University of Ostrava

Institute for Research and Applications of Fuzzy Modeling

Controlled Random Search with Competition: Application to Non-Linear Regression

Josef Tvrdík

Research report No. 90

2005

Submitted/to appear:

Will be included into the paper prepared for Computational Statistics and Data Analysis

Supported by:

Grant No. 201/05/0284 of Czech Grant Agency and MSM 6198898701

University of Ostrava
Institute for Research and Applications of Fuzzy Modeling
30. dubna 22, 701 03 Ostrava 1, Czech Republic

tel.: +420-59-6160234 fax: +420-59-6120 478
e-mail: josef.tvrdik@osu.cz

1 Introduction

Let us consider the continuous global optimization problem with box constraints, i.e. for a given objective function $f : D \to \mathcal{R},\ D \subset \mathcal{R}^d$, the point $\mathbf{x}^*$ is to be found such that $\mathbf{x}^* = \arg\min_{\mathbf{x}\in D} f(\mathbf{x})$. The point $\mathbf{x}^*$ is called the global minimum, D is the search space defined as $D = \prod_{i=1}^{d}[a_i, b_i]$, $a_i < b_i$, $i = 1, 2, \ldots, d$ and the objective function is computable, i.e. there is an efficient algorithm capable to evaluate $f(\mathbf{x})$ with sufficient accuracy in any point $\mathbf{x} \in D$.

The global optimization problem formed above is NP-hard in general but there are many stochastic algorithms where the search for the global minimum is heuristic. The stochastic algorithms are usually population-based and can be viewed as models of evolutionary processes in biological population [2]. Authors of evolutionary algorithms often claim the efficiency and the reliability of searching for the "true" global minimum. The "true" global minimum is found by a search algorithm if the point with minimal function value found in the process is sufficiently close to the global minimum. However, using such algorithms we face the problem of the setting their tuning parameters. The efficiency and the reliability of many algorithms are strongly dependent on the values of input parameters and recommendations of authors are rather vague, see e.g. [10], [18]. A user is supposed to be able to change parameter values according to partial results obtained in search process. Such attempt is not acceptable in tasks where the global optimization is one step only on the way to the solution of the problem and the user has no experience in fine art of parameter setting. Examples of these problems can be found in computational statistics like parameter estimation in non-linear regression models or maximization of likelihood functions.

It would be very useful to have a robust algorithm reliable enough at reasonable time-consumption without the necessity of fine setting its input parameters. Such an ideal algorithm cannot be found, see "No free lunch theorem" which states that any search algorithm cannot outperform the others for all objective functions [16]. But it is also known that some algorithms can outperform others for relatively wide class of problems both in convergence rate and in reliability of finding the global minimum. It is supposed that the self-adaptation is enhanced if cooperation or competition are involved in the algorithms. Such adaptive population-based algorithms with competition of heuristics were studied recently [11], [12], [13]. The aim of this paper is to provide a stochastic algorithm with high self-adaptation applicable in standard tasks of non-linear regression.

2 Controlled Random Search with Competing Heuristics

Controlled random search (CRS) is a simple stochastic population-based algorithm searching for the global minimum. CRS algorithm was proposed by Price [8] in 1977. There are several modifications of the CRS algorithm which were successfully used in solving the global optimization problems [1]. The CRS algorithm belongs to the claas of evolutionary algorithms, see Křivý et al. [4]. The CRS algorithm can be written in pseudo-code as follows:

```
1  generate P (population of N points in D at random);
2  find x_max (the point in P with the highest function value);
3  repeat
4    generate a new trial point y ∈ D by a heuristic;
5    if f(y) < f(x_max) then
6      x_max := y;
7      find new x_max;
8    endif
9  until stopping condition;
```

A heuristic used at line 4 is any non-determinstic rule generating a new trial point $\mathbf{y} \in D$. Price used the reflection in simplex known from simplex method proposed by Nelder and Mead [7]. The simplex is $d + 1$ point from P randomly chosen at each step. However, there are many different heuristics that can be used. Moreover, it is not necessary to use the same heuristic within the whole search process, several heuristics can alternate. Let us have h heuristics at our disposal and at each iteration step a heuristic is chosen at random with the probability $q_i,\ i = 1, 2, \ldots, h$.

