# A Support Vector Machine-Inspired Evolutionary Approach for Parameter Tuning in Metaheuristics (Working Paper)

Ruxandra Stoean[1], Thomas Bartz-Beielstein[2], Mike Preuss[3], and Catalin Stoean[1]

[1] University of Craiova, A. I. Cuza, 13, 200585, Craiova, Romania
{ruxandra.stoean, catalin.stoean}@inf.ucv.ro
[2] Cologne University of Applied Sciences, Steinmüllerallee 1, 51643 Gummersbach, Germany bartz@gm.fh-koeln.de
[3] Technical University of Dortmund, Otto-Hahn 14, 44221, Dortmund, Germany mike.preuss@tu-dortmund.de

**Summary.** The paper presents a novel, combined methodology to target parameter tuning. It uses Latin hypercube sampling to generate a diverse, large set of configurations for the variables to be set. These serve as input for the metaheuristic to be tuned and an extensive data set, with the parameter values and the success rate obtained by the algorithm, is formed. The collection is next subject to regression by means of a recent evolutionary engine for support vector machine learning. The investigations on tuning an evolutionary algorithm for function optimization led to interesting insights on a simple, unconstrained evolution of the structure and coefficients of the underlying regression function. The approach can be further improved in prediction accuracy, while also enhanced to target multiobjectivity and discovery of the best set of performing parameters for the metaheuristic to be tuned.

## 1 Introduction

Parameter tuning within any metaheuristic is an issue of essential importance that significantly accounts for the performance of the respective algorithm. Finding the proper values for the involved parameters in order to make a method perform well is imperative from several perspectives:

- The inner-workings of any technique depend on an appropriate choice for its intrinsic variables.
- Metaheuristics usually exhibit an intricate design and/or a large number of parameters. Both target a great computational effort in adjusting many variables possibly within a complex working scheme.

- There is a big number of options for the value of each possible variable and generally little knowledge on the effect these values have on the maintenance of equilibrium between exploration and exploitation of the search space which ultimately lead to the ability of the metaheuristic to converge to the (near) optimal solution.
- The parameters strongly depend on each problem instance that is currently at hand.
- The use of rough estimations for some parameter values that are given by different theoretically proven formulas do not always lead to the (near) optimal performance or have all the necessary knowledge on the problem to be accurate enough.

Traditionally, parameter tuning has been performed manually, although ways for automatic regulation had been reported since the 1980s. Nevertheless, recent years have demanded an aggressive shift from the time-wasting, precision-susceptible human adjustment of parameters to computationally sustained, rapid estimation of their appropriate values for success.

Distinguishing itself from the many current techniques, a novel methodology to estimate the direction of the relationship between the different parameters on the one hand and the algorithm response variable on the other hand is presented in this paper. Values for the parameters to be tuned are uniformly generated by Latin hypercube sampling in order to obtain a fair sample within the space of possible configurations. The response of the algorithm is computed by running it with the produced variables. The achieved regression data set is next subject to evolutionary support vector learning, whose result is the underlying decision function. The simple and proficient instrument is additionally able to provide a priori, instant information on the expected success rate when different configurations of parameters are appointed. Thus, valuable time that would have otherwise been spent on running the metaheuristic is furthermore saved. It eventually has to present itself as a low computational means, since runtime is especially important when tuning a metaheuristic in application to a real-world complex problem.

The paper is structured as follows. Section 2 briefly outlines the current status of research in the field of parameter tuning. Section 3 explains the rationale behind the new technique and details every step and meaning of its components. Experimental results are given in section 4 and the advantages offered by the approach are discussed. Finally, conclusions and future enhancements are presented in the enclosing section.

