Regressor Encapsulation using Fuzzy Control in Evolving Nonlinear Models

Alina Patelli*, Lavinia Ferariu*

**"Gh. Asachi" Technical University of Iasi, Department of Automatic Control and Applied Informatics 700050 RO (e-mail: apatelli@tuiasi.ro, lferaru@tuiasi.ro)*

Abstract: The paper suggests a customized evolutionary identification tool for the automatic generation of complex nonlinear systems' models. The proposed algorithm features a novel regressor encapsulation mechanism, based on fuzzy logic, aimed at guiding crossover cut point selection. To encourage the production of well adapted offspring, a fuzzy controller identifies the model terms which would bring the most relevant gain in fitness, if swapped via crossover. The authors contrast two methods of tuning the fuzzy membership functions' parameters, both in a static and a dynamic way, in order to accurately capture regressor relevance. In addition, the basic multiobjective evolutionary loop is upgraded with original similarity analysis and genetic material refreshment mechanisms aimed at preserving population diversity. The practical usefulness of the proposed tool is demonstrated within an experimental trial involving the identification of a complex nonlinear industrial system within the Sugar Factory of Lublin, Poland.

Keywords: evolutionary algorithms, nonlinear systems identification, multiobjective optimization, fuzzy logic.

1. INTRODUCTION

Obtaining accurate and parsimonious mathematical models for nonlinear systems is a key step to take in all automatic control applications, as an appropriate closed loop control law is difficult to determine otherwise. The most straightforward form a nonlinear model may be generated in is a polynomial equation, as it is fairly simple and easily exploited by numerical algorithms. In addition, it has been rigorously demonstrated that any continuous bounded function can be approximated by a polynomial model, to any desired degree of accuracy (Young, 2006), therefore extensive research has been conducted concerning **N**onlinear **L**inear in **P**arameter (NLP) models.

In short, NLP representations are linear combinations of nonlinear atoms called regressors. When given access to a representative set of experimental data, it is easy to include all possible combinations of regressors in a maximal model. The redundant terms will afterwards be eliminated via pruning techniques which are usually time consuming and may take up significant computational memory resources (Wey *et al*, 2004). Another possible approach is to incrementally build an NLP model by gradually adding regressors as result of blind search or other deterministic methods that will most likely provide a solution only after prolonged runtime (Nelles, 2001).

A feasible alternative to this line of research is represented by evolutionary techniques, namely **G**enetic **P**rogramming (GP). In the context of this approach, several potential models are generated across the search space in the form of tree encrypted individuals (Koza, 1992) which are subsequently evolved according to the Darwinian principle of the survival of the fittest. In consequence, GP based identification tools are robust, as they feature no derivatives and work on populations of candidate solutions, instead of singular individuals (Koza, 1992). Another useful feature is their ability to cope with scarce *a priori* information, as they can work with any type of search space landscape (Back *et al*, 2000). As far as initial algorithm configuring is concerned, most internal working parameters only require "trial and error" tuning at no significant supplementary computational costs (Back *et al*, 2000). All these facts make the GP approach suitable for cases where the shape and dimension of the final solution are not known beforehand, where there are unexpected relations between variables, otherwise difficult to capture, or where other analytical methods impose unrealistic working hypothesis (Poli *et al*, 2008).

Within the evolutionary framework described above, the authors suggest a novel identification tool enhanced to address the specific requirements of Multi Objective Optimization (MOO). Therefore, diversity preservation is encouraged by means of similarity analysis and fresh genetic material injection (Patelli and Ferariu, 2010), whereas the main contribution of the paper resides in the use of adaptive, fuzzy controlled encapsulation to guide the crossover operator in effectively selecting the regressors to be swapped. After identifying similar terms featured by several trees in the current population, the proposed procedure assigns an encapsulation probability to each of them according to a set of fuzzy rules. The employed membership functions may be configured statically or dynamically, for a more accurate regressor

adaptation assessment. This is meant to encourage the production of fitter offspring in relation to their parents, thus speeding up the search process and reducing the overall runtime. Additionally, the authors propose a memetic approach, residing in the symbiosis between structure selection via genetic operators and parameter computation by means of a deterministic local optimization plug-in based on QR decomposition. The reason behind this is that parameter wise linearity invites the use of a deterministic procedure for fast and accurate coefficient determination. Another advantage of this hybridization is the fact that structure selection and parameter computation are carried out in an interdependent manner, whilst other non-evolutionary identification procedures perform the two tasks independently.

