An Improved Quantum-behaved Particle Swarm Optimization Algorithm Based on Linear Interpolation

Shouyong Jiang
Centre for Computational Intelligence (CCI),
School of Computer Science and Informatics,
De Montfort University
The Gateway, Leicester LE1 9BH, U.K.
Email: shouyong.jiang@email.dmu.ac.uk

Shengxiang Yang
Centre for Computational Intelligence (CCI),
School of Computer Science and Informatics,
De Montfort University
The Gateway, Leicester LE1 9BH, U.K.
Email: syang@dmu.ac.uk

***Abstract*—Quantum-behaved particle swarm optimization (QPSO) has shown to be an effective algorithm for solving global optimization problems that are of high complexity. This paper presents a new QPSO algorithm, denoted LI-QPSO, which employs a model-based linear interpolation method to strengthen the local search ability and improve the precision and convergence performance of the QPSO algorithm. In LI-QPSO, linear interpolation is used to approximate the objective function around a pre-chosen point with high quality in the search space. Then, local search is used to generate a promising trial point around this pre-chosen point, which is then used to update the worst personal best point in the swarm. Experimental results show that the proposed algorithm provides some significant improvements in performance on the tested problems.**

I. Introduction

In many real-world applications, for example, applied sciences and engineering, the optimization function of interest may be non-linear, non-smooth, or simulation-based. It is with this view in mind that some search methods that do not require much information about the function were developed. They are known as meta-heuristic methods or nature-inspired algorithms. Unlike traditional gradient-based methods, these search methods use no properties of the function being optimized. The only requirement on the problem is that the fitness value $f(x)$ can be computed for any solution $x \in \Omega$, where Ω represents the search space. Additionally, they are also easy to implement. Thus, meta-heuristic algorithms are welcomed to solve real-world complex optimization problems and have received great popularity in the optimization arena.

Particle swarm optimization (PSO) is one among these meta-heuristic methods. It is inspired by the idea of the flight of birds and the way that they flock to find food that they have no previous knowledge of its location. Its population is called a swarm and each individual is called a particle [1]. Due to the features of simple implementation and rapid convergence, PSO has been widely used in a variety of optimization problems [2]–[4].

Similar to other population-based algorithms, several parameters, e.g., the initial weight and acceleration coefficients, are used to control the performance of PSO. However, a proper parameter setting is always required if one wants to achieve a desired optimization result, and this problem appears to be much more difficult to tackle for those who apply PSO to real-world optimization problems but are lack of experience in parameter settings. Another problem is that the global convergence of PSO cannot always be guaranteed because the diversity of population decreases with the evolution process [5].

To address the above drawbacks, one of recent PSO variants is the introduction of quantum mechanics theories, i.e., quantum-behaved PSO (QPSO) [6], where a quantum model is utilized to depict the state of particles, instead of position and velocity in the standard PSO algorithm. Unlike traditional PSO, there is only one parameter (contraction-expansion coefficient) in QPSO, which helps balance the local and global search capacities during the optimization process. As a consequence, QPSO is easier to implement in comparison with PSO. More importantly, QPSO provides a good method for avoiding getting trapped into local minima too early, thus rendering it a global convergence algorithm, which has been theoretically proved in [7]. In addition, some variants of QPSO have been proposed in order to improve its performance [8]–[10] further. Sun *et al.* [11] proposed a diversity-guided QPSO which imposes a mutation operator on the global best point. Wang and Zhou [12] developed a hybrid QPSO strategy to enhance the search quality by incorporating a local version of QPSO into a main QPSO. In [13], the influence of neighbours on a particle, deriving from a fuzzy membership function, was studied, thereby developing a fuzzy QPSO. Experimental results showed that the fuzzy QPSO could efficiently improve the population diversity. Coelho [14] studied that the chaos mutation operation can diversify the population of QPSO and thus improve its performance.

While existing empirical studies focus on the improvement in the search ability of QPSO, it is worth noting that the convergence speed of QPSO has been unexpectedly slowed down as the above studies try to maintain high diversity in the swarm. Therefore, a good version of QPSO with proper trade-off between fast convergence speed and high optimization precision is expected.

In this paper, we propose a new variant of QPSO, denoted LI-QPSO, which employs the model-based linear interpolation to update the personal best points by replacing the worst with

a much better trial point close to the optimum in the swarm during the evolution process, thereby exerting a beneficial influence on the value of the mean best position in QPSO. This modification helps in not only enhancing the local search ability but also improving the convergence performance and optimization precision. Experimental results show that the proposed method provides significant improvements in comparison with other algorithms on tested benchmark functions.

