Interval Search with Quadratic Interpolation and Stable Deviation Quantum-Behaved Particle Swarm Optimization (IQS-QPSO)

P Amini[1]*, A Bagheri[1], S Moshfegh[2]

1. Department of Mechanical Engineering, Faculty of Engineering, University of Guilan, Rasht, Iran

2. School of Business and Law, CQ University, Brisbane, Australia

ABSTRACT

In this article, in order to enhance the rate of convergence and scattering of particles at the same time, simple techniques are introduced. These techniques include: (1) Using the interval search to select a new particle candidate, (2) Replacement of three candidate particles instead to worst the particles in the population, (3) Using the best result of learning coefficients, (4) using a simple method to control the convergence of the algorithm in a high number of repetitions.

In this article, the performance of Quantum-Behaved Particle Swarm Optimization (QPSO) algorithm has been upgraded with using the interval search method. The proposed method of interval search of quantum-behaved particle swarm optimization algorithm has achieved better results than in the past with the use of quadratic interpolation recombination operator and stable deviation and interval search.

Moreover, the results of the proposed algorithm of Interval Search with Quadratic Interpolation and Stable Deviation Quantum-Behaved Particle Swarm Optimization (IQS-QPSO) is compared with the other former algorithms such as Quantum-Behaved Particle Swarm Optimization (QPSO), Quadratic Interpolation Quantum-Behaved Particle Swarm Optimization (Q-QPSO) and Stable Deviation Quantum-Behaved Particle Swarm Optimization (SD-QPSO). Then the performance improvement is reported. In order to compare the results of each algorithm, five famous functions are used and consequently the results are reported separately for each function.

1. INTRODUCTION

Optimizing is a methodical knowledge, with the ability to find out the optimal solution between all possible solutions for a problem. Optimality of solutions depends on the criteria that are usually related to that problem or the designer. For example, in designing of an engineering structure, solutions will depend on basic requirements of the constraints imposed by designer. Constraints imposed by designers or problems intrinsically decrease the number of solutions. If a problem is fully constrained, feasible solutions will be achieved. Of all the feasible solutions, global optimization will come up with the best solution. However, this best

*Corresponding Author: amini_pouriya@yahoo.com solution will not be always necessary or available. In some cases, local optimization is acceptable, especially on issues that the search space is complicated. Modelling of a problem is the first step in the optimization. At this step, the main challenge is mathematical modelling with respect to all existing constraints. Blocks of the structure that is candidates of solution are converted to numerical variables and solutions are demonstrated by numerical vectors. The optimum points or optimum results of an optimization problem are global minimum or maximum points of a function which is called the objective function. Every optimization problem can be considered as a minimization problem [1]. Finding the global minimum is the core of global optimization. By making the sign of an objective function negative, the optimization problem will find the maximum of the objective function. The objective function is the range including acceptable solutions. This range is determined by the constraints that must be accurately determined by use of equal or unequal signs mathematically. In the simplest possible form, constraints are determining the boundary conditions governing the problem. There will be complex relationships between the variables in complex problems [2]. The two main categories of optimization algorithm are deterministic and stochastic. Particle Swarm Optimization (PSO) is considered as a stochastic optimization method that is explored by many researchers. Stochastic optimization method is used to solve non-continues, non-convex and non-differentiable problems. on the other hand, the deterministic optimization method is suitable for continues, convex and differentiable problems [3].

2. PARTICLE SWARM OPTIMIZATION (PSO)

Kennedy and Eberhart developed the Particle Swarm Optimization (PSO) in 1995 as a random optimization model on basis of social simulation models [4]. The PSO algorithm searches a group of particles that are randomly moving in the search space. The best position achieved by each particle is called personal experience, which is simultaneously recorded. This experience in relation with the part or the whole population will determine the willingness of a group to move in a specific direction. This relationship can be fixed or adaptive and plays an important role in determining the convergence properties of an algorithm.

Development of particle swarm optimization is based on rules and concepts of organized societies in nature such as bird migration, movement of fish and flock of animals. In recent years, this method has been widely studied.

The results show that the proposed method could be compared with other smart direct search algorithms such as genetic algorithm [5,6].

3. MODELING OF PARTICLE SWARM OPTIMIZATION

The method of particle swarm optimization originated from simulating of social attribute of a population of birds. The main rule of finding food for each bird in the population is adapting with the speed and acceleration of the nearest neighbor bird based on the distance. After data simulation, the researchers [4] realized the ability of PSO model in optimization and they presented their optimization model in 1995.

By setting this theory in the context of mathematics, $A \subset R^n$ is considered as a search space and $f: A \rightarrow Y \subseteq R$ is considered as an objective function. In order to keep the definition as simple as possible, A is assumed as a feasible space of a problem on which any explicit constraint is not applied. Apart from search space and objective function, any other assumption is not required.

The mass of the components is called population and each component is called the particle. Population is defined as the following set: $S = \{x_1, x_2, \dots, x_N\}$, where N is number of particle (candidate solution) $x_i = (x_{i1}, x_{i2}, \dots, x_{in})^T \in A$, $i = 1,2, \dots, N$ user defines N for algorithm.

