Adaptive Discretization for Probabilistic Model Building Genetic Algorithms

Chao-Hong Chen, Wei-Nan Liu, Ying-Ping Chen

GECCO 2006

PMBGA

- "Linkage learning" in GAs is the identification of building blocks to be conserved under crossover.
- Probability model building genetic algorithms achieve linkage learning via exploring the probability models of genes.
- In the literature, several attempts to apply PMBGAs to problems in the continuous domain have been made, including continuous PBIL with Gaussian distribution, real-coded PBIL, BEA, and the real-coded BOA.

Discretization

- Most GAs work in discrete domain. If we want to use GAs to solve problems in continuous domain, some discretization must be made.
- In this paper, a new discretization scheme is proposed, which can be integrated into PMBGAs easily.
- ECGA is extended to real-coded ECGA as an example.

How to simulate the process of GA?

- Assume the population size n approaches infinite, and we want to perform simple GA on this population.
- Randomly pick S individual. The winner replace other S-1 loser.
- Perform n selection to generate n individuals.
- Repeat 2 and 3 until the population converges.

Procedure of cGA

1. Initialize an L-dimensional probability vector $P[]$ to 0.5
   $P = \{0.5, 0.5, 0.5, \ldots, 0.5\}$
2. S solutions are generated by polling this vector
3. The gene positions of the fittest of these S solutions are rewarded
4. Repeat step 2 and 3 until the $P[]$ vector implies a single solution

Example of cGA

1. Assume $L = 4, S = 4, E = 0.01$. Initialize $P = \{0.5, 0.5, 0.5, 0.5\}$. The fitness function is OneMax.
2. Generate 2 chromosomes:
   0111 fitness=3
   1010 fitness=2
   1000 fitness=1
   0100 fitness=1
3. Since best chromosome is 0111, $P = \{0.5 - 0.02, 0.5 + 0.02, 0.5 + 0.02, 0.5 + 0.03\}$
4. Repeat until all values in P are zeroes or ones.

Two main assertions

- Learning a "good" probability distribution is equivalent to learning linkage.
- One "good" distribution can be found by searching for a jointly small representation of two components:
  1. the compressed representation of the population under the given distribution
  2. the distribution's representation given the problem encoding

Probabilistic Optimization and Linkage Learning

1. GA's population can be interpreted as representing a probability distribution over the set of future solutions to be explored.
2. In the sense, the role of crossover can be played by a more direct representation of the distribution itself, like what the cGA does.
3. Therefore, a good probability distribution is equivalent to linkage learning.

Minimum Description Length Models

Mitchell, *Machine Learning*:
By reliance on Occam's Razor, good distribution are those under which the representation of the distribution using the current encoding, along with the representation of the population compressed under that distribution, is minimal.

Implements of MDL models

Model Complexity $=$

$$logN \times \sum_{i=1}^{n_s} 2^{s_i}$$

Compressed Population Complexity $= N \sum_{i=1}^{n_s} E_i$

$$E_i = \sum_{j=1}^{2^{s_i}} -p_j log_2 p_j$$

- E_i: the entropy of ith subset
- s_i: the size of ith subset
- N: population size

Marginal Product Model (MPM)

A marginal probability model partitions genes into several subsets.
Each subset has it own distribution.
Here is an example of MPM over four genes:

[0,3]		[1]		[2]	
00	0.5	0	0.5	0	0.6
01	0	1	0.5	1	0.4
10	0				
11	0.5				

Combine MPM and MDL model

Population: 1010, 1101, 0010, 1001

MPM 1

[0,3]		[1]		[2]	
00	1	0	3	0	2
01	0	1	1	1	2
10	1				
11	2				

MPM 2

[0,1]		[2]		[3]	
00	0	0	2	0	2
01	1	1	2	1	2
10	2				
11	1				

- Model complexities of both distributions are $log_2 4(2^2 + 2^1 + 2^1) = 16$.
- The population complexity of second distribution is 3.5, and the population complexity of first distribution is 3.3133.

Procedure of ECGA

1. Generate a random population of size N.
2. Perform tournament selection with tournament size s.
3. Model the population using a greedy MPM search.
   - Initial: assume that all genes are independent.
   - Attemp to merge all pairs of subsets, choosing the best merging result as new probability distribution.
   - Repeat until there's no imporvement.
4. Building-block-wise crossover.
5. Calculate fitnesses.