The rest of this paper is organized as follows. The basic PSO and QPSO algorithms are described in Section II. The improved QPSO algorithm is given in Section III. Experimental results are presented in Section IV. Finally, Section V concludes this paper with some discussions on relevant future work.

II. Quantum-behaved Particle Swarm Optimization (QPSO)

A. Basic PSO Algorithm

Kennedy and Eberhart first developed a PSO algorithm based on the behaviour of individuals, for example particles, in a swarm [1]. In the PSO algorithm, each individual represents a candidate solution to the optimization problem that needs to be solved. The optimum of the problem can be searched by employing the velocity and position of individuals in a form of iteration. In each iteration, the velocity $V_i = (v_1, v_2, \cdots, v_n)^T$ (n is the dimension of the search space) and position $X_i = (x_1, x_2, \cdots, x_n)^T$ of the ith individual respectively are updated towards its personal best position ($pbest$) and the global best position among the swarm ($gbest$) as follows:

$$\begin{aligned} V_i(t+1) = wV_i(t) + c_1R_1(P_i(t) - X_i(t)) \\ +c_2R_2(P_g(t) - X_i(t)) \end{aligned} \quad (1)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (2)$$

where w represents the inertia weight factor, usually set to a constant (e.g., 0.9), c_1 and c_2 are the acceleration constants (commonly $c_1 = c_2 = 2$), and R_1, and R_2 are two uniformly distributed random vector parameters in the range $[0, 1]^n$. The velocity update equation shows that the velocity of an individual at the $(t+1)$-th iteration depends on its previous velocity, the distance that the individual is from its personal best position (P_i) and the distance that the individual is from the global best position (P_g) in the swarm at iteration t.

B. Basic QPSO Algorithm

To overcome the disadvantage that global convergence cannot be guaranteed in PSO, Sun *et al.* proposed a variant of PSO based on quantum mechanics, i.e., QPSO. In QPSO, the particles are considered with quantum behaviour, and their states are depicted by a wave function $\Psi(x, t)$. Each particle only has its position information and no velocity information. A particle updates its position at each iteration as follows:

$$X_i(t+1) = \begin{cases} p+\beta*|mbest - X_i(t)| * ln(1/u), & \text{if } r \geq 0.5; \\ p+\beta*|mbest + X_i(t)| * ln(1/u), & \text{if } r < 0.5. \end{cases} \quad (3)$$

$$p = \alpha * P_i + (1-\alpha) * P_g \quad (4)$$

$$mbest = \frac{1}{N}\sum_{i=1}^{N} P_i = (\frac{1}{N}\sum_{i=1}^{N} P_{i1}, \cdots, \frac{1}{N}\sum_{i=1}^{N} P_{in}) \quad (5)$$

where $mbest$ is actually the mean of the $pbest$ positions of all particles, called the mainstream or mean best position. u, r and α are random numbers uniformly distributed in the range $[0, 1]$, respectively. Parameter β is called contraction-expansion coefficient, which helps to balance the local and global search of the algorithm. P_i and P_g are $pbest$ and $gbest$, respectively. N is the population size and n is the dimension of the search space.

III. Proposed QPSO Algorithm

The standard QPSO algorithm has good global search ability at the early stage; however, its drawbacks are manifest as the iteration increases, such as slow convergence speed, and unexpected low convergence precision. To tackle these problems, a model-based linear interpolation scheme is introduced into QPSO and the improved algorithm is denoted LI-QPSO. In the following, we first give a brief introduction of linear interpolation in Section III-A, and then present the LI-QPSO algorithm in details in Section III-B.

A. Linear Interpolation Method

Linear interpolation is also called linear model and can be used in model-based trust region methods [15]. So, it can exploit the smoothness in the objective function and attempt to preserve the convergence properties of its gradient-based counterparts. A model $s \mapsto m_c(x_c + s)$ is established to approximate the objective function as follows:

$$m_c(x_c + s) = f(x_c) + g_c^T s \quad (6)$$

where g_c is a n-dimensional vector to be determined. The model is also required to interpolate f at x_c, as well as at a set Ω_c of n additional sample points, i.e., $m_c(x_c) = f(x_c)$ and $m_c(x_i) = f(x_i)$ for all $x_i \in \Omega_c$. These interpolation conditions can be written as a linear system of equations as follows:

$$g_c^T s_i = f(x_i) - f(x_c), i = 1, \cdots, n \quad (7)$$

where s_i is the displacement from x_c to x_i, i.e., $x_i = x_c + s_i$ ($i = 1, \cdots, n$), x_c and x_n are the best point and worst point, respectively. It then follows from Eq. (7) that the linear model in Eq. (6) is uniquely determined if and only if the $n+1$ sample points $\{x_c\} \cup \Omega_c$ are set such that the set $\{s_i : i = 1, \cdots, n\}$ is linearly independent. Therefore, the vector g_c can be obtained by solving the linear system of Eq. (7).

