A Population-Based, Steady-State Procedure for Real-Parameter Optimization

Ankur Sinha, Santosh Tiwari and Kalyanmoy Deb
Kanpur Genetic Algorithms Laboratory (KanGAL)
Indian Institute of Technology Kanpur
Kanpur, PIN 208 016, India
Email: {ankursi,tiwaris,deb}@iitk.ac.in
URL: http://www.iitk.ac.in/kangal/pub.htm

Abstract- Despite the existence of a number of procedures for real-parameter optimization using evolutionary algorithms, there is still a need of a systematic and unbiased comparison of different approaches on a carefully chosen set of test problems. In this paper, we develop a steady-state, population-based optimization algorithm which allows the main search principles to be independently designed. The algorithm so developed is applied to a set of 25 test problems and results on 10 and 30 dimensions are presented. Although the proposed procedure cannot find the exact optimum within the specified number of function evaluations, in most problems, the algorithm shows steady progress towards the optimum. Moreover, it is also observed that the performance of the algorithm does not get affected by the rotation of the functions, discontinuity and embedded noise in function description.

1 Introduction

This paper is written for the special session devoted to comparing different real-parameter optimization methods on a set of 25 test problems described at `http://www.ntu.edu.sg/home/EPNSugan / index_files/CEC-0 5/CEC05.htm` . In this paper, we employ a population-based, steady-state optimization algorithm for the purpose. The algorithm is developed based on adaptation of a population-based algorithm-generator [1]. The generator requires specification of four plans of the optimization process: (i) selection plan, (ii) generation plan, (iii) replacement plan and (iv) update plan. These plans are designed based on essential aspects needed in solving uni-modal and multi-modal optimization problems, such as importance of diversity preservation and need for creation of offspring solutions based on diversity in parent solutions. This method has also been used in other successful real parameter optimization schemes, such as evolution strategy [2, 3] and differential evolution [4]. We pre-specify all the GA parameters (involving the four plans mentioned above) based on past studies and some experimentations. However, significantly better results can be obtained by fine-tuning the control parameters for a particular problem.

The test problems involve uni-modalities to multi-modalities, deterministic to noisy functions, low to high dimensionalities etc. It is our intuition that such wide variety of problems may be difficult to be solved to optimality using one single optimization algorithm. This is because for an efficient solution of a uni-modal problem, a *greedy* technique emphasizing the current best solution can be employed, whereas for an efficient solution of a multi-modal problem diversity preservation is an important task. It is also our believe that to solve such vagaries of problems to a reasonable level of satisfaction, the algorithm has to be simple and not specifically designed to solve a particular problem. In the following section, we describe the proposed procedure. In Section 3, we present the simulation results in tabular form and in Section 4, we discuss the performance of our algorithm on different test problems.

2 Description of the Algorithm

The optimization algorithm used here is derived from the population-based algorithm generator suggested elsewhere [1]. The algorithm-generator requires four plans to be specified and generates a steady-state optimization procedure:

- Selection Plan (SP): Strategy used to select a fixed number of parents for recombination from the current population.
- Generation Plan (GP): Methodology used to create offspring solutions from parents chosen in the selection plan.
- Replacement Plan (RP): Strategy used to select a fixed number of members from population that will compete with the newly generated offspring solutions for inclusion in the population.
- Update Plan (UP): Strategy used to decide the winners from a set consisting of offspring solutions and members obtained from replacement plan that will eventually get included in the current population.

The above division of an algorithm into different plans allows one to design each essential feature of an optimization task independently. With a set of population members, the first task (SP) is to choose a set of good solutions (parents) so that they can be utilized to create new solutions in GP. The use of a suitable probability distribution around parent solutions to create new offspring solutions would be one way to implement a GP. Once the offspring solutions are created they can be accepted in the fixed-size population by first choosing a set of possible population members for deletion using a RP and then designing a scheme for updating the population in UP. This is where an elite preserving strategy can be implemented. Here we design a suitable scheme for each of the four plans as described below.

The algorithm starts with an initial population (generated randomly) of size N. We then use the *Selection Plan* to choose μ parents from the initial population. In the present

0-7803-9363-5/05/$20.00 ©2005 IEEE.

selection scheme, we first sort the entire population in ascending order based on the function value. This requires $O(N\log(N))$ computations. We then divide the population into κ equal segments (based on the fitness value), where κ is a user-defined parameter (lying in the range 1 to N) indicating the extent of modality of the problem. For uni-modal problems, a small value and for a multi-modal problem a large value of κ is suggested. The best solution of each segment is picked and stored in $\mathcal{B}$. We, now randomly pick one solution from the set of best solutions $\mathcal{B}$ as the first parent. We also call this solution as the *index* parent and denote its index as p. Thereafter, the other $(\mu - 1)$ parents are picked randomly from the population.

In the *Generation Plan*, we create λ offspring solutions from the chosen μ parent solutions. We use the parent-centric recombination (PCX) operator [5] with modification for the purpose of recombination and produce λ offspring solutions. The modified PCX operator creates a new solution using a uniformly distributed random number u ($\in [0,1]$), as follows:

$$\vec{C} = \begin{cases} \vec{X_p} + \omega_\xi \vec{D} + \sum_{i=1,i\neq p}^{\mu} \omega_\eta d\vec{e_i} & \text{if } \left(u > 0.5\left(1 - \frac{1}{\kappa}\right)\right) \\ \vec{M} + \omega_\xi \vec{D} + \sum_{i=1,i\neq p}^{\mu} \omega_\eta d\vec{e_i} & \text{otherwise.} \end{cases} \quad (1)$$

The computational complexity of this crossover operator is $O(\mu)$. The terms used in above equation are defined as follows:

- $\vec{X_p} \in \mathcal{B}$ is *index* parent,
- $\vec{G}$ is the mean of μ parents,
- $\vec{D} = \vec{X_p} - \vec{G}$,
- $\vec{M}$ is the mean of entire population,
- d is the average of perpendicular distances from parents $\vec{X_i}$ $(i \neq p)$ to the line joining $\vec{X_p}$ and $\vec{G}$,
- $\vec{e_i}$ are $(\mu - 1)$ orthonormal vectors that span the subspace perpendicular to $\vec{D}$. For information on how to find $\vec{e_i}$, please refer to [5].

