



# Adaptive Differential Evolution: Application to Nonlinear Regression

Josef Tvrđík

University of Ostrava, Department of Computer Science \*  
30. dubna 22, 701 03 Ostrava, Czech Republic  
Josef.Tvrdik@osu.cz  
<http://albert.osu.cz/tvrdik>

**Abstract.** Adaptation of control parameters in differential evolution is considered. Five adaptive variants of differential evolution are applied to the estimation of parameters in nonlinear regression models and their performance is compared experimentally with the adaptive controlled random search algorithm tailored especially for these problems. The NIST nonlinear regression datasets are used as a benchmark. Two of five tested variants of adaptive differential evolution perform almost as reliable as the adaptive controlled random search algorithm and one of these two variants converges only slightly slower and its time requirements are almost comparable with the adaptive controlled random search.

## 1 Introduction

The global optimization problem in this paper follows the form:

$$\text{minimize } f(\mathbf{x}) \text{ subject to } \mathbf{x} \in D,$$

where  $\mathbf{x}$  is a continuous variable with the domain  $D \subset \mathbb{R}^d$ , and  $f(\mathbf{x}) : D \rightarrow \mathbb{R}$  is a continuous function. The domain  $D$  is defined by specifying lower ( $a_j$ ) and upper ( $b_j$ ) limits of each component  $x_j$ ,  $D = \prod_{j=1}^d [a_j, b_j]$ ,  $a_j < b_j$ ,  $j = 1, 2, \dots, d$ . This specification of  $D$  is also called the *box constraints*. The global minimum point  $\mathbf{x}^* = \arg \min_{\mathbf{x} \in D} f(\mathbf{x})$  is the solution of the problem.

Differential evolution (DE) introduced by Storn and Price [10] is simple but powerful evolutionary algorithm for global optimization over the box-constrained search space. The differential evolution has become one of the most frequently used algorithms for solving the continuous global optimization problems in recent years [8].

The aim of this paper is to compare numerically some self-adaptive variants of differential evolution with the specialized adaptive controlled random search algorithm [15] in the estimation of nonlinear-regression parameters. Section 2 concerns with the differential evolution and the setting of its control parameters. Principles of adaptive variants of differential evolution that have been used

\* Supported by the grant 201/05/0284 of the Czech Grant Agency and by MSM 6198898701 of the Institute for Research and Applications of Fuzzy Modeling.

in experimental comparison are described in Section 3. In Section 4 the additive nonlinear regression model is formed and the problem of parameter estimation is mentioned briefly. Numerical experiments on NIST nonlinear-regression benchmark tests are presented in Section 5. Some concluding remarks are made in Section 6.

## 2 Differential Evolution and its Control Parameters

The algorithm of differential evolution in pseudo-code is written as Algorithm 1.

**Algorithm 1.** Differential evolution

```

1  generate an initial population  $P = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N)$ ,  $\mathbf{x}_i \in D$ 
2  repeat
3    for  $i := 1$  to  $N$  do
4      generate a new trial vector  $\mathbf{y}$ 
5      if  $f(\mathbf{y}) < f(\mathbf{x}_i)$  then insert  $\mathbf{y}$  into new generation  $Q$ 
6      else insert  $\mathbf{x}_i$  into new generation  $Q$ 
7    endif
8  endfor
9   $P := Q$ 
10 until stopping condition

```

A new trial point  $\mathbf{y}$  is generated by using mutation and crossover. There are various strategies how to generate the mutant point  $\mathbf{u}$ . The most popular strategy called DE/rand/1/ generates the point  $\mathbf{u}$  by adding the weighted difference of two points

$$\mathbf{u} = \mathbf{r}_1 + F(\mathbf{r}_2 - \mathbf{r}_3), \quad (1)$$

where  $\mathbf{r}_1, \mathbf{r}_2$  and  $\mathbf{r}_3$  are three mutually distinct points taken randomly from  $P$  not coinciding with the current  $\mathbf{x}_i$ , and  $F > 0$  is an input parameter. Another strategy called DE/best/2/ generates the point  $\mathbf{u}$  according to formula

$$\mathbf{u} = \mathbf{x}_{\min} + F(\mathbf{r}_1 + \mathbf{r}_2 - \mathbf{r}_3 - \mathbf{r}_4), \quad (2)$$

where  $\mathbf{x}_{\min}$  is the point of  $P$  with minimal function value,  $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_4$  are four mutually distinct points taken randomly from  $P$  not coinciding with the current  $\mathbf{x}_i$  or  $\mathbf{x}_{\min}$ , and  $F > 0$  is an input parameter.

