QUANTUM-INSPIRED OPTIMIZATION APPROACH FOR ENGINEERING DESIGN

Leandro dos Santos Coelho
Industrial and Systems Engineering Graduate Program, PPGEPS
Pontifical Catholic University of Paraná
Rua Imaculada Conceição, 1155
80215-901, Curitiba, Paraná, Brazil
leandro.coelho@pucpr.br

Abstract. *Optimization problems are widely encountered in various fields of mechanical engineering. Sometimes such problems can be very complex due to the actual and practical nature of the objective function or the model constraints. During the history of science of computational intelligence, many evolutionary algorithms and swarm intelligence approaches were proposed having more or less success in solving various mechanical engineering optimization problems. In this context, the Particle Swarm Optimization (PSO) is a bio-inspired optimization mechanism based on the metaphor of social behavior of birds flocking and fish schooling in search for food. In PSO each member is seen as a particle, and each particle is a potential solution to the problem under analysis. In this context, each particle which moves through the space of the problem has a randomized velocity associated to it. Inspired by the classical PSO method and quantum mechanics theories, this work presents a new Quantum-behaved PSO (QPSO) approach using Gaussian probability distribution function. The simulation results demonstrate good performance of the QPSO approaches in solving a benchmark problem of tension/compression spring design.*

Keywords*: optimization, particle swarm optimization, mechanical design, quantum computing, Gaussian distribution.*

1. INTRODUCTION

Global optimization problems over continuous spaces arise throughout the scientific community in engineering design. The global optimization of nonlinear, non-convex and non-differential functions is still an open challenge for researchers since an analytical optimal solution is difficult to obtain even for relatively simple application problems.

In the recent four decades, many modern heuristic techniques have been proposed to meet this challenge. These methods are more flexible than the classical ones in dealing with a wide spectrum of problems because they do not require function properties such as differentiability or continuity any more. The well known heuristic search methods include genetic algorithms (Michalewicz, 1992), simulated annealing (Aarts and Korst, 1989), Nelder Mead simplex method (Wright, 1995), evolution strategies (Gomes and Saavedra, 2002), evolutionary programming (Fogel, 1994), differential evolution (Storn and Price, 1995), particle swarm optimization (PSO) (Kennedy *et al*., 2001), immune systems (Castro and Timmis, 2002), and others.

In this context, PSO is a stochastic population (a group of possible solutions to a problem) based optimization algorithm, firstly introduced by Kennedy and Eberhart in 1995 (Kennedy and Eberhart, 1995; Eberhart and Kennedy, 1995). PSO has shown to be an efficient, robust and simple optimization through individual improvement plus population cooperation and competition, which is based on the simulation of simplified social models, such as bird flocking, fish schooling, and the swarming theory. Nowadays PSO has gained much attention and wide applications in various fields (Kennedy *et al*., 2001).

In PSO algorithm, each member of the population is called a ‘‘particle’’, and each particle ‘‘flies’’ around in the multidimensional search space with a velocity, which is constantly updated by the particle’s own experience and the experience of the particle’s neighbours or the experience of the whole swarm. PSO starts with the random initialization of a swarm of particles in the search space and works on the social behavior of the particles in the swarm. Therefore, it finds the global best solution by simply adjusting the trajectory of each individual towards its own best location and towards the best particle of the swarm at each time step. However, the trajectory of each individual in the search space is adjusted by dynamically altering the velocity of each particle, according to its own flying experience and the flying experience of the other particles in the search space (Pan *et al*., 2006).

The advantages of PSO are that it is rapidly converging towards an optimum, simple to compute and easy to implement. However, PSO does exhibits some disadvantages: it sometimes is easy to be trapped in local optima, and the convergence rate decreased considerably in the later period of evolution; when reaching a near optimal solution, the algorithm stops optimizing, and thus the accuracy the algorithm can achieve is limited.

