MULTIPLE GENETIC PROGRAMMING BASED TECHNIQUES FOR NONLINEAR SYSTEMS IDENTIFICATION

BY

ALINA PATELLI and LAVINIA FERARIU

Abstract. The issue of nonlinear systems identification is addressed, in the following, by means of genetic programming, as this method is well known for providing a robust and unsupervised data driven model selection mechanism. The generated models are nonlinear, linear in parameter, a mathematical formalism that facilitates the use of a local optimisation procedure based on QR decomposition, to aid in the fast and accurate computation of model parameters. Model structure is configured by enhanced genetic operators, also designed by the authors to effectively exploit the considered parameter wise linearity. The two customisations work together, in an original symbiosis, for simultaneous structure and parameters selection. In order to refine the search for an accurate and compact model, the core algorithm is extended so that it evaluates potential solutions by making use of two separate objectives with dynamically adaptive priorities. A novel fitness assignment scheme is introduced, along with an adaptive threshold migration mechanism. The algorithm performances are then compared against the ones of an elitist procedure, within the experimental framework of an industrial system.

Key words: genetic programming, nonlinear systems, multiobjective optimisation, elitist procedure.

2000 Mathematics Subject Classification 93C30.

1. Introduction

Nonlinear systems are widely used in present day industry, hence the need for an increased efficiency in their exploitation, a requirement that cannot be met in the absence of a sound mathematical model. Obtaining such a model is a difficult task for a number of reasons. Firstly, most nonlinear systems are exceedingly complex, therefore determining the physical laws that govern their inner workings is a strenuous endeavour, sometimes offering only a limited, unsatisfactory approximation. That gives the automatic control engineer no choice but to turn towards black box type, data driven identification procedures. Furthermore, the input – output characteristic of such systems may contain
discontinuities and, of course, nonlinearities, which are particularly difficult to model. Another aspect to consider is that system inputs are usually time variant, each input value generating a different system response, making it challenging for one model to embody them all. Finally, there is the issue of noise. As most industrial environments superpose parasite signals to the system inputs and outputs, filtering the unwanted information is compulsory in order to obtain a good quality model.

Several attempts have been made so far to provide a robust identification tool capable of solving as many of the problems stated before as possible. The classical approach in systems identification makes use of test (probe) signals applied at the system inputs, and determines the most likely model structure based on the investigation of the system response. Afterwards, the model parameters are computed considering the preset structure, thus generating the final solution. Despite the fact that this method is easy to apply and computationally cheap, the models it generates are not always viable. No matter how effective the parameter computation technique is, the assumption made on the system structure may not be accurate enough, leading to poor model generalisation capabilities [1]. In the worst case scenario, such a structure might be extremely difficult to determine, should the system response to the probe signal not match any pre-selected configuration.

In 1985, Leontaris and Billings perfected a tool for generating NARMAX (Nonlinear AutoRegressive Moving Average for eXogenous inputs) models [2]. They started out with a set of terminals made out of lagged input and output terms. Those terms were used to create nonlinear atoms called regressors, which were afterwards assembled to form a linear combination. The general form of a NARMAX model is:

\[
\hat{y}(k) = c_0 + \sum_{i=1}^{n_u} c_i x_i(k) + \sum_{i_1=1}^{n_u} \sum_{i_2=1}^{n_u} \cdots \sum_{i_{n_y}=1}^{n_y} c_{i_1} \cdots c_{i_{n_y}} x_{i_1}(k) \cdots x_{i_{n_y}}(k),
\]

where: \( x(k) = (u(k), \cdots u(k-n_u)), y(k-1), \cdots y(k-n_y) \) is the terminal set, \( n_u \) and \( n_y \) are the input and output lags respectively, \( c_{i_j} \) are the model parameters and \( k \) is the current time instant. It is easy to observe that the structure employed by this method contains all possible regressor combinations, therefore there is no risk of omitting any model terms. However, the computational effort necessary for determining model parameters is considerable. In addition, most of them will probably result null, and their corresponding model terms will have to be eliminated from the model, post design.

