Particle Swarm Optimization Using Sobol Mutation

Millie Pant[1], Radha Thangaraj[1], Ajith Abraham[2] and Kusum Deep[1]
[1]Indian Institute of Technology Roorkee, India
[2]Machine Intelligence Research (MIR) Labs, Scientific Network for Innovation and Research Excellence (SNIRE), USA
e-mail: millifpt@iitr.ernet.in, t.radha@ieee.org, ajith.abraham@ieee.org, kusumfma@iitr.ernet.in

***Abstract*— Particle Swarm Optimization (PSO) is an evolutionary computation technique based on Swarm Intelligence. The PSO algorithm is simple in concept, easy to implement and computationally efficient. Many researchers have worked on improving its performance in various ways and have developed several interesting variants. In this paper, we present a new mutation operator called the Systematic Mutation (SM) operator for enhancing the performance of Basic Particle Swarm Optimization (BPSO) algorithm. The SM operator unlike most of its contemporary mutation operators do not use the random probability distribution for perturbing the swarm population, but uses a quasi random Sobol sequence to find new solution vectors in the search domain. The presence of SM operator makes the mutated particles to move systematically in the search space. The comparison of SMPSO is made with BPSO and some other variants of PSO on a set of 15 benchmark global optimization problems and three real life engineering design problems. The empirical results show that SM operator significantly improves the performance of PSO in terms of fitness function value.**

Keywords-Particle Swarm Optimization, Mutatation, Sobol Sequence, Low discrepancy sequence

I. INTRODUCTION

Evolutionary Algorithms (EAs) [1] are a broad class of stochastic optimization algorithms inspired by biological processes and, in particular, by those processes that allow populations of organisms to adapt to their surrounding environments: genetic inheritance and survival of the fittest. EAs have a prominent advantage over other types of numerical methods, among which the two most important ones are [2]:

(i) They can be applied to problems that consist of discontinuous, non-differentiable and non-convex objective functions and/or constraints.
(ii) They can easily escape from local optima.

EAs have been applied to a wide range of functions and real life problems [3] - [6]. Some common EAs are Genetic Algorithms (GA) [7], Evolutionary Programming (EP) [8], Particle Swarm Optimization (PSO) **[9]**, Differential Evolution (DE) [10] etc. All these algorithms have certain properties and special operators associated with them which make them different from each other. For example in Evolutionary Programming, mutation is the primary operator, in GA crossover, mutation and selection are the main operators. Similarly in DE also mutation, crossover and selection are prime operators. In this paper we have concentrated our study on PSO, which do not have any particular evolutionary operator but works on the socio-cooperative species shown by various species.

Particle Swarm Optimization (PSO) is relatively a newer addition to a class of population based search technique for solving numerical optimization problems. Metaphorically, PSO imitates the collective and cooperative behavior of species moving in groups. Some classic examples being a swarm of birds, school of fish, cooperative behavior of ants and bees etc.

In original PSO, developed by Kennedy and Eberhart in 1995 [9], each particle adjusts its position in the search space from time to time according to the flying experience of its own and of its neighbors (or colleagues). The particles or members of the swarm fly through a multidimensional search space looking for a potential solution. Researchers have shown that although PSO finds solutions much faster than most of the contemporary search techniques like Evolutionary and Genetic Algorithms, it usually do not improve the quality of solutions as the number of iterations increase and thus becomes a victim of premature convergence resulting in a suboptimal solution. This problem becomes more persistent in case of highly multimodal problems having several global and local optima. This drawback of PSO is due to the lack of diversity, which forces the swarm particles to converge to the global optimum found so far (after a certain number of iterations), which may not even be a local optimum. Thus without an effective diversity enhancing mechanism the PSO algorithm/ technique is not able to efficiently explore the search space.

One of the methods for maintaining the diversity of the population is inclusion of the concept of mutation (a phenomenon common to EP and GA). Ratnaweera *et al.* [11] state that lack of population diversity in PSO algorithms is understood to be a factor in their convergence on local minima. Therefore, the addition of a mutation operator to PSO should enhance its global search capacity and thus improve its performance. Most of the modern mutation operators defined in literature makes use of random probability distribution. Higashi et al. [12] use a mutation operator that changes a particle dimension value using a random number drawn from a Gaussian distribution. Stacey *et al.* [13] implement a mutation operator similar to that of Higashi et al. [12], but a Cauchy probability distribution is used instead. Esquivel et al. [14] incorporate a mutation operator into PSO that was developed by Michalewicz for use in real-valued Genetic Algorithms in [15]. This is called the Michalewicz's non-uniform mutation operator as the random numbers used to mutate values depends on the current algorithm iteration, with the probability of a value being mutated by a large amount being higher at the start of an optimization run.

In the present work, instead of using the random probability distribution, we have defined a SM operator which uses quasi random (Sobol) sequence for moving the particles of the swarm. The reason behind using quasi random sequence is that quasi random sequences cover the search domain more evenly in comparison to the random probability distributions, thereby increasing the chances of finding a better solution. The SM operator defined in this paper is applied to two versions of BPSO called SMPSO1 and SMPSO2. In SMPSO1, mutation is applied to the global best (gbest) particle, where as in SMPSO2, the worst particle of the swarm is mutated.

The remaining organization of the paper is as follows: Section II gives a brief review of Quasi Random Sequences (QRS) and Sobol Sequence. Section III describes the working of Basic Particle Swarm Optimization. In Section IV, we describe the proposed algorithms, Section V, gives the experimental settings and numerical results of some benchmark problems. Section VI gives three real life problems and their results. The paper finally concludes with Section VII.

II. QUASI RANDOM SEQUENCES (QRS)

QRS or low discrepancy sequences are less random than pseudorandom number sequences, but are more useful for computational methods, which depend on the generation of random numbers. Some of these tasks involve approximation of integrals in higher dimensions, simulation and global optimization. Some well known QRS are: Vander Corput, Sobol, Faure and Halton. These sequences have been applied to initialize the swarm and the numerical results show a marked improvement over the traditional BPSO, which uses uniformly, distributed random numbers [16], [17]**.** The numerical results showed that initializing the population with QRS significantly improve the performance of a population based search algorithms. Motivated by this fact we applied it as a mutation operator where the particles instead of moving randomly, move in a quasi random manner.