## 2 State-of-the-Art in Parameter Tuning

There have been numerous attempts to tackle the automatic setting of parameters that control the behavior of a metaheuristic. The appointment of values for the implicated variables can be performed a priori (offline) or during the

run (online) of the particular algorithm. Since the task of proposed algorithm refers to the former situation, it is further focused only on the actual stage of knowledge and research that deals with the correspondingly called parameter tuning. The number of approaches in this field is vast; as a result, attention is directed to the main classes of tuning methods and several representatives for each of those. Typically it is the area of Evolutionary Algorithms (EAs) that is the 'Swiss army knife' of metaheuristics, and, subsequently, many parameter tuning techniques generally address this type of algorithms. What is more, EAs exhibit many performance-powering parameters, which make them a suitable candidate for application of such techniques. The major categories of research in tuning metaheuristics can be appointed as follows:

- Parameter sweeping across the search space [11].
- Statistical analysis of parameters: The employment of response curves for the estimation of confidence intervals for parameter values [3], [4] or logistic regression [10].
- Meta-algorithms for calibrating algorithms, which range from the early meta genetic algorithm in [6] to the more contemporary work of [5], where parameter values result from optimization by evolutionary computation, and the recent relevance estimation and value calibration (REVAC) method that constructs a distribution over the range of each parameter that gives high probability to best values [7], [8].
- Combinations of methods: design of experiments (DOE), together with classification and regression trees (CART) and design and analysis of computer experiments (DACE) that led to the sequential parameter optimization (SPO) [1], [9].

A recent book [2] summarizes state-of-the-art approaches to parameter tuning.

## 3 Evolutionary Detection of a Regression Hyperplane for Variable Setting

The suggested technique lies on the border between statistics and meta-algorithms for the parameter tuning of algorithms. It addresses regression, where the explanatory variables correspond to the parameters to be tuned, while the response variable is the percentage of runs when the solution is attained. Regression is tackled by a recently elaborated combined approach which embraces the geometrical consideration of automatic, intelligent learning within the state-of-the-art technique of support vector machines (SVMs), while it considers the estimation of the coefficients of the decision surface through the direct search capabilities of the flexible and efficient framework of EAs. To the best of our present knowledge, there has been no similar attempt in addressing the issue of parameter tuning.

### 3.1 Motivation and Aims

Given the multitude of powerful existing methods in the area of automatic parameter tuning, here are the purposes that trigger the particular importance and contributions of the new technique:

- The fresh evolutionary approach to support vector machine training (ESVM) has been proven to be a competent, yet easily implementable and tunable metaheuristic for regression [13], [14], [15].
- The ESVM technique is unconstrained and can employ nonstandard decision functions. As a result and aiming for a better, unrestrained learning, the structure of the decision function is evolved alongside its coefficients.
- The prediction for a new test case can be done in a separate module from the training part.
- The outcome of the technique consists not only of test predictions but also provides insight into the relationships between investigated parameters and the success rate of the algorithm. This is possible since the ESVM is able to also output the form of the decision function.

As opposed to its most related technique for parameter tuning, i.e. the logistic regression tool in [10], the novel approach is substantially different through the following:

- The present method is driven by a new, competitive approach to regression.
- It is able to adjust as many parameters as needed, contrary to [10] where only two parameters specific to a given metaheuristic are addressed.
- It is intended to be a general tool, rather than one specialized to a certain algorithm. Hence, its mechanisms are independent of the chosen variables, which are generated by an autonomous sampling and given to the ESVM regression engine.
- The numerical collection of parameter values to which regression is applied is the result of a Latin hypercube sampling on every specific problem and not universally appointed. Moreover, it comprises of sufficient samples to draw an accurate model upon, as compared to the related instrument.

Present method thus aspires to approximate the correlation between the different parameters of a metaheuristic and its performance by a meta-algorithm that is able to resolve the complexity of the obtained regression problem through a direct and unconfined way.

### 3.2 Construction of the Data Set for Regression

The parameters involved in the mechanics of a metaheuristic denote the explanatory variables of the regression task, while the response variable of the given system addresses performance in terms of percentage of runs when the algorithm attains the solution.

The values for each parameter may be provided in a manner which is empirical and dependant on the actual metaheuristic/problem, just as it happens in [10]. However, this way is neither automatic nor very broad-spectrum. Moreover, the resulting data set is not significantly large and points are not equally distributed within the search space.