The paper initially browses through the most significant related work in the field. Section III describes the tree generation and evaluation mechanisms, whilst the following section details the enhanced genetic operators involved in offspring production. Section V summarizes the elite specific enhancements implemented by the authors. A representative experimental trial and a set of conclusions are included in the final two sections, respectively.

2. RELATED WORK

Early attempts of generating nonlinear models by evolving NLP compliant tree individuals employed a single optimization objective, accuracy, thus output only one system model (Flemming and Purshouse, 2002). Following in that trend, a transformation mechanism designed to assure parameter wise linearity for all involved trees was introduced (Madar *et al*, 2005). According to it, the ill positioned operator nodes featured by the randomly generated individuals would be identified and replaced by appropriate alleles. Although rapid and easy to implement, the procedure increases the risk of regressor bloat with a negative influence on model parsimony. In exchange, the authors suggest a different idea that promotes an even distribution of the trees in the initial population over the search space, and afterwards employs a transformation routine that guarantees mathematical equivalence with two immediate advantages: the preservation of the initial problem domain coverage and controlling regressor size.

As the area of automatic control imposes specific requirements in model quality assessment, an appropriate individuals' evaluation should be subject to multiple optimization criteria. Selecting the number and nature of the objectives to use is a delicate decision. Some researches employ an increased number of evaluation functions in the quest for a highly accurate tree assessment (Rodriguez-Vasquez *et al*, 2004). The main downside is that excessively harsh evaluation may prematurely exclude trees from the population, draining the pool of genetic material subject to the action of genetic operators. Balancing the importance of each objective is also difficult. Therefore the approach presented in this paper resorts to two assessment criteria: accuracy and parsimony.

After settling for the appropriate optimization objectives to use, the next step is combining all the evaluation related information into a single value, called fitness, later on used to select trees for the reproduction pool. To manage that, especially when dealing with conflicting objectives, Deb suggested assigning fitness via non-dominance analysis (Deb, 2001). The authors have proposed population clustering and adaptive migration as enhancements to Deb's approach to better suit the engineering related requirements of the identification problem (Ferariu and Patelli, 2009). Nondominated sorting, in Deb's view, is carried out in the objectives space which is easier to exploit than the decision space composed of individuals of various sizes (each individual features a different number of parameters). However, the literature contains references that employ neural networks to map the objectives space onto the search one (Adra *et al*, 2009). Furthermore, techniques have been documented that direct the search to specific areas of the objectives space, called Pareto knees, by dynamically aggregating the initial objective functions into new ones, adapted to the interest regions (Rachmawati and Srinivasan, 2009).

To speed up the search process, increasing selection pressure has been suggested, by storing external archives of elite individuals copies, used for fitness computation purposes only (Coello Coello *et al*, 2007). Other efforts aimed at limiting algorithm runtime involve hybridizing the evolutionary loop with deterministic procedures, like the **O**rthogonally **L**east **S**quare (OLS) tool that computes error ratios for each regressor to determine the least significant ones and eliminate them (Madar *et al*, 2005). To avoid the risk of premature model term elimination, brought on by the above procedure, the authors evaluate regressor relevance by means of a stochastic encapsulation procedure overlooked by an adaptive fuzzy controller, as detailed in paragraph IV.

