	6051.5639
AOA [33]	0.8303737	0.4162057	42.75127	169.3454	6048.7844
AO [19]	1.0540	0.182806	59.6219	38.8050	5949.2258
HAO	**0.810726461**	**0.400897167**	**42.16466765**	**175.8460143**	**5935.56831**

4.2. Three-bar truss design

The three-bar truss design is a two-dimensional constraint problem, it has only two non-equal constraints, yet a little difficult to find the candidate. The proposed HAO algorithm finds a better yet not the best option, as shown in Table 11, it failed to do better than the original version. However, the proposed HAO does find a better option than the original AO algorithm in our simultaneous comparison experiments.

Table 11. Best solutions for the three-bar truss design problem.

Algorithm	x_1	x_2	$f(x)$
AVOA [23]	0.788680394972034	0.408233412073586	263.895843396802
GOA [25]	0.788897555578973	0.407619570115153	263.895881496069
MBA [34]	0.7885650	0.4085597	263.8958522
SSA [35]	0.788665414	0.408275784444547	263.895843
PSO-DE [36]	0.7886751	0.4082482	263.8958433
DEDS [36]	0.78867513	0.40824828	263.8958434
MFO [28]	0.788244771	0.409466905784741	263.8959797
MVO [32]	0.78860276	0.408453070000000	263.8958499
CS [37]	0.78867	0.40902	263.9716
Tsai [38]	**0.788**	**0.408**	**263.68**
AOA [33]	0.79369	0.39426	263.9154
AO [19]	0.7926	0.3966	263.8684
HAO	0.788609979	0.408362638	263.8916603

4.3. Welded beam design

The welded beam design is a four-dimensional constraint problem. It has seven non-equal constraints. Results were shown in Table 12 and the proposed HAO also find a better result, yet not the best one. Same situation met in our experiment that the proposed HAO could find a better option than the original HAO, however, still worse than the reported result.

Table 12. Best solutions obtained for the welded beam design problem.

Algorithm	h	L	t	b	$f(x)$
GSA [32]	0.182129	3.856979	10.000	0.202376	1.87995
WOA [30]	0.205396	3.484293	9.037426	0.206276	1.730499
MVO [32]	0.205463	3.473193	9.044502	0.205695	1.72645
SSA [35]	0.2057	3.4714	9.0366	0.2057	1.72491
MPA [39]	0.205728	3.470509	9.036624	0.205730	1.724853
AVOA [23]	0.205730	3.470474	9.036621	0.205730	1.724852
AAO [40]	0.2057	3.4705	9.0366	0.2057	1.724
AOA [33]	0.194475	2.57092	10.000	0.201827	1.7164
SMA [41]	0.2054	3.2589	9.0384	0.2058	1.69604
AO [19]	**0.1631**	**3.3652**	**9.0202**	**0.2067**	**1.6566**
HAO	0.19952608	3.384869727	9.064048595	0.206681757	1.715727482

5. Discussion and conclusions

This paper reports a new improvement for the Aquila optimization (AO) algorithm, which was just proposed in literature with better performance. Considering the flat lines in latter convergence curves in the original paper, the original AO should have some defects in exploitation procedure. Inspired by the better performance of heterogeneous improvements, and the inspiration of multiple updating principle, the heterogeneous AO algorithm called HAO is proposed in this paper. The proposed HAO algorithm would not introduce other equations, and re-construct the four strategies with promising better performance.

Simulation experiments were carried out on either unimodal or multimodal benchmark functions. Results confirmed the better performance including most of the Wilcoxon rank sum test results.

Three real-world engineering problems were also included to test the capability of the proposed HAO algorithm. Only one result, specifically the pressure vessel design problem, succeeded in comparison with other reported results in literature, including the original AO algorithm. However, the rest two failed to be the best one, even worse than the reported results from the original version. But in our experiment, the proposed HAO could obtain better options than the original AO algorithm. We can find that the randomness could affect the results a lot and better results might be found with occasions.

We could find the proposed heterogeneous AO algorithm would outperform at most times, it has intensification capability with unimodal benchmark functions, diversification capability with multimodal benchmark functions, it would be faster in convergence rate and approach the global optima much closer. The scalability capability is also good and the rank sum test convinced such simulations.

Individuals in swarms with heterogeneous improvements should be accompanied with larger population size accordingly. In the future, the proposed HAO algorithm could be applied to solve other problems such as reducing dimensionality, figure segmentation for real applications.