Recently, the concepts of quantum mechanics and physics have motivated the generation of optimization methods, see Hogg and Portnov (2000), Protopescu and Barhen (2002) and Bulger *et al*. (2003). Inspired by the particle swarm optimization (PSO) and quantum mechanics theories, this work presents a new Quantum-behaved PSO (QPSO) approach with Gaussian operator (G-QPSO) to solve the optimization problems over continuous spaces. The optimization of engineering design evaluated here is the design of a tension/compression spring. This problem consists of minimizing the weight of a tension/compression spring subject to constraints on minimum deflection, shear stress, surge frequency, limits on outside diameter and on design variables. The design variables are the mean coil diameter, the wire diameter and the number of active coils.

The rest of the paper is organized as follows: section 2 describes the features of classical PSO for continuous optimization, while section 3 explains the QPSO method. Section 4 describes the procedure of constraints handling based on penalties. Section 5 then details a case study of engineering design and the simulation results are presented. Lastly, section 6 outlines the conclusion and future research.

2. CLASSICAL PARTICLE SWARM OPTIMIZATION

The proposal of PSO algorithm was put forward by several scientists who developed computational simulations of the movement of organisms such as flocks of birds and schools of fish. Such simulations were heavily based on manipulating the distances between individuals, i.e., the synchrony of the behavior of the swarm was seen as an effort to keep an optimal distance between them.

In theory, at least, individuals of a swarm may benefit from the prior discoveries and experiences of all the members of a swarm when foraging. The fundamental point of developing PSO is a hypothesis in which the exchange of information among creatures of the same species offers some sort of evolutionary advantage. PSO is a relatively new population-based heuristic optimization technique. It has been widely applied to optimization problems for simplicity and capability of finding fairly good solutions rapidly.

PSO is basically developed through simulation of bird flocking in two-dimension space (see Figure 1). The position of each agent (particle) is represented by *XY* axis position and the velocity is expressed by v_x (the velocity of *X* axis) and v_y (the velocity of *Y* axis). Modification of the agent position is realized by the position and velocity information (El-Zonkoly, 2006).

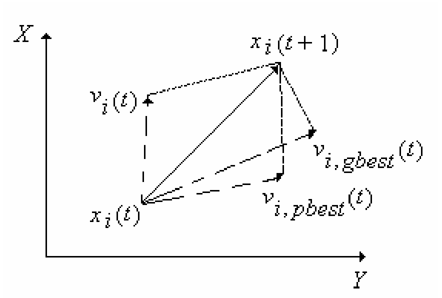

Figure 1. Concept of modification of a searching point of a particle by PSO.

In general terms, the PSO approach represents each potential solution of the target function by particles directly. Each particle has its own position and velocity, with initial values set randomly. Then, all particles ‘‘fly’’ through the solution space and update their positions until they find the optimal solution. During this iterative process, each particle’s velocity is adjusted according to its own experience and social cooperation (Xiang *et al.*, 2007).

The procedure for implementing the global version of PSO is given by the following steps:

Step 1. *Initialization of swarm positions and velocities*: Initialize a population (array) of particles with random positions and velocities in the *n* dimensional problem space using uniform probability distribution function.

Step 2. *Evaluation of particle’s fitness*: Evaluate each particle’s fitness value.

Step 3. *Comparison to pbest (personal best)*: Compare each particle’s fitness with the particle’s *pbest*. If the current value is better than *pbest*, then set the *pbest* value equal to the current value and the *pbest* location equal to the current location in *n*-dimensional space.

Step 4. *Comparison to gbest (global best)*: Compare the fitness with the population’s overall previous best. If the current value is better than *gbest*, then reset *gbest* to the current particle’s array index and value.

Step 5. *Updating of each particle’s velocity and position*: Change the velocity, v_i, and position of the particle, x_i, according to equations (1) and (2):