A. *Discrepancy of a Sequence*

Mathematically, discrepancy of a sequence is the measure of its uniformity. It is computed by comparing the actual number of sample points in a given volume of a multi-dimensional space with the number of sample points that should be there assuming a uniform distribution defined.

For a given set of points $x^1, x^2, \ldots, x^N \in I^S$ and a subset $G \subset I^S$, define a counting function $S_N(G)$ as the number of points $x^i \in G$. For each $x = (x_1, x_2, \ldots. x_S) \in I^S$, let G_x be the rectangular S dimensional region, such that $G_x = [0,x_1)$ x $[0,x_2)$ x…x$[0,x_S)$, with volume $x_1x_2\ldots x_N$. Then the discrepancy of points is given by $D^*_N(x^1, x^2, x^3 \ldots. x^N) = \text{Sup} \left| S_N(G_x) - N x_1 x_2 \ldots x_S \right|, x \in I^S$.

B. *Sobol Sequence*

The construction of the Sobol sequence [18] uses linear recurrence relations over the finite field, F2, where F2 = {0, 1}. If the binary expansion of the non-negative integer n is given by $n = n_1 2^0 + n_2 2^1 + \ldots.. + n_w 2^{w-1}$. Then the nth element of the jth dimension of the Sobol Sequence, $X_n^{(j)}$, can be generated by:

$$X_n^{(j)} = n_1 v_1^{(j)} \oplus n_2 v_2^{(j)} \oplus \ldots\ldots \oplus n_w v_w^{(j)}$$

where $v_i^{(j)}$ is a binary fraction called the ith direction number in the jth dimension.

These direction numbers are generated by the following q-term recurrence relation:

$$v_i^{(j)} = a_1 v_{i-1}^{(j)} \oplus a_2 v_{i-2}^{(j)} \oplus \ldots \oplus a_q v_{i-q+1}^{(j)} \oplus v_{i-q}^{(j)} \oplus (v_{i-q}^{(j)} / 2^q)$$

We have i > q, and the bit a_i , comes from the coefficients of a degree-q primitive polynomial over F2.

III. BASIC PARTICLE SWARM OPTIMIZATION

In this section we briefly describe the basic PSO. For a D-dimensional search space the position of the ith particle is represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. Each particle maintains a memory of its previous best position $P_i = (p_{i1}, p_{i2} \ldots p_{iD})$. The best one among all the particles in the population is represented as $P_g = (p_{g1}, p_{g2} \ldots p_{gD})$. The velocity of each particle is represented as $V_i = (v_{i1}, v_{i2}, \ldots v_{iD})$. In each iteration, the P vector of the particle with best fitness in the local neighborhood, designated g, and the P vector of the current particle are combined to adjust the velocity along each dimension and a new position of the particle is determined using that velocity. The two basic equations which govern the working of PSO are that of velocity vector and position vector given by:

$$v_{id} = w v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \qquad (1)$$

$$x_{id} = x_{id} + v_{id} \qquad (2)$$

The first part of equation (1) represents the inertia of the previous velocity, the second part is the cognition part and it tells us about the personal thinking of the particle, the third part represents the cooperation among particles and is therefore named as the social component. Acceleration constants c_1, c_2 and inertia weight w are the predefined by the user and r_1, r_2 are the uniformly generated random numbers in the range of [0, 1].

IV. Proposed Versions

The structure of PSO, as discussed in the previous section, is such that due to the fast information flow among the particles, the diversity is soon diminished and as a result it gets stuck in a suboptimal solution. To overcome this shortcoming we included a mutation operator to induce diversity in the population. The mutation operator presented in this article follows Sobol sequence and is called Sobol Mutation (SM).

The proposed SM operator is defined as:

$SM = R_1 + (R_2 / \ln R_1)$, Where R1 and R2 are random numbers in a Sobol sequence.

The proposed versions called SMPSO1 and SMPSO2 are an extension to the Basic Particle Swarm Optimization. These versions differ from each other in the sense that in SMPSO1, the best particle of the swarm is mutated, whereas in SMPSO2, the worst particle of the swarm is mutated. The quasi random numbers used in the SM operator allows the worst particle to move forward systemically and helps in exploring the search space more efficiently. As a result the probability of getting a better solution increases. Besides mutating the best particle of the swarm in SMPSO1, we have also mutated the worst particle in second version SMPSO2. The idea behind applying the mutation to the worst particle is to push the swarm from the back.

A. Computational steps of proposed Algorithms

The proposed algorithms start like the usual PSO algorithm up to the point of evaluating the position and velocity of the particles after which the systematic mutation is applied to produce a perturbation in the population. The computational steps of SMPSO1algorithm are given below.

```
Step1    Initialization.
For each particle i in the population:
Step1.1  Initialize X[i] with uniformly distributed random numbers
Step1.2  Initialize V[i] randomly.
Step1.3  Evaluate the objective function of X[i], and assigned the value to fitness[i].
Step1.4  Initialize $P_{best}[i]$ with a copy of X[i].
Step1.5  Initialize Pbest_fitness[i] with a copy of fitness[i].
Step1.6  Initialize $P_{gbest}$ with the index of the
          particle with the least fitness.
Step2    Repeat until stopping criterion is reached:
For each particle i:
Step 2.1 Update V[i] and X[i] according
          to equations (1) and (2).
Step2.2  Evaluate fitness of X[i].
Step2.3  If $f(X[i]) < f(P_{best}[i])$ then $P_{best}[i] = X[i]$,
Step2.4  if $f(P_{best}[i]) < f(P_{gbest})$ then $P_{gbest} = P_{best}[i]$
Step 3   // Sobol Mutation
         Find a new particle using SM operator
         Let this particle as TX
         $TX = R_1 + ( R_2 / \ln R_1)$
         If $f(TX) < f(P_{gbest})$ then $P_{gbest} = TX$
Step 4   Goto step 2
```

The computational steps of SMPSO2 are same as that of SMPSO1, except for the fact that the worst particle in the swarm is mutated instead of the best particle.