3. GENERATION, TRANSFORMATION AND EVALUATION OF TREES

A healthy evolution depends on the distribution of the initial genetic material over the search space. In response to that requirement, the authors suggest an upgrade of the classic tree generation routine based on random recursive insertion of nodes. The enhancement refers to a set of rules that mainly consist in including each available terminal (1) only once in each chromosome, in assuming a slightly higher probability for terminals insertion than the one of operator nodes, and in filling empty leaf slots with constants (Ferariu and Patelli, 2009). The result is the generation of individuals rich in non-redundant genetic material, evenly distributed across the problem domain, and encrypted by well balanced trees.

Due to the organization of the terminal set **x**, which contains lagged input u_i and output y_i values, n_u and n_y being the maximum allowed lags, the dynamic nature of the working models is implicitly guaranteed.

$$\mathbf{x}(k) = [u_i(k),..,u_i(k-n_u), y_j(k-1),..,y_j(k-n_y)],$$
$$i = \overline{1,m},\ j = \overline{1,n} \quad (1)$$

That reason, combined with the parameter wise linearity of NLP models leads to a minimally sufficient operator set O = {+, *}. The mandatory closure and sufficiency properties (Back *et al*, 2000) of sets **x** and O are further exploited by the three rule tree building algorithm for providing the search process with a good start.

Given the layout of the NLP formalism, compliant models can be described by the following matrix equation:

$$\begin{bmatrix} F_{i1}(\mathbf{x}(1)) & .. & \cdots & F_{ir}(\mathbf{x}(1)) \\ F_{i1}(\mathbf{x}(2)) & .. & \cdots & F_{ir}(\mathbf{x}(2)) \\ \vdots & \vdots & & \vdots \\ F_{i1}(\mathbf{x}(p)) & .. & \cdots & F_{ir}(\mathbf{x}(p)) \end{bmatrix} \begin{bmatrix} c_{i1} \\ c_{i2} \\ \vdots \\ c_{ir} \end{bmatrix} = \begin{bmatrix} \hat{y}_i(1) \\ \hat{y}_i(2) \\ \vdots \\ \hat{y}_i(p) \end{bmatrix},$$

$$\mathbf{F}_i \cdot \mathbf{c}_i = \hat{\mathbf{y}}_i, \quad \mathbf{F}_i \in \Re^{p \times r}; \mathbf{c}_i \in \Re^r; \hat{\mathbf{y}}_i \in \Re^p, i = \overline{1,n}, \quad (2)$$

where **F** is the regression matrix, **c** is the parameter vector, y_i denotes the i^{th} plant output and p stands for the training data set length. The suggested tree generation algorithm does not guarantee NLP compliance, so, in order to be able to attach an equation like (2) to each chromosome, a transformation procedure is necessary. To reposition the ill placed "+" operators inside the trees, while preserving mathematical equivalence with the original individuals, the nodes are reconfigured according to an elementary arithmetic identity, $a \cdot (b + c) = a \cdot b + a \cdot c$. This additional tree processing effort is directed towards facilitating the symbiosis with the deterministic QR local optimization procedure that can easily solve (2) and determine the optimal set of parameters for each tree encrypted structure.

Once their structure and parameters have been established, the trees in the initial population are ready for evaluation, via the two following objective functions:

$$SEF(M) = \frac{1}{2}\sum_{k=1}^{p}(y_i(k) - \hat{y}_i(k))^2, \quad k = \overline{1,p} \quad \text{and}$$
$$CF\ (M) = r + \frac{t}{n_u + n_y + 1} - \sum_{q=1}^{r} \lg c_{iq}\ . \quad (3)$$

The **S**quared **E**rror **F**unction (*SEF*) measures tree accuracy, whilst the **C**omplexity **F**unction (*CF*) is in charge of evaluating chromosome parsimony. Information about the total number of regressors, r, the terminals included in those regressors, relative to the number of available ones, t, as well as parameter relevance is reflected, in that order, by the three terms of the complexity criterion that the present approach employs. The last two components of *CF* are meant to penalize candidate models that contain a low number of large regressors as well as the ones featuring insignificant parameters, as neither have any practical significance.