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot ud \cdot (p_i(t) - x_i(t)) + c_2 \cdot Ud \cdot (p_g(t) - x_i(t)) \tag{1}$$

$$x_i(t+1) = x_i(t) + \Delta t \cdot v_i(t+1) \tag{2}$$

where w is the inertia weight; i=1,2,…,N indicates the number of particles of population (swarm); t=1,2,…t_{max} indicates the iterations, $v_i = [v_{i1}, v_{i2}, ..., v_{in}]^T$ stands for the velocity of the i-th particle, $x_i = [x_{i1}, x_{i2}, ..., x_{in}]^T$ stands for the position of the i-th particle of population, and $p_i = [p_{i1}, p_{i2}, ..., p_{in}]^T$ represents the best previous position of the i-th particle. Positive constants c_1 and c_2 are the cognitive and social components, respectively, which are the acceleration constants responsible for varying the particle velocity towards *pbest* and *gbest*, respectively. Index g represents the index of the best particle among all the particles in the swarm. Variables *ud* and *Ud* are two uniformly-distributed random numbers between 0 and 1. Equation (2) represents the position update, according to its previous position and its velocity, considering $\Delta t = 1$.

Step 6. *Repeating the evolutionary cycle*: Return to **Step 2** until a stop criterion is met, usually a sufficiently good fitness or a maximum number of iterations (generations).

3. QUANTUM-BEHAVED PARTICLE SWARM OPTIMIZATION

Many researchers have devoted to improving PSO performance in various ways and developed many interesting variations. In terms of classical mechanics, a particle is depicted by its position vector x_i and velocity vector v_i, which determine the trajectory of the particle. The particle moves along a determined trajectory in Newtonian mechanics, but this is not the case in quantum mechanics. In quantum world, the term *trajectory* is meaningless, because x_i and v_i of a particle can not be determined simultaneously according to uncertainty principle. Therefore, if individual particles in a PSO system have quantum behavior, the PSO algorithm is bound to work in a different fashion (Sun *et al*., 2004; Sun *et al*., 2005).

In the quantum model of a PSO called here QPSO, the state of a particle is depicted by wavefunction $y(x, t)$ (Schrödinger equation) (Schweizer, 2001; Levin, 2002), instead of position and velocity. The dynamic behavior of the particle is widely divergent form that of that the particle in classical PSO systems in that the exact values of x_i and v_i cannot be determined simultaneously. In this context, the probability of the particle's appearing in position x_i from probability density function $|y(x,t)|^2$, the form of which depends on the potential field the particle lies in Liu *et al*. (2005).

Employing the Monte Carlo method, the particles move according to the following iterative equation (Sun *et al.*, 2004; Sun *et al.*, 2005; Liu *et al.*, 2005):

$$\begin{cases} x_{i,j}(t+1) = p_j(t) + ß \cdot \left| Mbest_j(t) - x_{i,j}(t) \right| \cdot ln(1/u), \text{if } k \geq 0.5 \\ x_{i,j}(t+1) = p_j(t) - ß \cdot \left| Mbest_j(t) - x_{i,j}(t) \right| \cdot ln(1/u), \text{if } k < 0.5 \end{cases} \tag{3}$$

where b is a design parameter called contraction-expansion coefficient (Sun *et al*., 2005); u and k are values generated according to a uniform probability distribution in range [0,1].

The global point called *Mainstream Thought* or *Mean Best* (*Mbest*) of the population is defined as the mean of the *pbest* positions of all particles and it given by

$$Mbest_j(t) = \frac{1}{N} \sum_{j=1}^{N} p_{i,j}(t) \tag{4}$$

where N represents the population size of particles. In this case, the local attractor (Clerc and Kennedy, 2002) to guarantee convergence of the PSO presents the following coordinates:

$$p_j(t) = \frac{c_1 \cdot p_{k,j} + c_2 \cdot p_{g,j}}{c_1 + c_2} \tag{5}$$

where c_1 and c_2 are the cognitive and social components, respectively; $p_{k,j}$ (*pbest*) represents the best previous j-th dimension of the k-th particle and $p_{g,i}$ represents the j-th dimension of the best particle (*gbest*) of the population.

The procedure for implementing the QPSO is given by the following steps (Coelho, 2007a, 2007b):

Step 1. *Initialization of swarm positions*: Initialize a population (array) of particles with random positions in the n dimensional problem space using a uniform probability distribution function.

Step 2. *Evaluation of particle's fitness*: Evaluate the fitness value of each particle.