V. Experimental Settings, Benchmark Problems and Numerical Results

A. Experimental Settings

The main parameters of PSO algorithm are inertia weight *w* and acceleration constants c_1 and c_2. After performing a number of experiments and going through various PSO versions available in literature we took the following experimental settings: the inertial weight w is taken to be linearly decreasing from 0.9 to 0.5. Acceleration constants c1 and c2 are taken as 2.0 each and r_1, r_2 are taken as the uniformly distributed random numbers between 0 and 1. The proposed SMPSO has two added parameters R_1 and R_2, which are the random numbers following quasi random Sobol sequence.

For the purpose of comparison of the proposed SMPSO with other variants ([19] and [20]), we have kept the same parameter settings for a fair comparison and have considered the same set of problems as mentioned in literature. Each problem is evaluated for three different dimensions 10, 20 and 30 and size of the swarm is varied as 20, 40 and 80 for each of these population sizes. The stopping criteria is taken as the maximum numbers of generations reached which are 1000, 1500 and 2000 for dimensions 10, 20 and 30 respectively. The corresponding numerical results are given in Tables I, II and III. The proposed algorithms are compared with BPSO for a larger set of problems, where we have tested each problem for dimensions 10, 20 and 30. The size of the swarm is fixed as 20 and the stopping criteria is the maximum number of generations reached, which are 1000, 1500 and 2000 for dimensions 10, 20 and 30 respectively. The corresponding numerical results are given in Table IV.

B. Benchmark problems

In the present study we have taken 15 benchmark problems with box constraints. Mathematical models of the problems along with the true optimum value are given in Appendix. The collection of problems that we have taken though not exhaustive may be considered as a starting point for checking the credibility of any optimization algorithm.

The test bed consists of a variety of problems including multimodal, unimodal and noisy functions. Test functions f_1, f_2, f_5, f_{10}, f_{11}, f_{12}, f_{13}, f_{14} are multimodal in nature, where the complexity of the problem increases with the increase in the number of variables. Function f_6 is a noisy function where a uniformly distributed random noise is added. Due to the presence of this random noise, the optimum of the problem keeps on shifting from one position to another and it becomes a challenge for the global optimization algorithm to locate the optimum value. Functions f_4, f_7, f_8, f_9, f_{15} are unimodal in nature. Function f_4 is a simple sphere function and any decent optimization algorithm can solve it, however on increasing the number of variables sometimes the exact global optimal value is not located. Function f_7 has a discontinuity and function f_9 has plateau type region.

C. Numerical Results

SMPSO1 and SMPSO2 are compared with other versions for test functions $f_1 - f_3$. We have especially chosen these problems because they are common to the versions we have taken for comparison. From the corresponding numerical results we can see that both the proposed versions outperform the other algorithms by a significant difference. If we compare the proposed algorithms with each other we can see that SMPSO2 in which the worst particle is mutated is marginally better than SMPSO1. The performance of the proposed algorithms is further evaluated on a larger set of problems and the numerical results are compared with the BPSO. From the numerical results reported in Table IV, we can easily judge that for almost all the test problems taken in the present study, the proposed algorithms give a better performance in comparison to the BPSO for almost all the cases. The superior performance is more evident when the dimension of the problems is increased up to 30. The convergence graphs of the proposed algorithms with BPSO for all benchmark problems are illustrated in Figure I.

VI. Real life problems and results

The credibility of an optimization algorithm also depends on its ability to solve real life problems. In this paper we took three real life problems to validate the performance of the proposed SMPSO1 and SMPSO2 algorithms. Numerical results for the real life problems are listed in Table V. Numerical results show that in terms of average number of generations required to reach the optimum solution, the proposed SMPSO gave the best results. However in terms of function value all the algorithms gave more or less similar results

Gas transmission compressor design [21]:

Min $f(x) = 8.61\times10^5 \times x_1^{1/2} x_2 x_3^{-2/3} (x_2^2 - 1)^{-1/2}$

$+3.69\times10^4 \times x_3 + 7.72\times10^8 \times x_1^{-1} x_2^{0.219} - 765.43\times10^6 \times x_1^{-1}$

Subject to: $10 \le x_1 \le 55$, $1.1 \le x_2 \le 2$, $10 \le x_3 \le 40$

Optimal thermohydralic performance of an artificially roughened air heater [22]:

Max $L = 2.51*\ln e^+ + 5.5 - 0.1R_M - G_H$

Where $R_M = 0.95x_2^{0.53}$; $GH = 4.5(e^+)^{0.28}(0.7)^{0.57}$;

$e^+ = x_1 x_3 (\bar{f}/2)^{1/2}$; $\bar{f} = (f_s + f_r)/2$; $f_s = 0.079x_3^{-0.25}$;

$f_r = 2(0.95x_3^{0.53} + 2.5*\ln(1/2x_1)^2 - 3.75)^{-2}$;

Subject to: $0.02 \le x_1 \le 0.8$, $10 \le x_2 \le 40$,

$3000 \le x_3 \le 20000$

Optimal capacity of gas production facilities [21]:

Min $f(x) = 61.8 + 5.72x_1 + 0.2623[(40 - x_1)\ln(\frac{x_2}{200})]^{-0.85}$

$+0.087(40 - x_1)\ln(\frac{x_2}{200}) + 700.23x_2^{-0.75}$

Subject to: $x_1 \ge 17.5$, $x_2 \ge 200$; $17.5 \le x_1 \le 40$,

$300 \le x_2 \le 600$.