To compute a new iteration around the current iteration x_c, the model (6) should be minimized with respect to s subject to $\|s\|_2 \leq \rho_c$, where the radius ρ_c is defined by users. Thus, we have

$$\begin{aligned} &\min \quad m_c(x_c + s) = f(x_c) + g_c^T s \\ &\text{s.t.} \quad \|s\|_2 \leq \rho_c \end{aligned} \quad (8)$$

This problem can be easily solved, and the solution is $s^* = -(\rho_c/\|g_c\|)g_c$. Then, a step $s_c = s^*$ is generated to give a new trial point $\hat{x} = x_c + s_c$.

B. QPSO with Linear Interpolation (LI-QPSO)

In this section, we propose a new version of QPSO, i.e., the LI-QPSO, by introducing a local search technique based on the linear interpolation scheme to improve the performance of the basic QPSO algorithm. The main modification in QPSO is that, in each iteration, $n+1$ (n is the dimension of the search space) distinct good points, say $Z = \{z_1, z_2, \cdots, z_{n+1}\}$, are chosen randomly from the current personal best positions in the swarm. After that, the set of $n+1$ distinct points is used to find a model based on linear interpolation, which is to approximate the objective function around the best points in Z. This model is then minimized with respect to a given radius to give a step x_c, which is then added to the best point to generate a new trial point $\hat{x}$ in the swarm. Here, if the trial point $\hat{x}$ obtained from the linear interpolation scheme gives a function value $f(\hat{x})$ that is better than that of the worst of the personal best points, denoted as x_{worst}, then x_{worst} is replaced by $\hat{x}$.

The above idea is exactly in accordance with "survival of the fittest" in Darwinian theory [16]. This modification has an effect of adjusting the value of $mbest$ reasonably, thereby improving the local search ability and convergence performance of the QPSO algorithm. The only constraint in LI-QPSO is that the population size (N) must be larger than the dimension size of the search space, i.e., $N > n+1$. A complete description of LI-QPSO is presented in Algorithm 1.

Algorithm 1 LI-QPSO Algorithm

1: Initialize parameters of N, $Iteration$, ρ_c, and the population.
2: Calculate the fitness of particles and generate $pbest$ and $gbest$
3: Create a set of distinct points (say Z) chosen randomly from N $pbests$;
4: Select the best point in Z as x_c;
5: Update the worst of $pbests$ using the linear interpolation model shown in Eqs. (7) and (8);
6: **for** $t := 1$ to $Iteration$ **do**
7: **for** $i := 1$ to N **do**
8: Compute p and $mbest$ based on Eqs. (4) and (5), respectively;
9: Update $X_i(t)$ with Eq. (3);
10: Evaluate the fitness $f(X_i(t+1))$;
11: Update $pbest$;
12: **end for**
13: Update $gbest$;
14: Create a set of $n+1$ distinct points chosen randomly from N $pbests$;
15: Select the best point in Z as x_c;
16: Update the worst of $pbests$ using the linear interpolation model shown in Eqs. (7) and (8);
17: Update $gbest$;
18: **end for**
19: Output $gbest$.

IV. Experimental Study

A. Experimental Settings

In this section, five benchmark functions, shown in Table I, were used to test the performance of the proposed LI-QPSO algorithm. These functions are commonly used benchmark test functions from the literature of optimization. The first two functions and the Schwefel function 2.22 are unimodal while the rest are multimodal, containing a significant number of local optima. The numerical results of LI-QPSO are compared with the basic PSO and QPSO algorithms. The parameter settings of the three algorithms are described as follows: for PSO, the inertia weight was set to $w = 0.729$ and the acceleration coefficients were set to $c_1 = c_2 = 2$; for QPSO and LI-QPSO, the contraction-expansion coefficient β was recommended from Sun [6] with a linearly decreasing value from 1.0 to 0.5. In LI-QPSO, for our implementation, we calculated ρ_c every time the linear interpolation strategy was called. In particular, ρ_c was found as follows:

$$\rho_c = \min_{1 \le i \le n} |x_n^i - x_c^i| \tag{9}$$

where x_n^i and x_c^i are the ith component of x_n and x_c, respectively. x_c and x_n are the best and worst points respectively as defined in Section III-A. We also restricted ρ_c to $\rho_c \ge 10^{-3}$.