The two parameters ω_ξ and ω_η, describing the extent of variations in direction $\vec{D}$ and orthogonal to it, respectively, are defined as follows (after some experimentation):

$$\begin{aligned} \omega_\xi &= \kappa^{-\alpha} - \frac{1}{\kappa} + 0.2, & (2) \\ \omega_\eta &= \frac{\omega_\xi}{2}. & (3) \end{aligned}$$

Although we use a constant value of $\kappa = 10$ here in this study, for a better use of the proposed procedure, we suggest to set $\kappa = 1$ for uni-modal problems and a large value of κ for multi-modal problems. With a large value of κ, ω_ξ will also take a large value, thereby introducing a larger diversity among offspring solutions – a matter which is important in solving multi-modal problems. Another interesting aspect of the above equation is that the parameter α is defined as the fraction of function evaluations performed to the overall desired number of function evaluations. Thus, for any chosen κ value, the parameter α starts with a value close to zero and approaches one as the generation proceeds. In other words, at the initial generation, $\omega_\xi = 1.2 - 1/\kappa$ and as the generation proceeds this value monotonically reduces and at the end of the optimization process, $\omega_\xi = 0.2$. To allow a broad search early on and to make a focussed search later for convergence, such settings are desired. It is also interesting to note that equation 1 allows both parent and mean-centric version of the recombination operation based on the modality index κ. When $\kappa = 1$, the recombination operation is always parent-centric and for a large value of κ both parent and mean-centric versions occur with almost equal probability. Figure 1 depicts the crossover operator pictorially.

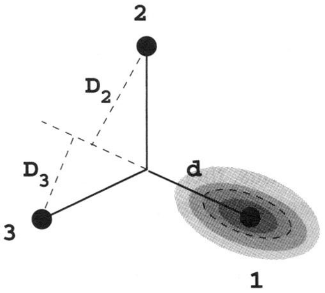

Figure 1: PCX Crossover

Next we use the *Replacement Plan* to choose r solutions from the population. In the present scheme, we choose these solutions randomly from the entire population.

We then form a pool (of size $(r + \lambda)$) consisting of r solutions chosen from the population by the replacement plan and λ newly created offspring solutions by the generation plan. The current population is then updated using the *Update Plan*, in which we replace the r solutions chosen in the replacement plan by the best r solutions of the pool. This operation ensures an elite-preservation strategy.

In performing a single iteration of above mentioned procedure, we have exhausted λ function evaluations (same as the number of offspring solutions produced). The iteration continues until a prescribed number of function evaluations is achieved or a pre-defined termination criterion is met. If at some instant, the diversity is lost in the population, we use cataclysmic mutation and choose the best individual obtained so far as the *index* parent and reproduce the population [6]. We use the polynomial mutation [7] as the mutation operator with mutation index $\eta_m = 1.0$ and mutation probability $p_m = 1/n$, where n is the number of real variables. For the purpose of simulation runs, we assume the diversity is lost whenever all the population members become identical.

3 Performance Benchmarks

We run all simulations using the following hardware and software: (i) Operating System Name: RedHat Linux 9.0 (i386 GNU/Linux), (ii) Machine Architecture: P-III 1.0 GHz, 256 MB RAM, (iii) Programming Language: ANSI-C, and (iv) Compiler Used: GCC version-3.2.2. Following parameters control the performance of the algorithm: (i) Population Size: N, (ii) Modality parameter: κ, (iii) Number of parents chosen for crossover: μ, (iv) Number of offspring solutions created: λ, and (v) Number of solutions chosen for replacement: r. We pre-set all the control parameters to some suitable values (with past experience of the authors and with some experimentations). We fix $N = 300$, $\kappa = 10$, $\mu = 3$, $\lambda = 2$ and $r = 1$ for performing the benchmark runs. Thus, there exists no parameter which is problem-dependent in this study.

3.1 Results of benchmark run for $D = 10$

Results for $D = 10$ and $MaxFES = 100,000$ are reported in tables 1 to 4.

3.2 Results of benchmark run for $D = 30$

Figures 2 to 6 are the convergence graphs for $D = 30$ and $MaxFES = 300,000$.

Results for $D = 30$ and $MaxFES = 300,000$ are reported in tables 5 to 8.

3.3 Algorithm running time benchmarks

Benchmarks for running time of algorithm are presented in Table 9. All times are reported in seconds.

Table 9: Computational Complexity of Algorithm

	T_0	T_1	T_2	$T_2 - T_1$	$(T_2 - T_1)/T_0$
D=10	0.24	1.25	34.37	33.12	138.0
D=30	0.24	24.60	105.75	81.15	338.12

3.4 Normalized value of convergence

We present here the degree of convergence achieved for all the test problems for $D = 10$ and $D = 30$. For every test problem, we have performed 25 independent simulation runs. We take the mean of best function value achieved in each simulation run and denote it as *Actual Mean* in table 10. For computing *Normalized Mean*, we first find the maximum absolute function value obtained from 10,000 randomly created solutions in the search range. For the purpose of normalization, we consider only the exponent of this maximum absolute function value for each function. We then divide the actual mean by the exponent of the maximum absolute function value and obtain the normalized mean. Table 10 presents the normalized mean values achieved for all test problems for $D = 10$ and $D = 30$. In table 10, *Problem Range* is the exponent of the maximum absolute function value obtained from 10,000 randomly created solutions. The cases where the achieved function value is within 1% of the maximum absolute function value of the search space, are shown in bold.

Table 10: Normalized mean value of convergence obtained for various test problems

Test Problem	Problem Range	Actual Mean		Normalized Mean	
		$D = 10$	$D = 30$	$D = 10$	$D = 30$
1	1.00e+05	8.71e-09	8.95e-09	**8.71e-14**	**8.95e-14**
2	1.00e+07	9.40e-09	1.44e-02	**9.40e-16**	**1.44e-09**
3	1.00e+11	3.02e+04	5.07e+05	**3.02e-07**	**5.07e-06**
4	1.00e+08	7.94e-07	1.11e+03	**7.94e-15**	**1.11e-05**
5	1.00e+06	4.85e+01	2.04e+03	**4.85e-05**	**2.04e-03**
6	1.00e+12	2.07e+01	9.89e+02	**2.07e-11**	**9.89e-10**
7	1.00e+05	6.40e-02	3.63e-02	**6.40e-07**	**3.63e-07**
8	1.00e+03	2.00e+01	2.00e+01	2.00e-02	2.00e-02
9	1.00e+04	1.19e-01	2.79e-01	**1.19e-05**	**2.79e-05**
10	1.00e+04	2.39e-01	5.17e-01	**2.39e-05**	**5.17e-05**
11	1.00e+03	9.11e+00	2.95e+01	**9.11e-03**	2.95e-02
12	1.00e+07	2.44e+04	1.04e+06	**2.44e-03**	1.04e-01
13	1.00e+08	6.53e-01	1.19e+01	**6.53e-09**	**1.19e-07**
14	1.00e+03	2.35e+00	1.38e+01	**2.35e-03**	1.38e-02
15	1.00e+04	5.10e+02	8.76e+02	5.10e-02	8.76e-02
16	1.00e+04	9.59e+01	7.15e+01	**9.59e-03**	**7.15e-03**
17	1.00e+04	9.73e+01	1.56e+02	**9.73e-03**	1.56e-02
18	1.00e+04	7.52e+02	8.30e+02	7.52e-02	8.30e-02
19	1.00e+04	7.51e+02	8.31e+02	7.51e-02	8.31e-02
20	1.00e+04	8.13e+02	8.31e+02	8.13e-02	8.31e-02
21	1.00e+04	1.05e+03	8.59e+02	1.05e-01	8.59e-02
22	1.00e+08	6.59e+02	1.56e+03	**6.59e-06**	**1.56e-05**
23	1.00e+04	1.06e+03	8.66e+02	1.06e-01	8.66e-02
24	1.00e+04	4.06e+02	2.13e+02	4.06e-02	2.13e-02
25	1.00e+04	4.06e+02	2.13e+02	4.06e-02	2.13e-02