VII. Conclusion

The purpose of the present article is to study the effect of a newly defined Sobol Mutation (SM) operator in the basic PSO. The SM mutation operator follows quasi random Sobol sequence which allows the mutated particles to move in a systematic manner. We have proposed two versions incorporating the SM operator, called SMPSO1 and SMPSO2 which differ from each other in the application of mutation. In SMPSO1 the best particle of the swarm is mutated while in SMPSO2, the worst particle is mutated The empirical studies on 15 bench mark problems and three real life problems show that the proposed versions are better than the BPSO and other recently modified versions quite significantly. In the present article we have used Sobol sequence, which gave better results than other quasi random sequences like Faure and Vander corput when used for initializing the swarm [16]. However a similar comparative study can be conducted for other quasi random sequences for mutation as well.

References

[1] T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds., *Handbook of EvolutionaryComputation*. New York: Inst. Phys. and Oxford Univ. Press, 1997.

[2] J. Zhang, J. Xu and Q. Zhou, "A New Differential Evolution for Constrained Optimization Problems", In Proc. of the sixth Int. conf. on Intelligent Systems, Design and Applications, 2006, pp. 1018 – 1023.

[3] X. Song, Y. Zhu, C. Yin, and F. Li, "Study on the combination of genetic algorithms and ant Colony algorithms for solving fuzzy job shop scheduling problems", In Proc. of IMACS Multiconference on Computational Engineering in Systems Applications, Vol. 2, 2006, pp. 1904 – 1909.

[4] A. Abbasy and S. H. Hosseini, "A Novel Multi-Agent Evolutionary Programming Algorithm for Economic Dispatch Problems with Non-Smooth Cost Functions, In Proc. of IEEE Power Engineering Society General Meeting, 2007, pp. 1 -7.

[5] C. Guangyi, G. Wei and H. Kaisheng, "On Line Parameter Identification of an Induction Motor Using Improved Particle Swarm Optimization", In Proc. of Chinese Control Conference, 2007, pp. 745 – 749.

TABLE I. COMPARISON OF PROPOSED SMPSO1 AND SMPSO2 VERSIONS WITH BPSO AND OTHER VERSIONS AVAILABLE IN LITERATURE FOR FUNCTION F_1 IN TERMS OF AVERAGE FITNESS FUNCTION VALUE

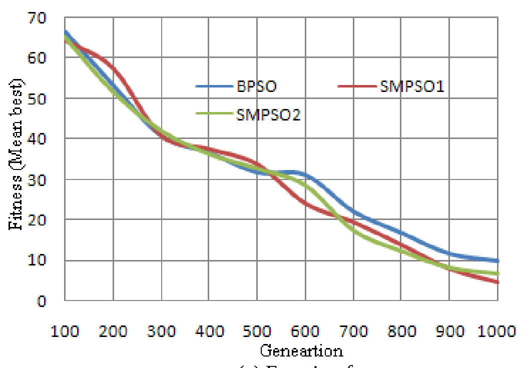
Pop	Dim	Gne	SMPSO1 (gbest)	SMPSO2 (gworst)	BPSO [19]	QPSO [19]	Mutation gbest [20]	Mutation gbest [19]
20	10	1000	0.881465	**0.641812**	5.5382	5.2543	5.2216	4.3976
	20	1500	5.014802	**4.52709**	23.1544	16.2673	16.1562	14.1678
	30	2000	13.152097	**12.669938**	47.4168	31.4576	26.2507	25.6415
40	10	1000	1.241561	**0.85634**	3.5778	3.5685	3.3361	3.2046
	20	1500	5.91223	**5.472557**	16.4337	11.1351	10.9072	9.5793
	30	2000	**13.005205**	14.523385	37.2896	22.9594	19.6360	20.5479
80	10	1000	1.182363	**0.813593**	2.5646	2.1245	2.0185	1.7166
	20	1500	5.501107	**4.97266**	13.3826	10.2759	7.7928	7.2041
	30	2000	**10.210538**	15.028891	28.6293	16.7768	14.9055	15.0393

TABLE II. COMPARISON OF PROPOSED SMPSO1 AND SMPSO2 VERSIONS WITH BPSO AND OTHER VERSIONS AVAILABLE IN LITERATURE FOR FUNCTION F_2 IN TERMS OF AVERAGE FITNESS FUNCTION VALUE

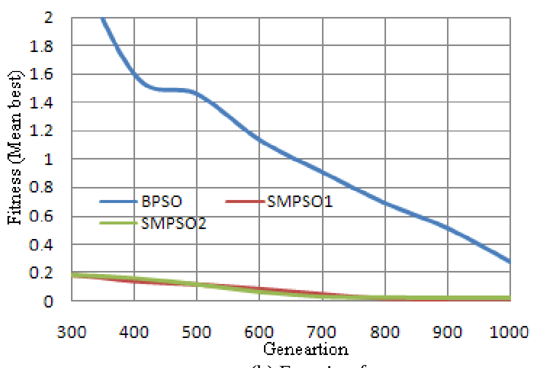
Pop	Dim	Gne	SMPSO1 (gbest)	SMPSO2 (gworst)	BPSO [19]	QPSO [19]	Mutation gbest [20]	Mutation gbest [19]
20	10	1000	**0.006896**	0.007877	0.09217	0.08331	0.0627	0.0780
	20	1500	0.009177	**0.008486**	0.03002	0.02033	0.0209	0.0235
	30	2000	0.025227	**0.014541**	0.01811	0.01119	0.0110	0.0099
40	10	1000	0.009677	**0.009515**	0.08496	0.06912	0.0539	0.0641
	20	1500	0.017195	**0.012269**	0.02719	0.01666	0.0238	0.0191
	30	2000	0.030103	**0.011066**	0.01267	0.01161	0.0119	0.0098
80	10	1000	0.00886	**0.006402**	0.07484	0.03508	0.0419	0.0460
	20	1500	**0.010828**	0.01296	0.02854	0.01460	0.0136	0.0186
	30	2000	0.024265_	**0.004692**	0.01258	0.01136	0.0120	0.0069