The probabilities are changing according to the successfulness of heuristics in previous steps of search process. The heuristic is successful in the current step of evolutionary process if it generates such a trial point $\mathbf{y}$ that $f(\mathbf{y}) < f_{\max}$, $f_{\max} = f(\mathbf{x}_{\max})$. Probabilities q_i can simply be evaluated as frequency of success

$$q_i = \frac{n_i + n_0}{\sum_{j=1}^{h}(n_j + n_0)}\ , \tag{1}$$

where $n_0 > 0$ is a constant. The setting $n_0 \geq 1$ prevents a dramatic change in q_i by one random successful use of the i-th heuristics. Other way is the weighting of success by

$$w_i = \frac{f_{\max} - \max(f(\mathbf{y}), f_{\min})}{f_{\max} - f_{\min}}. \tag{2}$$

Thus $w_i \in (0,\ 1]$ and probability q_i is evaluated as

$$q_i = \frac{W_i + w_0}{\sum_{j=1}^{h}(W_j + w_0)}\ , \tag{3}$$

where W_i is sum of w_i in previous search and $w_0 > 0$ is an input parameter of the algorithm. In order to avoid the degeneration of evolutionary process the current values of q_i are reset to their starting values ($q_i = 1/h$) when any probability q_i decreases below a given limit $\delta > 0$.

Four heuristics were used in implementations of the CRS algorithm with competing heuristics. Three of them are based on randomized reflection in the simplex S ($d+1$ points chosen from P) proposed in [3]. A new trial point $\mathbf{y}$ is generated from the simplex by the relation

$$\mathbf{y} = \mathbf{g} + U(\mathbf{g} - \mathbf{x}_H). \tag{4}$$

where $\mathbf{x}_H = \arg\max_{\mathbf{x}\in S} f(\mathbf{x})$ and $\mathbf{g}$ is the centroid of remaining d points of the simplex S. The multiplication factor U is a random variable distributed uniformly in $[s, \alpha - s)$, where α and s are input parameters, $0 < s < \alpha/2$. All the $d+1$ points of simplex are chosen at random in two heuristics with $\alpha = 2$, $s = 0.5$ and $\alpha = 5$, $s = 1.5$, resp. These heuristics are denoted Refl1 and Refl25 in the text. In the third heuristic denoted as Reflb one point of simplex is the point of P with the minimal function value and remaining d points of simplex S are chosen at random, $\alpha = 2$, $s = 0.5$ in the numerical experiments.

The fourth heuristic was based on differential evolution [10]. A point $\mathbf{u}$ is generated according to

$$\mathbf{u} = \mathbf{r}_1 + F(\mathbf{r}_2 - \mathbf{r}_3), \tag{5}$$

where $\mathbf{r}_1, \mathbf{r}_2$ and $\mathbf{r}_3$ are three distinct points taken randomly from P and $F > 0$ is an input parameter. The elements y_j, $j = 1, 2. \ldots, d$ of trial point $\mathbf{y}$ are built up by the crossover of randomly taken $\mathbf{x}$ (not coinciding with the current $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$) and $\mathbf{u}$ using the following rule:

$$y_j = \begin{cases} u_j & \text{if } U_j \leq C \quad \text{or} \quad j = l \\ x_j & \text{if } U_j > C \quad \text{and} \quad j \neq l, \end{cases} \tag{6}$$

where l is a randomly chosen integer from $\{1, 2, \ldots, d\}$, $U_1, U_2, \ldots, U_d$ are independent random variables uniformly distributed in $[0,\ 1)$, and $C \in [0, 1]$ is an input parameter influencing the number of elements to be exchanged by crossover. Ali and Törn [1] suggested to adapt the value of the scaling factor F within search process according to the equation

$$F = \begin{cases} \max(F_{\min}, 1 - |\frac{f_{\max}}{f_{\min}}|) & \text{if } |\frac{f_{\max}}{f_{\min}}| < 1 \\ \max(F_{\min}, 1 - |\frac{f_{\min}}{f_{\max}}|) & \text{otherwise,} \end{cases} \tag{7}$$

where $f_{\min}$, $f_{\max}$ are respectively the minimum and maximum function values in the population and $F_{\min}$ is an input parameter ensuring $F \in [F_{\min}, 1)$. The heuristic which uses Eq.(7) for evaluation of F with $F_{\min} = 0.4$ and $C = 0.9$ is denoted DERADP in the following text.