4 Inferences from benchmark runs

We have performed an exhaustive benchmark analysis of the algorithm on 25 synthetic test problems. Benchmark study shows mixed results.

- Test functions 1 and 2 are simple and are easily solved by the algorithm without even requiring to perform the prescribed maximum number of function evaluations.
- The algorithm also solves functions 4, 6, 7, 9, and 10 for a majority of simulation runs.
- Introduction of noise to test functions does not hamper the performance of the algorithm significantly as is evident from the results on test function 4 in which the algorithm has 84% convergence rate for $D = 10$.
- Rotating a test problem has little effect on the performance of the algorithm as is evident from the performance of the algorithm on test problems 9 and 10. Test problem 10 is a rotated version of test problem 9.
- The algorithm does not perform well on hybrid test problems which can be attributed to the fact that hybrid test problems have a complicated profile, a large number of local optima, narrow global basin surrounded by huge number of local maxima which can cause any algorithm to divert away from the global basin. We can infer from the performance of algorithm on test problems 21, 22, and 23 that introducing noise or making the function profile discontinuous does not hamper the performance of the algorithm and it achieves similar convergence rate for these functions, table 7.
- The proposed procedure finds problems 8, 15, and 22 most difficult to solve. Problem 8 is like a needle-in-a-haystock problem and will cause difficulty to most algorithms. Problem 22 has a high condition number causing non-uniform importance among variables. This may have caused some difficulty to the proposed algorithm.
- Problems 15 to 25 are hybrid problems having different landscapes at different locations in the search

Table 1: Error values achieved at $FES = 10^3, 10^4, 10^5$ for problems 1-9 ($D = 10$)

Problem FES		1	2	3	4	5	6	7	8	9
1e3	1st (Best)	3.67e+03	6.26e+03	2.23e+07	9.72e+03	7.30e+03	7.72e+08	4.59e+02	2.04e+01	6.58e+01
	7th	6.74e+03	9.70e+03	7.77e+07	1.53e+04	9.58e+03	1.20e+09	6.37e+02	2.06e+01	8.90e+01
	13th (Median)	8.29e+03	1.33e+04	9.42e+07	1.79e+04	1.10e+04	1.53e+09	7.59e+02	2.07e+01	9.60e+01
	19th	8.78e+03	1.64e+04	1.14e+08	2.02e+04	1.21e+04	2.22e+09	9.56e+02	2.08e+01	1.03e+02
	25th (Worst)	1.10e+04	2.06e+04	1.50e+08	2.91e+04	1.34e+04	3.97e+09	1.26e+03	2.09e+01	1.13e+02
	Mean	7.84e+03	1.31e+04	9.31e+07	1.81e+04	1.07e+04	1.82e+09	8.12e+02	2.07e+01	9.49e+01
	Std	1.92e+03	4.15e+03	3.03e+07	4.75e+03	1.86e+03	8.90e+08	2.29e+02	1.27e-01	1.26e+01
1e4	1st (Best)	1.43e-02	7.01e+01	7.75e+06	1.42e+02	1.41e+03	1.77e+05	8.82e+00	2.00e+01	3.28e+01
	7th	6.13e-02	1.56e+02	1.92e+07	2.79e+02	2.24e+03	3.82e+05	1.29e+01	2.00e+01	4.14e+01
	13th (Median)	7.53e-02	2.05e+02	2.50e+07	3.84e+02	2.43e+03	1.12e+06	1.71e+01	2.00e+01	4.55e+01
	19th	1.28e-01	2.98e+02	3.15e+07	5.05e+02	2.84e+03	1.80e+06	2.06e+01	2.00e+01	5.01e+01
	25th (Worst)	1.83e-01	4.33e+02	5.12e+07	8.57e+02	3.28e+03	4.94e+06	2.36e+01	2.02e+01	5.72e+01
	Mean	9.42e-02	2.33e+02	2.57e+07	4.08e+02	2.46e+03	1.24e+06	1.70e+01	2.00e+01	4.51e+01
	Std	5.10e-02	1.05e+02	1.11e+07	1.77e+02	4.68e+02	1.02e+06	4.45e+00	4.27e-02	6.62e+00
1e5	1st (Best)	3.76e-09T	7.57e-09T	5.90e+03	8.68e-09T	9.15e-01	8.30e-09T	5.27e-09T	2.00e+01	4.76e-09T
	7th	7.43e-09T	9.31e-09T	1.48e+04	9.83e-09T	1.52e+01	2.16e-04	3.20e-02	2.00e+01	7.41e-09T
	13th (Median)	8.31e-09T	9.54e-09T	2.04e+04	9.96e-09T	3.82e+01	3.99e+00	5.42e-02	2.00e+01	8.82e-09T
	19th	9.33e-09T	9.76e-09T	3.91e+04	2.95e-07	6.47e+01	8.63e+00	1.18e-01	2.00e+01	9.66e-09T
	25th (Worst)	9.89e-09T	9.88e-09T	9.87e+04	8.02e-06	2.55e+02	1.09e+02	1.53e-01	2.00e+01	2.98e+00
	Mean	8.71e-09T	9.40e-09T	3.02e+04	7.94e-07	4.85e+01	2.07e+01	6.40e-02	2.00e+01	1.19e-01
	Std	1.22e-09T	5.37e-10T	2.38e+04	1.97e-06	5.17e+01	3.56e+01	5.15e-02	3.25e-07	5.97e-01

Table 2: Error values achieved at $FES = 10^3, 10^4, 10^5$ for problems 10-17 ($D = 10$)