In the numerical experiments, three different dimension sizes, 10, 20 and 30, were tested. Taking into account the constraint of LI-QPSO ($N > n+1$), the population sizes were set to 40, 60 and 80 and the maximum generation was set to 1000, 1500, and 2000 corresponding to the dimensions 10, 20 and 30 for all functions and all three algorithms, respectively. The statistics results of the mean best fitness and standard deviation (St. Dev.) over a total of 30 runs on a computer with an Intel(R) Core(TM)2 Duo CPU E8135 @2.40GHz @2.40GHz, 4.00GB RAM and 64-bit Operating System in the software environment of MATLAB2013a for each experimental setting are reported in Tables II to VI. In all tables, 'N' represents the population size, 'n' represents the dimension, and 'G' represents the maximum allowable number of generations.

In order to determine whether the proposed algorithm produces statistically significant improvements in the optimization results. A t-test with a 95% significance level was performed over pairs of 30 trials regarding the mean best values. In the tables, '1' ('0') supports (rejects) the hypothesis that LI-QPSO has better performance than the compared algorithm.

B. Experimental Results and Analysis

Tables II to VI show the numerical results of the three algorithms on the tested functions. In Table II, the computational results indicate that LI-QPSO can find the optimal point of the Sphere function with a higher precision than PSO and QPSO and the standard deviation also shows that LI-QPSO has better stability due to its local search. The t-test results

TABLE I
THE TESTED BENCHMARK FUNCTIONS

Functions	Formulae	Range	Optimum
Sphere function	$f_1(x) = \sum_{i=1}^{n} x_i^2$	[-100, 100]	0
Rosenbrock function	$f_2(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	[-30, 30]	0
Rastrigrin function	$f_3(x) = \sum_{i=1}^{n} x_i^2 - 10cos(2\pi x_i) + 10)$	[-5.12, 5.12]	0
Schwefel function 2.22	$f_4(x) = \sum_{i=1}^{n} \|x_i\| + \prod_{i=1}^{n} x_i$	[-100, 100]	0
Griewank function	$f_5(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}) + 1$	[-600, 600]	0

TABLE II
COMPARISON RESULTS ON FUNCTION f_1

N	n	G	PSO			QPSO			LI-QPSO	
			t-test	Mean best	St. Dev.	t-test	Mean best	St. Dev.	Mean best	St. Dev.
40	10	1000	1	5.1994e-14	1.8135e-13	1	1.6488e-79	5.0220e-79	2.2444e-88	5.5448e-88
	20	1500	1	0.1183	0.0464	1	5.3733e-49	9.9638e-48	1.8302e-55	4.1184e-55
	30	2000	1	0.9548	0.4024	1	2.3771e-34	2.5671e-34	1.4891e-40	1.3858e-40
60	10	1000	1	4.4869e-27	2.5226e-27	1	1.0500e-97	6.8151e-98	6.8434e-104	1.5221e-103
	20	1500	1	1.1908e-04	1.4598e-04	0	4.1268e-65	1.2496e-67	3.1436e-68	4.3526e-68
	30	2000	1	1.3972e-03	3.5199e-04	1	1.0015e-44	5.6882e-47	1.3786e-50	1.1001e-48
80	10	1000	1	9.8934e-31	4.1503e-31	1	3.0174e-107	2.8467e-104	2.2126e-111	5.3352e-111
	20	1500	1	3.3873e-05	3.7078e-05	1	5.9596e-74	8.4093e-75	4.0731e-77	2.0815e-77
	30	2000	1	9.1224e-04	1.6837e-03	1	1.6524e-50	1.0700e-53	1.1830e-57	2.4015e-58