It is assumed that objective function $f(x)$ is available for all particles of A so that each particle has a unique function value $f_i = f(x_i)$. It is assumed that all particles of the search space are frequently moving. These movements can be defined with the help of velocity definition in the form of $v_i = (v_{i1}, v_{i2}, \dots, v_{in})^T, i = 1,2, \dots, N$

Velocity is being constantly updated so that particles can meet all the areas of A. t represents the number of repetitions. The position of the particle and its velocity are indicated as $x_i(t)$and $v_i(t)$.Velocity is updated based on the information obtained in the previous steps of the algorithm. This concept would be possible on basis of assigning a memory to each particle and save the best position of each particle.

Beside the set of S which includes the current positions of each particle, particle swarm optimization memorizes the positions of particles as $P = \{p_1, p_2, \dots, p_N\}$ that explains the best position that every particle has ever met.

The positions are in the form of $p_i = (p_{i1}, p_{i2}, \dots, p_{in})^T \in A, i = 1,2, \dots, N$ and $p_i(t) = arg_t min f_i(t)$ that t is the number of repetitions.

Particle swarm optimization is simulated based on social behavior. Therefore, PSO needs a mechanism enabling every particle to exchange the information and to connect with its experience.

The aim of the algorithm is to approximate the global minimum that has been observed by all particles during the run. Therefore, the exchange of such information is necessary. Considering g as the index of best position with the lowest amount of objective function in p, in a certain number of repetitions t , the formula is as below:

$$p_g(t) = arg_i min f(p_i(t))$$

As a result, the equation of particle swarm optimization is obtained as follows:

$$v_{ij}(t+1) = \mathrm{w} v_{ij}(t) + c_1 R_1 \left(p_{ij}(t) - x_{ij}(t)\right) + c_2 R_2 \left(p_{gj} - x_{ij}(t)\right) \tag{1}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \qquad ; i = 1,2, \dots, N, \quad j = 1,2, \dots, n \tag{2}$$

t in the above equations represents the number of repetitions, $R_1$and $R_2$are random values with the uniform distribution in the interval of [0,1] and $c_1$and c_2 are weighting factors which c_1 is a personal learning factor and c_2 is a social learning factor. In the primary sample of particle swarm optimization, equal acceleration coefficients were applied to the equation $c = c_1 = c_2$ in equation (1).

It should be noted that the next version of this equation is affected by each of c_1 and c_2 factors individually [7]. Inertia weight factor w is embedded in order to eliminate of the effect of velocity $v_{ij}(t)$ during the run of the algorithm. Therefore, it is desirable that the value of w to be decreased by time. as a common choice, the coefficient of w is taken greater than 1.0 at the start of running the algorithm in order to explore more particles (for example the amount of this coefficient is considered as 1.2), then by a linear decrease in the amount of w to zero the fluctuation of particles will be reduced. Usually a positive small value for example, 0.1 is considered as the lowest inertia weight factor so that the effect of previous velocity is not omitted from the equation.

In general, the linear reduction of factor w to be indicated in the form of $w(t) = w_{up} - \left(w_{up} - w_{low}\right)\frac{t}{T_{max}}$ where t represents the number of repetitions, w_{up} and $w_{low}$are lower and upper limits of w factor and T_{max} factor is the total number of repetitions. Factors of PSO are highly important in velocity, convergence and efficiency of the algorithm. Weight factors in the convergence of particles play a vital role. These factors adjust the contrast between the capability of population in local search and global search.

The large factors will help global search, while small amounts, make local search easy. Therefore, the suitable amount for inertia weight factor should maintain the balance between local and global search and subsequently, the optimal solution would be found at the lowest number of repetitions [2].

The effectiveness of primary method of PSO has attracted the attention of many researchers. Simplicity of PSO allowed the scientists of different disciplines with limited background in computers and programming skills to benefit from the PSO as an efficient optimization tool instead of previous inefficient methods. As a result, different optimization methods and compositions of PSO have been formed, which are generally divided into two major categories:

- Hybrid Approaches
- Changing Approaches, the Particle Swarm Optimization algorithm

Hybrid methods have combined PSO algorithm with one of the known algorithms and will lead to improvement of results previously obtained. In reference [8] PSO algorithm is combined with Ant Colony algorithm. In reference [9] the combination of PSO algorithm and Ant Colony algorithm is classified and considering enzymes, the results have been very promising.

In reference [10,11], combining of two different types of PSO algorithm and complementary differences algorithm has led to good results. In reference [12 and 13] Genetic Programming combined with PSO algorithm and provides the ability to train the particles.

Some methods have maintained the nature of PSO algorithm. They have made small changes in the structure of Particle Swarm Optimization algorithm and have improved it. Unified Particle Swarm Optimization (UPSO), Memetic Particle Swarm Optimization (MPSO), Guaranteed Convergence Particle Swarm Optimization (GCPSO), Cooperative Particle Swarm Optimization (CPSO), Niching Particle Swarm Optimization (NPSO) Quantum Particle Swarm Optimization (QPSO) are some of these methods [14, 15, 16, 17, 18, 19 and 20].