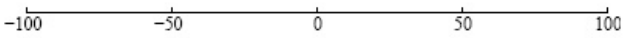
Discretization

1. Discretization is a must, but it's the most dangerous step.
2. Without discretization, the genotype space becomes very huge.
3. Fixed-length vs. varied-length coding.

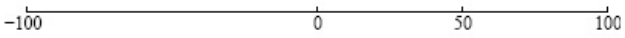
Code Tables

Dimension 1				Dimension 2			
Interval			Code	Interval			Code
-100	∼	-50	0	-100	∼	0	0
-50	∼	0	1	0	∼	50	1
0	∼	50	2	50	∼	100	2
50	∼	100	3				

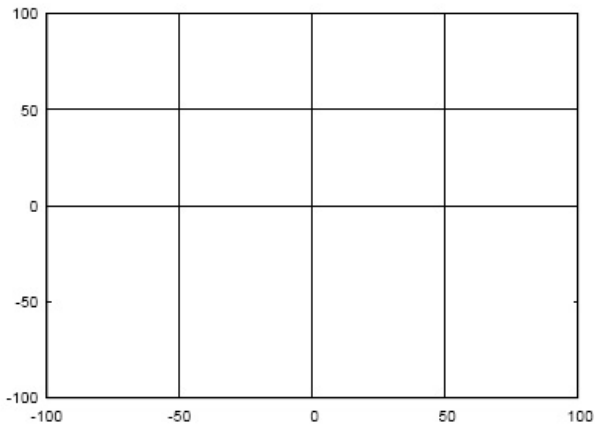
Figure 1: An example code table constructed by Split-on-Demand for a real-parameter optimization problem of two dimensions.



(a) Split configuration on dimension 1.



(b) Split configuration on dimension 2.



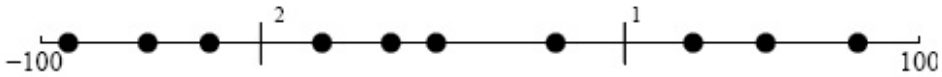
(c) Combined split configuration on both dimensions.

Figure 2: Illustration of the solution space split according to the code table given in Figure 1.

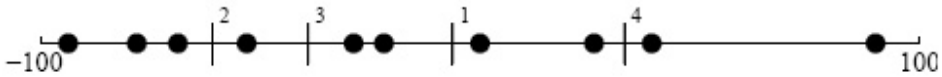
Split-on-Demand Process

1. Random choose a middle point between lower bound and upper bound. And split the dimension into two parts.
2. Count the number of individuals in two regions.
3. If the number of individuals is not less than $N \times \gamma$, split the region recursively.
4. Each dimension has its own partition.

Code Tables



(a) Population distribution and 2 split positions at generation 1. $\gamma = 0.5$. $10 \times \gamma = 5$.



