Forecasting time series with Hyper-Volume Error Separation (HVES)

Cyril Fillon Alberto Bartoli Carlo Poloni
University of Trieste, Via Valerio, 10, 34127 Trieste, Italy
{cfillon, bartolia, poloni}@univ.trieste.it

***Abstract*— Time series prediction is a crucial task in many areas but the development of effective modeling and simulation methods to understand or predict the behavior of time dependent phenomena remains particularly difficult. In this paper we propose to use a Genetic Programming (GP) approach as a robust method for coping with problems in which finding a solution and its representation is difficult but evaluating the performance of a candidate solution is reasonably simple.**

A new methodology is applied in synergy with the GP process. The original time series is transformed in a multidimensional input space where a variable is assigned to each distinct time delay. Then, the method deals with scalar functions of N variables and subdivides the input space of N dimensions in two input spaces. This subdivision is realized by a new algorithm called Hyper-Volume Error Separation (HVES), able to divide the original input space according to the errors made by the best individual found in the early steps of the GP process.

Our results show that coupling HVES with GP is an effective approach for this task and could be part of the toolbox of many analysts. Moreover the formulas obtained with the GP process could give better insights on time dependent phenomena.

I. Introduction

In the last decades, engineers and decision makers expressed a growing interest in the development of effective modeling and simulation methods to understand or predict the behavior of many phenomena in science and engineering. Many such methods are based on regression analysis of a data sample in order to construct mathematical models for convenience and ease of interpretation. It is usually assumed that the data points in the sample are related to a unique function. There are many applications, however, in which this assumption may constitute an oversimplification, such as signal processing, time series prediction, pattern recognition and so on. In such cases the data-set could span across portions of the input space that are to be modeled differently, which means that the symbolic regression should produce a discontinuous function.

Attacking this problem involves facing several challenging issues:

1) Localizing discontinuity boundaries in the data sample without preliminary knowledge about their number and location.
2) Partitioning the data-set according to such boundaries, in order to improve the fit on each partition.
3) Assembling the formulas found in each partition into a consistent hierarchy in order to provide a single discontinuous function.

Obviously, performing the above steps in a multidimensional space adds substantial further complexity.

In this work we describe a novel approach based on Genetic Programming (GP). GP is an automatic method for creating computer programs by means of artificial evolution [1]. GP may be a powerful means for coping with problems in which finding a solution and its representation is difficult but evaluating the performance of a candidate solution is reasonably simple [2]. GP is particularly suitable for symbolic regression problems, especially when the form of the approximating function is not known beforehand, because the process automatically optimizes both the functional form and the coefficient values of the formula.

With GP, a population of computer programs is generated at random. Each program is associated with a fitness value which depends on its ability to solve the problem. Fitter programs are selected for recombination to produce a new population by using genetic operators, such as crossover and mutation. This step is iterated for some number of generations until the termination criterion of the run has been satisfied — e.g. a program exhibiting the maximum possible fitness value has been found. The evolutionary cycle is illustrated in Figure 1.

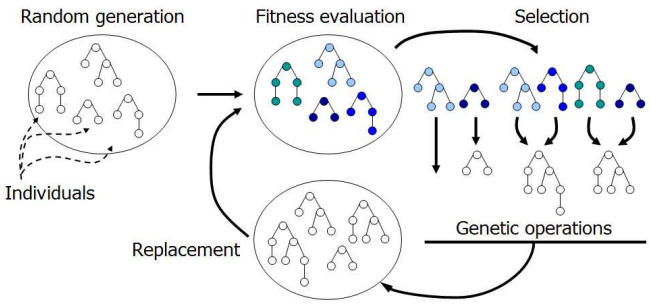

Fig. 1. Basic scheme for artificial evolution

Programs are usually represented as abstract syntax trees, where a branch node is an element from a *function set* which may contain arithmetic, logic operators, elementary functions with at least one argument. A leaf node of the tree is instead an element from a *terminal set*, which usually contains variables, constants and functions with no arguments.

Symbolic regression problems are usually faced with a function set including basic arithmetic operators (e.g., $+, -, \times, /$) and other elementary functions (e.g., $exp, log, cosine, sine$). As these basic elements are used by the GP process to evolve more elaborate programs — i.e., formulas — most of any resulting formula will be continuous and smooth. To improve accuracy when the underlying model is discontinuous, one can introduce in the function set conditional operators and

relational operators (*if*, $\leq, \geq, =$).

In this paper we propose an alternative approach, suitable for time series data-set and working as follows. We generate an initial population from scratch and let this population evolve for a small number of generations. We select the best individual and evaluate the error for each fitness case. This error is used by an algorithm developed by us, that we call *Hyper-Volume Error Separation (HVES)* and implements an heuristic for identifying the portions of the input space requiring different approximating functions. Next we reflect such partition of the input space on the data-set and run several preliminary evolutions, one for each partition. The populations resulting from such independent evolutions are finally merged and evolved again.