4. QUANTUM PARTICLES SWARM OPTIMIZATION (QPSO)

Quantum Particle Swarm Optimization (QPSO) was introduced by Sun et al in 2004 [20]. Although this method is considered as a kind of Particle Swarm Optimization, it has a different framework. PSO benefits from Newton's laws for the motion of the particles but QPSO benefits from quantum behavior of particles according to the quantum rules. Quantum computing are new theories, which are the result of computer science and quantum mechanics. The main objective is to review of all the possible answers regarding to the laws of quantum mechanics, and the software necessary to use over the past decade, Quantum computing are being paid more attention rather than classical calculations and proved an efficient tool used in problem solving. Quantum Particle Swarm Optimization is more convenient and less complex comparing with other methods. Moreover, QPSO is capable of parallel processing so it is appropriate for solving problems that have a wide area of solution.

In Classic PSO, a particle is known by its position vector $\mathrm{x}_{\mathrm{ij}}^{\mathrm{t}}$ and velocity vector $\mathrm{v}_{\mathrm{ij}}^{\mathrm{t}}$ that defines the trajectory of the particle. A particle moves on a path predetermined by Newtonian mechanics. It should be noted that if Quantum mechanics is taken into account, the trajectory would be nonsense because $\mathrm{x}_{\mathrm{ij}}^{\mathrm{t}}$and $\mathrm{v}_{\mathrm{ij}}^{\mathrm{t}}$ of a particle cannot simultaneously be determined in accordance with the principle of uncertainty. So if in a system, the particles have the quantum behavior, the effectiveness of classic PSO will be different from the classical PSO [21].

In quantum mechanics, time-dependent Schrödinger equation is as follows:

$$j\hbar\frac{\partial}{\partial t}\Psi(r,t) = \mathrm{H}(r)\Psi(r,t) \tag{3}$$

where:

$$\mathrm{H}(r) = -\frac{\hbar^2}{2m}\nabla^2 + V(r) \tag{4}$$

Where $\mathrm{H}(r)$is a time-dependent Hamiltonian operator, ħ is Planck's constant, m is mass of the particle and $V(r)$ is the potential energy distribution function.

Square range $Q = |\Psi|^2$of the wave function $\Psi(r,t)$in equation (3) acts as much as the probability of particle motion in the following normalization process:

$$\iiint |\Psi(r,t)\,|^2\, dxdydz = 1.0$$

In QPSO, the population is considered as a quantum system in which each particle moves towards p position based on quantum state of its wave function.

In reference [20],the weighted average of best position of a particle x_i is considered as the location of the particles p_i and the best position of the population, p_j , is calculated as follows:

$$p_j = \frac{\varphi_1 p_{ij} + \varphi_2 p_{gj}}{\varphi_1 + \varphi_2}, \qquad j = 1,2,\dots,n$$

Where $\varphi_1 = c_1 r_1$ and $\varphi_2 = c_2 r_2$, c_1 and c_2 are personal and collective learning coefficients of pso and r_1, r_2 are random numbers with the uniform distribution in the interval [0,1]. To indicate $QPSO$ performance in the simplest case, a one-dimensional model is considered. Assuming the centrality of p population in the equation (5), the ability of moving particles in that direction is calculated from the following formula [20]:

$$V(x) = -\gamma\delta(x - p) = -\gamma\delta(y) \tag{6}$$

Where $y = x - p$. Through mathematical calculations, the following wave equation is obtained [20]:

$$\Psi(y) = \frac{1}{\sqrt{L}} exp\left(-\frac{|y|}{L}\right) \tag{7}$$

And it is needed to calculate the probability as follows:

$$Q(y) = |\Psi(y)|^2 = \frac{1}{L} exp\left(-2\frac{|y|}{L}\right) \tag{8}$$

Where $L = h^2/m\gamma$. So far, a probability density of the particle position is obtained. Unless the exact location of the particle is obtained, the algorithm is not completely done. So, the position of the particle must be measured. This method is known as collapse of classical quantum positions. The collapse is possible through the Monte Carlo simulation.

For more explanation, S is considered as a random number with uniform distribution in the interval $[0,1/L]$. In this case, $\mathrm{S} = \frac{1}{L}\mathrm{u}$ where u is a random number with uniform distribution in the interval [0,1]. Replacing $Q(y)$in the left side of equation (8) with S, x in model $QPSO$ appears as follows:

$$\mathrm{x} = \mathrm{p} \pm \frac{\mathrm{L}}{2}\ln\left(\frac{1}{\mathrm{u}}\right) \tag{9}$$

equation (9) generates two new positions for a particle that are determined by the objective function. Reference [20] has demonstrated that this QPSO algorithm converges. The only controlling parameter appeared in updating of $QPSO$ equation, was the L coefficient.

The study on the effects of other factors can be a fascinating field in this method. Reference [20] includes $QPSO$ sensitivity study on changes of the L coefficient [2]. The basic equations are considered repeatable by the following forms:

$$x_{ij}^{t+1} = p_{ij}^{t} + \beta\left|mbest - x_{ij}^{t}\right| \ln\left(\frac{1}{u}\right) \quad if \quad k \geq 0.5 \tag{10}$$

$$x_{ij}^{t+1} = p_{ij}^{t} - \beta\left|mbest - x_{ij}^{t}\right| \ln\left(\frac{1}{u}\right) \quad if \quad k < 0.5 \tag{11}$$

Where

$$p_{ij}^{t} = \frac{p_{ij}^{t} c_1 + c_2 p_{gj}^{t}}{c_1 + c_2} \tag{12}$$

$$m_{best} = \frac{1}{M}\sum_{i=1}^{M} P_i = \left(\frac{1}{M}\sum_{i=1}^{M} P_{i1}\ , \frac{1}{M}\sum_{i=1}^{M} P_{i2}, \dots,\ \frac{1}{M}\sum_{i=1}^{M} P_{ij}\ \right) \tag{13}$$

In the original PSO, each particle independently converges to the global optimum position. But in the QPSO algorithm, in spite of the best overall position, particles cannot converge toward the best position without considering other particles and interaction.