Problem FES		10	11	12	13	14	15	16	17
1e3	1st (Best)	8.47e+01	9.42e+00	2.56e+04	8.93e+01	4.13e+00	7.38e+02	3.78e+02	4.07e+02
	7th	1.21e+02	1.09e+01	6.76e+04	4.86e+02	4.36e+00	8.21e+02	4.19e+02	4.80e+02
	13th (Median)	1.31e+02	1.16e+01	7.60e+04	1.96e+03	4.38e+00	8.50e+02	4.35e+02	5.25e+02
	19th	1.36e+02	1.20e+01	9.81e+04	3.47e+03	4.44e+00	8.76e+02	4.80e+02	5.76e+02
	25th (Worst)	1.56e+02	1.29e+01	1.32e+05	9.84e+03	4.63e+00	9.10e+02	5.59e+02	6.67e+02
	Mean	1.28e+02	1.14e+01	8.29e+04	2.38e+03	4.39e+00	8.41e+02	4.47e+02	5.27e+02
	Std	1.69e+01	8.96e-01	2.63e+04	2.43e+03	1.18e-01	4.74e+01	5.12e+01	6.60e+01
1e4	1st (Best)	4.12e+01	7.63e+00	2.56e+04	2.59e+00	3.67e+00	6.37e+02	1.73e+02	1.72e+02
	7th	4.70e+01	9.78e+00	4.03e+04	3.37e+00	4.15e+00	6.99e+02	1.85e+02	2.08e+02
	13th (Median)	5.30e+01	1.01e+01	4.55e+04	3.53e+00	4.27e+00	7.34e+02	1.95e+02	2.17e+02
	19th	5.73e+01	1.05e+01	5.63e+04	3.96e+00	4.37e+00	7.50e+02	2.02e+02	2.24e+02
	25th (Worst)	6.58e+01	1.18e+01	7.14e+04	4.66e+00	4.51e+00	7.88e+02	2.22e+02	2.42e+02
	Mean	5.25e+01	1.01e+01	4.78e+04	3.63e+00	4.25e+00	7.25e+02	1.95e+02	2.14e+02
	Std	6.80e+00	8.34e-01	1.19e+04	4.82e-01	1.65e-01	3.98e+01	1.32e+01	1.78e+01
1e5	1st (Best)	6.26e-09T	7.64e+00	1.98e+02	3.28e-01	1.53e+00	2.79e+02	8.75e+01	8.82e+01
	7th	7.69e-09T	8.80e+00	2.06e+04	4.74e-01	1.97e+00	4.14e+02	9.14e+01	9.42e+01
	13th (Median)	8.85e-09T	9.22e+00	2.50e+04	7.15e-01	2.40e+00	4.63e+02	9.37e+01	9.69e+01
	19th	9.73e-09T	9.42e+00	2.88e+04	8.01e-01	2.75e+00	6.38e+02	9.64e+01	9.94e+01
	25th (Worst)	2.98e+00	9.91e+00	4.00e+04	1.07e+00	3.30e+00	7.06e+02	1.13e+02	1.14e+02
	Mean	2.39e-01	9.11e+00	2.44e+04	6.53e-01	2.35e+00	5.10e+02	9.59e+01	9.73e+01
	Std	7.20e-01	5.27e-01	8.92e+03	2.06e-01	4.82e-01	1.27e+02	6.85e+00	5.55e+00

Table 3: Error values achieved at $FES = 10^3, 10^4, 10^5$ for problems 18-25 ($D = 10$)

Problem FES		18	19	20	21	22	23	24	25
1e3	1st (Best)	1.09e+03	1.11e+03	1.06e+03	1.22e+03	9.37e+02	1.25e+03	4.86e+02	5.53e+02
	7th	1.19e+03	1.19e+03	1.17e+03	1.31e+03	1.05e+03	1.39e+03	7.09e+02	7.05e+02
	13th (Median)	1.22e+03	1.23e+03	1.20e+03	1.35e+03	1.10e+03	1.41e+03	7.22e+02	7.73e+02
	19th	1.26e+03	1.24e+03	1.24e+03	1.37e+03	1.14e+03	1.42e+03	8.79e+02	8.69e+02
	25th (Worst)	1.30e+03	1.32e+03	1.34e+03	1.40e+03	1.22e+03	1.45e+03	9.95e+02	9.74e+02
	Mean	1.22e+03	1.22e+03	1.21e+03	1.34e+03	1.09e+03	1.40e+03	7.63e+02	7.72e+02
	Std	5.38e+01	5.03e+01	6.30e+01	4.62e+01	6.59e+01	4.96e+01	1.35e+02	1.20e+02
1e4	1st (Best)	8.48e+02	8.47e+02	8.60e+02	9.42e+02	5.50e+02	1.03e+03	4.08e+02	4.10e+02
	7th	8.76e+02	9.42e+02	9.23e+02	1.12e+03	5.77e+02	1.16e+03	4.10e+02	4.10e+02
	13th (Median)	9.47e+02	9.91e+02	9.77e+02	1.13e+03	7.95e+02	1.17e+03	4.12e+02	4.11e+02
	19th	1.02e+03	1.02e+03	1.04e+03	1.13e+03	9.00e+02	1.18e+03	4.12e+02	4.12e+02
	25th (Worst)	1.06e+03	1.08e+03	1.07e+03	1.14e+03	9.30e+02	1.20e+03	4.13e+02	4.13e+02
	Mean	9.52e+02	9.76e+02	9.74e+02	1.12e+03	7.43e+02	1.16e+03	4.11e+02	4.11e+02
	Std	7.16e+01	6.75e+01	6.53e+01	3.80e+01	1.54e+02	3.65e+01	1.44e+00	9.86e-01
1e5	1st (Best)	3.00e+02	3.00e+02	3.00e+02	5.00e+02	5.27e+02	5.59e+02	4.05e+02	4.05e+02
	7th	8.00e+02	8.00e+02	8.22e+02	1.06e+03	5.31e+02	1.10e+03	4.06e+02	4.06e+02
	13th (Median)	8.22e+02	8.24e+02	8.26e+02	1.08e+03	7.29e+02	1.10e+03	4.06e+02	4.06e+02
	19th	8.26e+02	8.27e+02	8.94e+02	1.08e+03	7.41e+02	1.10e+03	4.06e+02	4.06e+02
	25th (Worst)	9.51e+02	9.42e+02	9.59e+02	1.09e+03	8.69e+02	1.11e+03	4.07e+02	4.07e+02
	Mean	7.52e+02	7.51e+02	8.13e+02	1.05e+03	6.59e+02	1.06e+03	4.06e+02	4.06e+02
	Std	2.07e+02	2.04e+02	1.63e+02	1.16e+02	1.33e+02	1.50e+02	5.13e-01	4.88e-01