The algorithms evaluating probabilities q_i according to the Eq. (1) and Eq. (3) are denoted COMP1 and COMP4 resp.

3 Non-Linear Parameters Estimation

3.1 Non-Linear Regression Model

In non-linear regression model, we suppose that elements of random vector $\mathbf{Y}$ are expressed as follows

$$Y_i = g(\mathbf{x}_i, \boldsymbol{\beta}) + \varepsilon_i, \quad i = 1, 2, \ldots, n, \tag{8}$$

where $\mathbf{x}_i^T = (x_1, x_2, \ldots, x_k)$ is i-th row of regressor matrix $\mathbf{X}$, $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$ is vector of parameters and g is a given function. Estimation of parameters by the least squares method means to find such estimates that minimize the residual sum of squares $Q(\boldsymbol{\beta})$ given by the following equation

$$Q(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left[Y_i - g(\mathbf{x}_i, \boldsymbol{\beta})\right]^2 \tag{9}$$

3.2 Results of Standard Statistical Software

Reliability of standard statistical software was tested on eight non-linear regression tasks taken from NIST reference database [9]. All models with higher level of difficulty were selected. Additional information including source of data, starting values of parameters, certified values of parameters, and the corresponding standard deviations are also summarized in [9]. The tasks are not easy for standard algorithms, see [6]. Statistical packages use deterministic algorithms for least squares estimation of parameters, namely NCSS 2001, where Levenberg-Marquardt algorithm in NCSS 2001 and SPSS 10.0, Gauss-Newton algorithm in S-PLUS 4.5 and the simplex method in SYSTAT 8.0. Our results when several statistical packages were used are shown briefly in Table 1. The word "failure" means that algorithm stopped at local minimum different from the global one or no solution was found. The results are given in [14] in more detail. The failures of deterministic algorithms are very frequent in spite of the fact that the recommended starting values are very near to the certified values of estimates.

Table 1: Summarized results of statistical packages - eight difficult tasks form NIST

Task	NCSS 2001	SYSTAT 8.0	S-Plus 4.5	SPSS 10.0
Bennett5		failure		
BoxBOD	failure	failure		failure
Eckerle4	failure	failure	failure	
MGH09	failure		failure	
MGH10	failure	failure	failure	failure
Rat42		failure	failure	
Rat43	failure	failure	failure	
Thurber	failure		failure	failure

3.3 Results of COMP1 and COMP4 Algorithms

One hundred of repetitions were carried out for each task in numerical experimental tests with the stochastic algorithms. The search for the global minimum of residual sum of squares was marked successful if the agreement with its certified value was at least in seven digits. The reliability *RP* of the search is measured as percentage of successful searches, time consumption is expressed by average number $\overline{NE}$ of objective function evaluation (Eq. 9) needed to reach the stopping condition. The stopping condition was defined as

$$R^2_{\max} - R^2_{\min} < 1 \times 10^{-12},$$

$R^2_{\max}$ and $R^2_{\min}$ are the highest and the least determination index in population P, determination index R^2 is defined as

$$R^2 = 1 - \frac{Q(\boldsymbol{\beta})}{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}$$

Definition of search spaces is given in Tab. 2. The other tuning parameters of the algorithms in all the experiments were set to the following values:

- population size, $N = 10d$,
- constants used in Eq. 1 and Eq. 3, $n_0 = 1$ and $w_0 = 0.5$,
- limit value for the reset of probabilities q_i to initial values, $\delta = 1/32$

Table 2: search spaces D for test tasks - eight difficult tasks from NIST

		β_1	β_2	β_3	β_4	β_5	β_6	β_7
Bennett5	min	-5000	0	0.1				
	max	-1000	500	10				
BoxBOD	min	1	0.1					
	max	1000	2					
Eckerle4	min	0	1	400				
	max	10	10	500				
MGH09	min	0	0	0	0.01			
	max	100	100	100	100			
MGH10	min	0	0	0				
	max	100	1e+6	1e+5				
Rat42	min	0	0	0				
	max	1000	10	1				
Rat43	min	0	0	0	0.1			
	max	1000	100	1	10			
Thurber	min	0	0	0	0	0	0	0
	max	1e+4	5000	5000	1000	10	10	10

