Adaptive Multi-scale Quantum Harmonic Oscillator Algorithm Based on Evolutionary Strategy

Xinggui Ye
University of Chinese Academy of Sciences
Beijing, China
Chengdu Institution of Computer Application,
Chinese Academy of Sciences
Chengdu, China
yexingui@outlook.com

Peng Wang
School of Computer Science and Technology,
Southwest Minzu University
Chengdu, China
qhoalab@163.com

***Abstract*—This paper proposes a novel adaptive multi-scale quantum harmonic oscillator algorithm based on evolutionary strategies (AMQHOA-ES) for global numerical optimization. Since the original Multi-scale Quantum Harmonic Oscillator Algorithm (MQHOA) utilizes a fixed contraction factor to narrow the search scale, the searching step decreases too fast at the later stage of the evolution and is more likely to suffer premature convergence and stagnation. To improve the convergence performance, an adaptive attenuation mechanism of scaling is proposed to dynamically adjust the exploration and exploitation properties. Evolutionary strategies such as selection, crossover and DE/rand/1 mutation are implemented in the proposed algorithm to enhance the exploration and exploitation abilities. Experimental results evaluated on several unimodal and multimodal benchmark functions indicate the significant improvement of the proposed algorithm to the original MQHOA. Meanwhile, the experimental results compared with several state-of-the-art optimizers show the superiority or competitiveness of the proposed algorithm.**

***Index Terms*—Evolutionary strategy, differential evolution, adaptive mechanism, multi-scale quantum harmonic oscillator algorithm, population-based optimization**

I. Introduction

Evolutionary algorithms (EAs) have attracted extensive attentions for many decades. Generally, EAs are inspired by the biological evolution, utilizing some mechanisms from natural world: selection, mutation, recombination (crossover) and reproduction. EAs have been proved to be simple, but powerful and robust in solving most of the global numerical problems. EAs are stochastic algorithms, though they could not guarantee a single accurate output with the same inputs, they often perform well on approximating solutions to most type of problems. Typically, EAs are including Genetic Algorithm (GAs) [1], [2], Evolutionary Programming (EP) [3], Evolutionary Strategy (ES) [4], Differential Evolution (DE) [5], Particle Swarm Optimization (PSO) [6], Artificial Bee Colony (ABC) [7], Grey Wolf Optimizer (GWO) [8] and etc.

Unfortunately, there is no single algorithm that is able to perform well on all of the problems. Instead of improving the algorithm itself, more researchers are making their efforts to integrate different algorithms to enhance the adaptability and performance. In general, researchers are prone to integrate complementary characteristics from different algorithms into a hybrid version [9]–[11]. Another feasible way is to adaptively select the appropriate parameters or algorithms for the problems [12]–[14].

Multi-scale Quantum Harmonic Oscillator Algorithm (MQHOA) [15] is a population-based metaheuristic algorithm proposed recently. It utilizes overlapped quantum wavefunctions to track the global optimum of an optimization problem and has been proved to be effective and efficient to deal with unimodal and multimodal problems [15]–[17]. However, as MQHOA adopts a fixed contraction coefficient to narrow the search scale, it slows down the convergence speed at the later stage of iterations and easily falls into local optima. Inspired by the integration and adaptation mechanisms broadly used in EAs [10]–[14], in this paper, we propose an adaptive mechanism to improve the convergence performance of MQHOA. Meanwhile, to further enhance the exploration and exploitation abilities, "$DE/rand/1$" mutation strategy and selection are integrated into the original MQHOA. The new version of MQHOA is named Adaptive Multi-scale Quantum Harmonic Oscillator Algorithm based on Evolutionary Strategy (AMQHOA-ES). The proposed algorithm is executed on several well-defined benchmark functions. Experimental results reveal that AMQHOA-ES significantly outperforms the original MQHOA. The comparative results between the proposed algorithm and some state-of-the-art optimizers indicate the superiority or competitiveness of the proposed algorithm.

The remainder of this paper is organized as follows. Section II briefly introduces the related works about the research work. Followed by the demonstration of the proposed AMQHOA-ES in Section III. Section IV elaborates the experiments and compares the computational results with several popular optimizers. Finally, the conclusion and our future work are outlined in Section V.

II. Related Work

To the best of our knowledge, there is no single algorithm performs well on all function evaluations. In order to improve the performance of an algorithm on its effectiveness and efficiency, researchers have proposed a large number

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

of improvement strategies. One of the popular strategies is integration of different algorithms to enhance the adaptability and convergence performance. Another notable effort is to adaptively change the parameters in the course of evolution.

Differential evolution (DE) [5] is one of the most popular and broadly used evolutionary algorithms. By utilizing the differential mutation mechanism and crossover operations, DE is able to solve many unimodal and multimodal problems efficiently. The "$DE/rand/1$" mutation which is broadly utilized in literatures can be described as

$$Y = X_a + F(X_b - X_c) \tag{1}$$

where Y = $y_1, y_2, ..., y_n$ is the new candidate solution, n is the dimension size, X_a, X_b and X_c are three candidate solutions randomly selected from the population. F is the amplification factor of the differential vector. The differential mutation and crossover mechanisms significantly help to diversify the population and help the algorithm converge fast.

However, as DE utilizes the fixed amplification parameter F and the crossover control factor CR, it is not flexible to avoid premature convergence and getting trapped into local optima in the later iterations. In order to prevent DE from slowing down too much at the later evolution stage and getting stuck at local optima, several efforts have been made in literatures.

In order to flexibly and properly set the control parameters of DE, Janez Brest [12] proposed a self-adapting control parameter based on DE (jDE). The jDE utilizes adaptive parameters F and CR in the course of function evaluations.

$$F_{i,g+1} = \begin{cases} F_l + rand_1 * F_u, & if \quad rand_2 < \tau_1 \\ F_{i,g}, & otherwise \end{cases} \tag{2}$$

$$CR_{i,g+1} = \begin{cases} rand_3, & if \quad rand_4 < \tau_2 \\ CR_{i,g}, & otherwise \end{cases} \tag{3}$$

where $rand_i$ i=[1,2,3,4] are uniformed numbers from [0,1], τ_1 and τ_2 are the probabilities to adjust factor F and CR, respectively. F_l and F_u are the lower and upper values of F. New mechanisms of F and CR make the algorithm take an adaptive F from [0.1,1.0] and CR from [0,1]. The experimental results indicate the competitiveness of jDE [12].