To solve the model structure selection problem in a more flexible way, some researchers employed a GP (Genetic Programming) based technique, with one single objective: model accuracy. Based on Koza's tree like model encryption proposal [3], their approach employed the OLS (Orthogonally Least Square) method to determine which model regressor contributes less to the
overall tree accuracy and eliminate it [4]. As Matlab was used for implementation, their idea is very accurate mathematically, yet resource consuming.

Fonseca and Flemming introduced MOO (Multiple Objective Optimisation) GP in nonlinear systems identification in 1995. At first, they combined all considered objectives in a single one, using a set of fixed objectives weights. Soon after, they proposed a Pareto oriented analysis, based on a fitness assignment scheme guided by the number of individuals that the solution to be evaluated was dominated by [5]. This idea was later on improved by Deb by introducing a more refined fitness computation scheme aided by a niching mechanism to encourage diversity [6].

In 2004, Rodriguez Vasquez also used nonlinear, linear in parameter models, like the one in Eq. (1), evolved via MOOGP, yet enhanced her algorithm by externally introducing a goal vector containing preference information to help guide the search towards the interest zone of the Pareto front [7]. The method also employs a large number of objectives, some related to model accuracy and parsimony, others to nonessential information like model order or input and output lags, data that could easily be configured offline, by trial and error.

The solution proposed in this paper strives to provide answers for most of the downsides related to the previous approaches in the field. To begin with, an initial population of tree encrypted potential models is pseudo–randomly generated, each tree encoding a different possible structure configuration. For each individual, an optimum set of parameters is computed, then all individuals are evaluated according to the chosen optimisation criteria. The best candidates stand a better chance at producing offspring, and thus at forming the population of the following generation. As trees are reduced in dimension, the consequent parameter computation operation is cheaper and more accurate than in the case of NARMAX. Genetic operators, involved in reproduction, work on the tree structure, while a local optimisation procedure, based on QR decomposition, works on model parameters, in an repetitive symbiosis that takes place at each generation. Thus, model structure and parameters are evolved simultaneously, requiring no preset regressor configurations, or offline reduction of terms. Furthermore, as detailed in the paragraphs to follow, most of the algorithm parameters are self tuned, therefore, no information from an external, ‘wrapper’ layer is necessary to aid the search process.

An adaptation mechanism is also provided to assure that the trees in the population are linear relative to their symbolic coefficients before parameter computation stage. To speed up the evolution process, genetic operators have been trained to avoid altering regressors which are well adapted in terms of overall accuracy, and concentrate on improving the rest of the tree nodes (that other procedures identify using OLS). As far as MOO is concerned, the approach employs only two objectives: accuracy and parsimony, with dynamically adaptive priorities by means of a threshold based migration.
mechanism. Deb’s concept of dominance analysis has been refined by introducing a population clustering procedure to stress the importance of the accuracy objective whilst not completely ignoring the complexity one.

2. Nonlinear Models Linear in Parameters

A nonlinear input – output model, linear with respect to its parameters is of the following form:

\[
\hat{y}(k) = \sum_{i=1}^{r} c_i F_i(x(k)), \quad c_i \in R, \; k = 1, \ldots, p,
\]

where \( k \) stands for the current time instant, the system input and output are denoted with \( u \) and \( y \) respectively, \( \hat{y} \) indicates the estimated output provided by the designed model, \( x \) is a vector containing the current and the lagged values of plant input and output, \( n_u \) and \( n_y \) are the maximum permitted input and output lags. Nonlinear functions \( F_i \) are called regressors and represent atomic combinations (products) of terminals, namely \( x \) vector elements, considered to any exponent. The polynomial model can be rewritten in matrix-based formalism:

\[
\begin{bmatrix}
F_1(x(1)) & \cdots & F_r(x(1)) \\
F_1(x(2)) & \cdots & F_r(x(2)) \\
\vdots & \vdots & \vdots \\
F_1(x(p)) & \cdots & F_r(x(p))
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
\vdots \\
c_r
\end{bmatrix}
= \begin{bmatrix}
\hat{y}(1) \\
\hat{y}(2) \\
\vdots \\
\hat{y}(p)
\end{bmatrix},
\]

which better outlines its two main components: model structure encrypted by the regressor matrix \( F \), and model parameters encapsulated by the coefficients vector \( c \). The goal of the GP procedure is to obtain the optimal degree of regressor diversity, a task that falls under the care of specifically enhanced genetic operators, as well as compute an appropriate set of model coefficients, made possible by employing a QR local optimisation mechanism. By doing so, the generated models stand a good chance of featuring a simple structure and good accuracy on validation data sets.