TABLE III
COMPARISON RESULTS ON FUNCTION f_2

N	n	G	PSO			QPSO			LI-QPSO	
			t-test	Mean best	St. Dev.	t-test	Mean best	St. Dev.	Mean best	St. Dev.
40	10	1000	1	4.4231	2.3470	0	3.2511	1.1883	3.0884	0.7637
	20	1500	1	52.5378	29.1033	1	19.9140	18.1376	16.5226	1.5967
	30	2000	1	1.7999e+02	98.2430	1	54.9156	30.2420	28.0759	0.1783
60	10	1000	1	3.3943	2.1508	1	2.9311	0.4745	1.9758	0.9411
	20	1500	1	19.2339	1.1539	1	18.2442	17.8536	11.1841	3.2835
	30	2000	1	33.0560	3.6313	1	36.3375	26.3983	24.1895	1.7889
80	10	1000	0	1.6959	1.9257	0	1.9715	1.0372	1.3115	0.7765
	20	1500	1	18.4533	0.6591	1	11.9529	4.7639	7.9664	0.6772
	30	2000	1	29.6262	1.6826	1	23.8999	18.3767	21.3158	0.4986

also show that LI-QPSO is significantly better than PSO for all tested dimensions and better than QPSO for all tested dimensions except the combination of $N = 60$ and $n = 10$. As for the Rosenbrock function, whose statistics results are listed in Table III, PSO presents the worst performance in terms of the mean best fitness. However, LI-QPSO achieves excellent performance with a high stability. Judging from the t-test results, the use of linear interpolation seems to provide statistically significant improvements for the 20 and 30 dimensional cases. With regard to the Rastrigrin function in Table IV, LI-QPSO can still obtain the best results among the three algorithms. QPSO ranks the second in optimizing this problem, and PSO gets the worst performance. It means that PSO with quantum behavior has a better search ability than the standard PSO algorithm. The t-test results from Table IV further confirm these conclusions, implying QPSO with linear interpolation is significantly better PSO and QPSO.

In Table V, LI-QPSO achieves again the best results of the mean best fitness and standard deviation on Schwefel problem 2.22, and PSO performs the worst. The 10 and 20 dimensional cases show statistically significant improvements in the performance of LI-QPSO. The numerical results on the last function f_5 are given in Table VI, where we can find that all the three algorithms cannot get an excellent precision towards the optimal value, even though LI-QPSO achieves sort of better results than the other two algorithms. In the cases of $N = 60$, $n = 20$ and $N = 80$, $n = 20$ and $n = 30$, according to Table VI, QPSO fails to give better results than PSO, although QPSO has a strong search ability [7]. For f_5, according to the t-test results, QPSO with linear interpolation seems to have significantly better performance for the 20- and 30-dimensional cases but no statistical difference for the 10-dimensional case.

Figures 1-5 describe the dynamic performance regarding the mean best fitness from 50 to 1000 generations for five functions, where the population size was set to 40 and the dimension was 10. For a better visualization of the optimization process, log-lin plots are drawn in Figures 1 and 4, and a lin-lin plot is drawn in Figure 2 for a partial and enlarged view. As illustrated in Figures 1, 2 and 4, LI-QPSO apparently converges faster to the optimum and gets a higher precision than PSO and QPSO, and QPSO has a faster convergence

TABLE IV
COMPARISON RESULTS ON FUNCTION f_3

N	n	G	PSO			QPSO			LI-QPSO	
			t-test	Mean best	St. Dev.	t-test	Mean best	St. Dev.	Mean best	St. Dev.
40	10	1000	1	20.4961	8.5357	1	3.2879	2.6646	1.9379	1.1874
	20	1500	1	66.8054	11.0462	1	9.8447	2.8617	3.1959	1.8592
	30	2000	1	1.2110e+02	29.4380	1	18.7965	4.9188	7.6020	3.4911
60	10	1000	1	5.3728	2.3545	1	2.5681	1.8227	0.8492	0.7616
	20	1500	1	18.3658	7.4149	1	8.8537	2.3232	1.5415	0.8844
	30	2000	1	33.8217	8.4177	1	17.7346	4.8200	5.2559	3.1343
80	10	1000	1	4.9748	2.4371	1	1.5358	0.7289	0.5768	0.4461
	20	1500	1	18.2640	4.6801	1	10.7128	11.6441	0.8205	0.6394
	30	2000	1	25.6144	5.9568	1	12.6360	2.8932	2.2086	1.0293