Another adaptive DE optimizer which is named JADE implements the "DE/current-to-$p$best" mutation strategy was proposed in [13]. The algorithm utilizes an optional archive and adaptive mutation factor F and crossover rate CR. The CR is adaptively generated by

$$CR_i = randn_i(\mu CR, 0.1) \tag{4}$$

where CR_i is truncated to [0,1], μCR is updated by

$$\mu CR = (1 - c)\mu CR + c mean_A(S_C R) \tag{5}$$

where c is a positive constant within (0,1), and $mean_A$ is the arithmetic mean. And the mutation factor F is set according to a Cauchy distribution which helps to diversify the mutation factors and avoid premature convergence. The two adaptive parameters help the JADE perform more independently with optimization problems.

In [10], an adaptive variable difference algorithm based on particle swarm optimization algorithm (DEPSO) was proposed. The algorithm adopts an improved "$DE/rand/1$" mutation strategy to achieve stronger global exploration ability. Meanwhile, it utilizes PSO mutation strategy to obtain higher convergence ability. As a result, the population diversity can be maintained well at early stage of the evolution, and the faster convergence speed can be obtained at later stage of the evolution. DE mutation and PSO mutation mechanisms are selected according to

$$SP_g = \frac{1}{1 + e^{1-(g_{max}/g+1)^{\tau}}} \tag{6}$$

where τ is a positive constant, g is the generation number. If rand $< SP_g$, DE/rand/1 is selected; otherwise, PSO mutation is chosen. In addition, an elite archive is used in DEPSO to generate offsprings with $X^P_{r2,g}$-$X^Q_{r3,g}$, where P and Q are the elite population and non-elite population respectively. These operations help DEPSO to be a robust optimizer.

Inspired by the adaptivity and hybridization mechanisms in literatures [10]–[13], an adaptive scaling mechanism and some evolutionary strategies are applied in MQHOA to improve the convergence performance. The principles of MQHOA are detailed in [15]–[18], and this paper will not repeat them. The main contribution of this paper can be summarized as follows:

First, a novel adaptive multi-scale quantum harmonic oscillator algorithm with evolutionary strategy (AMQHOA-ES) is proposed. Second, an adaptive mechanism is proposed to dynamically adjust the search scale in every generation of iteration which helps enhance the exploration ability. Third, selection and DE/rand/1 mutation mechanisms are adopted to enhance the exploitation ability of the proposed algorithm. Fourth, a feedback of searching space expansion mechanism is proposed to strengthen the exploitation ability. Fifth, the performance of the proposed AMQHOA-ES is validated to be mostly superior to the original MQHOA and competitive to several popular optimizers.

III. AMQHOA-ES

As MQHOA utilizes a fixed contraction factor λ to reduce the search radius [15]–[17], it is effective and efficient in the evaluation of simple benchmark problems. However, when solving complex problems, especially for multimodal functions, MQHOA will easily fall into local optima. In order to maintain the effectiveness and efficiency of the algorithm, mechanisms for keeping the diversity of the particles and jumping out of the local optima should be well considered.

A. Evolutionary strategy

Inspired by the outstanding convergence properties of DE, the differential mutation mechanism is introduced and integrated to the proposed algorithm. As the "$DE/rand/1$" is mostly used, it is applied in the QHO process in MQHOA.

Within one iteration cycle, if the particles do not improve the fitness value, there is a chance (rand< D_r) to employ the DE/rand/1 mutation to generate new candidate solutions. D_r is defined as

$$D_r = 1 - iterNO/maxFE \tag{7}$$

where $iterNO$ is the generation number, $maxFE$ is the maximal runs. A new candidate solution is generated by

$$X_{i+1} = X_{r1} + F(X_{r2} - X_{r3}) \tag{8}$$

where $r1$, $r2$ and $r3$ are randomly selected numbers within [1,N_p], N_p is the number of the population. F is randomly generated by $rand(0,1)^D$, where D is the dimension.

In addition, in case of premature convergence, a small probability (rand $<$ C_r) of selecting the reset mechanism is applied to generate a new candidate solution.

$$X_{i+1} = x_l + (x_u - x_l)rand(D, N_p) \tag{9}$$

where C_r is a small positive constant, x_u and x_l are the upper and lower bound of the search space.

B. Adaptive contraction factor

In general, at the beginning of function evaluation, the algorithm needs to search for more landscapes as much as possible. Therefore, at this stage the search radius should be as large as possible. As the iteration proceeds, the particles gradually assemble near the local optimal landscape. At this moment, if the search radius is too large, it will be easy to ignore the potential global optimum and increase the search time. In this case, the search radius should be narrowed to increase the possibility of finding the global optimum.

Moreover, when the standard deviation of the population in every two successive iterations is large, it implies the search radius should be long enough to help the particles to have more chances to be close to the global optimum. Contrarily, if the standard deviation of the two successive generation is small, it means that the search step should be shorten to obtain more opportunities to be close to the global optimum. Based on the aforementioned analyses, the contractive factor is proposed as.

$$\lambda = \sqrt{\sigma_{old}/\sigma_{new}} + rand(0,1)/10 \tag{10}$$

where σ_{new} and σ_{old} are the standard deviation of current and last generation respectively, $rand(0,1)$ is a random number within (0,1) which is helpful to adjust the λ irregularly larger than 1, and hence increases the diversity of the population.

In addition, in order to prevent long period of premature convergence, a feedback mechanism is adopted with a selection mechanism. If the global fitness value does not be changed within the successive iterations, the feedback mechanism will be triggered. The current search scale will be redefined as

$$\chi_{i+1} = k_1 \cdot \chi_i \cdot \lambda \tag{11}$$

where k_1 is a constant ($k_1 > 1$), χ_i is the current search scale. It indicates that if the iteration stagnates for a long period, the current search scale will be enlarged and help the particles to jump out from local optima.

C. Framework of the proposed algorithm

In the proposed algorithm, the "$DE/rand/1$" mutation and selection mechanisms are applied in the QHO process in MQHOA [15]. The feedback mechanism based on adaptive search scale is adopted in the M process. The pseudocode of the proposed algorithm is demonstrated in Algorithm 1.