3. Genetic Programming Based Identification Approaches

The paper contrasts three optimisation alternatives: one based on a single evaluation criteria, another consisting in an extension of the first by adding a supplementary optimisation objective and a pair of extra customisations, and a third approach centred on the concept of elitism, used as a
benchmark in evaluating the first two methods’ experimental results. This final elitist procedure also serves as a starting point for future algorithm enhancements.

### 3.1. Single Objective Optimisation

The trees in the initial population are built using randomly selected terms from the terminal set (vector $\mathbf{x}$ in Eq. (1)) and operator one. As the nonlinear, linear in parameter mathematical convention that the generated trees comply with is quite straightforward, the operator set need only contain two elements: $O = \{+, \ast\}$. Provided that a sufficient number of lagged values are included in the terminal vector, the two sets meet the closure and sufficiency requirements that allow the generation of trees capable of encrypting a valid model.

After completion, the trees are sent to the reproduction pool where they undergo the action of genetic operators: crossover and mutation. As both of them have been significantly improved to increase their efficiency, a special paragraph is dedicated to their description. The resulted children are then reunited with their parents to form an intermediary population. At this point, given the random nature of the parents’ generation and the shuffle-like effect of the genetic operators, the individuals are most likely not linear in parameters, hence parameter computation via Eq. (4) is problematic. To settle this predicament, a transformation mechanism is provided, to ensure parameter wise linearity. The next step is to build the regressor matrix and the output vector in Eq. (4), an operation that requires assigning numerical values to the symbolic terminal nodes within each tree. To suit that purpose, a training data set $S = \{\mathbf{u}_i, \mathbf{y}_i\}$ must be provided, containing input–output data pairs that illustrate as wide a variety of system behaviours as possible. At this point, QR decomposition computes the optimum parameter vectors for each tree in the intermediary population, preparing them for performances assessment.

The SOO approach makes use of the $\text{SEF}$ (Squared Error Function) objective only, in evaluating trees. Mathematically put, $\text{SEF}$ is a number assigned to each potential model $M$, representing how accurately the model output follows the system output, relative to the entire data set.

$$\text{SEF}(M) = \frac{1}{2} \sum_{i=1}^{P} (y_i(p) - \hat{y}_i(p))^2$$

The individuals with the lowest $\text{SEF}$ values are the best in the population, therefore they will be assigned with the highest fitness values (selection probabilities), via a ranking procedure. Consequently, these particular individuals will stand a better chance of being reinserted in the parent population by means of a roulette based selection mechanism called stochastic universal sampling, than their less fit counterparts.


3.2. Multiobjective Optimisation

The first steps of the evolutionary procedure are similar with the ones implied by SOO, up to the point of individual evaluation. Here, a supplementary criteria is considered to assess the trees’ parsimony. This extra objective is called $CF$ (Complexity Function) and is defined as follows:

$$CF(M) = \frac{n}{n_a + n_y + 1} - \sum_{i=1}^{r} \lg c_i,$$

where $r$ represents the number of regressors in tree $M$, $t$ is the total number of terminals and $c_i$ stand for the model parameters. By cumulating all three terms in the definition of the complexity function, the trees are assessed more thoroughly. An individual may feature a reduced number of regressors, each of them containing a large number of terminals. If this should be the case, the tree is penalised by the second term of the $CF$. Furthermore, individuals that include regressors with very low coefficients, therefore with a small if any contribution to the global accuracy, are not to be encouraged, hence the third component of the complexity objective.

It is obvious that the $SEF$ and $CF$ objectives are conflicting as it is highly unlikely that a rudimentary tree, in terms of complexity, should encrypt an accurate model and vice versa. The most appropriate way to assign fair fitness values that take into account both conflicting objectives is to consider a Pareto wise analysis based on the concept of non domination. A solution in the current population is nondominated if it is better than all other individuals with respect to at least one objective function. At first, all nondominated solutions are separated from the population. In a graphical representation of the objectives space, they form the first order Pareto front. The dominance analysis continues, in a similar fashion, on the remaining trees in the population, to form the second order Pareto front, and so on, until all the individuals have been assigned to a front of a certain order (Fig.1).