Even though model parsimony is encouraged by other inherent components of the evolutionary algorithm (tree generations, enhanced genetic operators), considering a separate complexity objective is preferred, as the algorithm can output a whole set of nondominated solutions from which the designer can afterwards choose one or several tradeoff models, depending on the identification application's specific. However, not all solutions from the Pareto front are of practical use. The models situated on the extremes are either inaccurate or overfitted, with unsatisfactory generalization capacities. To avoid those areas and focus tree generation on the central Pareto front area, the authors suggest clustering procedures that target both regular trees (Ferariu and Patelli, 2009) and elites (section V).

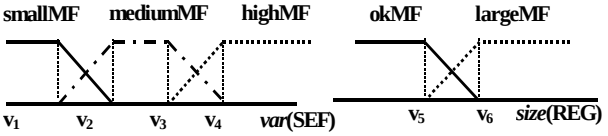
4. ADAPTIVE FUZZY ENCAPSULATION AND ENHANCED GENETIC OPERATORS

Towards the end of the evolutionary loop, the trees are expected to get closer to the optimal Pareto front. In consequence, it is highly probable for most of them to feature similar regressors that have survived over generations as well adapted model terms, which significantly contribute to increasing overall tree performances. Therefore they should be protected from division by crossover, so that offspring individuals should also benefit from them. Allowing well adapted model terms to pass on to future generations is called exploitation. On the other hand, population diversity must also be encouraged, as the search process should explore new areas of the problem domain by generating children trees which are better than their parents and significantly different at the same time. To do so, the authors suggest an original crossover operator designed to identify and avoid similar regressors, while selecting the remaining, non-resembling model terms instead. They are the ones to be swapped in an effort to generate better, yet diverse offspring, thus balancing between exploitation and exploration.

In practice, the first step to take is determining which similar regressors are truly well adapted model terms. The proposal is to employ a set of fuzzy rules. The trees in the current population are processed and all regressor featured by at least two individuals will be encapsulated with a probability that reflects how well adapted they truly are. Five fuzzy sets are used to that end. The first three are related to the fuzzy variable *var*(*SEF*) which stands for the variance of the *SEF* values (3) associated to all trees that feature the current regressor being encapsulated. They are labeled *low*, *medium* and *high* (Fig. 1). The remaining fuzzy sets are defined in relation to the size of the targeted regressor, *size*(*REG*), which can either be acceptable (fuzzy set is labeled *ok*) or not (fuzzy set is labeled *large*). The **M**embership **F**unctions (MF) associated to the two fuzzy variables and the five fuzzy sets are shown in Fig. 1.

For a complete definition of the membership functions, the values for the trapezes' vertices, v_i, i = 1, .., 6, must be accurately selected. The static configuration suggested by

the authors considers a uniform distribution of the v_i parameters. Therefore, $v_1 = 1/8m$, $v_2 = 1/6m$, $v_3 = 1/4m$, $v_4 = 1/2m$, where m denoted the mean *SEF* values achieved by the individuals in the early generations of the algorithm. The *size*(*REG*) related membership function parameters are $v_5 = m$ and $v_6 = 2m$, where m has the same significance as above, only in relation to the mean *CF* values. To improve the accuracy of the v_i, i = 1, .., 6 tuning process, the authors introduce a dynamic configuration as result of a separate learning process detailed later on in this section.



a. b.