In contrast to this mode, the proposed technique incorporates a generating engine that automatically provides a set (as large as necessary) of different configurations using Latin hypercube sampling (LHS). This statistical method is employed to generate a space-filling (fair) sample of the algorithm parameters. For small sample sizes, it is well-known to generate more even distributions than random sampling. For this reason, it is also the first step in the tuning algorithm SPO [1]. For each considered parameter setting, 30 repeated runs of the current metaheuristic are performed and the average computed result represents the value for the response variable.

The response variable (target) is appointed as the success rate of the algorithm, i.e. the percentage of runs out of the 30 ones in which the solution to a given problem is discovered. The metaheuristic terminates when an appropriate number of evaluations of the objective function is spent.

### 3.3 Evolution of the Regression Surface

The resulting data set, denoted by $\{(x_i, y_i)\}, i = 1, 2, ..., m$, where $x_i \in R^n$ represents a tuple of parameters and $y_i$ the response for each configuration are next subject to regression through the ESVM technique.

#### Optimization Statement

Drawing its scheme from the standard SVMs, the ESVM for regression attempts to find a function, $f_{w,b}(x) = \langle w, x \rangle - b$, $w \in R^n$ and $b \in R$, that has at most $\epsilon$ deviation from the actual targets of training samples and, simultaneously, is as flat as possible [12], i.e. it exhibits a minimal $\|w\|$.

In other words, the aim is to estimate the regression coefficients $w$ and $b$ of $f(x)$ with these requirements. Errors are therefore allowed as long as they are less than $\epsilon$. Also, resulting values of the regression coefficients may affect the model in the sense that it fits current training data but has low generalization ability, which would contradict the principle of structural risk minimization of statistical learning theory [16]. In order to overcome this limitation, it is required to choose the flattest function in the definition space. Another way to interpret the task for regression is that training data are constrained to lie on a hyperplane that allows for some error and, at the same time, has high generalization capacity.

Within the particular environment of ESVMs [13], [14], [15] the conditions for the flattest function (smallest slope) that approximates training data with $\epsilon$ precision can be translated into the general nonlinear optimization task (1). The kernel K translates the scalar product to a higher dimension where a

linear model fits the transposed data and the $\xi_i$'s stand for allowance to some deviation of samples from the regression hyperplane.

$$\begin{cases} \text{find } w \text{ and } b \text{ as to minimize } K(w,w) + C\sum_{i=1}^{m}(\xi_i + \xi_i^*) \\ \text{subject to } \begin{cases} y_i - K(w,x_i) - b \leq \epsilon + \xi_i \\ K(w,x_i) + b - y_i \leq \epsilon + \xi_i^*, \quad i = 1,2,...,m. \\ \xi_i, \xi_i^* \geq 0 \end{cases} \end{cases} \tag{1}$$

The ESVM then considers the adaptation of a flexible hyperplane to the given training data through the evolution of the optimal coefficients for its equation.

**Evolutionary Components**

An individual of the population (2) encodes the coefficients of the hyperplane, $w$ and $b$.

$$x = (w_1, w_2, ..., w_n, b) \tag{2}$$

The fitness assignment $f$ of an individual (3) derives from the objective function of the optimization problem and has to be minimized.

$$f(w,b) = K(w,w) + C\sum_{i=1}^{m}\xi_i \tag{3}$$

The function to be fitted to the data is thus still required to be as flat as possible and to minimize the errors of regression that are higher than the permitted $\epsilon$. The kernel $K$ may be either SVM-standard (restricted to obey Mercer's theorem for convergence), e.g. polynomial or radial, or may take any form whatsoever, as training is now performed under the adaptable, unconstrained EAs.

The minimization of the errors is resolved under the loss function [12]. It may take any formulation suitable for the problem at hand, from the straight expression to a least mean squares mode or more complex functions.