Acknowledgements

The authors would like to thank the supports of the following projects: 1) The scientific research team project of Jingchu University of technology with grant number TD202001. 2) the key research and development project of Jingmen with grant numbers 2019YFZD009. 3) the Outstanding Youth Science and Technology Innovation Team Project of Colleges and Universities in Hubei Province with grants T201923.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.
2. M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, *IEEE Trans. Syst. Man Cybern. B Cybern.*, **26** (1996), 29–41. https://doi.org/10.1109/3477.484436
3. R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science,* (1995), 39–43. https://doi.org/10.1109/MHS.1995.494215
4. M. Clerc, J. Kennedy, The particle swarm - explosion, stability, and convergence in a multidimensional complex space, *IEEE Trans. Evolut. Comput.,* **6** (2002), 58–73. https://doi.org/10.1109/4235.985692

5. R. C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, **1** (2000), 16–19. https://doi.org/10.1109/CEC.2000.870279
6. M. E. H. Pedersen, A. J. Chipperfield, Simplifying Particle Swarm Optimization, *Appl. Soft Comput.*, **10** (2010), 618–628. https://doi.org/10.1016/j.asoc.2009.08.029
7. G. I. Evers, M. B. Ghalia, Regrouping particle swarm optimization: A new global optimization algorithm with improved performance consistency across benchmarks, in *2009 IEEE International Conference on Systems, Man and Cybernetics*, (2009), 3901–3908. https://doi.org/10.1109/ICSMC.2009.5346625
8. F. v. d. Bergh, A. P. Engelbrecht, A new locally convergent particle swarm optimiser, in *IEEE International Conference on Systems, Man and Cybernetics*, **3** (2002). https://doi.org/10.1109/ICSMC.2002.1176018.
9. T. Xiang, X. Liao, K. W. Wong, An improved particle swarm optimization algorithm combined with piecewise linear chaotic map, *Appl. Math. Comput.*, **190** (2007), 1637–1645. https://doi.org/10.1016/j.amc.2007.02.103
10. H. Haklı, H. Uğuz, A novel particle swarm optimization algorithm with Levy flight, *Appl. Soft Comput.*, **23** (2014), 333–345. http://dx.doi.org/10.1016/j.asoc.2014.06.034
11. H. Garg, A hybrid PSO-GA algorithm for constrained optimization problems, *Appl. Math. Comput.,* **274** (2016), 292–305. https://doi.org/10.1016/j.amc.2015.11.001
12. N. Holden, A. A. Freitas, A hybrid PSO/ACO algorithm for discovering classification rules in data mining, *J.Artif. Evolut. Appl.*, (2008), 316145. https://doi.org/10.1155/2008/316145
13. A. P. Engelbrecht, Heterogeneous particle swarm optimization, in *Swarm Intelligence* (eds. M. Dorigo *et al.*), Springer Berlin Heidelberg, (2010), 191–202.
14. X. S. Yang, A New Metaheuristic Bat-Inspired Algorithm, in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, Springer Berlin Heidelberg, (2010), 65–74.
15. Z. M. Gao, J. Zhao, X. R. Li, Y. R. Hu, An improved sine cosine algorithm with multiple updating ways for individuals, *J. Phys. Conf. Ser.*, **1678** (2020), 012079. https://doi.org/10.1088/17426596/1678/1/012079
16. J. Zhao, Z. M. Gao, An improved grey wolf optimization algorithm with multiple tunnels for updating, *J. Phys. Conf. Ser.*, **1678** (2020), 012096. https://doi.org/10.1088/17426596/1678/1/012096
17. S. Mirjalili, SCA: A Sine Cosine algorithm for solving optimization problems, *Knowl. Based Syst.*, **96** (2016), 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
18. L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, A. H. Gandomi, The arithmetic optimization algorithm, *Comput. Methods Appl. Mech. Eng.*, **376** (2021), 113609. https://doi.org/10.1016/j.cma.2020.113609
19. L. Abualigaha, D. Yousrib, M. A. Elazizc, A. A. Eweesd, M. A. A. Al-qanesse, A. H. Gandomif, Aquila optimizer: a novel meta-heuristic optimization algorithm, *Comput. Indust. Eng.*, **157** (2021), 107250. https://doi.org/10.1016/j.cie.2021.107250
20. A. Fatani, A. Dahou, M. A. A. Al-qaness, S. Lu, M. Abd Elaziz, Advanced feature extraction and selection approach using deep learning and Aquila optimizer for IoT intrusion detection system, *Sensors,* **22** (2022), 140. https://doi.org/10.3390/s22010140