The distance between current position and the position of $mbest$ expressed in equations (10) and (11). This distance indicates scattering particles for the next iteration.

The average of the best of a population or $mbest$ is defined as the average of best positions of all particles in a population. Parameters of u, k, c_1 and c_2 in the equations of (10) to (13) are the random numbers selected in the interval [0,1]. β coefficient is considered as the contraction-expansion coefficient. As long as some particles are close to the total optimal position, the best position of several particles would be far from the total optimal position. Those far particles are named slow particles. The position of $mbest$ effected by the slow particles will have a distance from the total optimal position. When slower particles following their group converge to the total optimal position, the average position or $mbest$ will slowly converge to the total optimal position as well. It should be noted in this algorithm that the distance between the $mbest$ position and the best position of individual particles closed to the total optimal point will not be reduced quickly.

Particle's speed convergence to the total optimal point will be reduced only in such a way that the search around the total optimal point accelerates so that the slower particles become close to the total optimal point. So, in $QPSO$ algorithm with average optimized positioning strategy, bad or slower particles are never removed by the aggregation of other particles. This is an intelligent issue with more social organism to achieve the optimal solution based on the fact that all particles cooperate and participate. Therefore, it should be noted that with patience and movement of slow particles, the ability of global search in $QPSO$ algorithm will improve [22].

5. IQS-QPSO METHOD

In this paper, a simple but efficient version of $QPSO$ is presented. The main idea of this method is based on advanced algorithms of Q-$QPSO$ and SD-$QPSO$. Here, besides the functions defined in the previous, (Q-$QPSO$) one internal search function is considered to increase the accuracy and performance of the algorithm in the search area. The other change in this algorithm is using experience of other researchers particularly applying the best coefficients. Similar to Q-$QPSO$ and SD-$QPSO$, new particle swarm is generated in this algorithm. After comparing the results of the objective function, particles leading to better results will take the place of the worst particles. This process is maintained until the end of iteration. This strategy is repeated to view the best answer.

Quadratic operator is a nonlinear operator that present a new solution vector in the minimum point of the second-degree curve passing among the three particles selected from the aggregation.

These particles based on $Q\text{-}QPSO$ here named $\{a, b, c\}$ are selected as follows [23]:

- Choose the particle of the population which offers the minimum objective function value $\{a = X_{min}\}$.
- Randomly choose two particles from the remaining population $\{b, c\}$
-

It should be noted that a, b and c should be chosen differently. Since the variation in the search space for optimization algorithms is a very important indicator, the method presented in this paper to increase the chances of finding a new point is so effective. This is the basic idea of this method. Since the better objective function value is kept in this algorithm, it is likely to create a new vector of the solution that improve the objective function value than existing solutions.

Based on $Q\text{-}QPSO$ method and quadratic interpolation recombination operator, the first candidate particle $xpzt$ is as follows:

$$x_{pz}^{t} = \frac{1}{2}\frac{(b_z^2-c_z^2)f(a_z)+(c_z^2-a_z^2)f(b_z)+(a_z^2-b_z^2)f(c_z)}{(b_z-c_z)f(a_z)+(c_z-a_z)f(b_z)+(a_z-b_z)f(c_z)} \tag{14}$$

Moreover, a sustained deviation function in parallel to create a second particle x_{qz}^{t} candidate is considered as recombination operator as follows [22]:

$$x_{qz}^{t} = \frac{(b_z-c_z)^2f^2(a_z)}{(b_z-c_z)^2f^2(a_z)+1}\tanh\big(\big(\ (b_z-c_z)f(a_z)\big) + \frac{(c_z-a_z)^2f^2(b_z)}{(c_z-a_z)^2f^2(b_z)+1}\tanh\big((c_z-a_z)f(b_z)\big) + \frac{(a_z-b_z)^2f^2(c_z)}{(a_z-b_z)^2f^2(c_z)+1}\tanh\big((a_z-b_z)f(c_z)\big) \tag{15}$$

It should be noted that the equation (15) demonstrates a function which is very smoothly based on the value a_z, b_z and c_z. The result of this function is a numerical value in neighborhood of these three particles arbitrarily selected from population.

Then, the third candidate x_{wz}^{t} achieved through the internal search using the following algorithm:

1-Calculate the length of the search path

$$L = \propto_u - \propto_l$$

Where $\propto_u$ is a particle with maximum value and $\propto_l$ is a particle with a minimum value.

2- dividing the path by desired equal number of points

```
$For\ i = 1 : n$
    $x_{wz}^{t}(i) = \propto_l + \frac{i+l}{n-1}$
$end$
```

Where n is the number of dimensions of the problem.