Table 4: Number of FES required to achieve a given accuracy level for problems $1 - 25$ ($D = 10$)

Prob	1^{st}	7^{th}	13^{th}	19^{th}	25^{th}	Mean	Std	Succ. Rate	Succ. Perf.
1	14878	15204	15542	15764	16148	1.5494e+04	3.255e+03	100.00%	1.5509e+04
2	30708	33208	33950	34796	38982	3.4138e+04	6.930e+03	100.00%	3.4148e+04
3	-	-	-	-	-	-	-	0.00%	-
4	36044	39550	48878	57892	-	4.7705e+04	8.7414e+03	84.00%	5.6806e+04
5	-	-	-	-	-	-	-	0.00%	-
6	73188	90134	-	-	-	8.663e+04	8.473e+03	40.00%	2.16585e+05
7	31290	-	-	-	-	3.5534e+04	8.017e+03	20.00%	1.77682e+05
8	-	-	-	-	-	-	-	0.00%	-
9	42956	44616	48236	49476	-	4.7246e+04	9.8935e+03	96.00%	4.9226e+04
10	41722	45644	47570	52234	-	4.8381e+04	9.5963e+03	88.00%	5.4991e+04
11	-	-	-	-	-	-	-	0.00%	-
12	-	-	-	-	-	-	-	00.00%	-
13	-	-	-	-	-	-	-	0.00%	-
14	-	-	-	-	-	-	-	0.00%	-
15	-	-	-	-	-	-	-	0.00%	-
16	-	-	-	-	-	-	-	0.00%	-
17	-	-	-	-	-	-	-	0.00%	-
18	-	-	-	-	-	-	-	0.00%	-
19	-	-	-	-	-	-	-	0.00%	-
20	-	-	-	-	-	-	-	0.00%	-
21	-	-	-	-	-	-	-	0.00%	-
22	-	-	-	-	-	-	-	0.00%	-
23	-	-	-	-	-	-	-	0.00%	-
24	-	-	-	-	-	-	-	0.00%	-
25	-	-	-	-	-	-	-	0.00%	-

Table 5: Error values achieved at $FES = 10^3, 10^4, 10^5$ for problems 1-9 ($D = 30$)

Problem FES		1	2	3	4	5	6	7	8	9
1e3	1^{st} (Best)	6.43e+04	1.00e+05	7.17e+08	1.16e+05	3.50e+04	4.35e+10	6.73e+03	2.11e+01	4.43e+02
	7^{th}	9.72e+04	1.16e+05	1.60e+09	1.49e+05	3.98e+04	6.01e+10	8.14e+03	2.12e+01	4.92e+02
	13^{th} (Median)	1.07e+05	1.39e+05	1.95e+09	1.64e+05	4.12e+04	6.68e+10	8.77e+03	2.12e+01	5.07e+02
	19^{th}	1.13e+05	1.63e+05	2.07e+09	1.87e+05	4.33e+04	8.29e+10	8.99e+03	2.12e+01	5.17e+02
	25^{th} (Worst)	1.24e+05	3.11e+05	2.61e+09	2.40e+05	4.89e+04	1.05e+11	9.67e+03	2.13e+01	5.61e+02
	Mean	1.04e+05	1.46e+05	1.83e+09	1.69e+05	4.14e+04	7.06e+10	8.57e+03	2.12e+01	5.03e+02
	Std	1.54e+04	4.23e+04	4.89e+08	3.47e+04	3.28e+03	1.61e+10	7.87e+02	5.86e-02	2.87e+01
1e4	1^{st} (Best)	6.43e+04	1.00e+05	7.17e+08	1.16e+05	3.50e+04	4.35e+10	4.10e+03	2.03e+01	4.43e+02
	7^{th}	9.72e+04	1.16e+05	1.54e+09	1.49e+05	3.98e+04	6.01e+10	4.47e+03	2.11e+01	4.92e+02
	13^{th} (Median)	1.07e+05	1.39e+05	1.90e+09	1.64e+05	4.12e+04	6.69e+10	5.17e+03	2.11e+01	5.03e+02
	19^{th}	1.13e+05	1.63e+05	2.05e+09	1.87e+05	4.28e+04	8.29e+10	5.61e+03	2.11e+01	5.17e+02
	25^{th} (Worst)	1.24e+05	2.50e+05	2.61e+09	2.40e+05	4.89e+04	1.05e+11	7.28e+03	2.12e+01	5.61e+02
	Mean	1.03e+05	1.44e+05	1.75e+09	1.69e+05	4.12e+04	7.06e+10	5.19e+03	2.10e+01	5.02e+02
	Std	1.52e+04	3.33e+04	5.23e+08	3.47e+04	3.07e+03	1.61e+10	8.03e+02	1.78e-01	2.80e+01
1e5	1^{st} (Best)	8.79e+03	7.22e+04	7.17e+08	9.99e+04	2.18e+04	6.77e+09	1.32e+03	2.00e+01	2.91e+02
	7^{th}	9.94e+03	1.00e+05	1.04e+09	1.40e+05	2.72e+04	1.10e+10	2.05e+03	2.00e+01	3.53e+02
	13^{th} (Median)	1.31e+04	1.15e+05	1.16e+09	1.53e+05	2.89e+04	1.33e+10	2.17e+03	2.00e+01	3.62e+02
	19^{th}	1.52e+04	1.24e+05	1.45e+09	1.59e+05	3.04e+04	1.43e+10	2.41e+03	2.00e+01	3.84e+02
	25^{th} (Worst)	1.91e+04	1.61e+05	1.76e+09	1.84e+05	3.28e+04	2.25e+10	2.92e+03	2.00e+01	4.19e+02
	Mean	1.31e+04	1.13e+05	1.22e+09	1.47e+05	2.87e+04	1.30e+10	2.17e+03	2.00e+01	3.65e+02
	Std	3.14e+03	2.21e+04	2.74e+08	2.16e+04	2.89e+03	3.51e+09	3.63e+02	7.24e-07	2.65e+01
3e5	1^{st} (Best)	5.11e-09T	2.09e-03	2.27e+05	3.22e+02	1.31e+03	2.53e+01	9.86e-03	2.00e+01	8.27e-09T
	7^{th}	8.58e-09T	6.35e-03	3.63e+05	6.87e+02	1.80e+03	1.29e+02	9.86e-03	2.00e+01	9.68e-09T
	13^{th} (Median)	9.18e-09T	1.22e-02	4.70e+05	1.19e+03	2.06e+03	1.48e+02	2.95e-02	2.00e+01	1.16e-08
	19^{th}	9.57e-09T	2.14e-02	6.88e+05	1.34e+03	2.36e+03	8.48e+02	3.69e-02	2.00e+01	9.95e-01
	25^{th} (Worst)	9.96e-09T	3.89e-02	8.39e+05	2.17e+03	2.64e+03	6.45e+03	2.26e-01	2.00e+01	9.95e-01
	Mean	8.95e-09T	1.44e-02	5.07e+05	1.11e+03	2.04e+03	9.89e+02	3.63e-02	2.00e+01	2.79e-01
	Std	9.90e-10T	9.83e-03	1.86e+05	4.35e+02	3.81e+02	1.69e+03	4.77e-02	4.97e-07	4.56e-01