TABLE III. COMPARISON OF PROPOSED SMPSO1 AND SMPSO2 VERSIONS WITH BPSO AND OTHER VERSIONS AVAILABLE IN LITERATURE FOR FUNCTION F_3 IN TERMS OF AVERAGE FITNESS FUNCTION VALUE

Pop	Dim	Gne	SMPSO1 (gbest)	SMPSO2 (gworst)	BPSO [19]	QPSO [19]	Mutation gbest [20]	Mutation gbest [19]
20	10	1000	6.416553	**6.410466**	94.1276	59.4764	27.4620	21.2081
	20	1500	17.311169	**17.287586**	204.336	110.664	49.1176	61.9268
	30	2000	30.566478	**28.259791**	313.734	147.609	97.5952	86.1195
40	10	1000	6.4147	**6.401132**	71.0239	10.4238	7.8741	8.1828
	20	1500	**17.23444**	17.250421	179.291	46.5957	28.4435	40.0749
	30	2000	**28.114756**	28.640997	289.593	59.0291	62.3854	65.2891
80	10	1000	6.416151	**6.345346**	37.3747	8.63638	6.7098	7.3686
	20	1500	17.440593	**17.190714**	83.6931	35.8947	31.0929	30.1607
	30	2000	**28.324733**	30.153352	202.672	51.5479	43.7622	38.3036



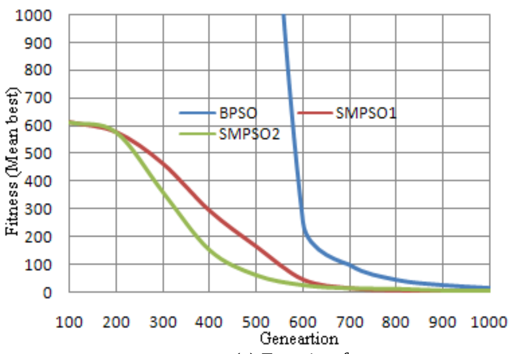
(a) Function f_1



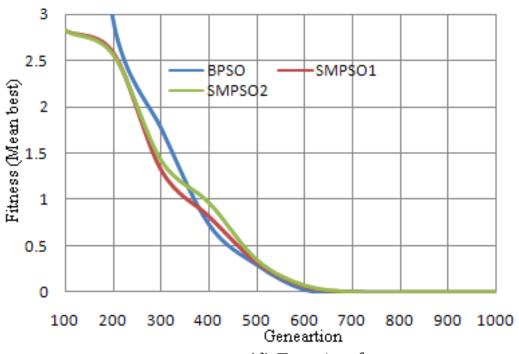
(b) Function f_2

TABLE IV. COMPARISON OF PROPOSED SMPSO1 AND SMPSO2 VERSIONS WITH BPSO FOR FUNCTIONS $F_4 - F_{15}$ IN TERMS OF AVERAGE FITNESS FUNCTION VALUE

Function	Dim	Gne	SMPSO1 (gbest)	SMPSO2 (gworst)	BPSO
f_4	10	1000	**1.62763e-10**	1.69783e-10	1.04431e-07
	20	1500	**0.000297**	0.000391	0.000801
	30	2000	0.004315	**0.003344**	0.009211
f_5	10	1000	0.000368	**0.000131**	0.003435
	20	1500	0.026621	**0.018378**	0.123742
	30	2000	**0.115094**	0.140612	1.31424
f_6	10	1000	**0.003347**	0.006410	0.008474
	20	1500	**0.023071**	0.031297	0.031376
	30	2000	**0.005992**	0.084939	0.089811
f_7	10	1000	9.25975e-05	**7.01387e-06**	0.000102
	20	1500	0.010336	**0.008547**	0.020129
	30	2000	**0.061249**	0.089932	0.174977
f_8	10	1000	**2.55247e-07**	2.90956e-06	0.001198
	20	1500	0.041532	**0.036888**	4.40991
	30	2000	**3.80048**	4.09788	271.793
f_9	10	1000	**0.000000**	**0.000000**	0.000000
	20	1500	**0.000000**	**0.000000**	0.000000
	30	2000	**0.000000**	**0.000000**	5.8
f_{10}	10	1000	3.17643e-09	**2.61458e-09**	6.13307e-07
	20	1500	1.7112e-05	**1.32619e-05**	0.083184
	30	2000	**0.000109**	0.000238	0.87767
f_{11}	10	1000	-1.15042	**-1.15044**	-1.15007
	20	1500	**-1.13147**	-1.12577	-0.813208
	30	2000	**-1.12011**	-1.04616	11.5649
f_{12}	10	1000	-3439.57	**-3456.7**	-3308.83
	20	1500	-6355.59	**-6593.98**	-6258.6
	30	2000	-9221.62	**-9830.23**	-8872.75
f_{13}	10	1000	-20.5621	**-20.604**	-20.346
	20	1500	-18.9347	**-19.0519**	-17.479
	30	2000	**-17.2635**	-16.1726	-13.4851
f_{14}	10	1000	0.346679	**0.29003**	0.612593
	20	1500	1.203	**1.07035**	2.41555
	30	2000	1.82546	**1.7637**	4.22028
f_{15}	10	1000	**-78.3323**	**-78.3323**	-78.0496
	20	1500	-75.6455	**-76.9185**	-74.9025
	30	2000	-75.4883	**75.4935**	-74.3619