3- Calculating the objective function for every point and comparison with the rest of the points with a minimum value of objective function:

$$F(x_{wz}^{t}) = minf(x_{wz}^{t}(i))$$

4- Introducing the point candidates x_{wz}^{t} with the lowest objective function value. In most cases of optimization problems using accumulation method or quantum particles, the convergence to the optimal answer in the beginning (early iterations of the algorithm) is more quickly done.

But on the other hand, usually the convergence rate near solution is slow so that the different techniques of modified algorithms are considered to accelerate this issue. The use of the proposed function and modified algorithm even in the neighborhood optimal answer with a corresponding distortion can assist to accelerate sudden convergence speed and increase the possibility of approaching the optimal solution.

The calculation in IQS-$QPSO$ algorithm includes the following steps:

Step 1: Input population with random numbers that are distributed uniformly in the search area.
Step 2: Calculate $mbest$ using equation (13)
Step 3: Find the position of the particles using equations (10) and (11)
Step 4: Obtain the value of each particle in the objective function.
Step 5: If the current value of the objective function is better than the best value in its history, update the $Pbest$ using the current value.
Step 6: Update the $Pgbest$ by comparing
Step 7: Using equation (14) to choose the first particle.
Step 8: Using equation (15) to choose the second particle.
Step 9: Use the internal search algorithm to choose the third particle.
Step 10: This step is to compare the three particles with the worst three particles in the population and replace them if they are better
Step 11: Repeat from step 2 until the number of iterations of the algorithm to finish or acceptable solution is reached.

5.1. Changes made in the algorithm coefficients

Discussions on the coefficients in PSO algorithm have been considered by many researchers and more advances obtained in this way also indebted to check these factors and their impact on the results.

With the use of the quantum principle in PSO and get the IQS-$QPSO$ algorithm and increase the number of coefficients, discussing the coefficients of the algorithm is necessary.

In 2002, Kennedy and Clerk performed extensive studies on the learning coefficients [24]. They were able to obtain optimum value of the coefficients ($c_1 = c_2 = 2.05$). To obtain better results in solving problems using a weight factor can change these two coefficients as follows:

$$\begin{gathered}
\text{phi1} = 2.05 \\
\text{phi2} = 2.05 \\
\text{phi} = \text{phi1} + \text{phi2} \\
\text{chi} = 2/(\text{phi} - 2 + \sqrt{(\text{phi}^2 - 4 \times \text{phi})}) \\
\text{w} = \text{chi} \\
\text{wdamp} = 0.99 \\
\text{c1} = \text{chi} \times \text{phi1} \\
\text{c2} = \text{chi} \times \text{phi2} \\
\text{w} = \text{w} * \text{wdamp}
\end{gathered}$$

c_1 and c_2 coefficients derived from the above, have the best performance of algorithms in solving problems. u coefficient is a random number in the range of [0,1]. β coefficient called expansion or contraction coefficient has a great impact on results. In fact, this factor can indicate the amount of allegiance of particles to its quantum algorithm.

Small amounts of this factor decrease the features of quantum behavior and its high value boost the role of $mbest$ and consequently the role of slow or bad particles.

According to quantum behavior of the IQS-$QPSO$ algorithm and the results obtained, the amount of this coefficient is considered as 1.45.

By examining results and comparing with each other, one strategy is to increase of u in order to improve the power of converging particles, especially in high repetitions, and consequently it is away from the quantum properties. This increase is performed by gradual increase of u that starts when the number of iterations reached 35% of the total number of repetitions during the implementation of software.

6. CHECK THE PERFORMANCE OF THE PROPOSED ALGORITHM IQS-QPSO

In the last section, the proposed IQS-$QPSO$ algorithm was introduced and the basis of applied changes was discussed. Now, the performance of IQS-$QPSO$ algorithm is reviewed and compared to the performance of former algorithms including Q-$QPSO$, $QPSO$ and SD-$QPSO$. In order to compare the results with each other, five common benchmark functions in optimization named Rastrigin, Griewank, Ackley, Sphere and Schwefel22 to be used. MATLAB code optimization algorithm has been developed to optimize the five function tests. In Figure 1, the five functions are illustrated.