The errors are given by providing the difference between the actual target and the predicted value that is obtained following the coefficients of the current individual (regression hyperplane) (4).

$$\delta_i = |K(w,x_i) + b - y_i|, i = 1,2,...,m. \tag{4}$$

The difference against the $\epsilon$ threshold may be subsequently tested through (5).

$$\begin{cases} \text{if } \delta_i < \epsilon & \text{then } \xi_i = 0 \\ \text{else} & \xi_i = \delta_i - \epsilon, \end{cases} i = 1,2,...,m. \tag{5}$$

The efficient tournament selection and the common genetic operators for real encoding, i.e. intermediate crossover and mutation with normal perturbation, are applied.

The EA stops after a predefined, appropriate number of fitness evaluations and outputs the optimal coefficients $w^*$ and $b^*$ for the estimated model for the data, which is given by expression (6).

$$f(x) = K(w^*, x) + b^*. \tag{6}$$

**Choosing a Kernel**

Since employing SVM kernels had proven unsuccessful in early experimentation, the use of a nonstandard decision function that is possible under ESVMs may conduct to a better fit to the data.

Therefore, a more complex individual is considered in the form (7).

$$x = (w_1, w_2, ..., w_n, w'_1, w'_2, ..., w'_n, b). \tag{7}$$

The expression of the kernel is experimentally chosen to a sum of weighted radials after every parameter to be tuned (8).

$$K(w, x) = \sum_{i=1}^{n} (w'_i \cdot e^{-2 \cdot (w_i - x_i)^2}). \tag{8}$$

### 3.4 Structural and Parameter Evolution

The ESVM can further profit from the flexibility of the EA in additionally generating the structure of a nonstandard kernel. This allows for learning with one kernel that does not necessarily obey Mercer's theorem and thus proves another advantage over SVMs that cannot converge with otherwise possibly well-performing decision functions. Moreover, such a broad evolution is able to independently adapt to the shape of each problem or algorithm.

The approach attempts a two-step evolutionary process. A hill-climber (HC) encodes a combination of different simple functions (9), i.e. linear, squared, exponential, logarithmic, square root, trigonometric etc. It moves through the search space of possible arrangements driven by small perturbations that are better than the current state.

$$x = (f_1, f_2, ..., f_n). \tag{9}$$

The coefficients of each hill-climbing kernel are again generated by the same inner EA in a fixed number of fitness evaluations. Both the fitness of the individuals carrying the weights of regression and that of the hill-climber address the training error rate. After a fixed number of repeats, the hill-climber stops and provides the kernel and its best found coefficients. The

form of the regression function is thus finally output in the form (10). For a quick reference, we will further abbreviate this approach as HC-ESVM.

$$f(x) = \sum_{i=1}^{n} (w_i^* \cdot f_i(x)) + b^*. \tag{10}$$

### 3.5 Interpreting the Resulting Function

Once the decision function is found, it can be used twofold:

- To interpret the direction of the relationship between the parameters and the response variable, the signs of the determined regression coefficients giving the information of whether the value of each parameter should be greater, lower or it is insignificant.
- To obtain a rapid approximation of the behavior of the metaheuristic if certain values are appointed for the involved parameters. This provides an efficient means of saving computational time when a certain configuration of parameters is tried.

## 4 Experimentation. Parameter Tuning of an Evolutionary Algorithm for Function Optimization

As a test metaheuristic, it is chosen to take a canonical EA. The reasons for this choice are the high number of parameters it makes use of and their importance when no special mechanisms are employed for tackling hard optimization problems.

### 4.1 Preexperimental Planning

The first runs of the algorithms had been conducted under the structure of a Mercer kernel. The polynomial and the radial had been tried, with high resulting errors of prediction. This did not come as a surprise, as the EA optimization is significantly conceptually different than the SVM customary training. A sum of weighted radials on every variable has been subsequently tried, resulting in a better outcome, however one that could not be further enhanced. This led to the decision of a second complete structural and parameter evolution.

### 4.2 Task

The aim of experimentation is to investigate whether prediction capability remains at least competitive to the related SVM, while learning involving non-standard kernels allows for further resulting information on parameters relationship. We also address for comparison another traditional artificial intelligence paradigm, namely regression trees (RTs).