Algorithm 1: AMQHOA-ES pseudocode

```
Input: k, X_i, ∈ [d_l, d_u]^D, (i = 1, 2, ..., k), ε, λ, c
Output: the global optimum f_best, the optimal position X_best
initialize k, ε, d_u, d_u, λ ;
randomly generate x_di, d = 1, 2, ..., D, i = 1, 2, ..., k in
 [d_l, d_u];
evaluate F_i = f(X_i), F^opt=min(F_i), X^opt = X_i,σ_n;
while (iterNO < maxFE) || (F^opt < ε) do
    while (σ_n < χ) do
        generate x_dj (j = 1, ..., k) from N(x_{d,i}, χ^2);
        if f(x_dj) < F_max then
            F_max = f(x_dj), X_max = x_dj
        else
            else if rand < D_r then
                generate a new candidate solution by (8)
            else if rand < C_r then  z_ij = x_dj ;
            z_ij = x_di ;
        end
        generate mean position X_m of the population ;
        replace the worst individual with X_m ;
        update standard deviation σ_n ;
        if σ_new < σ_old then
            update λ by (10) ;
            σ_old=σ_new ;
        end
        if σ_new < χ then
            sNO=sNO+1;
            else if sNO=20 then
                χ = k_1 * χ * λ
            else if sNO=30 then  χ = k_2 * χ * λ, sNO=0 ;
        end
    end
    χ_{i+1} = χ_i/λ
end
Output F^opt, X^opt ;
```

In Algorithm 1, k is the population size , ε is the calculation precision, λ denotes the scale reduction factor, f_{best} is the best fitness in current iteration cycle, X_{best} is the best solution, X^{opt} and F^{opt} are the global optimal solution and fitness value respectively. $iterNO$ is the number of function evaluation, σ_n is the current standard deviation of the population, χ is the current search length. $N(x_{di}, \chi^2)$ is a normal distribution, F_{max} is the largest fitness value in the current iteration cycle. D_r is defined in (7), C_r is a positive constant, F is adjusted by (8), k_1 and k_2 are two positive amplification factors.

IV. Experimental results and discussion

In this section, the effectiveness and efficiency of the proposed algorithm are evaluated. Several well-defined benchmark functions are utilized to compare the performances of AMQHOA-ES with the original MQHOA and MQHOA with truncated mean stabilization strategy (TSMQHOA) [16].

Meanwhile, several popular methods such as Stud Genetic Algorithm (StudGA) [19], Particle Swarm Optimization version 2011 (SPSO2011) [20], Comprehensive Learning Particle Swarm Optimization (CLPSO) [21], jDE [12], DEPSO [10], Imperialist Competitive Algorithm (ICA) [22], [23], are applied to compete with AMQHOA-ES.

A. Benchmark functions and parameter settings

In order to validate the characteristics of the proposed algorithm, several unimodal and multimodal benchmarks from IEEE Congress on Evolutionary Computation [24], [25] are executed under the same conditions. Function f_1-f_7 are unimodal benchmarks and f_8-f_{12} are multimodal functions. The test benchmark functions are listed in Table I.

Parameters used in every algorithm are set as same as possible. The number of particles (population size) is defined N_p=20. The maximum function evaluation (run times) is set according to the rule used in CEC2017 [25] and is defined as $maxFE$=10000 $* \, dimension$. The search space $[d_l, \, d_u]^D$ for each benchmark function is set according to Table I. The calculation accuracy (error) is set $\varepsilon = 0.000001$. Special parameters used in the compared algorithms are set accordingly.

For StudGA [19], the crossover probability is defined 1.0, the number of points in each crossover is 1, the mutation rate is set 0.01, maintaining 2 best individuals from one generation to the next. For SPSO2011 [20], the inertia weight $\omega = 1/2log(2)$, the learning factor $c1 = c2 = 0.5 + log(2)$. The parameters used in CLPSO [21] are the inertia weight linearly declines from 0.9 to 0.2, the accelerate constant $c1 = c2 = 1.49445$. For jDE [12], parameters in (2) and (3) are F_l=0.1, F_u=0.9, τ_1=τ_2=0.1. For DEPSO [10], the maximum of DE population and PSO population is 5. For ICA [23], the number of Empires is 10, the selection pressure α=1, the assimilation coefficient β=1.5, the revolution probability is defined 0.5, the revolution rate μ= 0.1, the colonies mean cost coefficient ζ=0.2. For TSMQHOA [16], the truncated probability is 10%. The contraction coefficient in MQHOA and TSMQHOA is set $\lambda = 2.0$. For AMQHOA-ES, C_r, k_1 and k_2 are experimentally defined 0.05, 1.2 and 1.6, respectively.

Meanwhile, the stopping criteria for all of the algorithms are uniformly defined as: the ε is less than 1e-6 or the $maxFE$ is larger than 10000*D. All of the algorithms are coded in Matlab R2016a and executed on the same personal computer with an Intel core(TM) i5-4200U 64 bit, 2.3 GHz and Windows 7 operation system.

B. Experimental results

1) Fitness computation: To estimate the effectiveness and efficiency of the proposed algorithm, several items are considered such as $Best$, the best fitness; $Mean$, the average of the fitness values obtained by the population; Std, the standard deviation of the fitness values; $Time$, the CPU run time (time from start to finish the function evaluation) and $IterNO$, the function evaluation number (FE). Without losing of generality, the experiments were executed on 100-dimensional function evaluations. Meanwhile, to reveal the differences among the AMQHOA-ES and some congeneric algorithms, the experimental results are compared with MQHOA, TSMQHOA [16] and several state-of-the-art optimizers. Parameters are set according to Section IV-A. The experimental results are the average of 50 independent trials and listed in Table II.