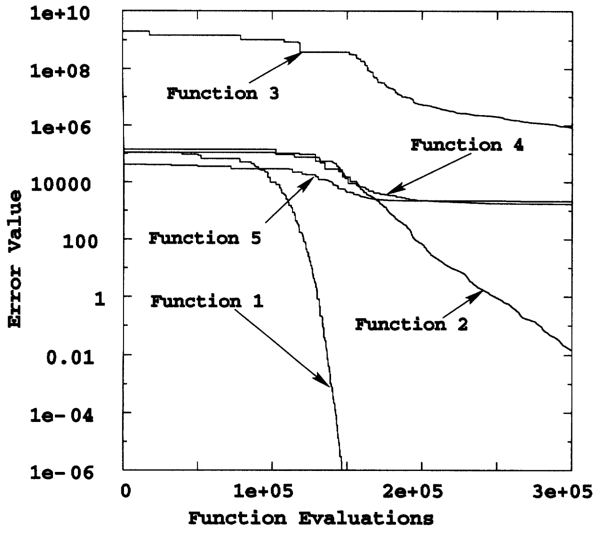

Figure 2: Convergence Graph for Problems 1 – 5 (D = 30)

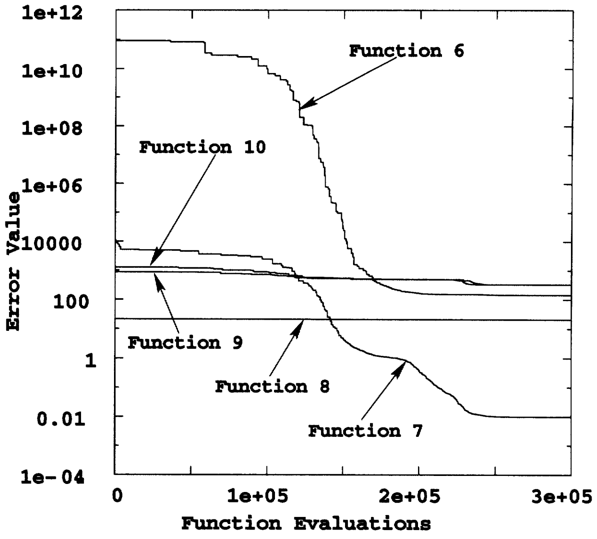

Figure 3: Convergence Graph for Problems 6 – 10 (D = 30)

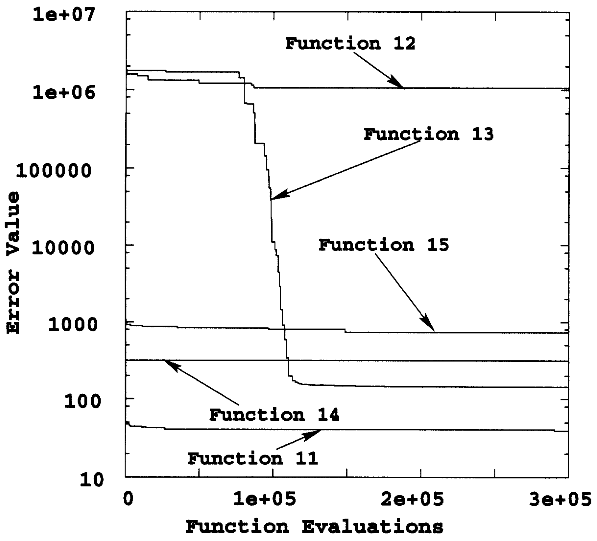

Figure 4: Convergence Graph for Problems 11 – 15 (D = 30)

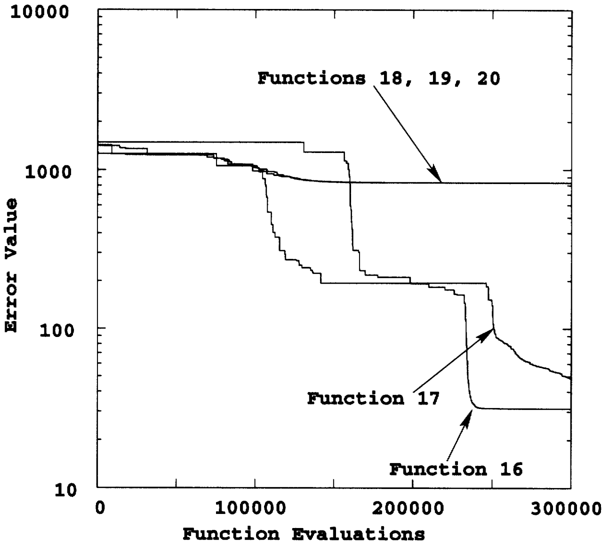

Figure 5: Convergence Graph for Problems 16 – 20 (D = 30)

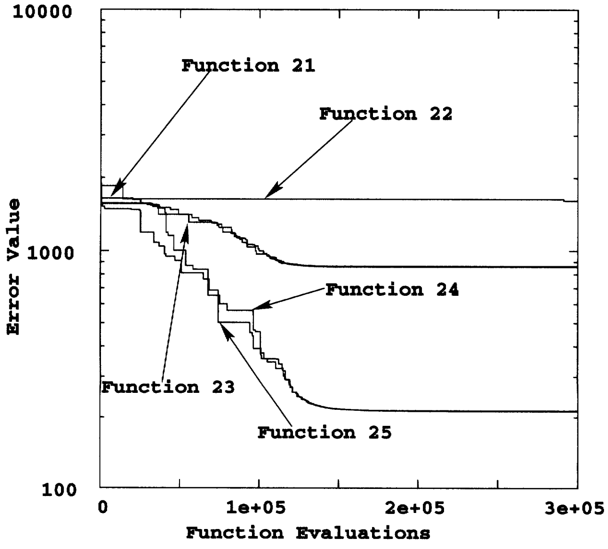

Figure 6: Convergence Graph for Problems 21 – 25 (D = 30)

Table 6: Error values achieved at $FES = 10^3, 10^4, 10^5$ for problems 10-17 ($D = 30$)