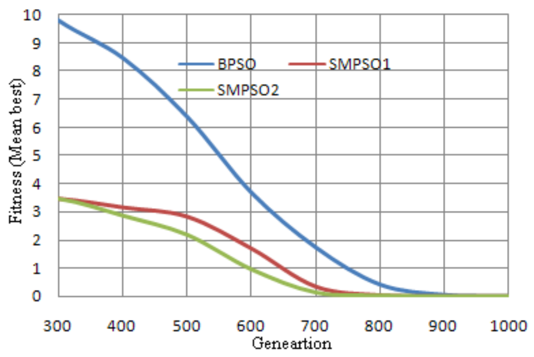
(c) Function f_3



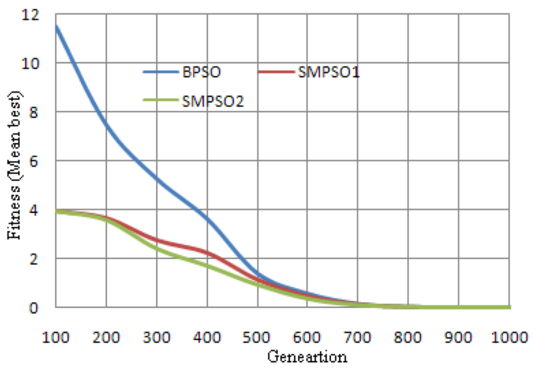
(d) Function f_4

TABLE V. NUMERICAL RESULTS OF REAL LIFE PROBLEMS

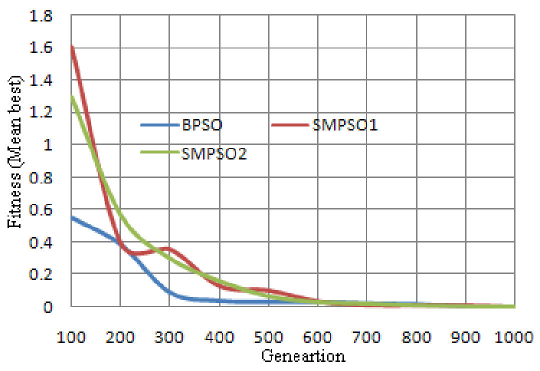
Gas Transmission Compressor Design			
Item	BPSO	SMPSO1(gbest)	SMPSO2(gworst)
x_1	55	51.9857	53.4474
x_2	1.19541	1.18335	1.1901
x_3	24.7749	24.7195	24.7186
f(x)	296.446e+004	296.448e+004	**296.436e+004**
G_{Avg}	786.7	146.4	**129.6**
f_{Eval}	23631	6205.1	**4422**
Optimal Thermohydralic Performance of an Artificially Roughened Air Heater			
Item	BPSO	SMPSO1(gbest)	SMPSO2(gworst)
x_1	0.05809	0.066242	0.032359
x_2	10	10	10
x_3	10400.2	7924.19	16643.4
f(x)	4.21422	4.21422	4.21422
G_{Avg}	205.9	196	**146**
f_{Eval}	6207	5911	**4425**
Optimal Capacity of Gas Production Facilities			
Item	BPSO	SMPSO1(gbest)	SMPSO2(gworst)
x_1	17.5	17.5	17.5
x_2	600	600	600
f(x)	169.844	169.844	169.844
G_{Avg}	10.4	**8**	9.8
f_{Eval}	342	**270**	324