Step 3. *Comparison to pbest (personal best)*: Compare each particle's fitness with the particle's *pbest*. If the current value is better than *pbest*, then set the *pbest* value equal to the current value and the *pbest* location equal to the current location in *n*-dimensional space.

Step 4. *Comparison to gbest (global best)*: Compare the fitness with the population's overall previous best. If the current value is better than *gbest*, then reset *gbest* to the current particle's array index and value.

Step 5. *Updating of global point*: Calculate the *Mbest* using equation (4).

Step 6. *Updating of particles' position*: Change the position of the particles where c_1 and c_2 are two random numbers generated using a uniform probability distribution in the range [0, 1].

Step 7. *Repeating the evolutionary cycle*: Loop to **Step 2** until a stop criterion is met, usually a sufficiently good fitness or a maximum number of iterations (generations).

3.1 QPSO with Gaussian operator

Various approaches using Gaussian, Cauchy and exponential probability distributions to generate random numbers to updating the velocity equation of classical PSO have been proposed (Kennedy, 2003; Secrest and Lamont, 2003; Krohling and Coelho, 2006). In this paper, following the same line of study, we present novel operator in QPSO using random numbers based on Gaussian probability distribution.

The design of methods to improve the convergence of QPSO is a challenging issue in the continuous optimization applications. A novel QPSO approach based on Gaussian distribution is proposed here.

Generating random numbers using Gaussian distribution sequences with zero mean and unit variance for the stochastic coefficients of QPSO may provide a good compromise between the probability of having a large number of small amplitudes around the current points (fine tuning) and a small probability of having higher amplitudes, which may allow particles to move away from the current point and escape from local minima.

In this work, firstly, random numbers are generated using the absolute value |·| of the Gaussian probability distribution with zero mean and unit variance, i.e., $|N(0,1)|$, and then mapped to a truncated signal given by $G=0.33\cdot|N(0,1)|$. The QPSO approach combined with Gaussian mutation operator (G-QPSO) is described as follows:

$$\begin{cases} x_{i,j}(t+1) = p_j(t) + ß \cdot \left| Mbest_j(t) - x_{i,j}(t) \right| \cdot ln(1/G_u), \text{if } k \geq 0.5 \\ x_{i,j}(t+1) = p_j(t) - ß \cdot \left| Mbest_j(t) - x_{i,j}(t) \right| \cdot ln(1/G_u), \text{if } k < 0.5 \end{cases} \tag{6}$$

$$p_j(t) = \frac{G_1 \cdot p_{k,j} + G_2 \cdot p_{g,j}}{G_1 + G_2} \tag{7}$$

where G_u, G_1 and G_2 are signals given by $0.33\cdot|N(0,1)|$. In this proposed G-QPSO approach, the parameter *u* of equation (3) is modified by the equation (6) and the parameters c_1 and c_2 of equation (5) are modified by the equation (7).

4. A PENALTY METHOD FOR CONSTRAINTS-HANDLING

The search space in constrained optimization problems consists of two kinds of solutions: feasible and unfeasible. Feasible points satisfy all the constraints, while unfeasible solutions violate at least one of them. Therefore, the solution or set of solutions obtained as the final result of an optimization method must necessarily be feasible, i.e., they must satisfy all constraints.

During the past decade, optimization algorithms combining evolutionary computation and constraint-handling techniques have shown to be effective to solve constrained optimization problems. For constrained optimization, the penalty function method has been regarded as one of the most popular constraint-handling technique so far, whereas its drawback lies in the determination of suitable penalty factors, which greatly weakens the efficiency of the method (Himmelblau, 1972; Rao, 1996). In this case, the constrained problem is transformed into an unconstrained one by penalizing the constraints and building a single objective function, which in turn is minimized using an unconstrained optimization algorithm. Over the last few decades, several methods have been proposed for constraints handling in evolutionary algorithms and swarm intelligence approaches. These methods can be grouped into four categories:

methods that preserve the feasibility of solutions, penalty-based methods, methods that clearly distinguish between feasible and unfeasible solutions, and hybrid methods (Michalewicz and Schoenauer, 1996).