![Fig.1 − Different order Pareto fronts and the Pareto optimal front (a); population clustering (b).](image)
The trees situated on Pareto fronts closer to the optimal one (depicted with a solid line in Fig. 1a) are assigned higher fitness values than the individuals further away. In other words, fitness decreases as the Pareto front order increases. In order to differentiate between the trees situated on the same Pareto front, a proximity analysis is conducted aimed at identifying the solitary individuals, like tree 1. Such trees will be awarded with slightly higher fitness values than the ones in clusters (trees 2, 3, 4) as to encourage the production of offspring in the depleted areas of the Pareto front, and implicitly generate as many objectives trade-offs as possible.

This fitness assignment scheme has been enhanced by introducing a clustering technique (Fig. 1b) and an adaptive threshold migration procedure, both detailed in section 6.

3.3. Elitist Approach

The elite based attempt to solving the identification problem is, in concept, a multiobjective optimisation procedure as well. Yet, the whole of the fitness assignment scheme is based on the idea of elite set, which is somewhat equivalent to the first order Pareto front in the previous subsection. After the initial population has been generated, a random individual is included in the elite set and assigned a random fitness value \( f \). All the remaining individuals in the population are then compared in terms of dominance against all elites.

Should a tree \( M \) in the regular population be nondominated in relation to all elites, it will be included in the elite set, its fitness computed as follows:

\[
FIT(M) = FIT(E_c) + d(M, E_c),
\]

where \( E_c \) is the elite that \( M \) is closest to, and \( d(M, E_c) \) is the distance in between the two individuals, computed within the objective space. If \( M \) is dominated by one or more of the elites, than it is not included in the elite set, and given a fitness of:

\[
FIT(M) = \minFIT(E_c) - d(M, E_c), 0),
\]

with the same notations as above. After each insertion of a new elite, the elite set is updated by removing all individuals in it that are dominated by the new comer, should there exist any such trees. After the elite set has been built, all the fitness values of the trees in it are levelled up to the highest one. Please note that once an individual is included in the elite set, it is not removed from the regular population, hence, after elite fitness maximisation, all elites will have two sets of fitness values: one as an elite and one as a regular individual.

The following evolutionary steps, selection, reproduction and reinsertion are conducted only on the regular population and are similar to the ones presented before. It is important to notice that, although computing fitness values based on Eq. (7) and Eq. (8) has a general positive effect on diversity
preservation, the efficiency of the scheme depends on the order of the trees in the regular population being considered for comparison. Unless a niching mechanism is introduced, there are cases in which the effect would be the opposite than the expected one as trees in clusters could be assigned higher fitness values than solitary ones. Nonetheless, the run speed of the procedure is not to be neglected, and its downsides represent challenges for future enhancements.

4. General Enhancements

As little a priori information is available pre design, there is no information about where the solution might be situated in the problem space, therefore the individuals in the initial population must be evenly spread across the search domain in order to ensure as good a coverage as possible. The best way to achieve that desiderate is to create the trees in a pseudo-random fashion by recursively inserting nodes selected from the terminal (x) and operator sets. There are some validity rules that guide the tree building process, the most important one being that each individual must contain all the nodes in the terminal set. This way, a rich storage of raw genetic material is provided in the initial phase for the algorithm to evolve. Secondly, if all terminals have been exhausted and the tree is not yet complete (there are still operator nodes with less than two successors), constant nodes are inserted in the empty slots. After construction, a tree in the initial population is similar to the one in Fig. 2 a.