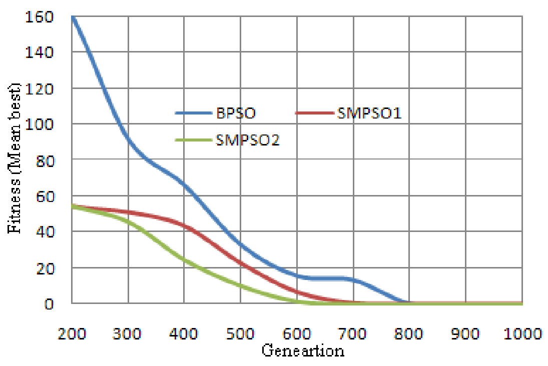
(e) Function f_5



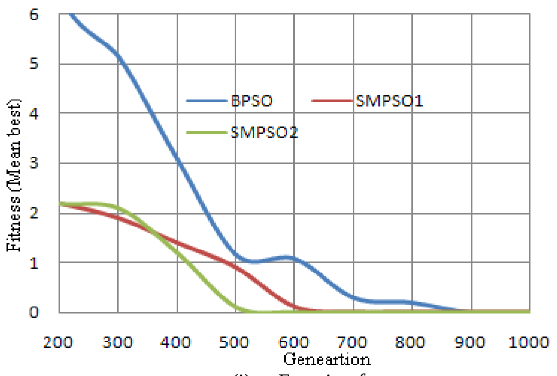
(g) Function f_7



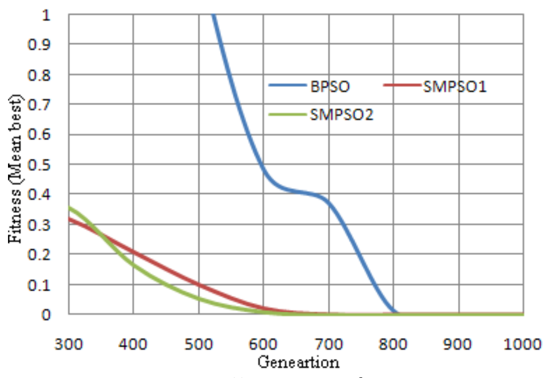
(f) Function f_6



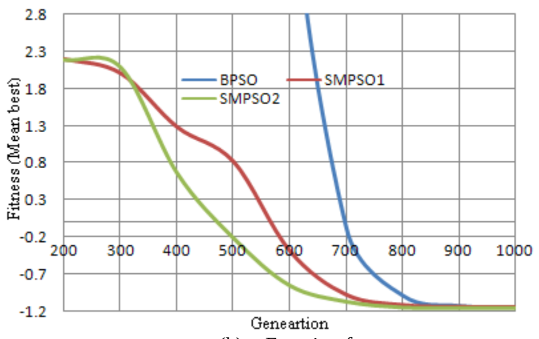
(h) Function f_8



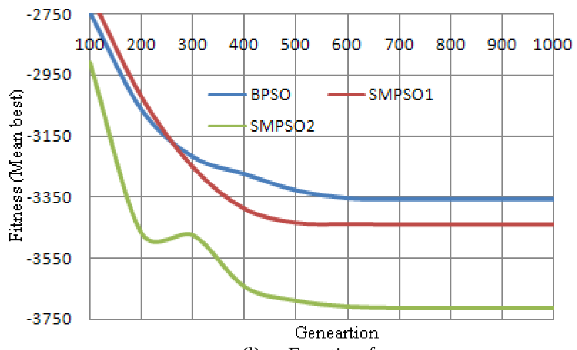
(i) Function f_9



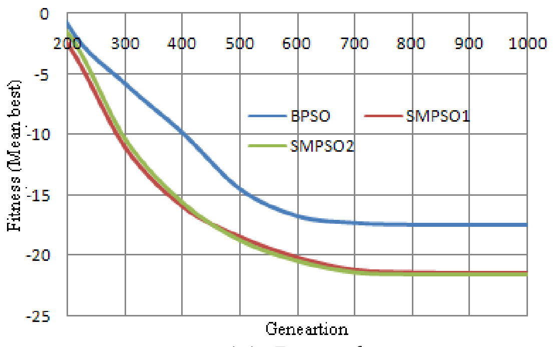
(j) Function f_{10}



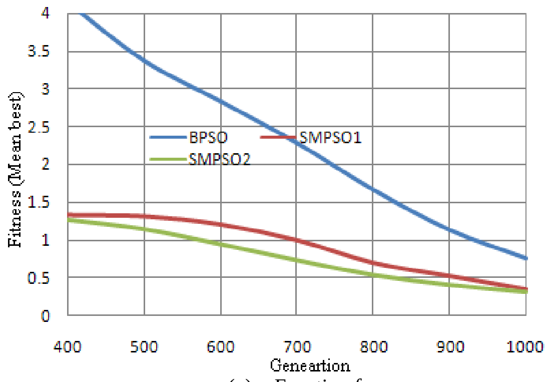
(k) Function f_{11}



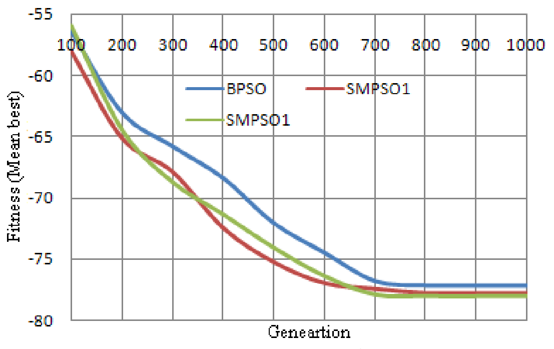
(l) Function f_{12}



(m) Function f_{13}



(n) Function f_{14}



(o) Function f_{15}

Figure 1. Performance curves of BPSO, SMPSO1 and SMPSO2 for all benchmark problems

[6] E. Cao and M. Lai, “An Improved Differential Evolution Algorithm for the Vehicle Routing Problem With Simultaneous Delivery and Pick-up Service”, In Proc. of Third International Conference on Natural Computation, Vol. 3, 2007, pp. 436 – 440.

[7] J. H. Holland, “Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence”, Ann Arbor, MI: University of Michigan Press, 1975.

[8] L. J. Fogel, A. J. Owens, and M. J. Walsh, “Artificial intelligence through a simulation of evolution”, In M. Maxfield, A. Callahan and L. J. Fogel, editors, Biophysics and Cybernetic systems, Proc. of the 2nd Cybernetic Sciences Symposium, pp. 131 – 155, Spartan Books, 1965.

[9] J. Kennedy and R. C., “Particle Swarm Optimization”, *IEEE International Conference on Neural Networks* (Perth, Australia), IEEE Service Center, Piscataway, NJ, 1995, pg. IV: 1942-1948.

[10] R. Storn and K. Price.: Differential Evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report, International Computer Science Institute, Berkley, 1995.

[11] A. Ratnaweera, S. K. Halgamuge, and H. C.Watson, “Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients,” *IEEE Transactions on Evolutionary Computation*, Vol. 8 (3), 2005, pp. 240–255.

[12] H. Higashi and H. Iba, “Particle Swarm Optimization with Gaussian Mutation”, *In Proc. of the IEEE swarm Intelligence Symposium*, 2003, pp. 72 – 79.

[13] A. Stacey, M. Jancic and I. Grundy, “Particle Swarm Optimization with Mutation”, *In Proc. of the IEEE Congress on Evolutionary Computation*, 2003, pp. 1425 – 1430.

[14] S. C. Esquivel and C. A. Coello Coello, “On the use of particle swarm optimization with multimodal functions”, *In Proceedings of the 2003 Congress on Evolutionary Computation*, IEEE Press, 2003, pp. 1130–1136.

[15] Z. Michalewicz, “Genetic Algorithms + Data Structures = Evolution Programs”, 3rd edition. Springer-Verlag, 1996.

[16] M. Pant, R. Thangaraj and A. Abraham, “Improved Particle Swarm Optimization with Low-Discrepancy Sequences”, In Proc. of IEEE Congress on Evolutionary Computation, 2008, pp. 3016 - 3023.

[17] X. H. Nguyen, Q. Uy. Nguyen, R. I. Mckay and P. M. Tuan, “Initializing PSO with Randomized Low- Discrepancy Sequences: The Comparative Results”, In Proc. of IEEE Congress on Evolutionary Algorithms, 2007, pp. 1985 – 1992.

[18] H. M. Chi, P. Beerli, D. W. Evans, and M. Mascagni, “On the Scrambled Sobol Sequence”, In Proc. of Workshop on Parellel Monte Carlo Algorithms for Diverse Applications in a Distributed Setting, LNCS 3516, Springer Verlog, 1999, pp. 775 – 782.

[19] J. Liu, J. Sun, W. Xu, “Quantum-Behaved Particle Swarm Optimization with Adaptive Mutation Operator”, ICNC 2006, Part I, Springer-Verlag, 2006, pp. 959 – 967.