TABLE V
COMPARISON RESULTS ON FUNCTION f_4

N	n	G	PSO			QPSO			LI-QPSO	
			t-test	Mean best	St. Dev.	t-test	Mean best	St. Dev.	Mean best	St. Dev.
40	10	1000	1	2.3124e-11	6.6416e-11	1	6.5064e-77	2.0574e-76	1.2455e-86	2.7363e-86
	20	1500	1	0.1544	0.1156	1	5.6298e-46	9.3446e-46	5.2300e-50	1.6167e-49
	30	2000	1	0.9942	1.0506	0	2.3258e-32	4.1158e-32	2.2365e-37	5.1728e-37
60	10	1000	1	9.7147e-25	1.1412e-24	1	4.7024e-95	1.4627e-94	2.3957e-101	7.4797e-101
	20	1500	1	0.0177	0.0287	1	1.5497e-61	3.8374e-61	2.4980e-64	4.8512e-64
	30	2000	1	0.2989	0.1835	1	1.4181e-44	1.6147e-44	9.1514e-48	2.8753e-47
80	10	1000	1	6.5675e-29	1.8672e-28	1	5.3575e-105	1.5900e-104	1.5640e-110	2.6335e-110
	20	1500	1	5.0293e-03	4.6196e-03	1	3.7647e-69	1.1874e-68	2.2150e-74	3.6790e-74
	30	2000	1	0.1556	0.1024	0	1.1054e-53	1.5494e-53	2.8033e-56	5.0096e-56

TABLE VI
COMPARISON RESULTS ON FUNCTION f_5

N	n	G	PSO			QPSO			LI-QPSO	
			t-test	Mean best	St.Dev.	t-test	Mean best	St.Dev.	Mean best	St.Dev.
40	10	1000	1	0.0485	0.0293	0	0.0274	0.0238	0.0221	0.0189
	20	1500	1	0.0632	0.0573	1	0.0495	0.0406	0.0391	0.0189
	30	2000	1	0.1076	0.1237	1	0.0713	0.0461	0.0593	0.0242
60	10	1000	1	0.0272	0.0113	1	0.0259	0.0179	0.0103	0.0128
	20	1500	1	0.0362	0.0207	1	0.0412	0.0494	0.0191	0.0277
	30	2000	1	0.0871	0.0740	0	0.0635	0.0222	0.0572	0.0498
80	10	1000	1	0.0189	0.0154	1	0.0285	0.0203	2.5568e-03	3.3201e-03
	20	1500	1	0.0238	0.0217	1	0.0314	0.0147	7.6753e-03	0.0126
	30	2000	1	0.5485	0.6112	1	0.0448	0.0162	0.0343	0.0229

speed than PSO. However, this trend seems to change in Figures 3 and 5, where PSO appears to get trapped into local minimum (due to premature convergence) at the early stage. While PSO converges to a local optimal point, QPSO and LI-QPSO continue their convergence towards the global optimum and LI-QPSO has an obvious advantage over the QPSO algorithm regarding the convergence speed. In addition, the introduction of linear interpolation into QPSO helps enhance the local search ability so that LI-QPSO can achieve a higher precision, which can be clearly observed in Figures 3 and 5.

The above experimental results show the improvement in the performance of the proposed LI-QPSO algorithm in comparison with PSO and QPSO algorithms. While LI-QPSO uses no diversity-maintenance methods, e.g., mutation operator, during the optimization process, it can still achieve encouraging results in terms of the given benchmark problems. This is probably due to the fact that LI-QPSO employs a strategy to improve the worst of the personal best particles by the

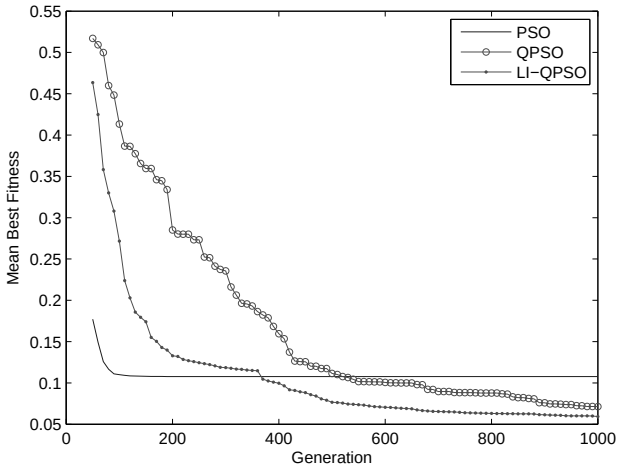

Fig. 1. The convergence characteristics of algorithms on function f_1.

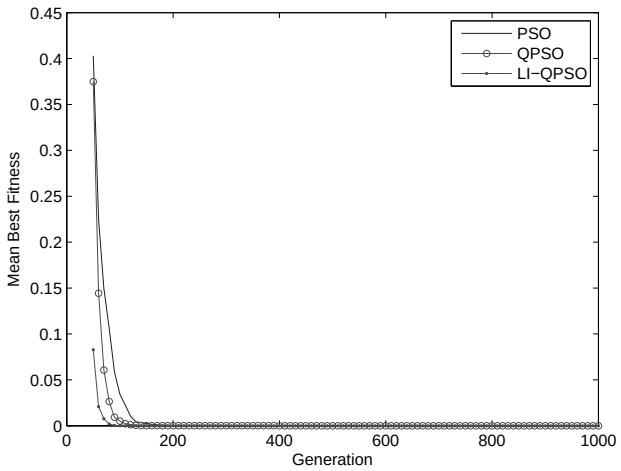

Fig. 2. The convergence characteristics of algorithms on function f_2.