Experimental results obtained with algorithms COMP1 and COMP4 are presented in Table 3. Symbol *vc* means variation coefficient (percentage), *suc* is relative frequency of function evaluations, if $f(\mathbf{y}) < f_{\max}$, R^2 is determination index, *rst* is average number of resets per 1000 objective function evaluations and *cpu* is average time per function evaluation in milliseconds (on very old PC 667 MHz). We see that the COMP4 algorithm was very reliable in finding the global minimum of the objective function, much more reliable than deterministic algorithms summarized in Table 1. Time consumption obtained by test program without any optimization of Matlab code [5] was also acceptable in most tasks. Moreover, as we can see in Figure 1, where the least and the most time-consuming tasks are compared, the algorithm is self-adaptive in selection of proper heuristics used for the search. These results are very promising for the common use of the COMP4 algorithm in estimation of non-linear regression parameters.

Table 3: CRS algorithm with competing heuristics - eight difficult tasks from NIST

	COMP1			COMP4						
Task	*RP*	$\overline{NE}$	*vc*	*RP*	$\overline{NE}$	*vc*	*suc*	1-R^2	*rst*	*cpu*
Bennett5	100	42367	33.0	100	37620	38.2	44	1.06e-5	12.1	3.23
BoxBOD	100	1023	9.4	100	1018	8.9	51	1.20e-1	21.8	2.65
Eckerle4	100	2089	5.9	100	2118	5.9	50	2.94e-3	18.7	2.70
MGH09	100	9552	10.2	100	9316	9.8	44	5.94e-3	14.8	2.69
MGH10	100	20709	7.6	100	21031	10.3	50	4.70e-8	8.5	2.50
Rat42	100	2357	5.7	100	2368	6.2	44	1.73e-3	16.2	2.51
Rat43	100	3777	4.8	100	3779	5.0	42	8.16e-3	16.5	2.76
Thurber	91	12343	3.7	97	12309	3.7	38	4.92e-4	13.9	3.21

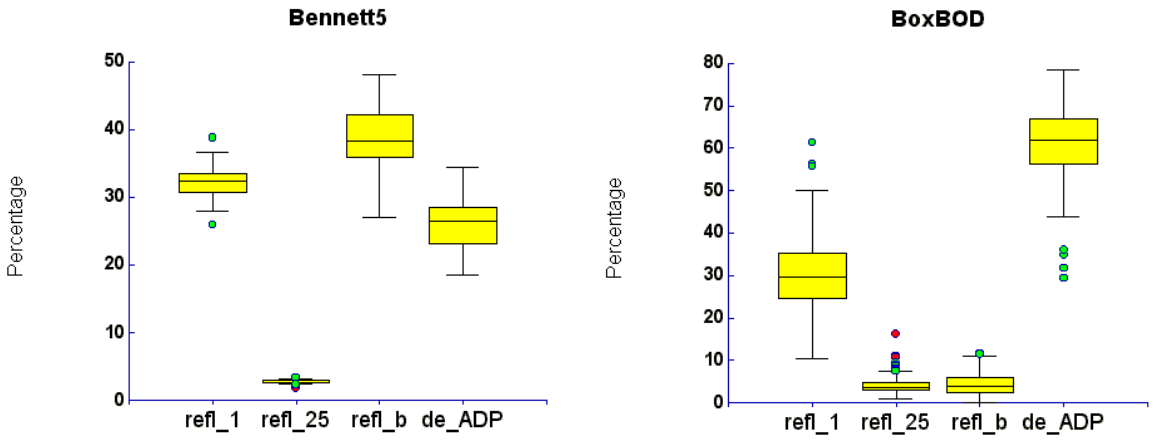

Figure 1: Relative frequencies of used heuristics

3.4 Other Tests

The COMP4 was tested on other tasks of non-linear regression. Input parameters of the algorithm were set the same as in the previous experiments, specifications of search spaces, data and model functions can be found on web page [15]. Fourteen tasks collected in Křivý et al. [4] gave results shown in Table 4. In comparison with the algorithm MCRS used in [4] the COMP4 algorithm has less time-consumption

Table 4: Algorithm COMP4, tasks from paper Křivý et al. [4]