Fig. 1. Fuzzy sets and membership functions

IF *var*(*SEF*) **IS** *small* **AND** *size*(*REG*) **IS** *ok* **THEN** *P*[*enc*] = 1
IF *var*(*SEF*) **IS** *medium* **AND** *size*(*REG*) **IS** *ok* **THEN** *P*[*enc*] = 0.8
IF *var*(*SEF*) **IS** *high* **AND** *size*(*REG*) **IS** *ok* **THEN** *P*[*enc*] = 0
IF *var*(*SEF*) **IS** *small* **AND** *size*(*REG*) **IS** *large* **THEN** *P*[*enc*] = 0.9
IF *var*(*SEF*) **IS** *medium* **AND** *size*(*REG*) **IS** *large* **THEN** *P*[*enc*] = 0.5
IF *var*(*SEF*) **IS** *high* **AND** *size*(*REG*) **IS** *large* **THEN** *P*[*enc*] = 0

Fig. 2. Fuzzy rules

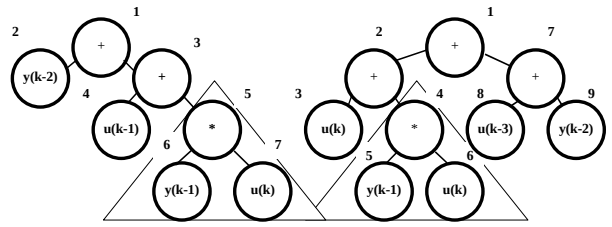


Fig. 3. Parents featuring one identical regressor

The encapsulation probability to be attached to the current similar regressors is computed by means of the fuzzy rules indicated in Fig. 2. To better understand the encapsulation process and the way it guides the crossover operator in selecting swap regressors, let us consider the example presented in Fig. 3. The two parents selected by crossover feature one identical regressor marked by a triangle (Fig. 3). Let us assume that, for all trees in the current population that feature the exact same regressor as the one inside the triangles, *var*(*SEF*) results *small* and that *size*(*REG*) is rather *large*. In that case, the fourth fuzzy rule (Fig. 2) is activated and the regressor is encapsulated with an encapsulation probability of 0.9 in all the trees that contain it, including the two parents presented in Fig. 3. The probability of selecting a cut point node within the identical regressor is 1 minus the encapsulation probability, which, for this example, is evaluated to 0.1. In other words, it is highly unlikely for crossover to divide this particular regressor, as the fuzzy control algorithm has found it useful. This example is meant to illustrate the opposing nature of encapsulation on the one hand and crossover on the other. The former is protective, meant to encourage exploitation, whilst the latter is dividing, aimed at favoring exploration.

As mentioned before, the encapsulation procedure is useful only in the final stages of the evolution process. In early generations, similar regressors are mostly coincidental and cannot be interpreted as a sign of tree adaptation. Note that, in the initial population, each tree includes all available terminals, however, the resulting similar regressors offer no indication as to adaptation. That is why encapsulation is applied only in the final *max_gen_no*/10 generations (*max_gen_no* stand for the maximum number of generations that the algorithm is allowed to run for). The dynamic tuning of the thresholds in Fig.1 considers that the initial generations represent a "training period", used to configure the parameters of the MF, v_i, i = 1, .., 6 (Fig. 1), by storing the variance of *SEF* values and the one of *CF* values for each population, in the vectors *var*(*SEF*) and *var*(*CF*). When the *max_gen_no*/10 threshold is reached, the mean value for each of the two vectors is computed and the parameters v_i, i = 1, .., 6 are evenly distributed as follows:

$$\begin{aligned} v_i &= [mean(\mathbf{var}(SEF))/4]*i, \quad i = 1,2,3,4, \\ v_i &= [mean(\mathbf{var}(CF))/2]*i, \quad i = 5,6 \end{aligned} \tag{4}$$

The other genetic operator, mutation is enhanced to avoid compensation (Ferariu and Patelli, 2009). Thus, both crossover and mutation, customized by the authors in the manner described above, have a double role: controlling tree complexity, as well as encouraging the generation of fitter offspring. The additional tree processing performed by the upgraded genetic operators is in fact computationally cheaper than generating poorly adapted children, a genuine risk when working with raw versions of the evolutionary tools.