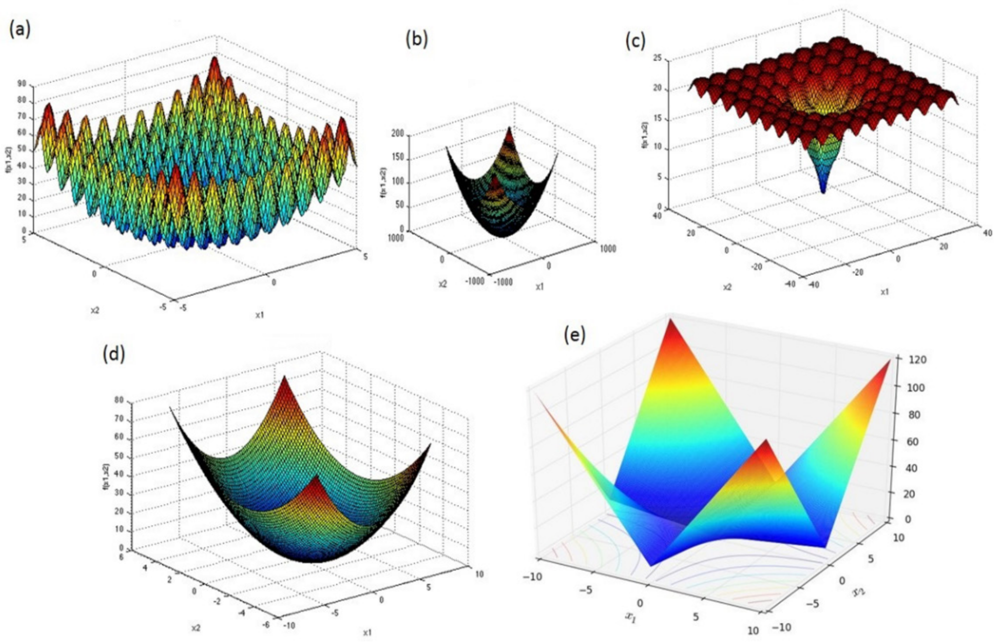

Fig.1. benchmark functions: (a) Rastrigin (b) Griewank (c) Ackley (d) Sphere (e) Schwefel22

All of these functions have a global minimum $f^* = 0$ in $x^* = (0,0,\dots,0)^T$. Five functions equation are shown in Table 1 [25].

Table1. Numerical benchmark problems.

Benchmark name	**Function**	**optimum**
Rastrigin	$f(x) = \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i) + 10]$	0
Griewank	$f(x) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	0
Ackley	$f(x) = -20exp\left(-0.02\sqrt{\frac{\sum_{i=1}^{n}x_i^2}{n}}\right) - exp\left(\frac{\sum_{i=1}^{n}\cos(2\pi i)}{n}\right) + 20 +$ $\exp(1)$	0
Sphere	$f(x) = x^t x = \sum_{i=1}^{n} x_i^2$	0
Schwefel22	$f(x) = \sum_{i=0}^{n-1}\lvert x_i\rvert + \prod_{i=0}^{n-1} x_i$	0

The results for each of these functions are separately shown in the following Tables 2 to 6.

QPSO:	quantum-behaved particle swarm optimization;
Q-QPSO:	quantum-behaved particle swarm optimization with quadratic interpolation recombination operator;
SD-QPSO:	stable deviation quantum- behaved particle swarm optimization;
IQS-QPSO:	interval search with quadratic interpolation and stable deviation quantum- behaved particle swarm optimization.

Table 2. Comparison results of Rastrigin function (mean best).

Pop size	Number of dimensions	Number of generations	QPSO	Q-QPSO	SD-QPSO	IQS-QPSO
20	11	50	33.2951	26.0285	5.8319	1.1234
	15	200	30.6552	19.5663	1.3162	0.9960
	20	700	21.8670	18.6481	0.7118	0.0544
40	11	50	15.8183	13.0594	4.0727	0.9967
	15	200	7.7326	8.8370	0.6661	0.0009
	20	700	11.3848	9.4721	0.4333	0.0014
80	11	50	8.6533	7.4627	3.5186	0.9966
	15	200	6.9078	6.8225	0.2991	2.7401e-5
	20	700	9.1539	8.6241	0.1990	6.7138e-7

Table 3. Comparison results of Griewank function (mean best).

Pop size	Number of dimensions	Number of generations	QPSO	Q-QPSO	SD-QPSO	IQS-QPSO
20	11	50	0.0315	0.0207	l.5171e-4	4.6745e-05
	15	200	0.0182	0.0081	7.4015e-15	7.5959e-07
	20	700	0.0061	0.0038	l.4433e-16	l.6031e-10
40	11	50	0.0115	0.0087	7.3334e-5	9.7197e-08
	15	200	0.0031	0.0012	2.1464e-16	4.063e-10
	20	700	1.3142e-4	2.8498e-5	8.0338e-17	3.8136e-13
80	11	50	0.0036	0.0022	l.0574e-5	3.2644e-10
	15	200	l.9504e-4	4.0409e-5	5.2217e-17	0
	20	700	1.3271e-7	l.0145e-7	7.9614e-18	0

QPSO:	quantum-behaved particle swarm optimization;
Q-QPSO:	quantum-behaved particle swarm optimization with quadratic interpolation recombination operator;
SD-QPSO:	stable deviation quantum- behaved particle swarm optimization;
IQS-QPSO:	interval search with quadratic interpolation and stable deviation quantum- behaved particle swarm optimization.

Table 4. Comparison results of Ackley function (mean best).

Pop size	Number of dimensions	Number of generations	QPSO	Q-QPSO	SD-QPSO	IQS-QPSO
20	11	50	2.2379	2.0908	0.5271	0.0883
	15	200	1.5936	0.9980	0.0851	0.0003
	20	700	0.4700	0.0600	3.6119e-15	1.5319e-09
40	11	50	1.4122	1.2795	0.3218	0.03836
	15	200	0.5274	0.2490	0.0635	9.3169e-09
	20	700	0.3514	0.0498	2.7830e-15	6.2172e-15
80	11	50	0.8723	0.5763	0.2681	0.0164
	15	200	0.3740	0.0697	9.0862e-8	3.7825e-11
	20	700	0.4314	0.0549	1.2633e-15	6.2172e-15

Table 5. Comparison results of Sphere function (mean best).