We applied our approach on 11 time series provided by the Artificial Neural Network & Computational Intelligence Forecasting Competition (*http://www.neural-forecasting-competition.com*).

This paper is organized as follows. In Section II we give an overview of the Genetic Programming strategy coupled with the HVES algorithm. Section III describes the experimental procedure used to forecast the 18 missing values. Section IV concludes and anticipates on further evolutions related to this new methodology.

II. Coupling GP with HVES: an overview

In this section we describe step by step the working principles of our approach. More details on the HVES algorithm may be found in *http://www.units.it/bartolia/download/HVES_tech_report.pdf*. Clearly, if the search finds an individual that solves the problem for a given data sample, then the search stops immediately. For ease of description, we omit this action from the description below.

A. Model description

We generate an initial population P_I from scratch and let this population evolve for a predefined number of generations. We select the best individual and evaluate the error for each fitness case. The resulting errors and the entire dataset D are given as parameters to our HVES algorithm. This algorithm partitions D in two subsets D_H and D_R according to an heuristic described later. Then we generate two further populations from scratch, say P_H and P_R, and let them evolve for a small and predefined number of generations on only part of the dataset: P_H is given only D_H whereas P_R is given only P_R (Figure 2).

Finally, we merge the evolved P_I, P_H, P_R and let the resulting final population P_F evolve for a predefined number of generations on the entire dataset D (Figure 3). We discovered in our early experiments that this merging step is very helpful. Each evolution phase consists of the same number of generations and involves a population of the same size.

Our HVES algorithm works as follows. It partitions the input space in several hyper-volumes whose boundaries are determined by discontinuities in the error function (i.e., the function associating the error of the best individual with each fitness case). Then, it selects the "most difficult" hyper-volume (see below) and partitions the dataset D in two regions: one D_H including all the fitness cases within this hyper-volume, and one D_R including all the remaining fitness cases. Recall that after HVES we focus the evolution of a population on D_H and of *another* population on D_R. The choice of the "most difficult" hyper-volume is made through an index describing a trade-off between number of fitness cases and resulting error — either few points with a large error, or many points with a small error. The rationale is that such hyper-volume should not contain any discontinuity.

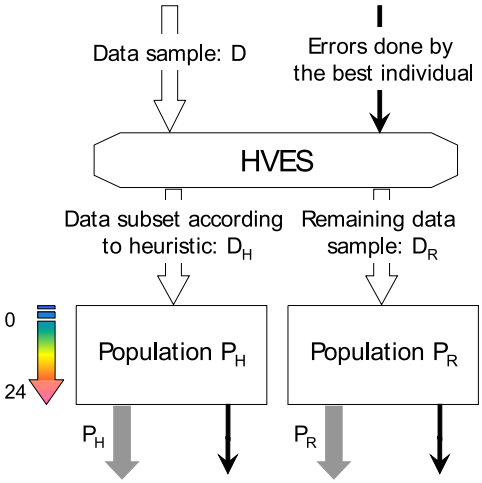

Fig. 2. GP with HVES in the division phase (thick gray arrows represent populations, thick empty arrows represent data-sets)

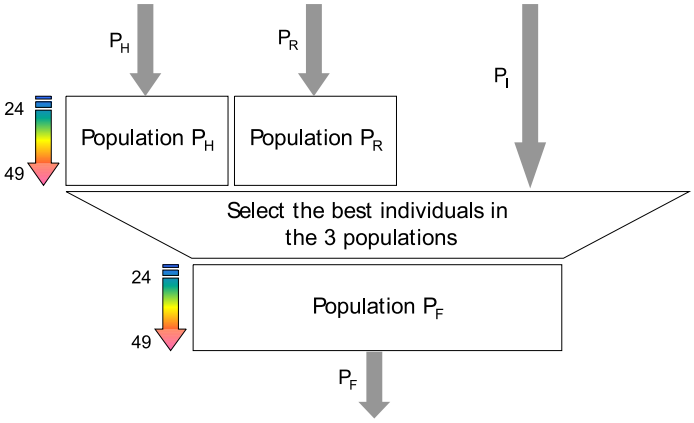

Fig. 3. GP with HVES in the merging phase

The algorithm is also applied *recursively* in between HVES and the merging phase, as follows (recursion is not shown in the figures, for clarity). The pair (P_H, D_H) produced by HVES plays the role of (P_I, D). The population P_H actually used for the merging phase is the one produced by this recursion. Recursion stops when one of the following conditions is satisfied.

(i) A maximum decomposition depth defined by the user is reached.

(ii) The HVES algorithm does not find any discontinuity boundaries.

