GA-P based search of structures and parameters of dynamical process models

López, A.M.[1], López[1], H., Sánchez, L.[2]

Oviedo University, Spain.
[1] Electrical Engineering Department. C. Universitario de Viesques.
Ed. Departamental N°.2, 33204, Gijón, Asturias.
e-mail: [antonio,hilario]@isa.uniovi.es
[2] Computer Science Department. C. Universitario de Viesques.
Ed. Departamental N°.1, 33204, Gijón, Asturias.
e-mail: luciano@lsi.uniovi.es

Abstract. The most effective approaches for evolutionary identifying dynamical processes depend on iterative trial-error searches in a hierarchical fashion: a new structure is proposed first; then, its set of parameters is numerically determined, and the process is repeated until a model accurate enough is found.
Canonical Genetic Programming has been used to automate this search; but its output can be difficult to interpret. Because of this reason, the use of hierarchical learning methods, that combine GP search of structures with deterministic optimization algorithms, has been proposed. We will show in this paper that the output of such methods can be further improved with non hierarchical algorithms. In particular, we will show that the use of GA-P improves the interpretability of the models and does a better model search than previous approaches.

Keywords: GA-P algorithms, Genetic Programming, System Identification, Hierarchical models

1 Introduction

Most of the evolutionary methods for system identification from sampled data focus in nonlinear state space-based models. For this kind of models, the objective of the learning process is the production of a set of difference equations defining the dynamics of the process. Unfortunately, for practical purposes, a set of equations that relate all state variables between them is hard to manage in all but small sized problems. Modular representations are usually preferred, because they allow to determine groups of variables affected by specific parameters.
Genetic Programming has been applied to learn such modular models. One of the first examples was given in [9], where a structured Genetic Algorithm, in a tree based representation, is used. The set of functions that was proposed contained only two-inputs quadratic functions, which are not the building blocks that control engineers expect to find in structured models. Some implementations nearer

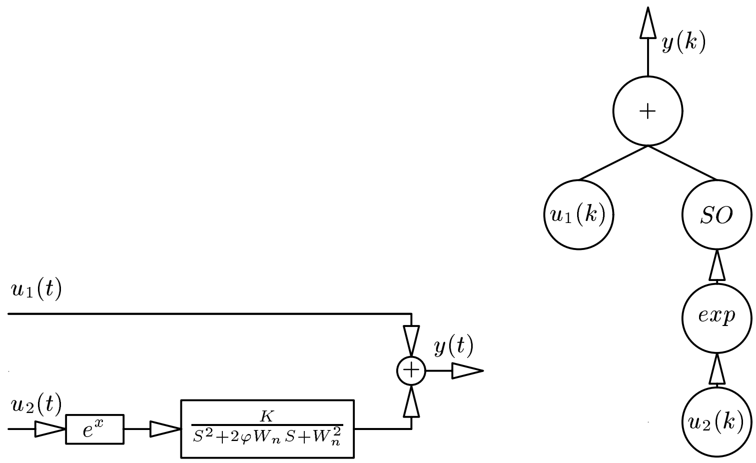


Fig. 1. Block diagram representation of a system (left) and its tree based representation (right.) "SO" stands for "Second Order" and "*exp*" for "exponential function".

to usual practice can be found in [2, 4–6, 10, 19, 23] and other, less common approaches to model the dynamics of a system, are described in [7, 17, 24]. Most of these schemes introduce dynamic considerations by means of extended terminal sets, that include either input and input-output delayed variables.

One of the most complete methods is described in SMOG [15, 16]. The problem is addressed there as a search of a diagram block based representation of a model of the process in a tree codification (see figure 1). The function set used includes continuous time blocks defined in the domain $\mathcal{S}$, making the dynamical considerations intrinsic to the search. Recently, a similar approach has been used for the induction of process controllers in [11].