Problem FES		10	11	12	13	14	15	16	17
1e3	1^{st} (Best)	7.57e+02	2.77e+01	1.39e+06	3.21e+05	1.40e+01	9.37e+02	9.82e+02	1.16e+03
	7^{th}	7.73e+02	3.10e+01	1.61e+06	8.10e+05	1.42e+01	9.97e+02	1.12e+03	1.27e+03
	13^{th} (Median)	8.56e+02	3.34e+01	1.77e+06	1.35e+06	1.44e+01	1.02e+03	1.18e+03	1.36e+03
	19^{th}	8.74e+02	3.48e+01	1.83e+06	1.70e+06	1.45e+01	1.04e+03	1.28e+03	1.42e+03
	25^{th} (Worst)	9.44e+02	3.87e+01	2.09e+06	2.35e+06	1.46e+01	1.07e+03	1.36e+03	1.55e+03
	Mean	8.36e+02	3.30e+01	1.74e+06	1.29e+06	1.44e+01	1.02e+03	1.20e+03	1.36e+03
	Std	5.94e+01	2.95e+00	1.62e+05	5.52e+05	1.54e-01	3.12e+01	1.07e+02	1.06e+02
1e4	1^{st} (Best)	7.57e+02	2.43e+01	9.99e+05	3.21e+05	1.40e+01	8.82e+02	9.82e+02	1.16e+03
	7^{th}	7.73e+02	2.67e+01	1.33e+06	8.10e+05	1.42e+01	9.65e+02	1.11e+03	1.27e+03
	13^{th} (Median)	8.56e+02	3.08e+01	1.44e+06	1.35e+06	1.44e+01	9.87e+02	1.18e+03	1.36e+03
	19^{th}	8.74e+02	3.27e+01	1.55e+06	1.70e+06	1.44e+01	1.00e+03	1.28e+03	1.42e+03
	25^{th} (Worst)	9.44e+02	3.67e+01	1.69e+06	2.35e+06	1.45e+01	1.03e+03	1.36e+03	1.55e+03
	Mean	8.36e+02	3.01e+01	2.69e+03	1.43e+06	1.43e+01	9.79e+02	1.19e+03	1.36e+03
	Std	5.94e+01	3.63e+00	3.52e+03	1.81e+05	1.40e-01	3.30e+01	1.07e+02	1.06e+02
1e5	1^{st} (Best)	3.81e+02	2.42e+01	6.74e+05	2.56e+03	1.40e+01	8.81e+02	4.18e+02	1.08e+03
	7^{th}	4.87e+02	2.67e+01	1.06e+06	8.54e+03	1.41e+01	9.04e+02	8.02e+02	1.25e+03
	13^{th} (Median)	5.14e+02	3.08e+01	1.15e+06	1.42e+04	1.42e+01	9.23e+02	1.05e+03	1.31e+03
	19^{th}	5.59e+02	3.27e+01	1.24e+06	2.43e+04	1.43e+01	9.36e+02	1.16e+03	1.39e+03
	25^{th} (Worst)	5.97e+02	3.67e+01	1.32e+06	4.97e+04	1.44e+01	9.72e+02	1.32e+03	1.52e+03
	Mean	5.16e+02	3.01e+01	1.12e+06	1.74e+04	1.42e+01	9.23e+02	9.45e+02	1.31e+03
	Std	4.98e+01	3.64e+00	1.50e+05	1.18e+04	1.07e-01	2.44e+01	2.79e+02	1.07e+02
3e5	1^{st} (Best)	3.71e-08	2.29e+01	6.74e+05	2.27e+00	1.32e+01	8.24e+02	1.85e+01	3.39e+01
	7^{th}	7.68e-08	2.67e+01	9.93e+05	1.26e+01	1.38e+01	8.66e+02	2.46e+01	4.67e+01
	13^{th} (Median)	1.73e-07	2.90e+01	1.06e+06	1.33e+01	1.39e+01	8.82e+02	3.30e+01	6.74e+01
	19^{th}	9.95e-01	3.15e+01	1.15e+06	1.38e+01	1.40e+01	8.87e+02	1.13e+02	1.57e+02
	25^{th} (Worst)	2.98e+00	3.67e+01	1.23e+06	1.49e+01	1.41e+01	9.02e+02	4.00e+02	5.45e+02
	Mean	5.17e-01	2.95e+01	1.04e+06	1.19e+01	1.38e+01	8.76e+02	7.15e+01	1.56e+02
	Std	7.11e-01	3.65e+00	1.32e+05	3.80e+00	1.90e-01	1.95e+01	8.10e+01	1.58e+02

Table 7: Error values achieved at $FES = 10^3, 10^4, 10^5$ for problems 18-25 ($D = 30$)