21. A. M. AlRassas, M. A. A. Al-qaness, A. A. Ewees, S. Ren, M. Abd Elaziz, Optimized ANFIS model using Aquila optimizer for oil production forecasting, *Processes,* **9** (2021). https://doi.org/10.3390/pr9071194
22. S. Mirjalili, The ant lion optimizer, *Adv. Eng. Soft.,* **83** (2015), 80–98. http://dx.doi.org/10.1016/j.advengsoft.2015.01.010
23. B. Abdollahzadeh, F. S. Gharehchopogh, S. Mirjalili, African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems, *Comput. Indust. Eng.*, **158** (2021), 107408. https://doi.org/10.1016/j.cie.2021.107408
24. A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, *Knowl. Based Syst.*, (2019), 105190. https://doi.org/10.1016/j.knosys.2019.105190
25. S. Saremi, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: theory and application, *Adv. Eng. Soft.*, **105** (2017), 30–47. https://doi.org/10.1016/j.advengsoft.2017.01.004
26. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer," *Adv. Eng. Soft.*, **69** (2014), 46–61. http://dx.doi.org/10.1016/j.advengsoft.2013.12.007
27. A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, *Future Gener. Comp. Syst.*, 2019. https://doi.org/10.1016/j.future.2019.02.028
28. S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, *Knowl. based syst.*, 89 (2015), 228. https://doi.org/10.1016/j.knosys.2015.07.006
29. K. Zervoudakis, S. Tsafarakis, A mayfly optimization algorithm, *Comput. Indust. Eng.*, **145** (2020), 106559. https://doi.org/10.1016/j.cie.2020.106559
30. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Softw.*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
31. Z. M. Gao, J. Zhao, *Benchmark functions with Python*, Golden Light Academic Publishing, (2020), 3–5.
32. S. Mirjalili, S. M. Mirjalili, A. Hatamlou, Multi-Verse Optimizer: a nature-inspired algorithm for global optimization, *Neural Comput. Appl.,* **27** (2016), 495–513. https://doi.org/10.1007/s00521-015-1870-7
33. A. D. Laith Abualigah, S. Mirjalilid, M. Abd Elazizf, A. H. Gandomih, The Arithmetic Optimization Algorithm, *Comput. Methods Appl. Mech. Eng.*, **376** (2021), 113609. https://doi.org/10.1016/j.cma.2020.113609
34. A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, *Appl. Soft Comput.*, **13** (2013), 2592–2612. https://doi.org/10.1016/j.asoc.2012.11.026
35. S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, *Adv. Eng. Softw.*, **114** (2017), 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
36. M. Zhang, W. Luo, X. Wang, Differential evolution with dynamic stochastic selection for constrained optimization, *Inform. Sci.*, **178** (2008), 3043–3074. https://doi.org/10.1016/j.ins.2008.02.014
37. V. Bhargava, S. E. K. Fateen, A. Bonilla-Petriciolet, Cuckoo Search: A new nature-inspired optimization method for phase equilibrium calculations, *Fluid Phase Equilibr.*, **337** (2013), 191–200. http://dx.doi.org/10.1016/j.fluid.2012.09.018

38. J. F. Tsai, Global optimization of nonlinear fractional programming problems in engineering design, *Eng. Optimiz.*, **37** (2005), 399–409. https://doi.org/10.1080/03052150500066737
39. A. Faramarzi, M. Heidarinejad, S. Mirjalili, A. H.Gandomic, Marine predators algorithm: A nature-inspired metaheuristic, *Expert Syst. Appl.*, **152** (2020), 113377. https://doi.org/10.1016/j.eswa.2020.113377
40. J. M. Czerniak, H. Zarzycki, D. Ewald, AAO as a new strategy in modeling and simulation of constructional problems optimization, *Simul. Model. Pract. Theory*, **76** (2017), 22–33. https://doi.org/10.1016/j.simpat.2017.04.001
41. S. Li, H. Chen, M. Wang, A. A. Heidari, S. Mirjalili, Slime mould algorithm: A new method for stochastic optimization, *Future Gener. Comput. Syst.*, **111** (2020), 300–323. https://doi.org/10.1016/j.future.2020.03.055