Task	*RP*	$\overline{NE}$	*vc*	*suc*	1-R^2	*rst*	*cpu*
Model 1	100	3177	5.2	48	3.45e-3	13.3	2.45
Model 2	80	3038	12.0	47	4.70e-8	12.1	2.46
Model 3	100	1404	6.3	56	8.08e-4	19.8	2.48
Model 4	100	3071	9.2	42	1.62e-3	14.5	2.34
Model 5	100	21590	6.6	47	6.20e-8	11.3	2.37
Model 6	100	1036	8.8	48	3.77e-1	25.0	2.40
Model 7	100	37061	1.9	48	2.38e-2	15.5	2.61
Model 8	100	32349	4.1	47	2.80e-6	15.6	2.73
Model 9	98	2051	13.1	41	3.84e-3	18.4	2.36
Model 10	100	30428	40.6	36	3.21e-5	12.8	2.80
Model 11	85	24485	33.6	35	3.70e-8	11.5	2.56
Model 12	98	2942	5.1	45	1.89e-3	16.3	2.54
Model 13	100	14297	3.9	42	1.11e-5	16.0	3.11
Model 14	100	19483	3.7	49	4.57e-4	13.5	2.47

in most tasks by about one third. Relatively less reliability in model 2 and model 11 can be explained by very small residual variability (see column $1 - R^2$).

Next set of tests was carried out on tasks of NIST [9], where the data were not generated but experimental. The results are shown in Table 5. Also in this case the results are relatively good with exception of the task Hahn1 where the reliability is 35 per cent only in spite of big time consumption. As it it shown in Figure 2 (*ne* is number of objective function evaluations, *rsucc* is relative frequency of success), the algorithm stopped at a local minimum due to the high preference of the most successful heuristics which do not allow to escape from the region of premature convergence. Time consumption needed for stopping at local minimum different from the global one is significantly less than the time needed for finding the global minimum.

Table 5: Tasks of NIST, low and medium difficulty, experimental data

Task	*RP*	$\overline{NE}$	*vc*	*suc*	1-R^2	*rst*	*cpu*
chwirut1	100	2423	5.1	52	2.00e-2	16.4	3.34
chwirut2	100	2382	6.0	52	1.40e-2	16.5	2.97
danwood	100	1278	9.0	44	5.67e-4	21.7	2.91
enso	91	19554	12.3	35	4.02e-1	12.6	4.44
hahn1	35	15844	4.9	40	1.96e-4	14.6	4.64
kirby2	100	6834	3.1	40	2.85e-5	14.1	3.44
misra1a	100	1802	8.0	44	1.84e-5	18.3	2.81
misra1b	100	1553	9.3	42	1.12e-5	17.4	2.95
misra1c	100	1743	9.8	44	6.10e-6	18.7	2.94
misra1d	100	1763	9.9	44	8.30e-6	16.6	2.79
nelson	100	5263	5.9	47	6.98e-2	12.7	3.18
roszman1	95	5145	4.5	41	1.59e-3	16.1	2.99