Problem FES		18	19	20	21	22	23	24	25
1e3	1^{st} (Best)	1.33e+03	1.33e+03	1.33e+03	1.39e+03	1.58e+03	1.38e+03	1.48e+03	1.52e+03
	7^{th}	1.39e+03	1.38e+03	1.38e+03	1.55e+03	1.79e+03	1.55e+03	1.56e+03	1.66e+03
	13^{th} (Median)	1.42e+03	1.41e+03	1.41e+03	1.57e+03	1.93e+03	1.56e+03	1.59e+03	1.68e+03
	19^{th}	1.45e+03	1.44e+03	1.44e+03	1.61e+03	2.02e+03	1.61e+03	1.62e+03	1.75e+03
	25^{th} (Worst)	1.48e+03	1.49e+03	1.49e+03	1.65e+03	2.29e+03	1.64e+03	1.65e+03	1.80e+03
	Mean	1.42e+03	1.42e+03	1.42e+03	1.56e+03	1.91e+03	1.56e+03	1.59e+03	1.69e+03
	Std	3.82e+01	4.07e+01	4.07e+01	6.10e+01	1.94e+02	6.25e+01	3.99e+01	6.68e+01
1e4	1^{st} (Best)	1.26e+03	1.28e+03	1.22e+03	1.39e+03	1.50e+03	1.38e+03	1.43e+03	1.10e+03
	7^{th}	1.34e+03	1.32e+03	1.31e+03	1.55e+03	1.73e+03	1.55e+03	1.55e+03	1.51e+03
	13^{th} (Median)	1.37e+03	1.38e+03	1.38e+03	1.57e+03	1.85e+03	1.56e+03	1.57e+03	1.56e+03
	19^{th}	1.40e+03	1.40e+03	1.40e+03	1.61e+03	1.99e+03	1.61e+03	1.61e+03	1.58e+03
	25^{th} (Worst)	1.48e+03	1.47e+03	1.47e+03	1.65e+03	2.18e+03	1.64e+03	1.65e+03	1.66e+03
	Mean	1.37e+03	1.37e+03	1.36e+03	1.56e+03	1.86e+03	1.56e+03	1.57e+03	1.52e+03
	Std	5.64e+01	4.76e+01	5.71e+01	6.10e+01	1.75e+02	6.25e+01	5.42e+01	1.18e+02
1e5	1^{st} (Best)	9.97e+02	9.83e+02	9.80e+02	9.71e+02	1.04e+03	9.78e+02	3.52e+02	3.05e+02
	7^{th}	1.03e+03	1.03e+03	1.04e+03	9.93e+02	1.63e+03	1.00e+03	3.87e+02	3.52e+02
	13^{th} (Median)	1.06e+03	1.05e+03	1.06e+03	1.00e+03	1.84e+03	1.02e+03	3.97e+02	3.61e+02
	19^{th}	1.07e+03	1.09e+03	1.09e+03	1.02e+03	1.94e+03	1.04e+03	4.42e+02	4.00e+02
	25^{th} (Worst)	1.14e+03	1.14e+03	1.14e+03	1.04e+03	2.05e+03	1.06e+03	5.09e+02	4.60e+02
	Mean	1.06e+03	1.06e+03	1.06e+03	1.01e+03	1.77e+03	1.02e+03	4.17e+02	3.75e+02
	Std	3.04e+01	4.40e+01	3.87e+01	1.91e+01	2.45e+02	2.13e+01	4.31e+01	4.14e+01
3e5	1^{st} (Best)	8.28e+02	8.28e+02	8.28e+02	8.58e+02	5.25e+02	8.65e+02	2.12e+02	2.12e+02
	7^{th}	8.30e+02	8.30e+02	8.30e+02	8.59e+02	1.50e+03	8.66e+02	2.13e+02	2.13e+02
	13^{th} (Median)	8.30e+02	8.31e+02	8.31e+02	8.60e+02	1.78e+03	8.66e+02	2.13e+02	2.13e+02
	19^{th}	8.31e+02	8.32e+02	8.32e+02	8.60e+02	1.91e+03	8.67e+02	2.13e+02	2.14e+02
	25^{th} (Worst)	8.37e+02	8.34e+02	8.33e+02	8.60e+02	1.96e+03	8.68e+02	2.14e+02	2.14e+02
	Mean	8.30e+02	8.31e+02	8.31e+02	8.59e+02	1.56e+03	8.66e+02	2.13e+02	2.13e+02
	Std	1.61e+00	1.47e+00	1.32e+00	5.44e-01	4.83e+02	8.07e-01	3.89e-01	5.50e-01

Table 8: Number of FES required to achieve a given accuracy level for problems $1-25$ ($D=30$)

Prob	1^{st}	7^{th}	13^{th}	19^{th}	25^{th}	Mean	Std	Succ. Rate	Succ. Perf.
1	146536	147008	147520	147820	148610	1.4753e+05	3.0022e+04	100.00%	1.4753e+03
2	-	-	-	-	-	-	-	0.00%	-
3	-	-	-	-	-	-	-	0.00%	-
4	-	-	-	-	-	-	-	0.00%	-
5	-	-	-	-	-	-	-	0.00%	-
6	-	-	-	-	-	-	-	0.00%	-
7	257332	273750	-	-	-	2.71361e+05	1.4685e+03	44.00%	6.1674e+05
8	-	-	-	-	-	-	-	0.00%	-
9	228064	236104	239896	-	-	2.3886e+05	1.8124e+04	72.00%	3.3176e+05
10	238612	249548	261126	-	-	2.5140e+05	5.0578e+04	56.00%	4.4894e+05
11	-	-	-	-	-	-	-	0.00%	-
12	-	-	-	-	-	-	-	0.00%	-
13	-	-	-	-	-	-	-	0.00%	-
14	-	-	-	-	-	-	-	0.00%	-
15	-	-	-	-	-	-	-	0.00%	-
16	-	-	-	-	-	-	-	0.00%	-
17	-	-	-	-	-	-	-	0.00%	-
18	-	-	-	-	-	-	-	0.00%	-
19	-	-	-	-	-	-	-	0.00%	-
20	-	-	-	-	-	-	-	0.00%	-
21	-	-	-	-	-	-	-	0.00%	-
22	-	-	-	-	-	-	-	0.00%	-
23	-	-	-	-	-	-	-	0.00%	-
24	-	-	-	-	-	-	-	0.00%	-
25	-	-	-	-	-	-	-	0.00%	-

space. For an algorithm to find the exact optimum, it needs to first locate the global basin by avoiding to get stuck at many local optimal basins. This feature of the problems makes it difficult for an algorithm to solve to optimality.

5 Conclusions

In this paper, we have developed a steady-state, population-based real-parameter optimization algorithm based on an algorithm-generator suggested elsewhere [1] and have attempted to solve 25 different benchmark test problems of dimensions 10 and 30. The proposed algorithm uses modified parent-centric recombination (PCX) operator and a polynomial mutation operator along with a niched-selection operator for creating offspring solutions. The algorithm has been developed with simple-minded yet essential aspects needed for solving uni-modal as well as multi-modal optimization problems. Extensive simulation results have demonstrated mixed performances on the test problems. Although in most problems, the desired accuracy could not be achieved, our algorithm has been able to find solutions *close* to the optimum function value, relative to the function values of the landscape. Moreover, our algorithm has not shown any remarkable sensitivity to (i) noise in the function values and rotation of the landscape. Comparisons with other algorithms should test the efficacy of the proposed procedure.

Bibliography

[1] K. Deb. A population-based algorithm-generator for real-parameter optimization. *Soft Computing*, in press.

[2] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strageties: The covariance matrix adaptation. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 312–317, 1996.

[3] H.-P. Schwefel. Collective intelligence in evolving systems. In W. Wolff, C. J. Soeder, and F. Drepper, editors, *Ecodynamics – Contributions to Theoretical Ecology*, pages 95–100. Berlin: Springer, 1987.

[4] R. Storn and K. Price. Differential evolution – A fast and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[5] K. Deb, D. Joshi, and A. Anand. Real-coded evolutionary algorithms with parent-centric recombination. In *Proceedings of the 2002 Congress on Evolutionary Computation (IEEE CEC-02)*, Honolulu, HI, USA, 2002.

[6] L. Eshelman. The CHC adaptive search algorithm. how to have safe search when engaging in nontraditional genetic recombination. *Foundations of Genetic Algorithms*, pages 265–283, 1991.

[7] K. Deb and M. Goyal. A combined genetic adaptive search (GeneAS) for engineering design. *Computer Science and Informatics*, 26(4):30–45, 1996.