Under the considered approach (see Figure 2), hierarchical evolutionary algorithms are applied: canonical GP is used for the evolution of model structures and combined with deterministic numerical optimization methods (Hooke and Jeeves algorithm) for parameter tuning. An iterative search of structure and parameters is done: each model considered is parametrically tuned by means of Hooke-Jeeves algorithm as a previous step to fitness evaluation. Genetic operators defined for evolution affects only the structure of the models.

We will show in this paper that, according to our experimentation, better results can be obtained if a new representation and a new set of genetic operators are used. The representation that is proposed in this paper is adapted from an idea first proposed in [8], and shares characteristics with GA and GP algorithms, being able to search in parallel in both structure and parameter spaces.

The focus will be put not only at the capabilities of the solutions to reproduce the sampled data used for training or validation. They will be also structurally

```
a. Initialize random population of models.
b. Tune parameters of models in the population (Hooke-Jeeves algorithm).
c. Calculate fitness.
d. Selection of models and application of genetic operators.
e. Go to b).
```

Fig. 2. SMOG evolution. Canonical GP is used for structural search and Hooke-Jeeves method is used for parameter tuning.

compared with a known model for the target system. This way, they can be analyzed as explaining methods of the underlying relationships in the data.

1.1 Structure of the paper

The outline of this paper is as follows: in section 2, the scope of application of this method is introduced. In section 3, the parallel search of parameters and structure done by the GA-P algorithm is described. Then (section 4) an experimental validation of our proposal is done, modeling both a synthetic and a real process and comparing the results with those obtained with previous works. The paper finishes with the concluding remarks and future work.

2 Scope of application

Our interest is focused over a class of physical systems involving common non-linear features, to which conventional methods are hardly applicable. Being a GP based modeling approach used, the definition of the functional set will define the scope of application of the algorithm. The GP will evolve a set of diagram block representations of the process. A diagram block is, in turn, a series, parallel or feedback association of subsystems. Series association is intrinsic in GP. Parallel association will be allowed by means of arithmetic operators, such as + and -, and feedback representation will be allowed by means of an special operator [1] described next.
Regarding the catalog of subsystems, we used only memoryless version of the common non-linear features of physical systems, such as dead zones or saturations [14, 20]. All the dynamic behavior is delegated to linear elements: we include in the function set a reduced group of linear models (first and second dumped order linear subsystems, unitary delay and static gain) such that it is possible to get higher order systems by means of series association.

3 Proposed algorithm

3.1 Drawbacks of the hierarchical learning process

There is an inherent drawback with the hierarchical learning: the searches of the numerical parameters best suited for the structures being produced by GP are,

$k_2+k_1*x = 4+3x$

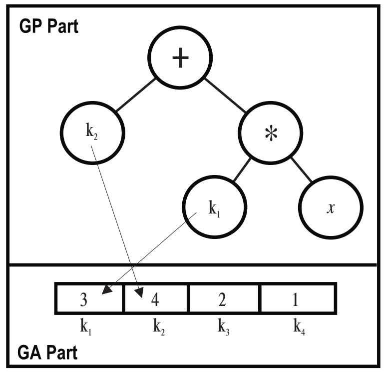


Fig. 3. Representation of a generic individual in GA-P algorithms. Individuals in GA-P have two parts: a tree based representation and a chain of numerical parameters.

themselves, multi modal problems [1]. Therefore, deterministic methods fall frequently into local minimum points and, as a consequence of this, a good structure can be assigned a low score in the search process. Despite this problem, hierarchical approaches are able to find good models because GP can produce several times the same structure with different initial values for the numerical parameters. Thus, the deterministic algorithm will eventually find the global minimum. But under this context the GP is not only being used to search different structures but also to search different numerical starting points, a problem in which GP is known not to perform too well.

In previous studies [1], we have tried the replacement of Hooke and Jeeves method with a real coded genetic algorithm, obtaining good numerical results. Anyway, such a hierarchical approach is a highly consuming task, because many resources are wasted in the identification of structurally invalid systems. An strategy that does not need the GA to converge before examining a new structure, and that does not discard too soon structures that may be valid, is needed.