Kaelo and Ali [4] proposed a slightly different attempt to generating a mutant point  $\mathbf{u}$  by (1). They called it random localization. The point  $\mathbf{r}_1$  is not selected randomly but it is tournament best among  $\mathbf{r}_1, \mathbf{r}_2$ , and  $\mathbf{r}_3$ , therefore  $\mathbf{r}_1 = \arg \min_{i \in \{1, 2, 3\}} f(\mathbf{r}_i)$ . Significant improvement in convergence (about 35% of function evaluations) with preserving the reliability of the search is reported for 50 test problems in [4].

The elements  $y_j$ ,  $j = 1, 2, \dots, d$  of the trial point  $\mathbf{y}$  are built up by the crossover of the current point  $\mathbf{x}_i$  and the mutant point  $\mathbf{u}$  using the following rule

$$y_j = \begin{cases} u_j & \text{if } U_j \leq CR \quad \text{or} \quad j = l \\ x_{ij} & \text{if } U_j > CR \quad \text{and} \quad j \neq l, \end{cases} \quad (3)$$

where  $l$  is a randomly chosen integer from  $\{1, 2, \dots, d\}$ ,  $U_1, U_2, \dots, U_d$  are independent random variables uniformly distributed in  $[0, 1)$ , and  $CR \in [0, 1]$  is an input parameter influencing the number of elements to be exchanged by crossover. Eq. (3) ensures that at least one element of  $\mathbf{x}_i$  is changed even if  $CR = 0$ . This kind of crossover according to (3) is called DE/././bin.

The efficiency of differential evolution is very sensitive to the setting of values  $F$  and  $CR$ . The values recommended in literature are  $F = 0.8$  and  $CR = 0.5$ , but even Storn and Price in their principal paper [10] use  $0.5 \leq F \leq 1$  and  $0 \leq CR \leq 1$  depending on the results of preliminary tuning. Several papers deal with the setting of control parameters. Ali and Törn [1] suggested to adapt the value of the scaling factor  $F$  within the search process. Zaharie [20] derived the critical interval for the control parameters. Teo [11, 12] suggested a self-adaptive mechanism changing population size during the search process. Some other attempts to the adaptation of DE control parameters are summarized in Liu and Lampinen [6].

### 3 Adaptive Variants of DE for Experiments

Evolutionary self-adaptation of control parameters  $F$  and  $CR$  was proposed by Brest et al. [3]. The values of  $F$  and  $CR$  are initialized randomly for each point in population and survive together with the individuals, but they can be changed randomly in each generation with respective probabilities  $\tau_1, \tau_2$ . New values of  $F$  are distributed uniformly in  $[F_l, F_u]$  and new values of  $CR$  are also uniform random values  $\in [0, 1]$ .

The competitive setting of the control parameters was introduced by Tvrdík [14]. Let us have  $H$  different settings of  $F$  and  $CR$  used in (3), (1), and (2) and choose among them at random with probabilities  $q_h$ ,  $h = 1, 2, \dots, H$ . The probabilities can be adjusted according to the success rate of the settings in the preceding steps of search process. The  $h$ -th setting is successful, if it generates such a trial point  $\mathbf{y}$  that  $f(\mathbf{y}) < f(\mathbf{x}_i)$ . When  $n_h$  is the current number of the  $h$ -th setting successes, probability  $q_h$  is evaluated as the relative frequency

$$q_h = \frac{n_h + n_0}{\sum_{j=1}^H (n_j + n_0)}, \quad (4)$$

where  $n_0 > 0$  is a constant. The setting of  $n_0 > 1$  prevents a dramatic change in  $q_h$  by one random successful use of the  $h$ -th parameter setting. To avoid process's degeneration, the current values of  $q_h$  are reset to their starting values  $q_h = 1/H$ , if any probability  $q_h$  decreases bellow a given limit  $\delta > 0$ . Several variants of such competitive differential evolution were implemented and tested on a benchmark in [16] and two best performing variants are included into the Matlab program library [17] for free use.

The mutation according to (1) or (2) does not guarantee that a new trial point  $\mathbf{y} \in D$ . In such a case either the point  $\mathbf{y}$  can be skipped and a new one generated or if any  $y_j < a_j$  or  $y_j > b_j$ , all such coordinates  $y_j$  can be reversed via flopping into  $D$  over the  $a_j$  or  $b_j$  by mirroring. The latter method is used in algorithms implemented for numerical tests.