As seen in Table II, in the evaluation of function f_1-f_5, the experimental results obtained by AMQHOA-ES are much smaller than TSMQHOA and MQHOA, and outperform several state-of-the-art optimizers. In the evaluation of function f_1, every algorithm is able to find the global optimum except for CLPSO and DEPSO, but AMQHOA-ES requires the least evaluation times (1.488E+04). The experimental results of function f_2 indicate that AMQHOA-ES obtains better records than MQHOA and TSMQHOA on the best fitness, the mean fitness, the standard deviation, the total evaluation number and CPU run time. In the evaluation of function f_3 and f_4, AMQHOA-ES obtains the best results of the mean fitness, the iteration times and CPU run time. In the evaluation of function f_5, AMQHOA-ES is the only technique which is able to find the global optimum. Although none of the algorithms is capable of locating to the global optimal landscape in the evaluation of function f_6, AMQHOA-ES gains the best results on the best fitness value and the mean fitness value. The evaluation results on f_7 indicate that jDE is the only technique which is able to find the global optimum, and AMQHOA-ES outperforms the rest of the algorithms.

The results in the evaluation of f_8 indicate that ICA and AMQHOA-ES are the only two algorithms which are capable of finding the global minimum. AMQHOA-ES obtains smaller fitness value and mean fitness value, and ICA requires smaller iteration number. In the evaluation of function f_9, AMQHOA-ES is the only algorithm which is able to find the global optimum before the stopping criteria are satisfied.

The results of function f_{10} show that jDE, ICA, TSMQHOA and AMQHOA-ES are the four techniques which are able to locate to the global optimal landscape. Meanwhile, AMQHOA-ES obtains the smallest mean fitness and iteration number, and TSMQHOA spends the least CPU run time. None of the algorithms is able to find the global optimum within the stopping criteria in the evaluation of function f_{11}. Comparatively, StudGA obtains better results with the best fitness of 1.982E-02, but other algorithms gain their best fitness values larger than that. In the evaluation of function f_{12}, most of the algorithms are able to find the global optimum except for CLPSO and DEPSO. The iteration numbers gained by AMQHOA-ES and TSMQHOA are much smaller than that of MQHOA and other algorithms.

2) Fitness-iteration comparison: In Fig.1, the X-axis represents the number of evaluation, the Y-axis indicates the fitness value (semilogy). As seen in Fig.1, overall, the fitness-iteration curves of AMQHOA-ES are the lowest in the evaluation of the benchmark functions except for Fig.1(h) and Fig.1(k), which indicates the superiority of AMQHOA-ES in the evaluation of function f_1-f_7, f_9-f_{10} and f_{12}. However, the curves of MQHOA are mostly on the top of other algorithms, which implies its hardness of converging. Meanwhile, in Fig.1

Table I Benchmark functions.

Function Name	Benchmark Function	D	Range	Optimum
Sphere	$f_1 = \sum_{i=1}^{n} x_i^2$	n	[-5.12,5.12]	$f(0,...,0) = 0$
Sum Squares	$f_2 = \sum_{i=0}^{n-1} ix_i^2$	n	[-10,10]	$f(0,...,0) = 0$
Rotated Hyper-Ellipsoid	$f_3 = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_i)^2$	n	[-65.54,65.54]	$f(0,...,0) = 0$
Ellipsoidal	$f_4 = \sum_{i=1}^{n} (x_i - i)^2$	n	[-100,100]	$f(1,2,...n) = 0$
Zakharov	$f_5 = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5ix_i)^2 + (\sum_{i=1}^{n} 0.5ix_i)^4$	n	[-5,10]	$f(0,...,0) = 0$
Rosenbrock	$f_6 = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$	n	[-5,10]	$f(1,...,1) = 0$
Bent Cigar	$f_7 = x_i^2 + 10^6 \sum_{i=2}^{n} x_i^2$	n	[-10,10]	$f(0,0) = 0$
Ackley	$f_8 = -20exp(-0.2\sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n} \sum_{i=1}^{n} cos(2\pi x_i)) + 20 + e$	n	$[-32.77, 32.77]$	$f(0,...,0) = 0$
Griewank	$f_9 = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	n	[-100,100]	$f(0,...,0) = 0$
Levy	$f_{10} = sin^2(\pi\omega_1) + \sum_{i=1}^{n-1} (\omega_i - 1)^2[1 + 10sin^2(\pi\omega_i + 1)] + (\omega_n - 1)^2[1 + sin^2(2\pi\omega_n)]$,where $\omega_i = 1 + \frac{x_i - 1}{4}$,for all $i = 1, ..., n$	n	[-10,10]	$f(1,...,1) = 0$
Rastrigin	$f_{11} = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]$	n	[-5.12,5.12]	$f(0,...,0) = 0$
Modified Schwefel	$f_{12} = 418.9829 \times D - \sum_{i=1}^{n} g(z_i) \quad z_i = x_i + 420.9687462275036, where$	n	[-5.12,5.12]	$f(0,...,0)$
	$g(z_i) = \begin{cases} z_i sin(\lvert z_i\rvert^{1/2}) & if \quad \lvert z_i\rvert \leq 500 \\ (500 - mod(z_i, 500))sin(\sqrt(\lvert 500 - mod(z_i, 500)\rvert) - \frac{(z_i - 500)^2}{10000n} & if \quad z_i > 500 \\ (mod(\lvert z_i\rvert, 500) - 500)sin\sqrt(\lvert mod(\lvert z_i\rvert, 500) - 500\rvert) - \frac{(z_i + 500)^2}{10000n} & if \quad z_i < -500 \end{cases}$			=0.000012727*D

(a), (c), (d), (i), (j) and (l), the fitness-iteration lines of AMQHOA-ES are much lower than other techniques, followed by TSMQHOA, which indicates the better convergence performance of AMQHOA-ES compared with that of TSMQHOA and other algorithms in the evaluation of function f_1, f_3, f_4, f_9, f_{10} and f_{12}.

The curves of AMQHOA-ES and TSMQHOA in Fig.1(b) are much lower than other algorithms within the first 3.0×10^4 iterations. However, after that, both AMQHOA-ES and ICA outperform TSMQHOA and other methods. In the evaluation of function f_5, AMQHOA-ES is the only method which is able to converge to the global optimum within the stopping criteria. The curves of ICA and SPSO2011 are much higher than AMQHOA-ES but lower than MQHOA, jDE, CLPSO, TSMQHOA and DEPSO in Fig.1(e). As seen in Fig.1(f), none of the algorithms is able to find the global minimum before the stopping criteria are met. Comparatively, the fitness-iteration lines of AMQHOA-ES and TSMQHOA outperform other methods. The fitness-iteration line of AMQHOA-ES in Fig.1(g) remains the lowest from the beginning to the end. Although the curve of jDE is higher than TSMQHOA, jDE and SPSO2011, it gradually goes down to be lower than TSMQHOA, jDE and SPSO2011 after 2.2×10^5 iterations.