Due to the random nature of tree generation, some individuals will not comply with the linear in parameter formalism described above. To solve that problem, an adaptation mechanism is provided to “lift” all the “+” nodes situated below “*” nodes, so that the result resembles the one in Fig. 2 b. Other approaches (Madar) address this issue by simply swapping all problematic “+” nodes with “*” nodes, which is simpler in terms of implementation, yet can lead to oversized regressors should the tree root happen to be a “*” node. The

\[
\begin{align*}
  a(k) & = b(k-1) \\
  y(k) & = a(k) \\
  y(k-1) & = b(k) \\
  u(k) & = c(k-2) \\
  u(k-1) & = d(k-1)
\end{align*}
\]

Fig. 2 − Trees in the initial population: a – raw form b – regressor based form.
Another consequence of the random generation of the primary genetic material is that most trees feature unsatisfactory performances in terms of objective function values. Therefore, they are improved by means of genetic operators. The enhanced crossover operator proposed by this paper was configured to spot which subtrees would bring the highest gain in accuracy if swapped. To fit that end, the two parents about to exchange genetic material are investigated prior to the actual swap to determine any similar subtrees. Especially towards the end of the evolutionary loop, when the first order Pareto front comes close to the optimal one, well adapted identical nonlinear atoms start to appear in many trees (nodes 5, 6, 7 in the first parent and nodes 4, 5, 6 in the second parent in Fig. 3). When two such trees become parents, the similar regressors that they contain should not be considered for swapping, as the increase in children accuracy would be null. Fig. 3 illustrates such a case, where all the nodes within the identical subtrees, along with the nodes on the path to the parents’ roots are eliminated from the cut point list, in order to protect the well adapted regressors.

After the potential cut point list simplification, the only valid remaining options are node 4 in the first parent and node 3 and 8 in the second. Thus, the cut point search that blind cross over would have performed is significantly simplified.

As far as the second genetic operator is concerned, mutation has been trained to modify not only the terminal names, but also their exponents. This behaviour is targeted at dealing with an unwanted phenomenon called compensation that occurs when a model features an increased number of regressors while the same accuracy could be achieved by a solution with fewer nonlinear atoms, yet to higher exponents.

5. Multiobjective Related Enhancements

The multiobjective algorithm benefits from all the enhancements presented in the previous section plus two extra customisations targeted directly at the multiobjective paradigm. The population evolved by MOO is dual; after
generation it is split into two subpopulations: one will undergo a SOO technique, the other a MOO alternative.

The MOO evolved subpopulation is separated into two groups based on its average performances in terms of $SEF$ and $CF$. The individuals in the first group (Fig. 1b) are the best relative to both objectives and their fitness will be assigned based on the accuracy criterion only. The ones in the second group are subject to a dominance analysis based fitness assignment scheme like the one described in section 4. The purpose of this clustering mechanism is to dynamically establish the objectives weights in tree evaluation, based on the mean population performances and, more importantly, to emphasise the importance of the $SEF$ objective without completely ignoring the complexity one. Accuracy is of great interest in the field of automatic control, therefore this enhancement is aimed at encouraging the production of models in the interest zone of the Pareto optimal front (see models marked with “o” in Fig. 1b).

To increase the $SEF$ related selection pressure even more, the two subpopulations exchange their “champions” every few generations. The exact number of individuals that get swapped during migration is determined dynamically, based on the average complexity of the trees in the two subpopulations. Three preset migration rates are considered, as follows: rate1 = 10%, rate2 = 20% and rate3 = 25%. If the average complexity of the SOO population is below that of the MOO population, then many of the simple and accurate trees evolved solely via $SEF$ have a good chance of being included in the first group (Fig. 1b). Therefore, the maximum allowed percentage (25%) of SOO trees will migrate to the MOO population, whilst only 10% will be sent the opposite direction. The second possibility is that of an average SOO complexity comparable to the MOO one, leading to a 20% migration rate in both directions. Finally, if the SOO trees are substantially more complex than the MOO ones, then their chances of being included in group 1 and of populating the interest region of the first order front are slim. Only 10% of the SOO trees will migrate towards the MOO population, while 25% simpler, yet less accurate trees will be sent the opposite direction.

### 6. Comparative Experimental Results

All three takes on solving the identification problem have been deployed to find a model for the steam subsection of the sugar factory in Lublin, Poland, a complex nonlinear industrial plant featuring steam pressure as input and steam temperature as output [8]. All three algorithms use the same configurations, for the relevance of the results comparison (Table 1).