## 4 Nonlinear Regression Model and Estimation of its Parameters

The additive nonlinear regression model is formed as follows:

$$Y_i = g(\mathbf{z}_i, \boldsymbol{\beta}) + \varepsilon_i, \quad i = 1, 2, \dots, n, \quad (5)$$

where  $Y_i$  are random variables,  $\mathbf{z}_i^T = (z_1, z_2, \dots, z_k)$  are rows of regressor matrix  $\mathbf{Z}$ ,  $\boldsymbol{\beta}$  is vector of unknown parameters,  $g$  is a given function nonlinear in parameters, and all  $\varepsilon_i$  are independent and identically distributed (i.i.d.) random variables with zero mean values. The estimation of parameters by the least squares method means to find such estimates  $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_d)^T$  that minimize the residual sum of squares  $f(\hat{\boldsymbol{\beta}})$  given by the equation

$$f(\hat{\boldsymbol{\beta}}) = \sum_{i=1}^n \left[ Y_i - g(\mathbf{z}_i, \hat{\boldsymbol{\beta}}) \right]^2. \quad (6)$$

Unlike the residual sum of squares in linear regression models, the function  $f(\hat{\boldsymbol{\beta}})$  is not convex in general. Due to this fact the estimation of the parameters in nonlinear regression models is the global optimization problem. Iterative deterministic algorithms used for the estimation of parameters in standard statistical packages like Levenberg-Marquardt or Gauss-Newton methods searching for a local minimum fail sometimes, see e.g. [13]. A specially tailored controlled random search algorithm with an adaptive stopping condition was proposed for the estimation of the parameters in nonlinear regression models [15]. Four local-search heuristics compete in this controlled random search algorithm, and the stopping condition is changed adaptively according to the value of determination index (often called R-squared), as achieved in the search. For details see [15], the implementation of this algorithm in Matlab is included into the Program Library [17] as well.

## 5 Experiments and Results

One of the criteria for examining the reliability of algorithm is the comparison of the computed values with the “certified values” from sources that are considered to be reliable. NIST collection of datasets [7] contains 27 nonlinear regression tasks categorized according to their level of difficulty (lower – 8 tasks, average – 11 tasks, and higher – 8 tasks). The nonlinear regression models in the collection are of exponential or rational types, the number of parameters ranging from 2 to 9. The total number of observations varies in a wide range, from 6 to 250. The details about the models are available at NIST web site [7]. The certified values are reported to 11 decimal places for each task. The accuracy of the experimental results obtained by the algorithms is evaluated according to the number of duplicated digits when compared with the certified results. The

number of duplicated digits  $\lambda$  can be calculated via log relative error [9] as

$$\lambda = \begin{cases} 0 & \text{if } \frac{|m-c|}{|c|} \geq 1 \\ 11 & \text{if } \frac{|m-c|}{|c|} < 1 \times 10^{-11} \\ -\log_{10} \left( \frac{|m-c|}{|c|} \right) & \text{otherwise,} \end{cases} \quad (7)$$

where  $c \neq 0$  denotes the certified value and  $m$  denotes the computed value. According to [7], excluding the cases where the certified value is essentially zero (e.g. for the three Lanczos problems), a good nonlinear least squares procedure should be able to duplicate the certified results to at least 4 digits.

Six algorithms were compared in numerical experiments, namely the controlled random search with competing local-search heuristics and adaptive stopping condition tailored for the estimation of regression parameters [15], denoted CRS4e in the text, and five self-adaptive variants of differential evolution. The same adaptive stopping condition was used in all algorithms and tests that were carried out on all 27 NIST nonlinear regression tasks [7]. The algorithms were tested in 100 repeated runs for each task. Such number of repetitions ensures that the asymptotical confidence intervals for the probability of stopping at a good approximation of the global minimum point are acceptably narrow (about 20% or less in the case of 95% confidence interval). Domains  $D$  for all the tasks are given in [15], as well as the values of control parameters for CRS4e.