3.2 GA-P algorithms

GA-P [8] is an hybrid between genetic algorithms and genetic programming, that was first used in symbolic regression problems. Individuals in GA-P have two parts: a tree based representation and a chain of numerical parameters. Different from canonical GP, the terminal nodes of the tree never store numbers but linguistic identifiers that are pointers to the chain of numbers (see Figure 3).

The behavior of the GA-P algorithm is mainly due to its crossover operator. Later in this section it will be described in detail how we adapted it to the problem at hand; let us say for the time being that either the tree parts or the chains of parameters may be selected and crossed, thus the GA search of the parameters and the GP search of the structures are being done in parallel. This way, individuals structurally fitted will have more possibilities to undergo an intensive parameter optimization, while those structurally unfitted will tend to disappear. A niche strategy [21] is used in the evolutionary process, preventing the search to fast fall into local minimum points.

Representation Tree based representation makes it impossible to model a wide set of systems, such as those involving nested or not unitary feedbacks. The reason is that a block diagram is not a tree when it includes feedback, but a directed graph. Structure and parameters parts of the representation are defined as follows:

- Structural component. The proposed representation (see figure 4), mixes a link nodes list with ideas from [16] and [22]. A special feedback node is used. Both input and the feedback branches originate in it. The terminal nodes of the feedback branch (marked as "**") are recessive. This way, standard structural modification operators can be applied at any point in the individual to evolve structures.
  It also contains a third link to another node in the graph from which the feedback signal will be taken. This pointed node must be contained in the path between the feedback node and the output node of the system. Otherwise, feedback node looses its significance. This consideration must be present in the creation and modification of individuals as a consequence of structural genetic operators. When an individual does not accomplish this condition after an structural modification, invalid feedback nodes are reinitialized.
  Algebraic loops are neglected by means of the implicit inclusion of a unit delay in the feedback branch. To prevent series associations of delays, dynamic blocks used respond instantly. But, known the fact that physical systems never respond instantly to an excitation, a unit delay is also implicitly linked to the output of the model.
- Parameters component. It contains a vector of values with the parameters of the model to be evolved by the GA component of the algorithm. It is used a real value codification based on [3].

Genetic Operators Two sets of operators are applied in the evolutionary process:

- Structural Genetic Operators. Subtree crossover [13] and internal crossover [12] are used. Subtree, node and a special operator for feedback mutation operators are also used. This set of operators only affect the structural component of the individuals involved, not the parametric one.
  All of the structural operators act over tree based representations. Therefore, feedback links are inhibited during the process.
- Parameter Genetic Operators. Two structurally identical individuals are selected from the population for each application of this set of operators. They only affect their parameter component, not the structural one. Real based