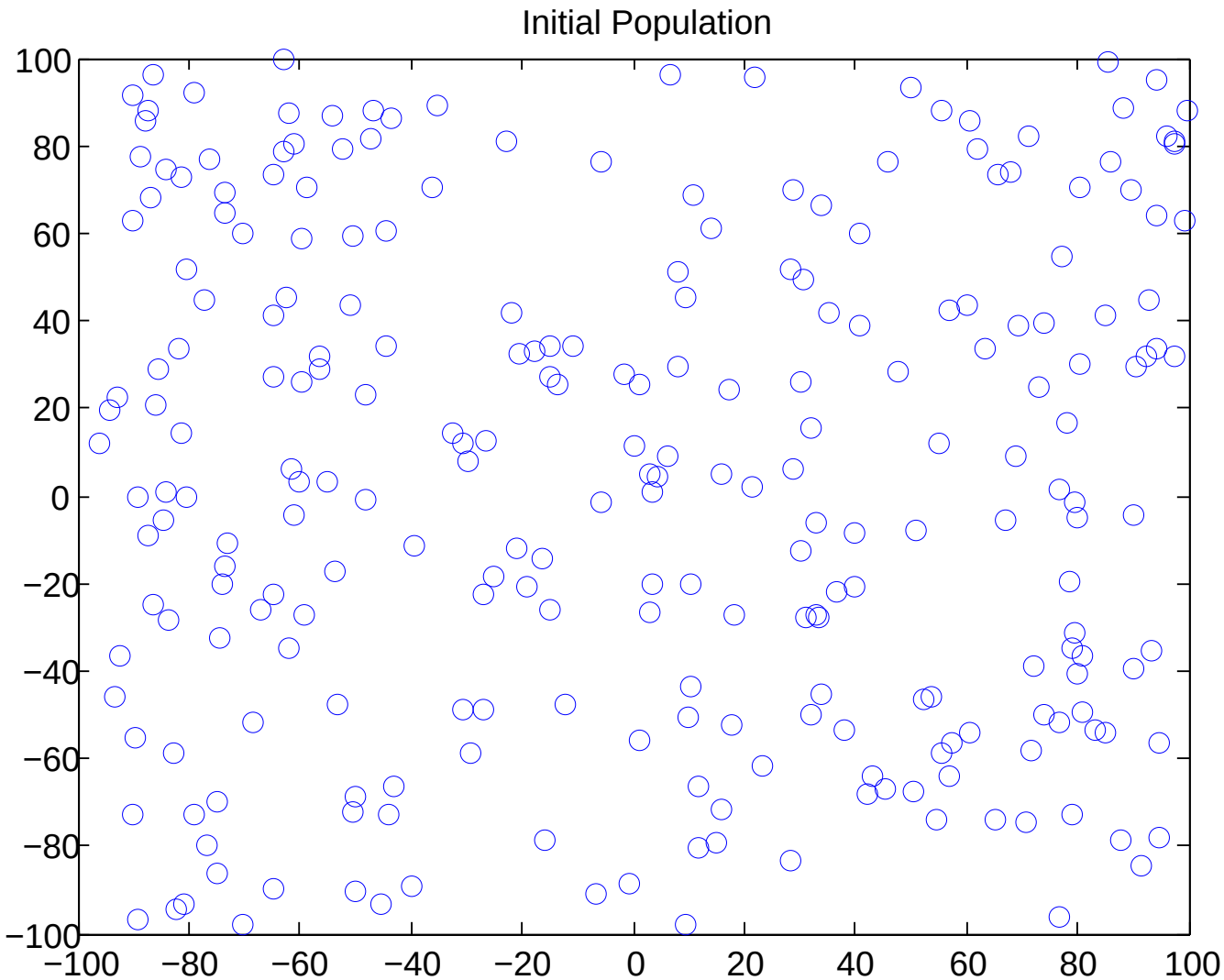
(b) Population distribution and 4 split positions at generation 10. $\gamma = 0.4$. $10 \times \gamma = 4$.

(c) Population distribution and 5 split positions at generation 20. $\gamma = 0.3$. $10 \times \gamma = 3$.