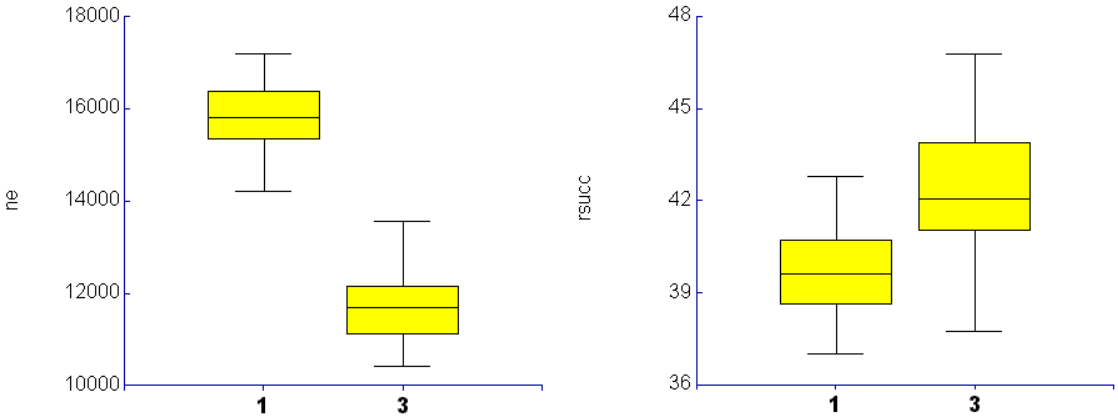

Figure 2: Comprarison of successful (1) and unsuccessful (3) runs - Hahn1

4 Conclusions

In spite of the promising results on eight difficult tasks of NIST the other tests showed that the COMP4 is not reliable enough to recommend it for a blind application to non-linear model parameter estimation. But the obtained results proved substantially higher reliability of the algorithm in comparison with deterministic ones used in standard statistical software and the COMP4 can be used at least as an alternative more reliable way of regression parameter estimation. The use of evolutionary algorithms do not need the specification of starting values of estimates which is often fine art deeply influencing the convergence of deterministic algorithms in standard software. Implementation of COMP4 in Matlab (not very friendly yet) is available on web site [15]. The research of self-adaptive algorithms and their implementation for convenient use in computational statistics will continue.

References

[1] Ali, M.M., Törn, A.: Population set based global optimization algorithms: Some modifications and numerical studies, Computers and Operations Research **31** (2004) 1703 – 1725.

[2] Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, New York (1996)

[3] Křivý, I., Tvrdík, J.: The Controlled Random Search Algorithm in Optimizing Regression Models. Comput. Statist. and Data Anal. **20** (1995) 229 – 234.

[4] Křivý I., Tvrdík J. Krpec, R.: Stochastic algorithms in nonlinear regression. Comput. Statist. and Data Anal. **33** (2000) 278 – 290.

[5] MATLAB, version 6.5.1, The MathWorks, Inc. (2003).

[6] McCullough, B.D., Wilson, B.: On the accuracy of statistical procedures in Microsoft Excel 97. Comput. Statist. and Data Anal. **31** (1999) 27 – 37.

[7] Nelder, J.A., Mead, R.: A Simplex Method for Function Minimization. Computer J. **7** (1964) 308 – 313.

[8] Price, W. L.: A Controlled Random Search Procedure for Global Optimization. Computer J. **20** (1977) 367 – 370.

[9] Statistical Reference Datasets. Nonlinear regression. NIST Information Technology Laboratory. http://www.itl.nist.gov/div898/strd/. December 1, 2001.

[10] Storn, R., Price, K.: Differential evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. J. Global Optimization **11** (1997) 341 – 359.

[11] Tvrdík, J., Křivý, I., Mišík, L.: Evolutionary Algorithm with Competing Heuristics. In: Ošmera, P. (ed.): MENDEL 2001, 7th International Conference on Soft Computing. Technical University, Brno (2001) 58 – 64.

[12] Tvrdík, J., Mišík, L., Křivý, I.: Competing Heuristics in Evolutionary Algorithms. In: Sinčák, P. et al. (eds.): Intelligent Technologies – Theory and Applications. IOS Press, Amsterdam (2002) 159 – 165.

[13] Tvrdík, J.: Generalized controlled random search and competing heuristics. In: Matoušek, R. and Ošmera, P. (eds): MENDEL 2004, 10th International Conference on Soft Computing. Technical University, Brno (2004) 228 – 233.

[14] Tvrdík, J., Křivý, I.: Comparison of algorithms for nonlinear regression estimates. In: Antoch, J. (ed.): COMPSTAT 2004. Physica-Verlag (2004) 1917 – 1924.

[15] Tvrdík, J.: Global Optimization, Evolutionary Algorithms and their Application to Computational Statistics, WEB: http://albert.osu.cz/tvrdik/down/global_optimization.html, update November 22, 2005

[16] Wolpert, D.H., Macready, W.G. No Free Lunch Theorems for Optimization. IEEE Transactions on Evolutionary Computation **1** (1997) 67 – 82.

[17] Zelinka, I., Lampinen, J.: SOMA – Self-Organizing Migrating Algorithm. In: Ošmera, P. (ed.): MENDEL 2000, 6th Int. Conference on Soft Computing. Technical University, Brno, Brno (2000) 177 – 187.

[18] Zelinka, I.: Artificial intelligence in global optimization problems. (in Czech) BEN, Praha (2002)