Population size for all the DE variants in experiments was set to  $N = \max(20, 5d)$ . Among adaptive variants of DE in testing there were four variants with the competition of various ( $F$ ,  $CR$ ) settings. Parameters for the control of competition were set to  $n_0 = 2$ , and  $\delta = 1/(5H)$  in all tasks. These competitive variants of DE are denoted as follows:

- DER9 uses 9 settings from all combinations of values ( $F = 0.5$ ,  $F = 0.8$ ,  $F = 1$ ) and ( $CR = 0$ ,  $CR = 0.5$ ,  $CR = 1$ ). The vector  $\mathbf{y}$  is generated by the standard DE/rand/1/bin strategy according to (1) and (3),
- DER9rl employs the same 9 settings of  $F$  and  $CR$  like DER9, but the mutant vector  $\mathbf{u}$  is generated by using random localization after [4] described in Section 2,
- DEBR18 with 18 settings (a pair of nine combinations  $F$  and  $CR$  used in DER9). For nine settings the trial vector  $\mathbf{y}$  is generated by the standard DE/rand/1/bin strategy, and for the other nine settings by the DE/best/2/bin strategy according to (2) and (3),
- DEBR18rl uses 18 settings of  $F$  and  $CR$  like DEBR18, but with random localization in DE/rand/1/bin strategy.

The last variant of adaptive DE denoted BREST was implemented after [3] with control parameters set up to the values recommended by authors:  $\tau_1 = 0.1$ ,  $\tau_2 = 0.1$ ,  $F_l = 0.1$ , and  $F_u = 0.9$ .

The results of the numerical comparison for each of 27 tasks are presented in Table 1. The time requirement of the task is expressed by the average number of the objective function evaluations ( $ne$ ) needed to reach the stopping condition.

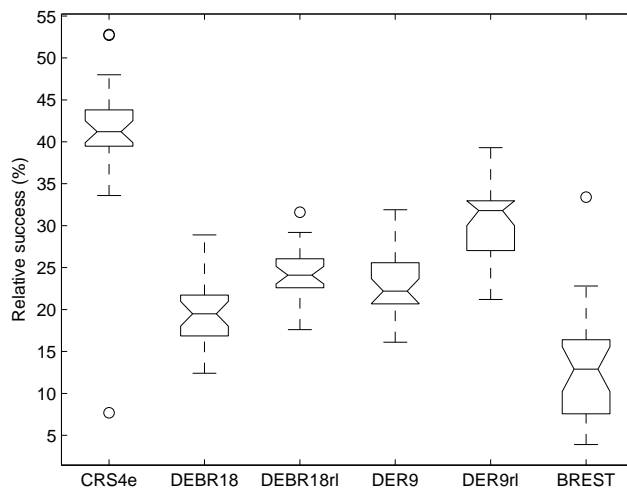
**Table 1.** Comparison of algorithms in estimation of nonlinear-regression parameters (tasks are ordered in groups from low to higher level of difficulty)

Algorithm	CRS4e		DEBR18		DEBR18rl		DER9		DER9rl		BREST	
Task	<i>ne</i>	<i>RP</i>	<i>rne</i>	<i>RP</i>	<i>rne</i>	<i>RP</i>	<i>rne</i>	<i>RP</i>	<i>rne</i>	<i>RP</i>	<i>rne</i>	<i>RP</i>
Chwirut1	1945	100	80	100	41	100	90	100	27	100	207	100
Chwirut2	1934	100	73	100	41	100	90	100	25	100	201	100
DanWood	1173	100	98	100	68	100	89	100	41	100	242	100
Gauss1	9133	100	104	100	46	100	82	100	18	100	87	100
Gauss2	9452	98	134	79	55	78	98	79	20	76	144	88
Lanczos3	30347	100	95	99	60	100	378	99	277	93	675	5
Misra1a	1794	100	110	99	71	99	139	100	96	97	463	95
Misra1b	1512	100	104	100	72	100	171	99	90	99	379	100
ENSO	13375	86	144	100	83	100	111	100	38	99	83	99
Gauss3	10391	99	124	100	39	100	104	100	21	99	273	100
Hahn1	12267	93	39	90	-2	83	49	96	1	85	548	92
Kirby2	6532	100	34	100	-5	100	52	100	-9	100	350	100
Lanczos1	209448	100	-57	99	-68	100	-21	100	-36	100	14	1
Lanczos2	30463	100	96	100	54	100	357	100	253	100	679	7
MGH17	9045	100	40	100	14	100	135	100	77	100	486	100
Misra1c	1868	100	110	100	72	100	284	97	125	97	833	88
Misra1d	1794	100	109	100	69	100	284	99	118	99	736	91
Nelson	4876	100	100	100	67	100	320	94	172	96	536	100
Roszman1	3379	100	22	100	-9	100	13	100	-23	100	163	100
Bennett5	36585	100	103	96	104	91	174	2	135	1	216	1
BoxBOD	828	100	120	100	89	100	99	100	58	100	172	100
Eckerle4	1705	100	47	100	25	100	34	100	5	100	29	100
MGH09	8823	100	38	100	16	100	120	100	71	100	429	100
MGH10	21025	100	90	100	79	100	436	74	340	100	458	9
Rat42	1921	100	72	100	32	100	56	100	8	100	176	100
Rat43	2941	100	28	100	-6	100	18	100	-21	100	151	100
Thurber	9671	100	50	100	2	100	53	100	0	100	475	100
Average		99.1	78	98.6	41	98.2	141	94.0	71	94.1	341	80.6