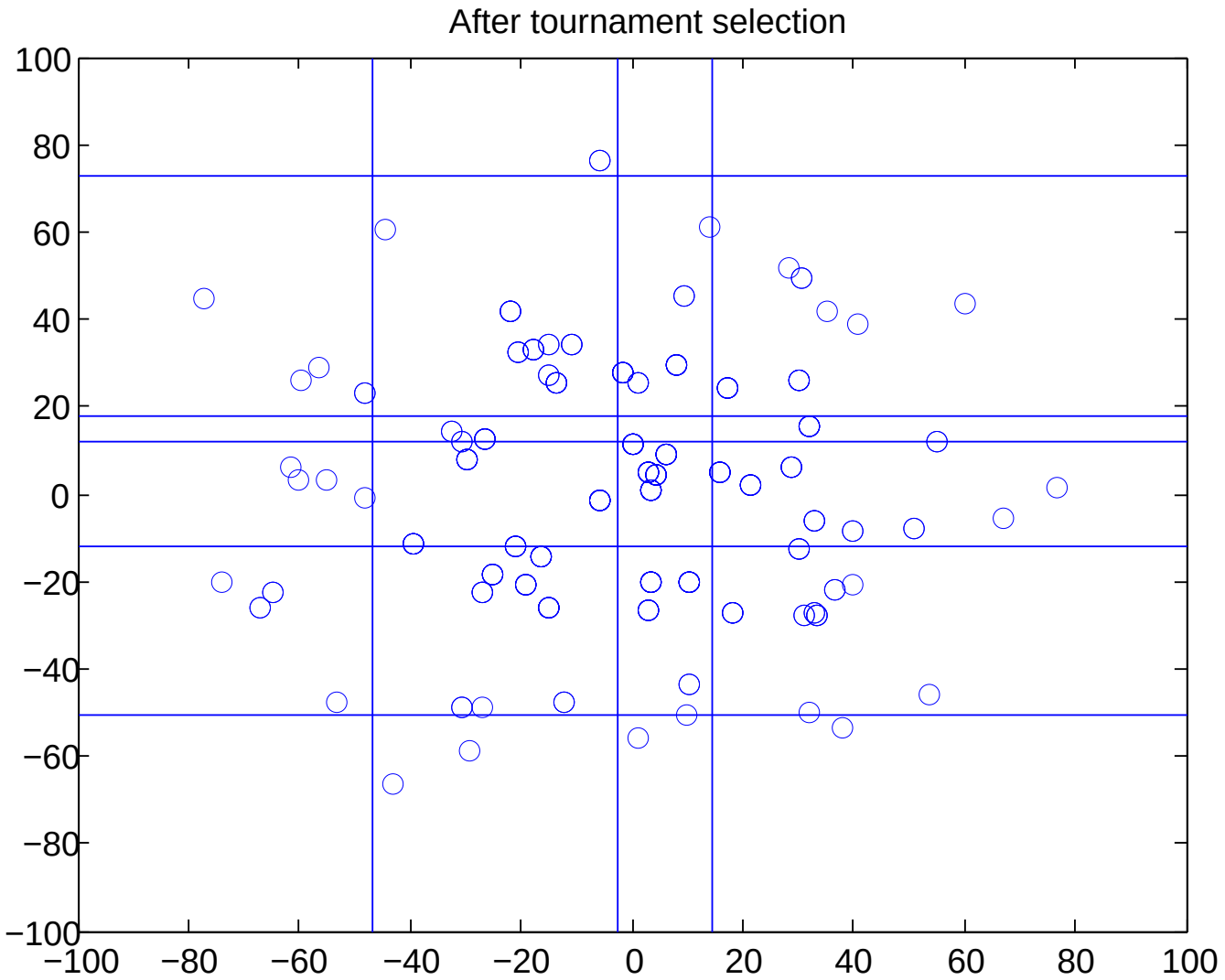
real-coded ECGA

1. Generate a random population of size N
2. Tournament selection
3. Use SoD to encode each dimension
4. Model the population with MPM search
5. Perform a BB-wised crossover
6. Every L generations, perform local search
7. Return to step 2