Pop size	Number of dimensions	Number of generations	QPSO	Q-QPSO	SD-QPSO	IQS-QPSO
20	11	50	1.4226	1.1825	0.5241	0.0121
	15	200	1.2018	0.8737	0.1611	0.0020
	20	700	0.9192	0.6084	0.1485	0.0040
40	11	50	0.4779	0.3298	0.1895	0.0053
	15	200	0.0960	0.0892	0.0124	4.8601e-6
	20	700	0.0142	0.0191	0.0011	3.2189e-14
80	11	50	0.3543	0.3087	0.1860	0.0018
	15	200	0.0357	0.0191	0.0066	9.0031e-7
	20	700	0.0034	0.0015	1.4438e-4	2.5286e-17

QPSO:	quantum-behaved particle swarm optimization;
Q-QPSO:	quantum-behaved particle swarm optimization with quadratic interpolation recombination operator;
SD-QPSO:	stable deviation quantum- behaved particle swarm optimization;
IQS-QPSO:	interval search with quadratic interpolation and stable deviation quantum- behaved particle swarm optimization.

Table 6. Comparison results of Schwefel22 function (mean best).

Pop size	Number of dimensions	Number of generations	QPSO	Q-QPSO	SD-QPSO	IQS-QPSO
20	11	50	1.5962	1.4673	0.3471	0.0988
	15	200	0.4758	0.3832	3.2816e-118	0.0075
	20	700	0.6803	0.0392	0	0.0042
40	11	50	0.9302	0.7725	0.1711	0.0601
	15	200	0.1518	0.1085	1.2347e-76	3.0819e-05
	20	700	0.1410	0.0017	0	0.0002
80	11	50	0.3806	0.3631	0.1437	0.0221
	15	200	0.2665	0.1886	1.3059e-8	3.6878e-08
	20	700	0.0990	0.0238	1.1698e-214	1.4938e-05

In table 2, for Rastrigin function when the population size is selected 20 and the number of production and dimension of problems to be respectively 50 and 11, the difference between results of the two optimized algorithms of QPSO and Q-QPSO is equal to 7.2666 unit.

Reply difference between SD-QPSO algorithm and two algorithms of Q-QPSO and QPSO is 20.1966 and 27.4632 respectively.

Reply difference between IQS-QPSO algorithm and QPSO, Q-QPSO and SD-QPSO is respectively, 32.1717, 24.9051 and 4.7085. On the other hand, when the population size is 80, number of dimension is 20 and number of production in function is 700, the difference between the algorithm IQS-QPSO and QPSO, Q-QPSO and SD-QPSO is 9.1539, 8.6241and 0.1990 .

In the latter case, increase in the performance of the algorithm up to three million times is also visible. This means dramatic increase in performance optimization algorithm IQS-QPSO with rising population and the number of productions compared to the previous algorithms.

By examining Table 3 in population size 20, dimension number 11 and generation number 50, IQS-QPSO algorithm generates the results that is 673 times better than results of QPSO. Same comparison for the results of IQS-QPSO is respectively 442 times and 3 times better than Q-QPSO and SD-QPS that is due to the absolute superiority of the IQS-QPSO algorithm than the other three algorithms.

By a closer look at this table, interesting results to be revealed. Considering the size of 20, by increase of dimension and generation, IQS-QPSO algorithm loses its superiority to SD-QPSO algorithm. The main reason for the decline in quality is high number of repetitions. By the increase of the number of repetitions, particles would be very small amounts and the quality and speed of convergence of the algorithm has been reduced. However, by the increase of population, the superiority of this algorithm is proved, and better results would be obtained, so that considering the population size of 80, IQS-QPSO algorithm would achieve the precise result of zero.

Table 4 shows that considering the population size 20, dimension number 11 and generation number 20, the results of IQS-QPSO algorithm are improved 25, 23 and 6 times comparing to QPSO, Q-QPSO and SD-QPSO. Similar to results of Table 3, by increase of repetitions, IQS-QPSO algorithm loses its superiority to SD-QPSO. However, the study of all results points out that the performance of IQS-QPSO algorithm is considerably better than other algorithms.

Table 5 illustrates the better results of the proposed algorithm of IQS-QPSO compared to the results of the previous algorithms of QPSO, Q-QPSO and SD-QPSO. For example, considering population of 100, dimension of 20 and generation of 250, the results of IQS-QPSO algorithm compared with the results of QPSO, Q-QPSO algorithms are 1.3446e14, 5.9321e13 and 5.7099e12 times better.

In Table 6, with the population size 20, dimension size 11 and generation 20, IQS-QPSO algorithm is 16, 14 and 3 times better than QPSO, Q-QPSO and SD-QPSO respectively. Similar to the results extracted from table 3 and 4, by increase in number of generation and constant population size, IQS-QPSO algorithm loses its absolute superiority to the SD-QPSO algorithm. In this particular issue, by increase in the number of dimensions and repetitions, the results of the IQS-QPSO algorithm are extremely degraded and the algorithm poorly acts.

7. RESULTS AND DISCUSSIONS

In this paper, a new version of the quantum-behaved PSO algorithm called (IQS-QPSO) is introduced. The main goal is to improve the performance of the algorithm using the simplest methods and ideas with the preservation of the quantum properties of the algorithm.