The iteration curves of AMQHOA-ES in the evaluation of function f_8 and f_{11} are not the lowest but still competitive. In Fig.1(h), most algorithms are not able to find the global optimum before meeting the stopping criteria, except for ICA and AMQHOA-ES. The curve of ICA is declining smoothly in the first 1.6×10^5 runs, but after that, it goes down rapidly. Similar situation happens to AMQHOA-ES which drops slowly in the first 2.5×10^5 iterations, but declines sharply after that. In the evaluation of function f_{11} in Fig.1(k), though AMQHOA-ES and TSMQHOA outperform other algorithms in the first 5.0×10^4 iterations, their priorities do not persist for long. The curves of StudGA, jDE and SPSO2011 drop quickly while that of AMQHOA-ES and TSMQHOA decline smoothly.

3) Wilcoxon rank-sum test: Further, we apply two-tail Wilcoxon rank-sum tests to calculate p-values of AMQHOA-ES and other algorithms (significant level α=0.05) to evaluate the distributional differences between each other. The compared values of each algorithm are the best fitness obtained from 50 independent trials. The null hypothesis is that H_0: the observations come from differently distributed populations.

The experimental results in Table III reveal that the p-values of AMQHOA-ES and other algorithms are mostly less than 0.05, which indicates rejecting the null hypotheses that the paired samples are not distributed significantly different. Conversely, the p-values of AMQHOA-ES v.s MQHOA in f_1 (6.763E-02), f_2 (1.670E-01), f_5 (5.771E-01), f_9 (5.808E-01) and f_{10} (5.390E-01) are larger than 0.05 which indicate accepting the null hypotheses that the paired samples are distributed significantly different.

C. Brief discussion

Based on the experimental results in Table II, Fig.1 and Table III, we can draw the conclusion that AMQHOA-ES outperforms MQHOA and TSMQHOA in the trialed benchmark function evaluations. The results in Table II show that AMQHOA-ES is effective and efficient in most cases. The fitness-iteration curves in Fig.1 indicate that AMQHOA-ES is superior or at least competitive in the evaluation of most applied benchmark functions. Further, the results in Table III validate the significant improvements of the proposed algorithm compared with MQHOA and TSMQHOA.

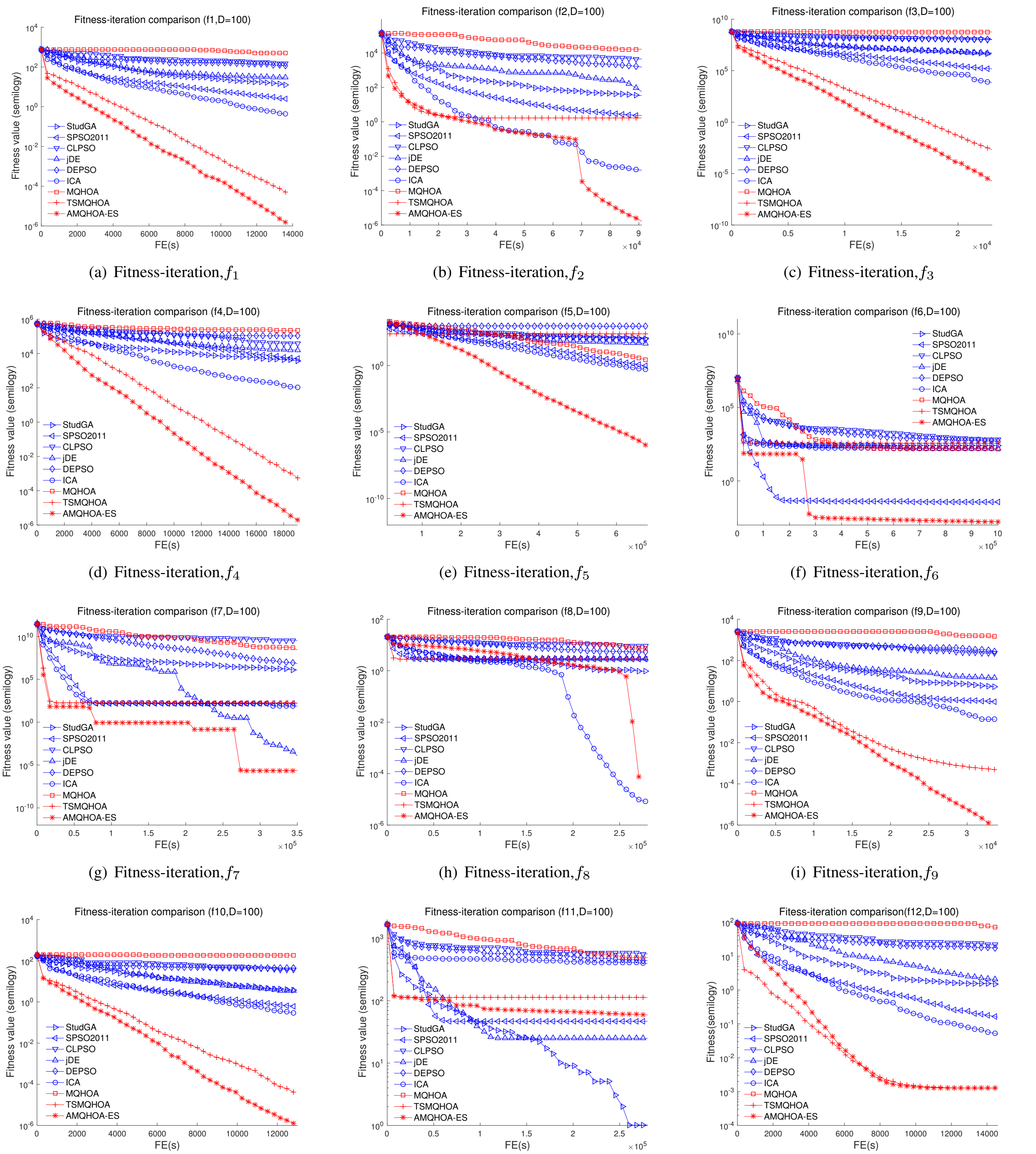
V. Conclusion

This paper proposes an adaptive multi-scale quantum harmonic oscillator algorithm based on evolutionary strategies. The evolutionary strategies such as selection and $DE/rand/1$ help to diversify the population and enhance the exploration ability of the proposed algorithm. The adaptive scaling mechanism improves the exploitation ability of AMQHOA-ES. The experimental results are compared with the original MQHOA

Table II Detailed computational results obtained by StudGA, SPSO2011, CLPSO, jDE, DEPSO, ICA, MQHOA, TSMQHOA and AMQHOA-ES. The records are the average from 50 independent trials.