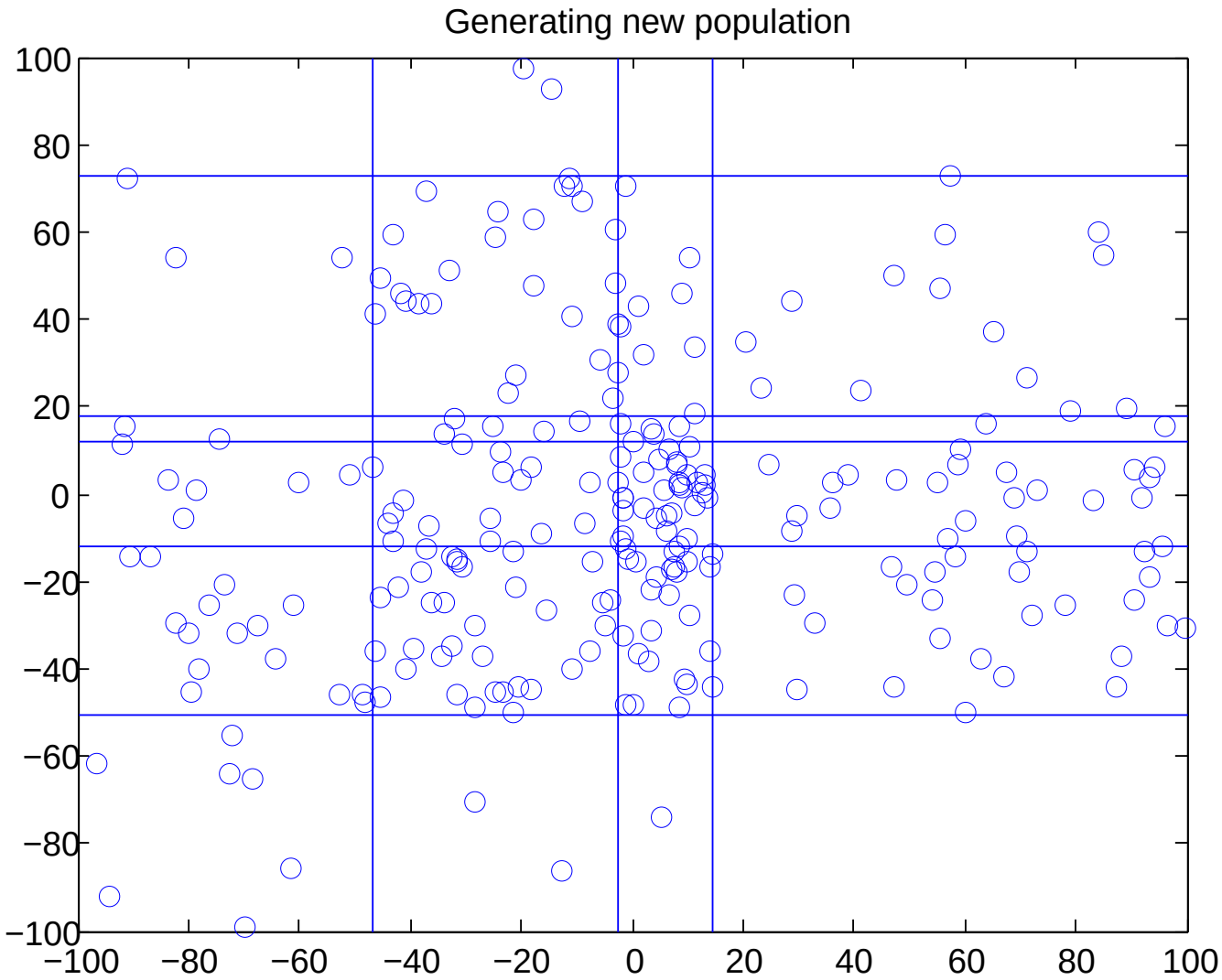
Example on Sphere Model



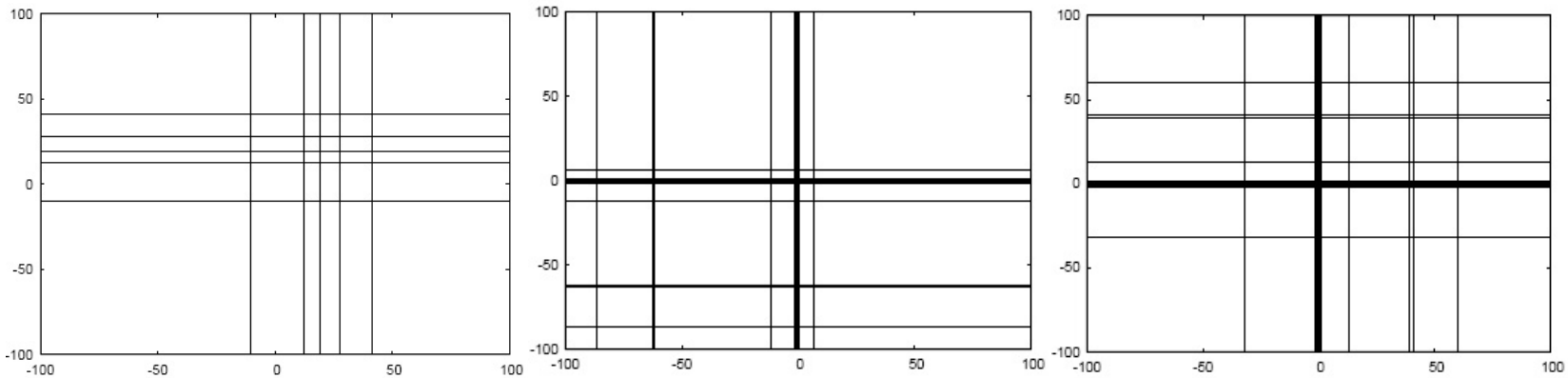
Example on Sphere Model



Example on Sphere Model



Example on Sphere Model



Experiments

Population size $= 250$, $P_c = 0.975$, tournament size $= 8$, $\gamma = 0.5$, $\epsilon = 0.998$, $L = 5$