For easier comparison and better readability, columns denoted *rne* contain the relative changes of *ne* in percents when compared with CRS4e. Therefore, the negative value of *rne* means less time consumption with respect to CRS4e, the positive value means a higher one, the value *rne* = 100 means twice greater time demand in comparison with CRS4e. The reliability of the search in each run was evaluated by the number of duplicated digits in residual sum of squares (6). The percentage of runs with  $\lambda > 4$  (except for Lanczos1 task, where the value of objective function in the global minimum point is essentially zero, the percentage of runs with  $\lambda > 2.4$  is reported) is given in the columns *RP*. As it is shown in Table 1, the tested variants of DE do not outperform the CRS4e algorithm either in the average reliability or in the average convergence. However, the average reliability of DEBR18 and DEBR18rl does not differ significantly from

the reliability of CRS4e. The convergence of DEBR18rl is only slightly slower than CRS4e, with increase of time demands 41% in average. All DE variants used in the tests outperform the specialized CRS4e algorithm in the reliability on the ENSO task, but this better reliability is paid by slower convergence.

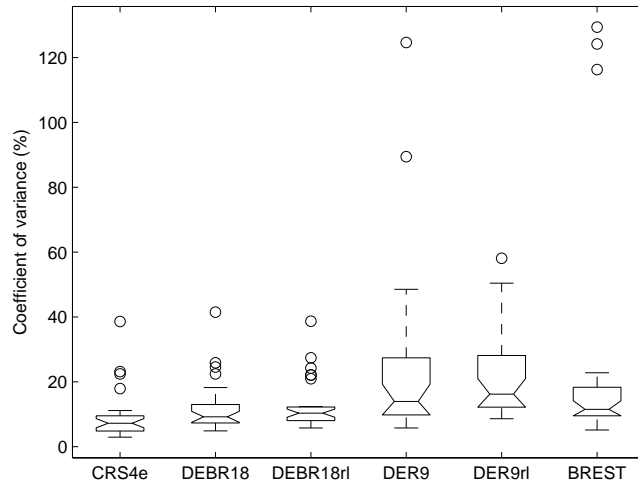
BREST variant of DE shows the worst performance in the estimation of nonlinear regression parameters among all the algorithms in tests. This is unexpected finding because in numerical comparison of the same algorithms on the six composite functions [5], BREST variant achieved very good performance [18].



**Fig. 1.** Boxplots of average relative success in generating the new trial point  $\mathbf{y}$

The best convergence of CRS4e can be explained by its highest relative success in generating the new trial point  $\mathbf{y}$ . Boxplots of average relative success are displayed for all tested algorithms in Fig. 1. The boxplots are based on 27 averages of relative success computed from 100 runs for each task. The values of the relative success for all the variants of DE are significantly less than for CRS4e. Median of the relative success for CRS4e is approximately 42%, which is higher value than maximum of all the DE variants. When comparing rl-variants of DE with their standard counterparts, random localization increases the relative success and in consequence improves convergence as well, but the average relative success of rl-variants remains significantly smaller than the relative success of CRS4e.

Variabilities in number of function evaluations needed to reach the stopping condition are compared in Fig. 2. The variability is evaluated by the coefficient of variance  $vc$ , defined by expression  $vc = 100 \times stdev / mean$ . The boxplots are also based on average coefficients of variability for the tasks like in the case of



**Fig. 2.** Variability in number of function evaluation – coefficients of variability

previous figure. As it is seen in the boxplots, the less reliable algorithms have also higher variability of time demands, mostly with several heavy outliers.

## 6 Conclusions

Adaptive variants of differential evolution intended as general heuristics for the box-constrained optimization problems were compared experimentally with CRS4e algorithm tailored for the estimation of parameters in nonlinear regression models. We can conclude from the numerical results that the algorithm tailored for a certain class of problems can outperform the other algorithms.