5. ELITE RELATED UPGRADES

To increase algorithm search speed, the authors suggest an elitist approach upgraded by several original enhancements meant to focus tree generation on the feasible area of the Pareto front (denoted group I in Fig. 4) and to encourage diversity within each population. At each generation, copies of the nondominated trees are stored in an external archive and used as reference in computing fitness values for the dominated individuals (Patelli, 2010). Once a new nondominated set has been inserted in the archive, also called elite set, the latter is updated by eliminating the eventual dominated trees. However, a snapshot of the global elite set at each generation, after the update stage, is stored for later use.

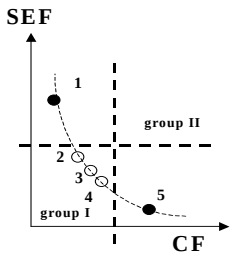


Fig. 4. Global elite set clustered to highlight feasible solutions

The feasible region of the Pareto front contains candidate models of practical significance (group I in Fig. 4). In order to identify it, the average performances of the population are computed relative to each objective (Patelli and Ferariu, 2010), thus partitioning the elite set in two groups. The individuals within the first, namely elites 2, 3 and 4 in Fig. 4, are allowed to generate offspring of their own, separately from the trees in the regular population. The resulting children are then injected among the common individuals, to draw the trees closer to the interest area of the Pareto front.

Note, however, that elites are merely copies of nondominated individuals and there is a good chance that some of the originals still exist and produce offspring in the general population, although that is not a certainty. Ergo, some of the elite children being injected might resemble already existing trees. To prevent the insertion of redundant genetic material, the authors have implemented a similarity analysis based on *SEF* variance, to validate elite children inclusion in the common population (Patelli and Ferariu, 2010). If, in spite of that precaution, overall diversity drops under a certain level, the current genetic pool is refreshed by including one of the elite set snapshots from a previous generation, determined dynamically in accordance with the magnitude of diversity decrease (Patelli and Ferariu, 2010).

6. APPLICATION

The described multiobjective, elitist, GP based tool, featuring **S**imilarity **A**nalysis and **D**ynamic fuzzy **E**ncapsulation (SADE) was employed to identify a complex industrial nonlinear system, namely the steam subsection of the evaporation station within the sugar factory of Lublin, Poland. The targeted system features one input (steam temperature) and one output (steam pressure) and has no available mathematical model. The SADE algorithm was run for five times, each considering a different size of the initial population. A reference MOO approach (RMOO), featuring none of the suggested upgrades was also run under the same initial conditions, for the sake of comparison. The results, obtained on a 190 data point training set and validated on a 290 entry set, are shown in Table 1.

Table 1. RMOO and SADE performances

run	*ind_ no*	*RMOO*				*SADE*			
		gen	*elite_no*	*var* (*SEF*)	*mean* (*reg*)	*gen*	*elite_no*	*var* (*SEF*)	*mean* (*reg*)
R1	20	75	5	0.820	5	23	18	0.623	7
R2	75	100	7	1.003	12	55	23	0.429	9
R3	150	100	12	7.115	7	72	34	0.515	6
R4	300	100	15	21.31	35	63	29	0.329	8
R5	570	100	13	105.7	41	71	31	0.418	9

ind_no - initial population size (number of trees);
gen - number of generations in which the final elite set is output;
var(*SEF*) - *SEF* variance for the final elite set, computed on the validation data;
mean(*reg*) - average number of regressors in the trees in the final elite set.

Table 2. RMOO and SADE performances

gen	static v_i			dynamic v_i		
	reg(0)	*reg*(1)	*var* (*SEF*)	*reg*(0)	*reg*(1)	*var* (*SEF*)
60	7	5	0.920	13	12	0.725
65	12	3	0.325	5	9	0.255
70	13	7	0.213	3	19	0.203

gen - current generation
reg(0) - number of regressor with encapsulation probability 0
reg(1) - number of regressor with encapsulation probability 1
var(*SEF*) - *SEF* variance