In this work, PSO, QPSO and G-QPSO handle the constraints using the concept of stationary penalty functions (that penalize infeasible solutions). That is, one tries to solve an unconstrained problem in the search space *S* using a modified fitness function, such as:

$$eval(x) = \begin{cases} f(x), & \text{if } x \in F \\ f(x) + penalty(x), & \text{otherwise} \end{cases} \tag{8}$$

where *penalty(x)* is zero if no constraint is violated, and is positive otherwise. The penalty function is usually based on a distance measured to the nearest solution in the feasible region *F* or on the effort to repair the solution.

The methodology proposed for constraint handling is divided into two steps. The purpose of the first step is to find solutions for the decision variables that lie within user-defined upper (*ub*) and lower (*lb*) bounds, i.e., $x \in [ub, lb]$. Whenever a lower bound or an upper bound restriction is not satisfied, a repair rule is applied, according to equations (9) and (10), respectively:

$$x_i = x_i + w \cdot rand[0,1] \cdot \{ub(x_i) - lb(x_i)\} \tag{9}$$

$$x_i = x_i - w \cdot rand[0,1] \cdot \{ub(x_i) - lb(x_i)\} \tag{10}$$

where $w \in [0,1]$ is a user-defined parameter and $rand[0,1]$ is a uniformly distributed random value between 0 and 1. In this work, a *w* equals to 0.005 is adopted.

In the second step decision variables are considered inequalities ($g_i(x) \leq 0$). In this work we minimize the objective function, so the equation of the objective function is rewritten as:

$$eval(x) = \begin{cases} f(x), & \text{when } g_i(x) \leq 0 \\ f(x) + r \cdot q \cdot \sum_{i=1}^{n} g_i(x), & \text{when } g_i(x) > 0 \end{cases} \tag{11}$$

where *q* is a positive constant (arbitrarily set to 50,000), *n* is the number of constraints, and *r* is the number of constraints $g_i(x)$ that were not satisfied.

5. A CASE STUDY AND ANALYSIS OF OPTIMIZATION RESULTS

5.1 Description of case study: Minimization of the weight of a tension/compression spring

This problem was described by Arora (1989) and Belegundu (1982) and it consists of minimizing the weight of a tension/compression spring subject to constraints on minimum deflection, shear stress, surge frequency, limits on the outside diameter, and on design variables. The design variables are the mean coil diameter *D* (x_1), the wire diameter *d* (x_2), as shown in Figure 2, and also the number of active coils *N* (x_3).

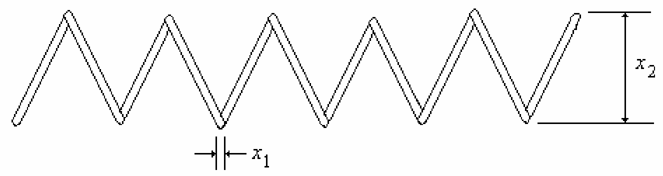

Figure 2. Tension/compression string used in case study.

The design optimization problem involves three continuous variables and four nonlinear inequality constraints, such that

$$\min \mathrm{f}(\mathbf{X}) = (x_3 + 2) \cdot x_1 \cdot (x_2)^2 \tag{12}$$

subject to the following constraints:

$$g_1(\mathbf{X}) = 1 - \frac{x_1^3 x_3}{71{,}785 x_2^4} \leq 0 \tag{13}$$

$$g_2(\mathbf{X}) = \frac{4x_1^2 - x_1 x_2}{12{,}566\left(x_1 x_2^3 - x_2^4\right)} + \frac{1}{5{,}108 x_2^2} - 1 \leq 0 \tag{14}$$

$$g_3(\mathbf{X}) = 1 - \frac{140.45 x_2}{x_1^2 x_3} \leq 0 \tag{15}$$

$$g_4(\mathbf{X}) = \frac{x_1 + x_2}{1.5} \leq 0 \tag{16}$$

The following ranges of the design variable were used: $0.25 \leq x_1 \leq 1.3$, $0.05 \leq x_2 \leq 2.0$, and $2 \leq x_3 \leq 15$ (Ray and Liew, 2003).