Function		1†	2†	3†	4†	5‡
1e3	Best	1.30e+03	2.79e+03	1.07e+07	4.37e+03	3.70e+03
	Median	2.31e+03	5.76e+03	4.13e+07	8.14e+03	6.86e+03
	Worst	3.62e+03	1.22e+04	9.05e+07	1.45e+04	9.03e+03
	Mean	2.34e+03	6.40e+03	4.33e+07	8.33e+03	6.77e+03
1e4	Best	3.71e-02	1.31e+01	9.15e+04	2.37e+01	5.43e+00
	Median	2.42e-01	5.81e+01	1.01e+06	7.49e+01	4.22e+01
	Worst	1.39e+00	1.52e+02	3.52e+06	3.08e+02	2.21e+02
	Mean	3.37e-01	5.92e+01	1.22e+06	1.03e+02	6.24e+01
1e5	Best	5.68e-14	1.14e-13	1.14e-13	6.06e-08	1.23e-04
	Median	5.68e-13	1.08e-12	2.16e-12	5.98e-05	7.44e-04
	Worst	2.33e-12	2.26e-11	2.58e+02	1.40e-02	5.48e+00
	Mean	7.16e-13	2.37e-12	1.03e+01	1.25e-03	3.78e-01

† Considered solved according to the given accuracy.
‡ Comparable to the results obtained by other algorithms.

Table 1: Error values for function 1–5.

Function		16	17	18	19	20‡
1e3	Best	2.17e+02	2.88e+02	1.07e+03	1.01e+03	1.08e+03
	Median	3.30e+02	3.61e+02	1.12e+03	1.10e+03	1.12e+03
	Worst	4.13e+02	4.69e+02	1.16e+03	1.16e+03	1.16e+03
	Mean	3.22e+02	3.62e+02	1.12e+03	1.10e+03	1.12e+03
1e4	Best	1.01e+02	1.17e+02	4.15e+02	3.83e+02	4.48e+02
	Median	1.29e+02	1.58e+02	9.17e+02	8.01e+02	8.93e+02
	Worst	1.90e+02	2.21e+02	1.01e+03	9.69e+02	1.02e+03
	Mean	1.30e+02	1.60e+02	8.34e+02	8.14e+02	8.36e+02
1e5	Best	9.87e+01	1.08e+02	3.00e+02	3.00e+02	3.00e+02
	Median	1.23e+02	1.29e+02	9.08e+02	8.00e+02	8.88e+02
	Worst	1.55e+02	1.51e+02	1.00e+03	9.64e+02	1.01e+03
	Mean	1.22e+02	1.30e+02	7.79e+02	7.95e+02	7.73e+02

† Considered solved according to the given accuracy.
‡ Comparable to the results obtained by other algorithms.

Table 4: Error values for function 16–20.

Function		6†	7†	8‡	9†	10
1e3	Best	3.58e+07	1.42e+03	2.05e+01	4.87e+01	5.95e+01
	Median	1.08e+08	1.64e+03	2.08e+01	6.78e+01	8.52e+01
	Worst	3.98e+08	1.83e+03	2.09e+01	8.12e+01	1.05e+02
	Mean	1.19e+08	1.66e+03	2.08e+01	6.66e+01	8.53e+01
1e4	Best	8.83e+01	1.23e+03	2.04e+01	3.75e-01	9.32e+00
	Median	4.06e+02	1.24e+03	2.05e+01	5.46e+00	1.47e+01
	Worst	3.41e+03	1.25e+03	2.06e+01	1.13e+01	3.49e+01
	Mean	8.00e+02	1.24e+03	2.05e+01	5.65e+00	1.82e+01
1e5	Best	3.41e-13	9.86e-03	2.00e+01	1.14e-13	4.98e+00

Function		21‡	22	23‡	24‡	25
1e3	Best	1.14e+03	9.64e+02	1.02e+03	9.34e+02	1.82e+03
	Median	1.30e+03	1.01e+03	1.31e+03	1.19e+03	1.86e+03
	Worst	1.35e+03	1.08e+03	1.35e+03	1.34e+03	1.89e+03
	Mean	1.28e+03	1.02e+03	1.29e+03	1.17e+03	1.86e+03
1e4	Best	5.00e+02	7.75e+02	5.60e+02	2.00e+02	1.69e+03
	Median	8.50e+02	7.87e+02	5.60e+02	2.01e+02	1.75e+03
	Worst	1.14e+03	9.05e+02	1.25e+03	2.05e+02	1.78e+03
	Mean	7.72e+02	7.96e+02	8.48e+02	2.02e+02	1.75e+03
1e5	Best	3.00e+02	7.32e+02	5.60e+02	2.00e+02	1.49e+03