As the number of trees in the initial population increases, the RMOO algorithm finds it difficult to handle the excess genetic material and cannot complete the evolution process (R2→R5 exit by reaching the maximum number of generations). The number of elite individuals on the final non-dominated fronts seems to increase when the size of the initial population is greater, which is not a desired behavior, as the identification problem remains the same, and should have resembling solutions, regardless of algorithm configuration. In addition, elite accuracy and complexity values are scattered over the objectives space, even in the regions of no practical relevance, as shown by the high values of the variance and average indicators. On the other hand, the SADE alternative manages to meet the accuracy termination criterion before the maximum number of generations expires, due to its elitist nature and the preservation of well adapted regressors via encapsulation. Due to the diversity conservation enhancements, the elites in the final set are much more evenly distributed than in the case of RMOO, as indicated by the low *SEF* variance values. Complexity is quasi constant on all runs, as a positive side effect of the two enhanced genetic operators.

To illustrate the contribution of the dynamically computed MF parameters, the SADE algorithm was compared against a similar version with the sole difference that the latter used static v_i values (Table II). When static MF parameters are used, there seems to be no dependency between the number of regressors considered to be well adapted, *reg*(1), and the evolution of *SEF* variance, which is an indicator of a high miss rate throughout the encapsulation process. The SADE version does better. As *var*(*SEF*) drops, which illustrates the generation of more accurate populations, ergo an increase in well adapted terms occurrence, the number of 0 labeled regressors also decreases, while the 1 labeled ones become more frequent.

The final nondominated sets generated by the RMOO and SADE algorithms, as result of a separate experiment that targeted the steam subsection described above, are presented in Fig. 5a. Fig. 5b depicts the generalization capacities of the most accurate elite situated on the SADE generated Pareto front. The considered size of the initial population is 150 individuals, and the maximum allowed number of generations is 250. The raw tool, featuring

none of the suggested enhancements, has produced tradeoff solutions throughout the entire span of the Pareto front, including the nonfeasible extremities. The generated models are clustered, leaving parts of the interest zone uncovered. The SADE alternative offers a much better distributed set of solutions, located only in the feasible region of the Pareto front. Also note that the proposed identification tool provides the tradeoff set in less than half the run time of the RMOO, measured in generations, thus illustrating the practical benefits of the implemented upgrades.

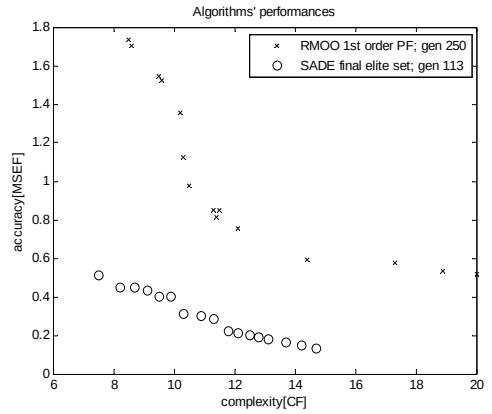


Fig 5a. RMOO and SADE generated Pareto fronts

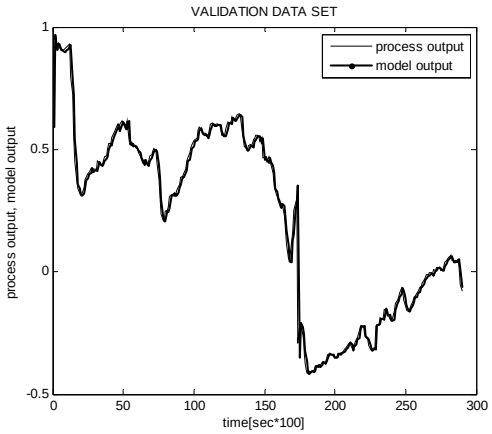


Fig 5b. Validation set performances of a SADE generated model

7. CONCLUSION

The nonlinear systems identification tool suggested in this paper is based on an elitist, multiobjective GP algorithm, enhanced with a fuzzy controlled dynamic encapsulation procedure and similarity analysis. The implemented upgrades are aimed at controlling model complexity, whilst balancing exploration and exploitation tendencies.