Firstly, the performances of ESOO (Enhanced SOO) have been analysed and compared against EMOO (Enhanced MOO) and EEMOO (Elitist EMOO). The experimental conclusions are summarised in Table 2. Note that the results displayed for the EMOO procedure refer to the best tree, in terms of accuracy, situated on the first order Pareto front. In the case of EEMOO, the
tree the data in the tables refers to is the most accurate elite in the final set generated by the algorithm.

The ESOO algorithm assesses the performances of the current population’s individuals only with respect to their accuracy, completely ignoring the complexity criterion. As a direct consequence, higher maximum lags lead to an increased number of regressors. That behaviour may tend to become disturbing, as the solutions’ generalisation capabilities decrease. Models M1 through M3 show poorer SEF values over the validation data set, caused by overfitting. Another important setback of the SOO procedure is the lack of any diversity preservation mechanism that causes the final generations to get saturated with fairly identical trees. At this point, no matter how many evolutionary steps are considered, the SEF value stops improving (M4→M6). The elitist alternative (EEMOO) converges faster than its two counterparts, yet the saturation setback remains, and is accentuated even more due the algorithm’s intense selective nature.

<table>
<thead>
<tr>
<th>Model ID</th>
<th>Lags</th>
<th>Number of individuals/number of generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1 197</td>
<td>$n_u=3$, $n_y=3$</td>
<td>50/50</td>
</tr>
<tr>
<td>M2 197</td>
<td>$n_u=5$, $n_y=4$</td>
<td>50/50</td>
</tr>
<tr>
<td>M3 197</td>
<td>$n_u=7$, $n_y=6$</td>
<td>50/50</td>
</tr>
<tr>
<td>M4 200</td>
<td>$n_u=3$, $n_y=2$</td>
<td>60/80</td>
</tr>
<tr>
<td>M5 297</td>
<td>$n_u=3$, $n_y=2$</td>
<td>60/100</td>
</tr>
<tr>
<td>M6 350</td>
<td>$n_u=3$, $n_y=2$</td>
<td>80/200</td>
</tr>
</tbody>
</table>

Fig. 4 — Model validation.

The EMOO procedure is more flexible than the standard MOO as it features two extra tuning mechanisms. The first one refers to the special fitness assignment scheme. As a general rule, too simple trees cannot encrypt accurate models. Therefore, if the CF goal value is just as restrictive as the SEF goal value, the population will rapidly be taken over by rudimentary models, unable to provide good accuracy. Results are better as the CF goal laxity is increased (M7→M9). The second tuning mechanism refers to the migration technique. Once every $No.m$ generations the two subpopulations exchange genetic material. If the migration rate is too high, the selection pressure imposed by the individuals with low SEF values from the SOO subpopulation will counterbalance the efforts of the MOO complementary procedure (M10), leading to results similar to the standard SOO (accurate, but complex). A rare migration produces results which resemble the classic MOO case (M11).
To better illustrate the quality of the models produced by the EMOO procedure, an accuracy graph is provided (Fig. 4), contrasting the output of the model with the best accuracy within the Pareto first order front generated for M1 (Table 1), and the actual output of the real system. The model in question features a mean SEF value of 0.720 over the validation data set and a complexity of 4 regressors.

7. Conclusions

1. All three genetic programming based algorithms described in this paper represent innovative instruments in system identification. Their ability to provide accurate yet compact models with good performances both on training and validation sets, without any prerequisites other than training/validation data, makes them able to cope with the special requirements of complex systems featuring dynamic nonlinearities. The search procedure is unsupervised and simultaneously improves the model structure and parameters, excluding the need of any off-line term reduction mechanisms.

2. Due to the linear in parameters formalism that the generated models are compliant with, a hybridisation with a local optimisation procedure is facilitated. Along with the random distribution of the initial population, the enhanced genetic operators guarantee a speedy convergence and a good variety of the population at any generation. By assessing the generated models with respect to multiple objectives, the genetic search becomes more refined. Clustering the individuals by considering adjustable objective goals, and evolving each cluster in a manner compatible to their average performances balances out the priorities of the accuracy and parsimony criteria in a dynamic way. Moreover, the separation in two subpopulations, evolved simultaneously, one according to a simple objective technique (accuracy) and the other to a
multiobjective technique (accuracy and complexity) allows the generated models to embody the advantages of both evolutionary procedures.