5.2 Simulation results

To study the performance of the PSO, QPSO and G-QPSO approaches, an example of well-studied continuous optimization problem of minimizing the weight of a tension/compression spring were solved and the best results obtained through the optimization approaches in 30 trials were compared with those reported in the literature.

PSO, QPSO and G-QPSO methods were implemented using Matlab 6.5 to run on a PC compatible with Pentium IV, a 3.2 GHz processor and 2 GB of RAM. A total of 4,000 cost function evaluations were made with each optimization approach in each run.

In all the experiments, to start the PSO, QPSO and G-QPSO approaches, the setup parameters used were: number of particles (swarm population size): 20, and stop criterion: 200 generations. The classical PSO uses also the inertia weight with a linear reduction equation with initial and final values of 0.7 and 0.4, respectively.

Table 1 summarizes the simulation results obtained by applying the PSO, QPSO and g-QPSO approaches. Table 2 shows the statistics for the 30 independent runs, including the best objective value found, mean, standard deviation, and worst value (maximum) found.

The best mean from the 30 runs was *f*(**X**)=0.0102631 using QPSO. The worst solution (mean of 30 runs) found was *f*(**X**) = 0.0120383 using classical PSO, and the best solutions obtained were *f*(**X**)= 0.0098832 using G-QPSO. The best result obtained in this study by G-QPSO for the case study is compared with results reported in the literature and are presented in Table 3. The best solution from the 30 runs obtained by G-QPSO is better than the best solution previously reported in the literature for five other optimization approaches.

Table 1. Results of optimization approaches.

Optimization Method Tested Approaches	Mean Time of Each Run (in sec)	Objective Function in 30 Runs				
		Worst (Maximum)	Best (Minimum)	Mean	Median	Standard Deviation
PSO	0.7	0.0120383	0.0099499	0.0108238	0.010661	0.000547
QPSO	1.1	0.0109443	0.0098971	**0.0102631**	**0.010202**	0.000264
G-QPSO	1.3	**0.0109388**	**0.0098832**	0.0103043	0.010284	**0.000260**

Table 2. Best results of the PSO, QPSO and G-QPSO approaches.

Design Variables	Parameters	PSO	QPSO	G-QPSO
x_1	*D*	0.0500000	0.0500000	0.0500000
x_2	*d*	0.3724629	0.373906	0.3744313
x_3	*N*	8.6854992	8.587813	8.5586635
g_1	eq. (13)	$-2.9980 \cdot 10^{-4}$	$-5.8819 \cdot 10^{-4}$	$-1.2596 \cdot 10^{-3}$
g_2	eq. (14)	$-5.0166 \cdot 10^{-3}$	$-1.3422 \cdot 10^{-3}$	$-4.9356 \cdot 10^{-5}$
g_3	eq. (15)	-4.8282	-4.8490	-4.8531
g_4	eq. (16)	-0.7184	-0.7174	-0.7171
f(**X**)	eq. (12)	0.0099499	0.0098971	**0.0098832**

Table 3. Comparison of results for the case study.

Design Variables	Parameters	Arora (1989)	Belengundu (1982)	Coello (2000)	Ray and Saini (2001)	Ray and Liew (2003)	Proposed G-QPSO
x_1	D	0.053396	0.0500000	0.051480	0.050417	0.0521602	0.0500000
x_2	d	0.399180	0.3159000	0.351661	0.321532	0.3681587	0.3744313
x_3	N	9.185400	14.250000	11.632201	13.979915	10.648442	8.5586635
g_1	eq. (13)	0.000019	$-1.2672\cdot10^{-3}$	$-3.3366\cdot10^{-3}$	$-1.925\cdot10^{-3}$	$-7.4527\cdot10^{-9}$	$-1.2596\cdot10^{-3}$
g_2	eq. (14)	-0.000018	-0.1490	-0.1357	-0.1556	-0.1314	$-4.9356\cdot10^{-5}$
g_3	eq. (15)	-4.123842	-3.9383	-4.0263	-3.8994	-4.0758	-4.8531
g_4	eq. (16)	-0.698283	-0.7561	-0.7312	-0.7520	-0.7198	-0.7171
f(**X**)	eq. (12)	0.0127302737	0.01273027	0.0127047834	0.0130602	0.012669249	**0.0098832**

6. CONCLUSION AND FUTURE RESEARCH

Inspired by the PSO and quantum mechanics theories, this work presents a new G-QPSO approach with Gaussian operator to solve a constrained optimization problem. Simulated experiments for the optimization of a mechanical design show that the G-QPSO approach displays a satisfactory performance, with respect to both the quality of its evolved solutions and the computational requirements.