The protection of well adapted nonlinear terms is achieved by evaluating regressor usefulness via a fuzzy controller. As far as regressor adaptation assessment is concerned, the superiority of dynamically tuning membership functions parameters is outlined by comparison against a static approach. The resulting encapsulation probabilities are attached as labels to each of the targeted regressors and afterwards used to guide crossover in its search for appropriate cut nodes. Search space exploration is also encouraged by means of discarding redundant genetic material with the help of similarity analysis. Solution diversity is upkept by refreshing the population with specifically chosen individuals, whenever necessary.

As the generated solutions are compliant with the NLP formalism, a demonstrated universal approximator, formally compatible with numerical applications, the suggested algorithm is a valid approach for nonlinear model generation, in the framework of automatic control problems. The QR hybridization, as well as the featured upgrades targeted at increasing search speed and reducing computational resource consumption, recommends the proposed approach for solving complex problems with reduced pre-design available information and difficult search space landscapes.

ACKNOWLEDGEMENTS

The authors would like to thank the Romanian National Center for Programs Management, SICONA research grant 2010, for their financial support.

REFERENCES

Adra S. F., Dodd T., Griffin I. A., Fleming P., Convergence Acceleration Operator for Multiobjective Optimisation, IEEE Transactions on Evolutionary Computation, 13 (4), 825-846, 2009

Back T., D. Fogel, Z. Michalewicz, Evolutionary Computation 2.Advanced Algorithms and Operators. US: Institute of Physics Publishing, 2000.

Coello Coello C.A., G.B. Lamont, D.A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, second edition, Springer, 2007.

Deb K., Multi - Objective Optimization using Evolutionary Algorithms, Wiley, USA, 2001

Ferariu L., A. Patelli Multiobjective Genetic Programming for Nonlinear Systems Identification, in curs de publicare, Lecture Notes on Compuer Science, Springer Verlag, 2009.

Flemming P. J., R. C. Purshouse "Evolutionary Algorithms in Control Systems Engieering: A Survey", Control Engineering Practice 10, 2002, pp. 1223-1241.

Koza J. R, Genetic Programming – On the Programming of Computers by Means of Natural Selection, Cambridge, MA: MIT Press, pp. 73-190, 1992

Madar J., J. Abonyi, F. Szeifert Genetic Programming for System Identification [Online]. Available: http://www.fmt.vein.Hu/softcomp/ isda04_gpolsnew.pdf, 2005.

Nelles O., Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models, Springer-Verlag, 2001.

Patelli A., L. Ferariu, Elitist Multiobjective Nonlinear Systems Identification with Insular Evolution and Diversity Preservation, IEEE World Congress on Computtaional Intelligence, July, Barcelona, 2010 - accepted paper.

Poli R., W.B. Langdon, N.F. McFee, J.R. Koza, A Field Guide to Genetic Programming. Published via http://lulu.com and freely available at http://www.gp-field-guide.org.uk, 2008.

Rachmawati L., D. Srinivasan, Multiobjective Evolutionary Algorithm with Controllable Focus on the Knees of the Pareto Front, IEEE Transactions on Evolutionary Computation, 13 (4), 810-824, 2009.

Rodriguez-Vasquez K., C. M. Fonseca, P. J. Flemming Identifying the Structure of Nonlinear Dynamic Systems Using Multiobjective Genetic Programming. IEEE Transactions on Systems Man and Cybernetics, Part A – Systems and Humans, 34, 531-534, 2004.

Wey H., S. A. Billings, J. Lui, "Term and Variable Selection for Nonlinear Models", *Int. J. Control* 77, 2004, pp. 86-110.

Young M., The Stone-Weierstrass Theorem, MATH 328 Notes, Queen's University at Kingston, winter term, 2006. Available: http://www.mast.queensu.ca/~speicher/Section14.pdf.