The same applies to (P_R, D_R). Recursion turns out to be helpful for finding all discontinuity boundaries and for improving the accuracy on difficult regions of the data sample.

III. Experimental setup

The data-sets proposed contain 126 values except for the time series numbered 104, 108 and 109 which contains 116, 116 and 123 values respectively. We used the first third of the data as cross validation set in order to evaluate the generalization capabilities of the candidates solutions. The training set consists in the remaining values. For each data-set we transform the original input space in a multidimensional input space where a variable is assigned to each distinct time delay. For our experiments we used all values of time delay ranging in the interval $[1, 18]$. The functions and terminals set used in each case are shown in Table I.

TABLE I
TERMINALS AND FUNCTIONS SET

Terminals set	**Functions set**
Time series 101 to 111 with time delay $t \in [1..18]$	$+, -, /, \times$

For all data-sets we used as fitness function the scaled mean of the squared distances between the expected values f_i and the values g_i obtained by the individual:

$$SMSE = \frac{1}{m} \sum_{i=1}^{m} [g_i - (a + bf_i)]^2 \quad (1)$$

where

$$a = \bar{g} - b\bar{f}$$

and

$$b = \frac{cov(g, f)}{var(f)}$$

This fitness function is described more in details in [3].

All the parameters are summarized in Table II. Whenever GP-HVES runs an evolution, the maximum number of generations is set to 100.

TABLE II
PARAMETER SETTINGS

Parameter	**Setting**
Population size	2000
Initialization method	Ramped Half-and-Half
Initialization depths	2-5 levels
Max depth for trees	10
Selection	Tournament of size 7
Elitism	5
Node bias for crossover	90% internals, 10% terminals
Duplication rate	5%
Crossover rate	85%
Mutation rate	15%
Recursion depth (HVES)	2

We used our own Genetic Programming API. This API is implemented in Java and is based on a modular architecture where each step of the GP process may be delegated to a plugin. Plugins for all common algorithms for tree generation, fitness evaluation, selection and variation are provided. Moreover this API is based on a strongly typed GP approach similar to [4].

For each time series we perform 100 independent executions. Each execution starts with a different seed for the random number generator but we used the same seeds for each test. We ran all simulations on a PC based on a processor Intel Xeon 3.20 GHz with 2 GB of RAM. Each execution takes approximately 20 minutes to be completed. At the end of the process we keep the candidate solution which minimize the fitness function on the complete data-set associated with the time series.

IV. Conclusions and future work

In this paper we introduced a new approach based on GP for symbolic regression of time series data-sets where the underlying phenomenon is completely unknown.

In our approach we execute a preliminary evolution and use the error exhibited by the best individual in order to infer discontinuity boundaries in the data sample. Then we apply an algorithm designed by us for selecting an Hyper-Volume in the input space whose boundaries approximately follow the discontinuities. This Hyper-Volume partitions the fitness cases in two sets that are used for driving further preliminary evolutions. The best individuals found by these independent evolutions are then merged and evolved again. The process is also applied recursively until either no further discontinuities are found or a predefined recursion depth is reached.

In the future, we plan to investigate other heuristics for discontinuity detection in noisy data-sets. We are currently studying the work done in computer vision, signal processing and statistics fields in [5] [6] [7] [8]. The challenge will be to adapt these techniques for multidimensional spaces since these approaches usually work in one or two dimensions only.

Acknowledgments

This work was supported by the Marie-Curie RTD network AI4IA, EU contract MEST-CT-2004-514510 (December 14th 2004)

References

[1] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.

[2] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic Programming˜: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications*. Heidelberg and San Francisco CA, resp.: dpunkt – Verlag für digitale Technologie GbmH and Morgan Kaufmann Publishers, Inc., 1998.

[3] M. Keijzer, "Scaled symbolic regression," *Genetic Programming and Evolvable Machines*, vol. 5, no. 3, pp. 259–269, 2004.

[4] D. J. Montana, "Strongly typed genetic programming," *Evolutionary Computation*, vol. 3, no. 2, pp. 199–230, 1995.

[5] D. Lee, "Coping with discontinuities in computer vision: their detection, classification, and measurement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 4, pp. 321–344, Apr. 1990.

[6] D. L. and W. G.W., "Discontinuity detection and thresholding-a stochastic approach," in *Proceedings of the 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, Eds. IEEE Press, 3-6 June 1991, pp. 208–214.

[7] A. W. Bowman, A. Pope, and B. Ismail, "Detecting discontinuities in nonparametric regression curves and surfaces," *Statistics and Computing*, vol. 16, no. 4, pp. 377–390, 2006.

[8] P. Qiu, "Discontinuous regression surfaces fitting," *Annals of Statistics*, vol. 26, no. 6, 1998.