Func.	Item	StudGA	SPSO2011	CLPSO	jDE	DEPSO	ICA	MQHOA	TSMQHOA	AMQHOA-ES
f_1	Best	**0.000E+00**	9.798E-07	9.226E-01	9.348E-07	1.650E-03	9.475E-07	9.759E-07	9.653E-07	9.360E-07
	Mean	1.247E+01	1.204E-06	2.993E+01	1.079E-06	1.650E-03	4.488E-01	1.101E-06	1.091E-06	**9.724E-07**
	Std	1.183E+01	1.091E-07	3.244E+01	6.438E-08	**0.000E+00**	1.391E+00	5.103E-08	4.502E-08	2.510E-08
	IterNO	2.892E+05	7.298E+04	1.000E+06	1.985E+05	1.000E+06	5.600E+04	8.958E+04	1.668E+04	**1.488E+04**
	Time	2.629E+02	2.033E+01	5.670E+01	3.828E+00	2.645E+01	5.493E+00	2.927E+00	**5.747E-01**	9.855E-01
f_2	Best	**0.000E+00**	3.355E+00	7.052E+01	9.437E-07	1.534E-01	9.220E-07	1.047E-06	3.641E+00	1.000E-06
	Mean	2.394E+03	3.416E+00	3.539E+03	1.572E-06	1.534E-01	9.655E+01	1.060E-06	3.641E+00	**1.001E-06**
	Std	2.625E+03	2.596E-02	4.253E+03	4.318E-07	**2.776E-17**	3.008E+02	6.932E-09	8.657E-16	1.116E-09
	IterNO	3.936E+05	1.000E+06	1.000E+06	2.900E+05	1.000E+06	**1.025E+05**	8.596E+05	1.000E+06	3.977E+05
	Time	3.226E+03	2.038E+03	1.336E+02	**6.366E+00**	3.170E+01	1.180E+01	7.113E+01	3.867E+01	2.836E+01
f_3	Best	5.452E+02	1.236E-03	6.958E+05	9.993E-07	1.731E+01	**9.497E-07**	9.796E-07	9.632E-07	9.985E-07
	Mean	8.314E+06	1.560E-03	2.549E+07	1.042E-06	1.731E+01	3.466E+05	1.133E-06	1.152E-06	**1.024E-06**
	Std	8.547E+06	1.391E-04	2.883E+07	2.812E-08	**3.553E-15**	1.153E+06	8.813E-08	7.191E-08	1.913E-08
	IterNO	1.000E+06	1.000E+06	1.000E+06	3.769E+05	1.000E+06	1.180E+05	1.494E+05	2.932E+04	**2.336E+04**
	Time	8.080E+03	2.089E+03	2.532E+02	4.718E+01	1.341E+02	7.419E+01	3.086E+01	4.138E+00	**3.994E+00**
f_4	Best	8.326E-01	1.154E-05	5.257E-03	9.999E-07	1.148E+03	**7.983E-07**	9.775E-07	9.892E-07	9.592E-07
	Mean	8.834E+03	1.434E-05	5.101E-01	1.009E-06	1.148E+03	1.768E+02	1.092E-06	1.088E-06	**9.970E-07**
	Std	1.032E+04	1.240E-06	1.180E+00	3.215E-09	**4.548E-13**	5.722E+02	4.525E-08	4.960E-08	2.076E-08
	IterNO	1.000E+06	1.000E+06	1.000E+06	3.840E+05	1.000E+06	1.571E+05	1.160E+05	2.542E+04	**1.964E+04**
	Time	7.461E+03	6.969E+02	1.426E+02	8.101E+00	3.009E+01	1.399E+01	8.326E+00	1.389E+00	**1.306E+00**
f_5	Best	1.036E+02	2.478E+02	8.292E+04	4.698E+01	9.238E+02	2.987E-02	1.429E-04	2.077E+02	**9.989E-07**
	Mean	6.334E+09	8.737E+08	1.309E+10	4.705E+01	9.238E+02	2.957E+07	1.448E-04	2.077E+02	**1.010E-06**
	Std	1.975E+10	5.249E+09	4.226E+10	5.234E-02	4.548E-13	1.254E+08	9.321E-07	**3.363E-14**	7.521E-09
	IterNO	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	8.776E+05	1.000E+06	**6.249E+05**
	Time	1.002E+03	1.646E+03	8.409E+01	**2.003E+01**	3.010E+01	8.888E+01	3.204E+01	3.822E+01	4.450E+01
f_6	Best	3.302E+02	6.810E+01	1.123E+03	8.480E+01	6.320E+02	1.560E+02	9.871E+01	1.850E+02	**7.912E-01**
	Mean	1.709E+05	6.810E+01	6.977E+04	8.480E+01	6.320E+02	3.056E+02	9.871E+01	1.850E+02	**1.202E+01**
	Std	2.984E+05	6.762E-05	8.609E+04	9.263E-04	1.137E-13	4.715E+02	7.058E-04	**3.048E-14**	4.896E+01
	IterNO	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06
	Time	8.307E+02	3.695E+02	1.357E+02	**9.323E+01**	1.032E+02	1.709E+02	1.094E+02	1.168E+02	1.587E+02
f_7	Best	8.000E+04	9.199E+00	1.927E+08	**9.858E-07**	2.387E+03	9.841E+01	1.100E+04	3.510E+02	1.919E-05
	Mean	3.656E+09	9.199E+00	8.600E+09	**1.288E-06**	2.387E+03	9.440E+07	1.110E+04	3.510E+02	4.472E+06
	Std	4.208E+09	9.245E-07	9.364E+09	1.702E-07	4.548E-13	2.819E+08	3.427E-03	**1.219E-13**	1.949E+07
	IterNO	1.000E+06	1.000E+06	1.000E+06	**4.553E+05**	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06
	Time	7.680E+02	2.881E+02	5.054E+01	**9.645E+00**	3.164E+01	9.070E+01	3.180E+01	3.919E+01	7.791E+01
f_8	Best	3.463E-01	2.479E+00	4.784E+00	4.436E+00	3.007E+00	9.924E-07	1.094E-02	2.426E+00	**9.845E-07**
	Mean	5.926E+00	2.479E+00	9.778E+00	4.436E+00	3.007E+00	6.403E-01	1.140E-02	2.426E+00	**1.007E-06**
	Std	3.756E+00	4.567E-15	4.182E+00	8.882E-16	8.882E-16	1.773E+00	1.495E-03	**8.189E-16**	5.498E-09
	IterNO	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	**2.946E+05**	1.000E+06	1.000E+06	3.718E+05
	Time	2.091E+03	2.306E+02	9.461E+01	9.780E+01	9.840E+01	1.235E+02	**6.640E+01**	9.991E+01	1.278E+02
f_9	Best	1.332E-01	4.089E-03	3.661E+00	1.601E-01	2.092E-04	4.188E-02	4.534E-03	1.124E-02	**9.984E-07**
	Mean	3.803E+01	4.220E-03	1.051E+02	1.601E-01	2.092E-04	4.673E+00	4.535E-03	1.124E-02	**1.017E-06**
	Std	3.838E+01	6.476E-05	1.147E+02	8.327E-17	1.314E+01	4.823E+00	2.800E-08	**2.420E-17**	5.421E-09
	IterNO	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	**3.332E+04**
	Time	7.786E+03	2.024E+03	2.638E+02	2.663E+01	3.996E+01	9.335E+01	9.743E+01	4.681E+01	**3.932E+00**
f_{10}	Best	6.250E-04	2.716E+00	4.942E+00	**8.746E-07**	3.631E-04	9.441E-07	4.527E+00	9.980E-07	9.748E-07
	Mean	9.728E+01	2.716E+00	2.107E+02	1.285E-06	3.631E-04	1.050E-01	4.527E+00	1.063E-06	**1.002E-06**
	Std	9.501E+01	2.809E-06	2.338E+02	1.794E-07	**1.084E-19**	3.360E-01	1.576E-06	2.658E-08	1.183E-08
	IterNO	1.598E+07	4.946E+06	1.000E+06	1.746E+05	1.000E+06	5.174E+04	7.101E+06	1.630E+04	**1.284E+04**
	Time	1.367E+05	9.506E+03	1.135E+04	1.053E+01	6.865E+01	3.764E+01	2.245E+03	**1.306E+00**	1.478E+00
f_{11}	Best	**1.982E-02**	1.095E+02	4.191E+02	9.081E+01	2.533E+02	2.627E+02	7.677E+02	7.263E+01	1.538E+02
	Mean	**3.051E+01**	2.570E+02	8.345E+02	9.081E+01	2.533E+02	2.660E+02	7.677E+02	7.263E+01	1.573E+02
	Std	2.468E+01	1.968E+02	3.003E+02	0.000E+00	2.842E-14	8.377E+00	1.716E-13	**0.000E+00**	1.372E+01
	IterNO	**4.141E+05**	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06	1.000E+06
	Time	3.510E+02	1.492E+03	7.154E+01	2.374E+01	3.519E+01	9.603E+01	3.871E+01	4.363E+01	7.968E+01
f_{12}	Best	**1.273E-03**	**1.273E-03**	1.066E-01	**1.273E-03**	1.570E-03	**1.273E-03**	**1.273E-03**	**1.273E-03**	**1.273E-03**
	Mean	1.544E+00	**1.273E-03**	3.751E+00	**1.273E-03**	1.570E-03	1.514E-01	**1.273E-03**	**1.273E-03**	**1.273E-03**
	Std	1.468E+00	2.588E-08	4.118E+00	6.333E-09	**0.000E+00**	6.499E-01	1.941E-08	2.308E-09	1.064E-09
	IterNO	3.031E+05	7.090E+04	1.000E+06	1.933E+05	1.000E+06	6.134E+04	8.613E+04	**2.192E+00**	2.273E+00
	Time	2.452E+03	8.698E+01	6.464E+02	2.004E+01	1.106E+02	**8.225E+00**	5.128E+01	1.804E+04	1.462E+04