The new G-QPSO method improves the performance of both convergence and results' precision to solving a well-studied continuous optimization problem of minimizing the weight of a tension/compression spring. The potential performance of G-QPSO indicates that it could be a viable optimization approach for nonlinear optimization.

The aim of future research is to investigate the use of G-QPSO combined with other computational intelligence methodologies, such as fuzzy systems and artificial neural networks for constrained nonlinear optimization problems.

7. ACKNOWLEDGEMENTS

This work was supported by the National Council of Scientific and Technologic Development of Brazil — CNPq — under Grant 309646/2006-5/PQ. The author would also like to thank the reviewers for their insightful comments and suggestions.

8. REFERENCES

Aarts, E. and Korst, J., 1989, "Simulated Annealing and Boltzmann Machines", John Wiley & Sons, Hoboken, NJ, USA.

Arora, J.S., 1989, "Introduction to Optimum Design", MGraw-Hill, New York, NY, USA.

Belengundu, A.D., 1982, "A Study of Mathematical Programming Methods for Structural Optimization", Department of Civil and Environmental Engineering, University of Iowa, Iowa, USA.

Bulger, D., Baritompa, W.P. and Wood, G.R., 2003, "Implementing Pure Adaptive Search with Grover's Quantum Algorithm", Journal of Optimization: Theory and Applications, Vol. 116, No. 3, pp. 517-529.

Clerc, M. and Kennedy, J.F., 2002, "The Particle Swarm: Explosion, Stability and Convergence in a Multi-Dimensional Complex Space", IEEE Transactions on Evolutionary Computation, Vol. 6, No. 1, pp. 58-73.

Coelho, L. S., 2007a, "Novel Gaussian Quantum-behaved Particle Swarm Optimiser applied to Electromagnetic Design", IET Science, Measurement, and Technology, Vol. 1, No. 5, pp. 290-294.

Coelho, L.S., 2007b, "A Quantum Particle Swarm Optimizer with Chaotic Mutation Operator", Chaos, Solitions, and Fractals (accepted to future publication).

Coello, C.A.C., 2000, "Use of a Self-Adaptive Penalty Approach for Engineering Optimization Problem", Computers in Industry, Vol. 4, No. 2, pp. 113-127.

De Castro, L.N. and Timmis, J., 2002, "An Artificial Immune Network for Multimodal Function Optimization, Proceedings of IEEE Congress on Evolutionary Computation, Hawaii, HI, USA, pp. 699-674.

Eberhart, R.C. and Kennedy, J.F., 1995. "A New Optimizer Using Particle Swarm Theory", Proceedings of International Symposium on Micro Machine and Human Science, Japan, pp. 39-43.

El-Zonkoly, A.M., 2006, "Optimal Tuning of Power Systems Stabilizers and AVR Gains Using Particle swarm Optimization", Expert Systems with Applications, Vol. 31, No. 3, pp. 551-557.

Fogel, D.B., 1994, "Applying Evolutionary Programming to Selected Control Problems", Computers & Mathematics with Applications, Vol. 27, No. 11, pp. 89-104.

Gomes, J.R. and Saavedra, O., 2002, “A Cauchy-based Evolution Strategy for Solving the Reactive Power Dispatch Problem”, International Journal of Electrical Power Energy Systems, Vol. 24, No. 4, pp. 277-283.

Himmelblau, D.M., 1972, “Applied Nonlinear Programming”, McGraw-Hill, New York, NY, USA.

Hogg, T. and Portnov, D.S., 2000, “Quantum Optimization”, Information Sciences, Vol. 128, No. 3-4, pp. 181-197.