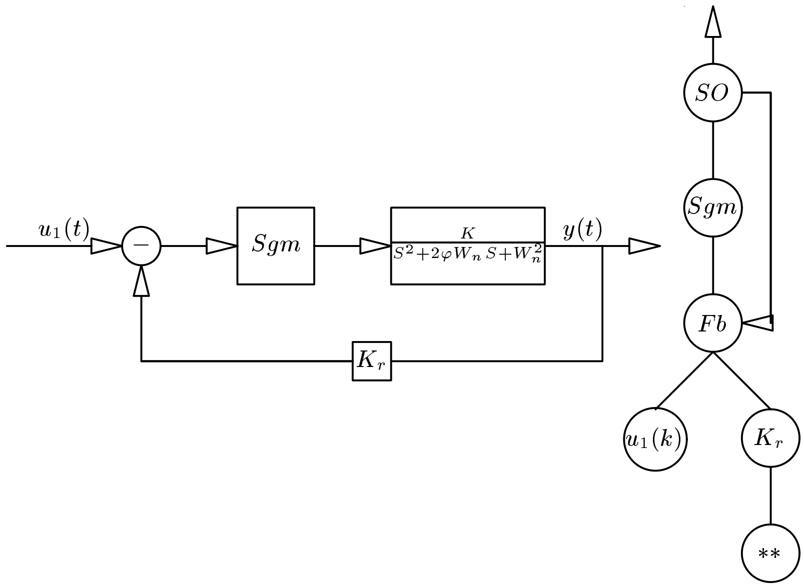


Fig. 4. Block diagram representation of a feedback system (left) and its genetic representation (right.) "SO" stands for "Second Order" and "Sgm" for "sigmoid function". Also, "**" stands for a recessive terminal.

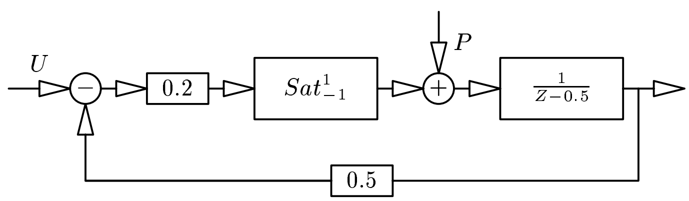


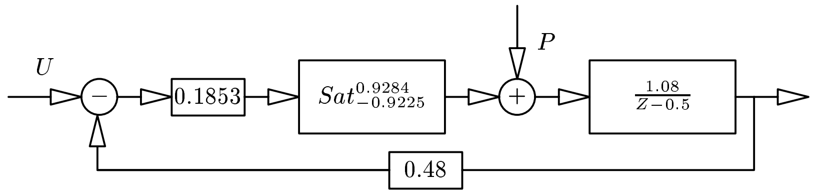
Fig. 5. Target model. "Sat_a^b" stands for "saturation" block with limits in a, b.

genome crossover operator is defined for the parameters of the model as a random movement of a vector in the direction of the other. After crossover, a mutation, a direct search or both can be applied to the resulting offsprings depending on predefined probabilities. Mutation is defined as a crossover with a randomly generated individual. Direct search is performed by means of Nelder & Mead algorithm [18] run for a few iterations.

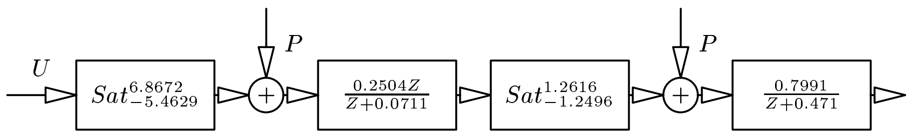
4 Numerical results

4.1 Modeling an empirical system

To validate our approach, an empirical control system of a first order process with a proportional saturated controller and a sensor without dynamics (see figure



(a) GA-P best model



(b) Hierarchical approach best model

Fig. 6. Differences in the structures of the learned models. "Sat_a^b" stands for "saturation" block with limits in a, b.

5) was modeled by means of the defined GA-P strategy. It was also compared with a hierarchical process. Both approaches were stopped after certain number of evaluations of the objective function.

Experiments were repeated 10 times each. Figure 7 contains validation errors for each experiment. Observe that GA-P improves slightly the results, but the differences are not significant. The gain with GA-P is in the identified structure (see Figure 6, where the best models obtained by both approaches are shown.) Observe that only little deviations are present in the parameters values, a problem which could be easily solved by the application of more intensive optimization procedure over that structure. In this case, GA-P found exactly the structure of the target model, explaining very well the data relationships. In contrast, the hierarchical method was trapped in a local minimum of the structure. It is only capable of fitting the sampled data.

4.2 Modeling a real process

As a final test, a real process was modeled by means of the proposed scheme. A DC motor was selected, in order to have information enough to contrast the GA-P solution with a known model for the process (usually a first or second order dumped linear subsystem with a non-linear dead zone component).