The **bold** mark indicates that they are the best results among the algorithms.



(a) Fitness-iteration, f_1 (b) Fitness-iteration, f_2 (c) Fitness-iteration, f_3

(d) Fitness-iteration, f_4 (e) Fitness-iteration, f_5 (f) Fitness-iteration, f_6

(g) Fitness-iteration, f_7 (h) Fitness-iteration, f_8 (i) Fitness-iteration, f_9

(j) Fitness-iteration, f_{10} (k) Fitness-iteration, f_{11} (l) Fitness-iteration, f_{12}

Fig. 1: Fitness-iteration evaluation on 100-dimensional benchmark functions. The maximal iteration in the horizontal axis is defined as the minimal iterations obtained by any algorithm when it finds the global optimum.

and TSMQHOA, showing the superiority of the proposed algorithm in most cases. Several well-known numerical global optimizers such as the StudGA, SPSO2011, CLPSO, jDE, DEPSO and ICA are compared with AMQHOA-ES. The

Table III Wilcoxon rank-sum test on benchmark functions among StudGA, SPSO2011, CLPSO, jDE, DEPSO, ICA, MQHOA, TSMQHOA and AMQHOA-ES. Stopping criteria: $\sigma_s \leqslant 0.000001$ or the generation count $\geqslant maxFE$. The sample values for each of the compared algorithm are from 50 independent trials.