Good performance of the CRS4e algorithm is achieved due to high success rate in generating of the new trial point  $\mathbf{y}$ , about 40% or higher for most tasks. This success rate is approximately twice greater than 20% used for correction of control parameters in evolutionary strategy (“the rule of one fifth”, for details see e.g. [2]). This high success rate achieved by CRS4e is a bit risky and can result in premature convergence, particularly if the algorithm is used for the functions of different shape than the algorithm was tailored. General adaptive heuristics like adaptive variants of differential evolution are supposed to perform better on such functions.

The worst performing BREST algorithm in the estimation of nonlinear-regression parameters is the best on the composite functions [5] and vice versa, the CRS4e is the worst on these composite functions, see the experimental results in [18]. Although such finding seems paradoxical, it is in agreement with the implications resulting from the No Free Lunch Theorems [19] stating that no stochastic search algorithm can outperform others for all objective functions.

## 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. Brest J., Greiner S., Boškovič B., Mernik M., Žumer V.: *Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems*, IEEE Transactions on Evolutionary Computation **10** (2006) 646–657.
4. Kaelo P., Ali M. M.: *A numerical study of some modified differential evolution algorithms*, European J. Operational Research, **169** (2006) 1176–1184.
5. Liang J. J., Suganthan P. N., Deb K.: *Novel Composition Test Functions for Numerical Global Optimization*, IEEE Swarm Intelligence Symposium, (2005) 68–75. Matlab codes available at [http://www.ntu.edu.sg/home/EPNSugan/index\\_files/comp-functions.htm](http://www.ntu.edu.sg/home/EPNSugan/index_files/comp-functions.htm)
6. Liu J., Lampinen J.: *A Fuzzy Adaptive Differential Evolution Algorithm*, Soft Computing **9** (2005) 448–462.
7. NIST. *Statistical Reference Datasets. Nonlinear regression*, NIST Information Technology Laboratory. <http://www.itl.nist.gov/div898/strd/> (2001).
8. Price K. V., Storn R., Lampinen J.: *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag (2005).
9. McCullough B. D., Wilson B.: *On the accuracy of statistical procedures in Microsoft Excel 2003*, Comput. Statist. and Data Anal. **49** (2005) 1244–1252.
10. Storn R., Price K. V.: *Differential Evolution—a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces*, J. Global Optimization **11** (1997) 341–359.
11. Teo J.: *Differential evolution with self-adaptive populations. Knowledge-Based Intelligent Information and Engineering Systems*, 9th International Conference, KES 2005, Melbourne, Australia, September 14-16, 2005, Proceedings, Part I, Lecture Notes in Computer Science **3681** (2005) 1284–1290.
12. Teo J.: *Exploring Dynamic Self-adaptive Populations in Differential Evolution*, Soft Comput. **10** (2006) 673–686.
13. Tvrdík J., Křivý I.: *Comparison of algorithms for nonlinear regression estimates*, In: Antoch J. (ed) COMPSTAT 2004. Physica-Verlag, Heidelberg New York, (2004) 1917–1924.
14. Tvrdík J.: *Competitive Differential Evolution*, MENDEL 2006, 12th International Conference on Soft Computing (Matoušek R. and Ošmera P. eds). University of Technology, Brno (2006) 7–12.
15. Tvrdík J., Křivý I., Mišík L.: *Adaptive Population-based Search: Application to Estimation of Nonlinear Regression Parameters*, Computational Statistics and Data Analysis (2007) doi 10.1016/j.csda.2006.10.014 (in print).
16. Tvrdík J.: *Differential Evolution with Competitive Setting of its Control Parameters*, TASK Quarterly **11** (2007) 169–179.
17. Tvrdík J., Habiballa H., Pavliska V.: *Matlab Program Library for Box-Constrained Global Optimization*, APLIMAT 2007, Part I, 6th International Conference on Applied Math. Slovak University of Technology, Bratislava (2007) 463–470, Program Library available on <http://albert.osu.cz/oukip/optimization/>
18. Tvrdík J.: *Adaptation in Differential Evolution: A Comparison on Composite Test Functions*, MENDEL 2007, 13th International Conference on Soft Computing. (Matoušek R. ed.). University of Technology, Brno (2007) 1–6.

19. Wolpert D. H., Macready W. G.: *No Free Lunch Theorems for Optimization*, IEEE Transactions on Evolutionary Computation **1** (1997) 67–82.
20. Zaharie D.: *Critical Values for the Control Parameter of the Differential Evolution Algorithms*, MENDEL 2002, 8th International Conference on Soft Computing (Matoušek R. and Ošmera P. eds). University of Technology, Brno (2002) 62–67.