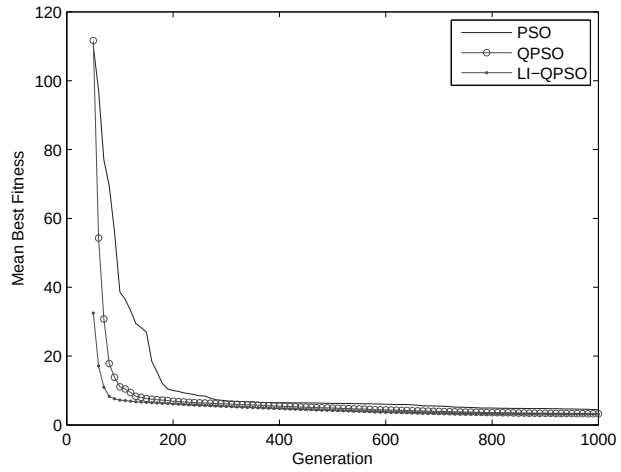

Fig. 3. The convergence characteristics of algorithms on function f_3.

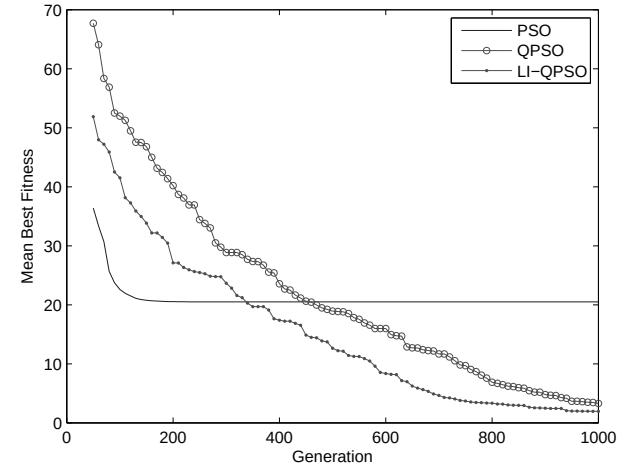

Fig. 4. The convergence characteristics of algorithms on function f_4.

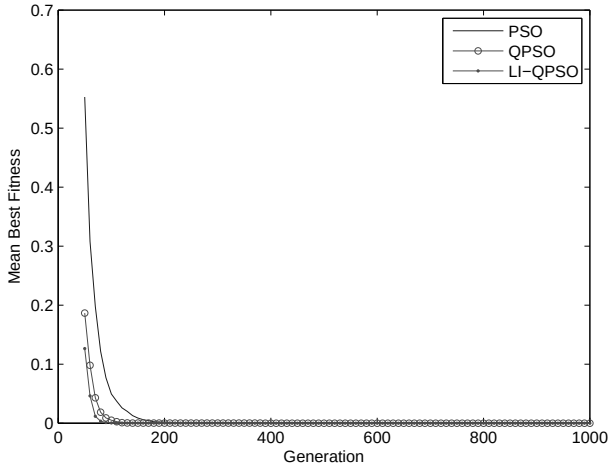

Fig. 5. The convergence characteristics of algorithms on function f_5.

trial point generated by the model-based linear interpolation. Another contributing factor may be that model-based linear interpolation helps QPSO in not only enhancing the local search ability, but also accelerating a particle to converge to its local attractor.

C. Discussions

In this section, we discuss the computational cost and convergence performance of the proposed LI-QPSO.

1) Computational cost: Let G be the maximum generation and N be the population size. In each generation, the linear interpolation model provides a trial point for the population, which requires only one more function evaluation. Thus, the overall time-complexity of LI-QPSO is $O(G(N+1))$, which is slightly higher than that of QPSO ($O(GN)$). Besides, the calculation of g_c in Eq. (7) may be the main time-consuming component of linear interpolation since it is closely associated with the efficiency of solving the linear system of equations. Thus, the processing time of LI-QPSO is higher than that of PSO and QPSO.