Func.	StudGA	SPSO2011	CLPSO	jDE	DEPSO	ICA	MQHOA	TSMQHOA
f_1	3.284E-20	5.785E-03	6.990E-18	3.360E-01	7.022E-18	3.257E-01	6.763E-02	4.780E-02
f_2	1.071E-20	2.981E-18	2.973E-18	9.293E-15	2.981E-18	4.890E-11	1.670E-01	2.981E-18
f_3	6.742E-18	7.030E-18	6.988E-18	3.501E-01	7.030E-18	5.035E-01	1.292E-01	3.378E-01
f_4	7.012E-18	7.041E-18	7.023E-18	3.766E-03	7.049E-18	4.704E-02	2.568E-02	2.977E-01
f_5	6.683E-18	6.683E-18	6.642E-18	7.097E-18	6.683E-18	5.449E-10	5.771E-02	6.683E-18
f_6	7.491E-18	6.428E-12	7.041E-18	1.272E-12	7.055E-18	3.772E-10	9.895E-13	1.425E-16
f_7	1.967E-16	4.793E-05	1.429E-14	7.781E-13	1.435E-14	5.586E-10	1.432E-14	6.619E-12
f_8	4.291E-19	7.055E-18	7.049E-18	7.449E-09	7.066E-18	6.863E-18	5.553E-12	7.057E-18
f_9	6.988E-18	1.042E-02	6.966E-18	4.571E-10	2.352E-02	8.334E-01	5.808E-01	1.319E-02
f_{10}	4.521E-11	7.505E-01	2.971E-11	7.115E-01	2.995E-11	4.915E-01	5.390E-01	7.448E-01
f_{11}	4.732E-20	1.959E-09	6.999E-18	8.406E-07	7.064E-18	7.947E-18	7.042E-18	2.321E-17
f_{12}	NAN*	2.628E-23	3.296E-20	NAN	3.310E-20	2.628E-23	2.628E-23	NAN

*NAN indicates the p-value is too small and out of calculation precision.

comparative results reveal the superiority or competitiveness of the proposed algorithm. In the near future, more complementary techniques will be considered to improve the performance of AMQHOA-ES. Meanwhile, the application of the proposed algorithm to real-world optimization problems is on our schedule.

ACKNOWLEDGMENT

This work is supported in part by the Fundamental Research Funds for the Central UniversitiesSouthwest Minzu University (2019NYB22), the National Natural Science Foundation of China (60702075).

REFERENCES

[1] J. H. Holland, "Genetic algorithms and the optimal allocation of trials," *SIAM J. Comput.*, vol. 2, no. 2, pp. 88–105, 1973.

[2] J. H.Holland, "Erratum: Genetic algorithms and the optimal allocation of trials," *SIAM J. Comput.*, vol. 3, no. 4, p. 326, 1974.

[3] C. Y. Lee and X. Yao, "Evolutionary programming using mutations based on the levy probability distribution," *Ieee Transactions on Evolutionary Computation*, vol. 8, no. 1, pp. 1–13, 2004.

[4] K. Y. Lee and F. F. Yang, "Optimal reactive power planning using evolutionary algorithms: A comparative study for evolutionary programming, evolutionary strategy, genetic algorithm, and linear programming," *Ieee Transactions on Power Systems*, vol. 13, no. 1, pp. 101–108, 1998.

[5] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *Ieee Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526–553, 2009.

[6] R. E. James Kennedy, "Particle swarm optimization," *IEEE Int. Conf. on Neural Networks*, 1995.

[7] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," *Technical Report TR06*, 2005.

[8] S. M. Mirjalili and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[9] M. Q. Hu, T. Wu, and J. D. Weir, "An adaptive particle swarm optimization with multiple adaptive methods," *Ieee Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 705–720, 2013.

[10] S. H. Wang, Y. Z. Li, and H. Y. Yang, "Self-adaptive mutation differential evolution algorithm based on particle swarm optimization," *Applied Soft Computing*, vol. 81, 2019.

[11] X. H. Chu, F. L. Cai, C. Cui, M. Q. Hu, L. Li, and Q. D. Qin, "Adaptive recommendation model using meta-learning for population-based algorithms," *Information Sciences*, vol. 476, pp. 192–210, 2019.

[12] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *Ieee Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[13] J. Zhang and A. C. Sanderson, "Jade: Adaptive differential evolution with optional external archive," *Ieee Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[14] A. Aleti and I. Moser, "A systematic literature review of adaptive parameter control methods for evolutionary algorithms," *Acm Computing Surveys*, vol. 49, no. 3, 2016.

[15] P. Wang, X. Ye, B. Li, and K. Cheng, "Multi-scale quantum harmonic oscillator algorithm for global numerical optimization," *Applied Soft Computing*, vol. 69, pp. 655 – 670, 2018.

[16] X. Ye, P. Wang, G. Xin, J. Jin, and Y. Huang, "Multi-scale quantum harmonic oscillator algorithm with truncated mean stabilization strategy for global numerical optimization problems," *Ieee Access*, vol. 7, pp. 18 926–18 939, 2019.

[17] X. Ye and P. Wang, "Impact of migration strategies and individual stabilization on multi-scale quantum harmonic oscillator algorithm for global numerical optimization problems," *Applied Soft Computing*, vol. 85, p. 105800, 2019.

[18] P. Wang, K. Cheng, Y. Huang, B. Li, X. Ye, and X. Chen, "Multiscale quantum harmonic oscillator algorithm for multimodal optimization," *Comput Intell Neurosci*, vol. 2018, no. 95001, pp. 1–12, 2018.

[19] W. Khatib and P. J. Fleming, "The stud ga: A mini revolution?" in *International Conference on Parallel Problem Solving from Nature*. Springer, Berlin, Heidelberg, 1998.

[20] M. Omran. (2011) Spso 2011, matlab. [Online]. Available: http://www.particleswarm.info/Programs.html

[21] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *Ieee Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[22] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition," in *2007 IEEE Congress on Evolutionary Computation, CEC 2007*, vol. 7, 10 2007, pp. 4661 – 4667.

[23] S. Hosseini and A. Khaled, "A survey on the imperialist competitive algorithm metaheuristic: Implementation in engineering domain and directions for future research," *Applied Soft Computing*, vol. 24, pp. 1078–1094, 11 2014.

[24] N. H. Awad, M. Z. Ali, P. N. Suganthan, and R. G. Reynolds, "An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems," in *IEEE Congress on Evolutionary Computation, CEC 2016, Vancouver, BC, Canada, July 24-29, 2016*. IEEE, 2016, pp. 2958–2965.

[25] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective real-parameter numerical optimization," Nanyang Tech. Univ., Singapore and Jordan Univ. of Science and Tech., Jordan and Zhengzhou Univ., Zhengzhou China, Tech. Rep., 2016.