Experimental conditions were the same as in the preceding section. Table 7(c) contains the numerical validation errors for each experiment. From it, it can be concluded that the best result was found at experiment 10, shown in figure 8(a). Solution is close to a known model for the system. It is capable of capturing the

Experiment	Error
1	0.00017
2	0.0004
3	0.00005
4	0.00004
5	0.00019
6	0.00005
7	0.00005
8	0.00006
9	0.00029
10	0.00007
Average	0.00014

(a) GA-P modeling errors, synthetic problem

Experiment	Error
1	0.00206
2	0.00301
3	0.00129
4	0.00184
5	0.00287
6	0.00112
7	0.00111
8	0.00107
9	0.00147
10	0.00263
Average	0.00185

(b) Hierarchical approach modeling errors, synthetic problem

Experiment	Error
1	0.9196
2	0.7755
3	0.7354
4	0.8433
5	0.9223
6	0.9259
7	1.1809
8	1.0134
9	1.0976
10	**0.6933**

(c) Numerical modeling errors for the direct current motor

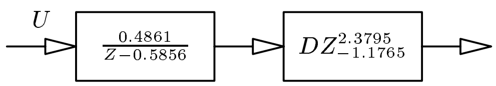
Fig. 7. Numerical modeling errors.

most significant relationships in the data. Finally, in Figure 8(b), a comparison between the motor and the model responses is shown. Observe that the behavior is correctly reproduced and the noise is smoothed as expected.

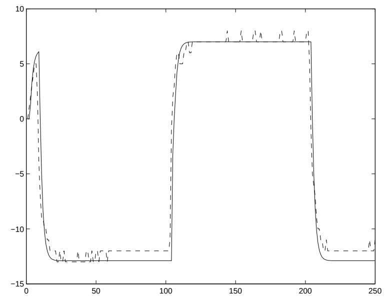
5 Concluding remarks and future work

The identification of nonlinear systems from sampled data is a multimodal problem either in structure and parameter spaces. We have shown that “state of the art” hierarchical learning algorithms can be trapped in these minimum points and be unable to find the right structure in certain cases. We have solved this problem by introducing a parallel evolutive search of parameters and structure that does not waste time optimizing parameters for invalid structures neither discards structures too early.

While being able to process more complex problems than its predecessors, this learning algorithm is not complete. In practical situations we need to be able to incorporate expert knowledge to the system, either in the form of structural restrictions or by means of closed submodels with known expression around which a joint model should be evolved. In a near future, we plan to incorporate a measure of structural quality to the fitness function and use multicriteria evolutionary algorithms to obtain a family of solutions with balanced precision and complexity from which the control engineer can choose.



(a) Direct current motor model. "DZ_a^b" stands for "dead zone" block with limits in a, b.



(b) Model (continuous line) and system (plotted line) responses

Fig. 8. Direct current motor modeling

References

1. H. Lopez A.M. Lopez and L. Sanchez. Graph based gp applied to dynamical systems modeling. *Connectionist Models of Neurons, Learning Processes and Artificial Intelligence*, pages 725–732, 2001.
2. V M Babovic, B Brozkova, M Mata, V Baca, and S Vanecek. System identification and modelling of vltava river system. In *Proceedings of the 2nd DHI Software User Conference*, 1991.
3. Lawrence Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, NY, 1991.
4. Saso Dzeroski, Ljupeo Todorovski, and Igor Petrovski. Dynamical system identification with machine learning. In Justinian P. Rosca, editor, *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, pages 50–63, Tahoe City, California, USA, 9 July 1995.
5. Anna I. Esparcia Alcazar and Ken C. Sharman. Some applications of genetic programming in digital signal processing. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1996 Conference*, pages 24–31, Stanford University, CA, USA, 28–31 July 1996. Stanford Bookstore.
6. Gary J. Gray, David J. Murray-Smith, Yun Li, and Ken C. Sharman. Nonlinear model structure identification using genetic programming. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1996 Conference*, pages 32–37, Stanford University, CA, USA, 28–31 July 1996. Stanford Bookstore.
7. Hugo Hiden, Mark Willis, Ben McKay, and Gary Montague. Non-linear and direction dependent dynamic modelling using genetic programming. In John R. Koza, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max Garzon, Hitoshi Iba, and Rick L. Riolo, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 168–173, Stanford University, CA, USA, 13-16 July 1997. Morgan Kaufmann.
8. Les M. Howard and Donna J. D'Angelo. The GA–P: A genetic algorithm and genetic programming hybrid. *IEEE Expert*, 10(3):11–15, June 1995.