Kennedy, J.F., 2003, “Bare Bones Particle Swarms”, Proceedings of the IEEE Swarm Intelligence Symposium, Indianapolis, IN, USA, pp. 80-87.

Kennedy, J.F., Eberhart, R.C. and Shi, Y., 2001. “Swarm Intelligence”, Morgan Kaufmann Pub, San Francisco, CA, USA.

Kennedy, J.F. and Eberhart, R.C., 1995. “Particle Swarm Optimization”, Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942-1948.

Krohling, R. and Coelho, L.S., 2006, “Coevolutionary Particle Swarm Optimization Using Gaussian Distribution for Solving Constrained Optimization Problems”, IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, Vol. 36, No. 6, pp. 1407-1416.

Levin, F.S., 2002, “An Introduction to Quantum Theory”, Cambridge University Press.

Liu, J., Xu, W. and Sun, J., 2005, “Quantum-behaved Particle Swarm Optimization with Mutation Operator”, Proceedings of 17th International Conference on Tools with Artificial Intelligence, Hong Kong, China.

Michalewicz Z. and Schoenauer, M. (1996). “Evolutionary Algorithms for Constrained Parameter Optimization Problems”, Evolutionary Computation, Vol. 4, No. 1, pp. 1-32.

Michalewicz, Z., 1992, “Genetic Algorithms + Data Structures = Evolution Programs”, Springer-Verlag, AI Series, New York, NY, USA.

Pan, H., Wang, L. and Liu, B., 2006, “Particle Swarm Optimization for Function Optimization in Noisy Environment”, Applied Mathematics and Computation, Vol. 181, No. 2, pp. 908-919.

Protopescu, V. and Barhen, J., 2002, “Solving a Class of Continuous Global Optimization Problems Using Quantum Algorithms”, Physics Letters A, Vol. 296, No. 1, pp. 9-14.

Rao, S.S., 1996, “Engineering Optimization”, John Wiley & Sons, Hoboken, NJ, USA, 3rd edition.

Ray, T. and Liew, K.M., 2003, “Society and Civilization: An Optimization Algorithm Based on the Simulation of Social Behavior”, IEEE Transactions on Evolutionary Computation, Vol. 7, No. 4, pp. 386-396.

Ray, T. and Saini, P., 2001, “Engineering Design Optimization Using a Swarm with an Intelligent Information Sharing Among Individuals”, Engineering Optimization, Vol. 33, No. 3, pp. 735-748.

Schweizer, W., 2001, “Numerical Quantum Dynamics”, Hingham, MA, USA.

Secrest, B.R. and Lamont, G.B., 2003, “Visualizing Particle Swarm Optimization - Gaussian Particle Swarm Optimization”, Proceedings of the IEEE Swarm Intelligence Symposium, Indianapolis, IN, USA, pp. 198-204.

Storn, R. and Price, K., 1995, “Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces”, Technical Report TR-95-012, International Computer Science Institute, Berkeley, CA, USA.

Sun, J., Feng, B. and Xu, W., 2004, “Particle Swarm Optimization with Particles Having Quantum Behavior”, Proceedings of Congress on Evolutionary Computation, Portland, Oregon, USA, pp. 325-331.

Sun, J., Xu, W. and Feng, B., 2005, “Adaptive Parameter Control for Quantum-behaved Particle Swarm Optimization on Individual Level”, Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Big Island, HI, USA, pp. 3049-3054.

Wright, M.H., 1995, “Direct Search Methods: Once Scorned, Now Respectable”, in D. F. Griffiths and G. A. Watson (eds.), Proceedings of the Dundee Biennial Conference on Numerical Analysis, Addison-Wesley Longman, Harlow, UK, pp. 191-208.

Xiang, T., Wong, K.-W. and Liao, X., 2007, “A Novel Particle Swarm Optimizer with Time-delay”, Applied Mathematics and Computation, Vol. 186, No. 1, pp. 789-793.

9. RESPONSIBILITY NOTICE

The author is the only responsible for the printed material included in this paper.