[20] J. Liu, W. Xu, J. Sun “ Quantum-Behaved Particle Swarm Optimization with Mutation Operator”, In Proc. of the 17th IEEE Int. Conf. on Tools with Artificial Intelligence, Hong Kong (China), 2005, pp. 237 - 240.

[21] Beightler C. S. and Phillips DT, “Applied Geometric Programming”, Jhon Wiley and sons, New York, 1976.

[22] Prasad and Saini, “Optimal Thermo Hydraulic Performance of Artificially Roughened Solar Air Heaters”, *J. Solar Energy*, 1991, pp. 91 – 96.

APPENDIX

BENCHMARK PROBLEMS

The complete formulation of the test of problems is given below:

1. Rastringin function

$$\min_x \; f_1(x) = \sum_{i=1}^{n} (x_i^{\,2} - 10\cos(2\pi x_i) + 10)\,,$$
$-5.12 \le x_i \le 5.12$, $x^* = (0,0,...,0)$, $f_1(x^*) = 0$.

2. Griewank function

$$\min_x \; f_2(x) = \frac{1}{4000}\sum_{i=0}^{n-1} x_i^{\,2} - \sum_{i=0}^{n-1}\cos(\frac{x_i}{\sqrt{i+1}}) + 1\,,$$
$-600 \le x_i \le 600$, $x^* = (0,0,...,0)$, $f_2(x^*) = 0$.

3. Rosenbrock function

$$\min_x \; f_3(x) = \sum_{i=0}^{n-1} 100(x_{i+1} - x_i^{\,2})^2 + (x_i - 1)^2\,,$$
$-30 \le x_i \le 30$, $x^* = (1,1,...,1)$, $f_3(x^*) = 0$.

4. Sphere function

$$\min_x \; f_4(x) = \sum_{i=1}^{n} x_i^{\,2}\,,\; -5.12 \le x_i \le 5.12\,,\; x^* = (0,0,...,0)\,,$$
$f_4(x^*) = 0$.

5. Ackley’s function

$$\min_x \; f_5(x) = 20 + e - 20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^{\,2}})$$
$$-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i))\,,\; -32 \le x_i \le 32\,,\; x^* = (0,0,...,0)\,,$$
$f_5(x^*) = 0$.

6. Dejong’s function with noise

$$\min_x \; f_6(x) = (\sum_{i=0}^{n-1}(i+1)x_i^{\,4}) + rand[0,1]\,,$$
$-1.28 \le x_i \le 1.28$, $x^* = (0,0,...,0)$, $f_6(x^*) = 0$.

7. Schwefel function 1

$$\min_x \; f_7(x) = \sum_{i=0}^{n-1} |\, x_i \,| + \prod_{i=0}^{n-1} |\, x_i \,|\,,\; -10 \le x_i \le 10\,,$$
$x^* = (0,0,...,0)$, $f_7(x^*) = 0$.

8. Schwefel’s function 1.2

$$\min_x \; f_8(x) = \sum_{i=0}^{n-1}(\sum_{j=0}^{i} x_i)^2\,,\; -100 \le x_i \le 100\,,$$
$x^* = (0,0,...,0)$, $f_8(x^*) = 0$.

9. Step function

$$\min_x \; f_9(x) = \sum_{i=0}^{n-1} \lfloor x_i + 1/2 \rfloor^2\,,\; -100 \le x_i \le 100\,,$$
$x^* = (0,0,...,0)$, $f_9(x^*) = 0$.

10. Generalized penalized function 1

$$\min_x \; f_{10}(x) = \frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(y_{i+1}\pi)]$$
$$+ (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)\,,\text{ where } y_i = 1 + \frac{1}{4}(x_i + 1)$$
$-50 \le x_i \le 50$, $x^* = (0,0,...,0)$, $f_{10}(x^*) = 0$.

11. Generalized penalized function 2

$$\min_x \; f_{11}(x) = (0.1)\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}((x_i - 1)^2(1+\sin^2(3\pi x_{i+1})))$$
$$+(x_n - 1)(1+\sin^2(2\pi x_n))\} + \sum_{i=0}^{n-1} u(x_i, 5, 100, 4),\; -50 \le x_i \le 50,$$
$x^* = (1,1,\ldots,-4.76)$, $f_{11}(x^*) = -1.1428$.

12. Schwefel function

$$\min_x \; f_{12}(x) = -\sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|}),\; -500 \le x_i \le 500,$$
$x^* = (420.97, 420.97, \ldots, 420.97)$, $f_{12}(x^*) = -418.9829 * n$.

13. Levy function

$$\min_x \; f_{13}(x) = \sin^2(3\pi x_1) + \sum_{i=1}^{n-1}((x_i - 1)^2(1+\sin^2(3\pi x_{i+1})))$$
$$+(x_n - 1)(1+\sin^2(2\pi x_n)),$$
$-10 \le x_i \le 10$, $x^* = (0,0,\ldots,-9.7523)$, $f_{13}(x^*) = -21.5024$.

14 Circle function

$$\min_x \; f_{14}(x) = (\sum_{i=1}^{n} x_i^2)^{1/4}[\sin^2(50(\sum_{i=1}^{n} x_i^2)^{1/10}) + 1.0],$$
$-32.767 \le x_i \le 32.767$, $x^* = (0,0,\ldots,0)$, $f_{14}(x^*) = 0$.

15 Test2N function

$$\min_x \; f_{15}(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i^4 - 16x_i^2 + 5x_i),$$
$-5 \le x_i \le 5$, $x^* = (-2.9053, -2.9053, \ldots, -2.9053)$,
$f_{15}(x^*) = -78.3323$

In problem 10 and 11 the value of penalty function u is given by,

$u(x,a,b,c) = b(x-a)^c$, if $x > a$

$u(x,a,b,c) = b(-x-a)^c$, if $x < -a$

$u(x,a,b,c) = 0$, if $-a \le x \le a$.