### 4.3 Experimental Setup

The experiments involve arrangements on several stages:

1. The canonical EA to be parameterized is allowed 5000 fitness evaluations to reach the solution. A solution is presumed to be reached if $|solution_{known} - solution_{found}| < \epsilon$, where the threshold is different for every considered function (Table 1).
2. Three functions are taken for optimization via the EA, namely Schwefel ($F1$), Ackley ($F2$) and Rastrigin ($F3$), each with two variables (Table 1). They all exhibit one global optimum out of many local ones and its reach is difficult for a canonical EA, as the local optima obstruct the search. That is the reason for their selection, as herein the EA variables play a crucial role for the detection of the optimum.
3. The parameters to be tuned are the population size ($pop\_size$), the crossover probability ($pc$), the probability of mutation ($pm$) and the mutation strength ($ms$).
4. 300 diverse configuration settings for these variables are generated by the LHS.
5. 30 runs of the EA are performed and its success rate is computed and given as the response variable.
6. ESVMs are applied to the obtained regression data set. For the straight ESVM, 20 000 fitness evaluations are appointed for each run, while for the HC-ESVM, the HC stops after 1000 steps and the ESVM after 1000 fitness evaluations.
7. The HC uses 12 simple functions for the combinatorial optimization.
8. 200 data are considered as training cases and the rest of 100 are appointed to the test collection.
9. The root mean squared error (RMSE) on the test set gives the final prediction ability (11). The prediction error is computed in mean after 30 runs of each algorithm.

$$RMSE = \sqrt{\frac{1}{p}\sum_{i=1}^{p}(y_i^{(pred)} - y_i)^2} \tag{11}$$

10. Given the observed ease of parameterization and the actual task of ESVM, only manual tuning is performed on its own parameters. For an objective comparison, variables of SVMs and RTs are also manually chosen.

### 4.4 Results

The resulting prediction errors in mean after 30 repeats of the each algorithm are outlined in Table 2.

The HC-ESVM also outputs the structure of the best-performing kernel. For the three functions to be optimized, the mostly found decision function (structure and parameters) proved to be:

- $f_{F1}(x) = 0.58 \cdot \sqrt{pop\_size} + 0.29 \cdot \sqrt{pc} + 0.01 \cdot \ln(pm) + 0.48 \cdot \sqrt{ms} - 0.41$
- $f_{F2}(x) = 0.08 \cdot \ln(pop\_size) + 0.09 \cdot \ln(pc) - 0.29 \cdot pm^2 + 0.30 \cdot \cos(ms) + 0.73$
- $f_{F3}(x) = 0.67 \cdot \sqrt{pop\_size} + 0.18 \cdot ln(pc) - 0.22 \cdot pm^2 - 0.10 \cdot ln(ms) + 0.34$

**Table 1.** Functions optimized by the canonical EA to be tuned and the threshold for the found solution claim.

| Function | Threshold |
|---|---|
| $F1(a,b) = 418.9820 \cdot 2 - a\sin(\sqrt{\lvert a\rvert}) - b\sin(\sqrt{\lvert b\rvert})$ $-500 \leq a, b \leq 500$ | $\epsilon = 10^{-1}$ |
| $F2(a,b) = a^2 + b^2 - 10 \cdot (\cos(2\Pi a) + \cos(2\Pi b)) + 20$ $-5.12 \leq a, b \leq 5.12$ | $\epsilon = 10^{-5}$ |
| $F3(a,b) = 20 + e - 20e^{-0.2\sqrt{\frac{a^2+b^2}{2}}} - e^{\frac{\cos(2\pi a)+\cos(2\pi b)}{2}}$ $-1 \leq a, b \leq 1$ | $\epsilon = 10^{-4}$ |

**Table 2.** Average prediction errors for ESVMs, HC-ESVMs, SVMs and RTs for regression data sets connected to each function $F1$ - $F3$.