After presenting the IQS-QPSO algorithm, it is applied on five test functions. The results obtained from this algorithm are compared with the former algorithms such as QPSO, Q-QPSO and SD-QPSO on basis of different population size ranging from 20 to 200, different generation ranging from 20 to 1500 and different dimensions ranging from 11 to 20.

According to tables 2-6, IQS-QPSO algorithm obviously results in near optimum solution in all the cases. The results of simulations show the fast converging and high performance of IQS-QPSO algorithm. Therefore, the superiority of IQS-QPSO algorithm is proven. The IQS-QPSO algorithm is suggested as an advanced optimization algorithm that can solve more complex problem.

REFERENCES

[1] Nariman-Zadeh, N., Atashkari, K., Jamali, A., Pilechi, A. and Yao, X., 2005. Inverse modelling of multi-objective thermodynamically optimized turbojet engines using GMDH-type neural networks and evolutionary algorithms. Engineering Optimization, 37(5), pp.437-462.

[2] Parsopoulos, K.E. ed., 2010. Particle swarm optimization and intelligence: advances and applications: advances and applications. IGI global.

[3] Mahmoodabadi, M.J., Mottaghi, Z.S. and Bagheri, A., 2014. HEPSO: high exploration particle swarm optimization. Information Sciences, 273, pp.101-111.

[4] Kennedy, J., & Eberhart, R. C.,1995. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948.

[5] Kwok, N.M., Liu, D.K. and Dissanayake, G., 2006. Evolutionary computing based mobile robot localization. Engineering Applications of Artificial Intelligence, 19(8), pp.857-868.

[6] Mouser, C.R. and Dunn, S.A., 2005. Comparing genetic algorithms and particle swarm optimisation for an inverse problem exercise. ANZIAM Journal, 46, pp.89-101.

[7] Engelbrecht, A.P., 2007. Computational intelligence: an introduction. John Wiley & Sons.

[8] Dorigo, M. and Thomas, S., 2004. Ant Colony Optimization. Cambridge, vol. 9, Dec. 2002.

[9] Holden, N. and Freitas, A.A., 2005, June. A hybrid particle swarm/ant colony algorithm for the classification of hierarchical biological data. In Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE (pp. 100-107). IEEE.

[10] Hendtlass, T., 2001, June. A combined swarm differential evolution algorithm for optimization problems. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 11-18). Springer Berlin Heidelberg.

[11] Zhang, W.J. and Xie, X.F., 2003, October. DEPSO: hybrid particle swarm with differential evolution operator. In Systems, Man and Cybernetics, 2003. IEEE International Conference on (Vol. 4, pp. 3816-3821). IEEE.

[12] Poli, R., Langdon, W.B. and Holland, O., 2005, March. Extending particle swarm optimisation via genetic programming. In European Conference on Genetic Programming (pp. 291-300). Springer Berlin Heidelberg.

[13] Poli, R., Di Chio, C. and Langdon, W.B., 2005, June. Exploring extended particle swarms: a genetic programming approach. In Proceedings of the 7th annual conference on Genetic and evolutionary computation (pp. 169-176). ACM.

[14] Parsopoulos, K.E. and Vrahatis, M.N., 2004. On the computation of all global minimizers through particle swarm optimization. IEEE Transactions on evolutionary computation, 8(3), pp.211-224.

[15] Parsopoulos, K.E. and Vrahatis, M.N., 2001. Particle swarm optimizer in noisy and continuously changing environment. methods, 5(6), pp289-294.

[16] Dawkins, R., 1976. The selfish gene New York: Oxford Univ.

[17] van den Bergh, F. and Engelbrecht, A.P., 2002, October. A new locally convergent particle swarm optimiser. In Systems, Man and Cybernetics, 2002 IEEE International Conference on (Vol. 3, pp. 6-pp). IEEE.

[18] Clearwater, S.H., Hogg, T. and Huberman, B.A., 1992. Cooperative problem solving. Computation: The micro and the macro view, 275, pp.33-70.

[19] Brits, R., Engelbrecht, A.P. and Van den Bergh, F., 2002, November. A niching particle swarm optimizer. In Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning (Vol. 2, pp. 692-696). Singapore: Orchid Country Club.

[20] Sun, J., Feng, B. and Xu, W., 2004, June. Particle swarm optimization with particles having quantum behavior. In Evolutionary Computation, 2004. CEC2004. Congress on (Vol. 1, pp. 325-331). IEEE.

[21] Feng, B. and Xu, W., 2004, December. Adaptive particle swarm optimization based on quantum oscillator model. In Cybernetics and Intelligent Systems, 2004 IEEE Conference on (Vol. 1, pp. 291-294). IEEE.

[22] Moghaddam, J.J. and Bagheri, A., 2015. A novel stable deviation quantum-behaved particle swarm optimization to optimal piezoelectric actuator and sensor location for active vibration control. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 229(6), pp.485-494.

[23] Pant, M., Thangaraj, R. and Abraham, A., 2008, July. A new quantum behaved particle swarm optimization. In Proceedings of the 10th annual conference on Genetic and evolutionary computation (pp. 87-94). ACM.

[24] Clerc, M. and Kennedy, J., 2002. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE transactions on Evolutionary Computation, 6(1), pp.58-73.

[25] Storn, R. and Price, K., 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4), pp.341-359.