2) Convergence performance: As has been illustrated above, the convergence characteristics of LI-QPSO seems to be statistically better than QPSO on five tested functions. One possible reason is that QPSO has strong randomicity so that it may perform excessive exploration and probably miss the chance of exploitation during the evolution. LI-QPSO, however, can save some redundant exploration and improve the exploitation due to the use of linear interpolation. Therefore, LI-QPSO can achieve better convergence performance than QPSO. The analysis also suggests that linear interpolation may be applicable to algorithms with strong randomicity.

V. Conclusions

In this paper, we propose an improved quantum-behaved particle swarm optimization algorithm, denoted LI-QPSO, where the model-based linear interpolation is introduced to enhance local search by generating a trial point to update

the personal best points at present. The proposed LI-QPSO algorithm was tested on several standard benchmark problems and the experimental results were compared with the basic PSO and QPSO algorithms.

The results from 30 runs presented in this paper demonstrated that the proposed LI-QPSO algorithm provides significantly better improvements regarding the convergence performance and precision. However, LI-QPSO is only applied to numerical optimization and has not been extended to real-world engineering optimization problems. Moreover, the completely theoretical analysis of LI-QPSO is also absent in our work. These issues will be our future research on LI-QPSO.

ACKNOWLEDGMENT

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of UK under Grant EP/K001310/1.

REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. 1995 IEEE Conference on Neural Networks*, 1995, pp. 1942–1948.

[2] C. Coello, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[3] X. Li, Q. Yan, and D. Yu, "PEMFC model parameter optimization based on a hybrid PSO algorithm," *Journal of Computational Information Systems*, pp. 479–486, 2011.

[4] Rajni and A. Tayal, "Step size optimization of LMS algorithm using particle swarm optimization algorithm in system identification," *International Journal of Computer Science and Network Security*, vol. 13, no. 6, pp. 125–130, 2013.

[5] F. Van den Bergh, "An analysis of particle swarm optimizers," Ph.D. Thesis, University of Pretoria, November 2001.

[6] J. Sun, B. Feng, and W. B. Xu, "Particle swarm optimization with particles having quantum behavior," in *Proc. 2004 IEEE Congress on Evolutionary Computation*, 2004, vol. 1, pp. 325–331.

[7] J. Sun, X. Wu, V. Palade, W. Fang, C. H. Lai, and W. Xu, "Convergence analysis and improvements of quantum-behaved particle swarm optimization," *Information Sciences*, vol. 193, no. 1800, pp. 81–103, Jun. 2012.

[8] M. Xi, J. Sun, and W. Xu, "An improved quantum-behaved particle swarm optimization algorithm with weighted mean best position," *Applied Mathematics and Computation*, vol. 205, no. 2, pp. 751–759, Nov. 2008.

[9] M. Jamalipour, R. Sayareh, M. Gharib, F. Khoshahval, and M. R. Karimi, "Quantum behaved Particle Swarm Optimization with Differential Mutation operator applied to WWER-1000 in-core fuel management optimization," *Annals of Nuclear Energy*, vol. 54, pp. 134–140, 2013.

[10] D. Tang, Y. Cai, and X. Cai, "Improved quantum-behaved particle swarm optimization algorithm with memory and singal step searching strategy for continuous optimization problems," *Journal of Computational Information Systems*, vol. 9, no. 2, pp. 493–501, 2013.

[11] J. Sun, W. B. Xu, and W. Fang, "A diversity-guided quantum-behaved particle swarm optimization algorithm," *Lecture Notes in Computer Science*, vol. 4287, pp. 497–504, 2006.

[12] J. Wang, and Y. Zhou, "Quantum-behaved particle swarm optimization with generalized local search operator for global optimization," in *Proc. 3rd International Conference on Intelligent Computing, ICIC 2007*, LNCS, vol. 4682, pp. 851–860, 2007.

[13] W. Zhao, Y. San, and H. Shi, "Fuzzy quantum-behaved particle swarm optimization algorithm," in *Proc. 2010 International Symposium on Computational Intelligence and Design*, vol. 1, pp. 49–52, 2010.

[14] L. Coelho, "A quantum particle swarm optimizer with chaotic mutation operator," *Chaos, Solitons and Fractals*, vol. 37, no. 5, pp. 1409–1418, 2008.

[15] M. Marazzi and J. Nocedal, "Wedge trust region methods for derivative free optimization," *Mathematical Programming*, vol. 91, no. 2, pp. 289-305, Jan. 2002.

[16] C. Darwin, *On the Origin of Species by Means of Natural Selection, Or, The Preservation of Favoured Races in the Struggle for Life*, 1st ed. London, UK: John Murray, 1859.