| | ESVMs | HC-ESVMs | SVMs | RTs |
|---|---|---|---|---|
| $F1$ | 11.40 | 11.65 | 10.36 | 12.85 |
| $F2$ | 18 | 19.03 | 13.59 | 18.74 |
| $F3$ | 16.91 | 15.86 | 11.85 | 17.21 |

### 4.5 Observations

The ESVM technique performs worse than the traditional SVMs, while remains close to the RTs. When comparing the two versions of the ESVM, i.e. the one with a fixed nonstandard kernel and the other with a HC generated underlying function, the prediction errors seem similar. However, the bias may remain towards the structural and parameter evolution if an independent adaptation to any problem/algorithm is considered as more important. The importance of such obtained decision functions is twofold:

- They provide a final formula of parameter interactions. Therefore, when a new set of parameters is tested, a simple computation takes place instead of a complete run of the metaheuristic to be tuned. Once the desired variables are introduced into the equation, the result is already the predicted success rate of the technique.
- The importance of each variable and their relationship can be straightforwardly seen. For example, it can be noticed that a high population size (following a square root function) is needed for both $F1$ and $F3$, a large

mutation strength is necessary for the discovery of the optimum in $F1$, while the probability of mutation (pursuing a squared formulation) must be small for $F2$ and $F3$.

### 4.6 Discussion

The analysis of the evolutionary process within ESVM raises the possibility that search is hindered by local optima of the complex task of evolving both the underlying function and its variables. A restart HC and/or a niching EA instead of the canonical formulation may overcome this and allow the convergence to the global optimum.

Although prediction power is not at its peak yet, several advantages already derive from current approach:

- The possibility to allow for any form of well-performing kernels, either manually constructed or evolved.
- The evolved kernel is able to independently adapt to the provided data.
- A miscellaneous structure can be possible and self-discovered.
- All these are impossible under SVMs. All kernels must obey Mercer's theorem in order for the technique to converge. While restricting the employment to only some kernels, the condition is additionally very difficult to be checked.
- The form of the decision function allows for a separate, instant testing of new possibilities of parameter configurations. At the same time, the SVM methodology is not always able to also output the learning function together with the target of new test samples after training is finished. Thus, the hyperplane is usually not known and consequently unavailable to be independently used in a distinct testing module.
- Having achieved the decision hyperplane, a new component can be further added. A separate EA can evolve an optimal configuration of parameters to maximize the obtained function.

The realism of proposed solution can be potentially questioned as regards the criticism of using a metaheuristic like ESVM to tune parameters of another metaheuristic [7], since the high level algorithm has parameters of its own to be set. This can be replied by two arguments that are also true for the evolutionary support vector training. Firstly, ESVM has been experimentally seen to function well with many possible configurations of parameters on a diverse set of problems, while the variables to be adjusted in the low level metaheuristic can be of a completely different order or difficulty. Secondly, even if at some point it will be arrived at the task of calibrating even ESVMs, there is no obstacle in the way of the algorithm to parameterize itself.

## 5 Conclusions and Outlook

The paper presents a novel approach for parameter tuning in metaheuristics. This is constructed by combining an initial LHS for generating a diverse collection of parameter values of an EA to be tuned with the recent ESVM technique for regression on the obtained data set. After having compared the regression power to that of classical methodologies, the ESVM is closer (but better) in performance to RTs and still remotely accurate from the traditional SVMs. However, a deeper investigation into the convergence issue, namely a restart combined with a niching strategy, will most likely help surpass this lower threshold. Also, the check the goodness of fit of the model and the testing of the statistical significance of the estimated parameters must be eventually conducted.

Furthermore, the potential of the novel technique can be enhanced through some additions:

- Comparison and interaction to the results of the state-of-the-art in parameter tuning SPO [1] should be interesting and beneficial for both approaches.
- A most beneficial solution to the problem of tuning parameters of a metaheuristic would be achieved through the attendance of more than one response variable. Success rate and the average number of evaluations of the objective function are both very important to a certain technique, therefore, their inclusion in a transformed multivariate regression task would be the next level of the proposed approach. The ESVM algorithm can be easily adjusted to treat the new regression formulation through the replacement of the canonical EA inside by a multiobjective alternative.