9. Hitoshi Iba, Takio Karita, Hugo de Garis, and Taisuke Sato. System identification using structured genetic algorithms. In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, pages 279–286, University of Illinois at Urbana-Champaign, 17-21 July 1993. Morgan Kaufmann.
10. Martin A. Keane, John R. Koza, and James P. Rice. Finding an impulse response function using genetic programming. In *Proceedings of the 1993 American Control Conference*, volume III, pages 2345–2350, Evanston, IL, USA, 1993.
11. Martin A. Keane, Jessen Yu, and John R. Koza. Automatic synthesis of both topology and tuning of a common parameterized controller for two families of plants using genetic programming. In Darrell Whitley, David Goldberg, Erick Cantu-Paz, Lee Spector, Ian Parmee, and Hans-Georg Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, pages 496–504, Las Vegas, Nevada, USA, 10-12 July 2000. Morgan Kaufmann.
12. Kenneth E. Kinnear, Jr. Alternatives in automatic function definition: A comparison of performance. In Kenneth E. Kinnear, Jr., editor, *Advances in Genetic Programming*, chapter 6, pages 119–141. MIT Press, 1994.
13. John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.
14. P.H. Lewis and C. Yang. *Basic control systems engineering*. Prentice Hall, Englewood Cliffs, New Jersey, 1997.
15. Peter Marenbach. Using prior knowledge and obtaining process insight in data based modelling of bioprocesses. *System Analysis Modelling Simulation*, 31:39–59, 1998.
16. Peter Marenbach, Kurt D. Betterhausen, and Stephan Freyer. Signal path oriented approach for generation of dynamic process models. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 327–332, Stanford University, CA, USA, 28–31 July 1996. MIT Press.
17. B. McKay, B. Lennox, M. J. Willis, G. W. Barton, and G. A. Montague. Extruder modelling: A comparison of two paradigms. Technical report, Chemical Engineering, Newcastle University, UK, 1996. Appears in Control '96.
18. R. Mead and J. A. Nelder. A simplex method for function minimization. *Computer Journal*, 7(4):308–313, 1965.
19. Howard Oakley. Two scientific applications of genetic programming: Stack filters and non-linear equation fitting to chaotic data. In Kenneth E. Kinnear, Jr., editor, *Advances in Genetic Programming*, chapter 17, pages 369–389. MIT Press, 1994.
20. K. Ogata. *Modern Control Engineering*. Prentice-Hall, Englewood Cliffs, New Jersey, 1996.
21. L.A. Sanchez and J.A. Corrales. Niching scheme for steady state ga-p and its application to fuzzy rule based classifiers induction. *MATHWARE*, 7(2-3):337–350, 2000.
22. Ken C. Sharman and Anna I. Esparcia-Alcazar. Genetic evolution of symbolic signal models. In *Proceedings of the Second International Conference on Natural Algorithms in Signal Processing, NASP'93*, Essex University, UK, 15-16 November 1993.
23. A. H. Watson and I. C. Parmee. Identification of fluid systems using genetic programming. In *Proceedings of the Second Online Workshop on Evolutionary Computation (WEC2)*, number 2, pages 45–48, 1996.
24. Mark Willis, Hugo Hiden, Mark Hinchliffe, Ben McKay, and Geoffrey W. Barton. Systems modelling using genetic programming. *Computers in Chemical Engineering*, 21:S1161–1166, 1997. Supplemental.