Received: January 19, 2009

“Gheorghe Asachi” Technical University of Iași, Department of Automatic Control and Applied Informatics
e-mails: apatelli@ac.tuiasi.ro
lferaru@ac.tuiasi.ro

REFERENCES


TEHNICI MULTIPLE BAZATE PE PROGRAMARE GENETICĂ PENTRU IDENTIFICAREA SISTEMELOR NELINIARE

(Rezumat)

Trei tehnici evolutive, îmbunătățite cu mecanisme originale menite să le crească performanțele, au fost folosite în identificarea de sisteme neliniare industriale. Optimizarea simplu obiectiv, cea multiobiectiv și cea multiobiectiv elitistă sunt descrise în lucrare atât în forma de bază cât și în cea îmbunătățită, expunere susținută de o analiză comparativă a rezultatelor experimentale obținute în contextul identificării subsecțiunii “aburi” a stației de evaporare din cadrul fabricii de zahăr din Lublin, Polonia.

Un avantaj general al celor trei metode constă în capacitatea de a rezolva probleme de identificare în absența unor date bogate despre funcționarea sistemului,
singurele informații necesare fiind cuprinse de seturile de date de antrenare respectiv validare culese în timpul funcționării sistemului șintă. În plus, metodologia prezentată e capabilă să conducă o optimizare nesupervizată și simultană a structurii și parametrilor modelului, fără a fi necesare simplificări sau adăugiri ulterioare de termeni. Datorită codificării flexibile, sub formă de arbore, a modelelor potențiale evolute de algoritmul propus, și rafinării evaluării acestora, atât din punct de vedere al acurateții cât și al complexității, soluțiile generate înregistrează o bună capacitate de generalizare, fiind bine adaptate cerințelor aplicațiilor de automatizare.

Îmbunătățirile aduse algoritmilor de bază constau într-un mecanism evasatoriu de generare a populației inițiale, asigurându-se astfel o bună acoperire a spațiului de căutare și un material genetic de start îndeajuns de bogat pentru obținerea unui model de calitate. Transformarea arborilor din forma brută în cea bazată pe regresori, proces menit a asigura compatibilitatea indivizilor cu formalismul matematic al modelelor nelineare, liniare în parametri, asigură o bună hibridizare cu o metodă rapidă de calcul al parametrilor, bazată pe descompunerea QR. Operatorii genetici au fost de asemenea antrenați suplimentar pentru a conduce eficient cu procedura de optimizare locală și pentru a minimiza timpul de căutare a subarborilor respectiv nodurilor a căror modificare ar aduce cele mai semnificative câștiguri în ceea ce privește obiectivele considerate.

Îmbunătățirile aduse algoritmilor multiobiectiv sunt menite să încurajeze menținerea diversității populației și să accentueze în mod dinamic prioritatea obiectivului acuratețe, fără a ignora complet importanța complexității. Astfel, populația e împărțită în două subpopulații: una e îmbunătățită numai în ceea ce privește acuratețea, cealaltă ține cont și de acuratețe și de complexitate, obiective gestionate din prisma unei analize de dominantă. Cea de-a doua subpopulație (optimizată multiobiectiv) e împărțită în două grupe în funcție de performanțele medii în materie de acuratețe și complexitate. Primul grup, conținând arborii cei mai bine adaptați, va fi optimizat doar pe bază de acuratețe, al doilea grup fiind supus unei analize multiobiectiv. În plus, o dată la câteva generații, cele două subpopulații schimbă material genetic. Numărul exact al arborilor care migrează dintr-o subpopulație în cealaltă e dinamic, fiind configurat în funcție de complexitatea medie a arborilor.

Rezultatele experimentale evidențiază o creștere în performanța a metodelor îmbunătățite față de variantele lor de bază, recomandând metoda propusă pentru rezolvarea problemelor de identificare ce nu dispun de un set bogat de date despre evoluția internă a procesului, dar impun cerințe stricte de acuratețe a modelului sistemului.