## References

1. T. Bartz-Beielstein, Experimental Research in Evolutionary Computation - The New Experimentalism, Natural Computing Series, Springer, Berlin, 2006.
2. T. Bartz–Beielstein, M. Chiarandini, L. Paquete, and M. Preuss - Empirical Methods for the Analysis of Optimization Algorithms, Springer, Berlin, 2009.
3. A. Czarn, C. MacNish, K. Vijayan, B. Turlach, and R. Gupta. Statistical Exploratory Analysis of Genetic Algorithms, IEEE Transactions on Evolutionary Computation, Vol. 8, No. 4, pp. 405-421, 2004.
4. O. Francois and C. Lavergne, Design of Evolutionary Algorithms - A Statistical Perspective. IEEE Transactions on Evolutionary Computation, Vol. 5, No. 2, pp. 129-148, 2001.
5. B. Freisleben, Advanced Techniques in Evolutionary Computation, Metaevolutionary approaches, chapter 7.2, pages 212-223, Oxford University Press, 1997.
6. J. Grefenstette, Optimization of control parameters for genetic algorithms, IEEE Transactions on Systems Man and Cybernetics, Vol. 16, No. 1, pp. 122-128, 1986.

7. W. A. de Landgraaf, A. E. Eiben, and V. Nannen, Parameter Calibration using Meta-Algorithms, IEEE Congress on Evolutionary Computation (CEC 2007), pp. 71-78, 2007.
8. V.Nannen, S. K. Smit, A. E. Eiben, Costs and Benefits of Tuning Parameters of Evolutionary Algorithms, International Conference on Parallel Problem Solving from Nature (PPSN 2008), pp. 528-538, 2008.
9. M. Preuss, T. Bartz-Beielstein, Sequential Parameter Optimization Applied to Self-adaptation for Binary-coded Evolutionary Algorithms, F. G. Lobo, C. F. Lima, Z. Michalewicz (Eds.) Parameter Setting in Evolutionary Algorithms, Studies in Computational Intelligence, Springer, pp. 91-119.
10. I. C. O. Ramos, M. C. Goldbarg, E. G. Goldbarg, A. D, Neto, Logistic regression for parameter tuning on an evolutionary algorithm, Congress on Evolutionary Computation (CEC 2005), pp. 1061-1068, 2005.
11. M. E. Samples, J. M. Daida, M. Byom, M. Pizzimenti. Parameter sweeps for exploring GP parameters, Conference on Genetic and Evolutionary Computation (GECCO 2005), Vol.2, pp. 1791-1792, 2005.
12. A. J. Smola, B. Scholkopf, A tutorial on support vector regression. Technical report, University of London, 1998.
13. Ruxandra Stoean, Support Vector Machines. An Evolutionary Resembling Approach, Research Center for Artificial Intelligence, Computer Science Series, Universitaria Publishing House, Craiova, 132 pages, ISBN: 978-606-510-161-6, 2008.
14. Ruxandra Stoean, Mike Preuss, Catalin Stoean, Elia El-Darzi, D. Dumitrescu, An Evolutionary Approximation for the Coefficients of Decision Functions within a Support Vector Machine Learning Strategy, Foundations on Computational Intelligence, Studies in Computational Intelligence, Springer, Aboul Ella Hassanien and Ajith Abraham (Eds.), Vol. 1, pp. 83-114, ISSN 1860-949X, 2009.
15. Ruxandra Stoean, Mike Preuss, Catalin Stoean, Elia El-Darzi, D. Dumitrescu, An Evolutionary Resemblant to Support Vector Machines for Classification and Regression, Journal of the Operational Research Society, Palgrave Macmillan, Vol. 60, Issue 8 (August 2009), Special Issue: Data Mining and Operational Research: Techniques and Applications, Guest Editors: Kweku-Muata Osei-Bryson and Vic J Rayward-Smith, pp. 1116-1122, ISSN 0160-5682, 2009.
16. V. Vapnik, Statistical Learning Theory, Wiley, 1998.