Investigating the Use of Geometric Semantic Operators in Vectorial Genetic Programming

Irene Azzali[1(✉)], Leonardo Vanneschi[2,3], and Mario Giacobini[1]

[1] DAMU - Data Analysis and Modeling Unit, Department of Veterinary Sciences, University of Torino, Turin, Italy
{irene.azzali,mario.giacobini}@unito.it
[2] NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, 1070-312 Lisbon, Portugal
lvanneschi@novaims.unl.pt
[3] LASIGE, Departamento de Informática, Faculdade de Ciências, Universidade de Lisboa, 1749-016 Lisbon, Portugal

Abstract. Vectorial Genetic Programming (VE_GP) is a new GP approach for panel data forecasting. Besides permitting the use of vectors as terminal symbols to represent time series and including aggregation functions to extract time series features, it introduces the possibility of evolving the window of aggregation. The local aggregation of data allows the identification of meaningful patterns overcoming the drawback of considering always the previous history of a series of data. In this work, we investigate the use of geometric semantic operators (GSOs) in VE_GP, comparing its performance with traditional GP with GSOs. Experiments are conducted on two real panel data forecasting problems, one allowing the aggregation on moving windows, one not. Results show that classical VE_GP is the best approach in both cases in terms of predictive accuracy, suggesting that GSOs are not able to evolve efficiently individuals when time series are involved. We discuss the possible reasons of this behaviour, to understand how we could design valuable GSOs for time series in the future.

Keywords: Vector-based genetic programming · Time series · Sliding windows · Geometric semantic operators

1 Introduction

A *panel dataset* is a dataset consisting of observations collected during time from multiple subjects [11]. As such, these datasets combine static features with time series data. An important issue involving panel datasets arises when we face the problem of predicting one of the time series variables. Table 1 shows a simple example of a panel dataset for three stores in U.S. regions over the course of several weeks, in which the data include the fuel price in the region (Fuel_pr),

T. Hu et al. (Eds.): EuroGP 2020, LNCS 12101, pp. 52–67, 2020.
https://doi.org/10.1007/978-3-030-44094-7_4

the unemployment rate of the region (Unempl_r), and the distance of the store from the nearest metro station (Dist_M). The goal is to predict the total sales (*Sales*) of each week, based on the explanatory variables introduced before.

Table 1. Example of a standard panel dataset.

Store ID	Fuel_pr	Unempl_r	Dist_M	Week	*Sales*
1	2.7	5.4	1800	12	3000
1	2.6	5.4	1800	13	1750
2	2.3	6.2	1400	11	440
2	2.3	6.2	1400	12	4100
2	2.6	6.5	1400	13	1800
3	2.1	2.2	8000	10	650

Classical machine learning (ML) techniques such as neural networks, random forests and genetic programming (GP) can be applied to forecast panel datasets, but their performance might suffer from considering independently each observation, therefore losing information on their temporal order. This would result in difficulties caused by the lack of meaningful predictive characteristics of time series such as peaks and regularities. In this perspective, besides known advanced ML techniques such as recurrent neural networks, a new approach of GP called vectorial genetic programming (VE_GP) was recently proposed in [4]. VE_GP extends the terminal set to vectors, providing a suitable representation for time series. In this way, the time series variables collected from each subject can be kept intact, making it possible to fully exploit the knowledge about the behaviour of the series. To clarify, Table 2 shows how the panel dataset of Table 1 changes representation in order to feed a VE_GP algorithm.

Table 2. The same data as in Table 1, but with the representation used for VE_GP.

Store ID	Fuel_pr	Unempl_r	Dist_M	Week	*Sales*
1	[2.7, 2.6]	[5.4, 5.4]	1800	[12, 13]	[3000, 1750]
2	[2.3, 2.3, 2.6]	[6.2, 6.2, 6.5]	1400	[11, 12, 13]	[440, 1100, 1800]
3	[2.1]	[2.2]	8000	[10]	[650]

To efficiently use the information contained in time series, VE_GP includes aggregation functions as primitives, as well as new strategies in the different steps of the classical GP search process.

VE_GP has already revealed advantages in benchmark problems [4], but noteworthy are the results on a real prediction of panel data [3]. In this last work, the predictive accuracy and the generalization ability of VE_GP were ascribed

to the key feature of keeping together ordered sequences in vectors. This representation, in fact, lets the evolution discover the most informative aggregation functions to be used in the predictive model, which are responsible of inferring information on the time series behaviour.

Nonetheless, one of the major advantages of VE_GP was claimed to be its ability to evolve the window of time where the new aggregation functions are applied. VE_GP, in fact, adds to all the aggregation functions their parametric version, so that they can be applied only on a portion of the whole vector. The search for the best parameters, the ones that determine the most informative portion of the vector, is part of the evolutionary process, thanks to the introduction of a parameter mutation operator.

In recent years, the use of geometric semantic operators (GSOs) in GP [18] became popular and showed some interesting advantages with respect to GP with classical genetic operators [7,8,10,17]. GSOs, thus, deserve to be explored even in VE_GP approach to see if they still bring advantages in panel data forecasting, although they can not include a semantic parameter mutation. In this article, we investigate the use of GSOs in VE_GP by presenting a comparative study of GP techniques on two panel data forecasting problems. The first one consists in predicting mosquito abundance from climatic and environmental factors, a problem recently approached with standard GP in [12]. This dataset allows the inclusion of parametric aggregation functions as primitives, offering the possibility to explore the role of the windows evolution for the accuracy of predictions. The second dataset deals with the prediction of ventilation flow of running people, based on physiological parameters including the heart rate flow. In this case, the fact that the time series among different subjects have different lengths suggests the use of aggregation functions without parameters. Further explanations on the different use of aggregation functions will be provided in Sect. 5. The methods we compare are VE_GP and classical GP (ST_GP), both using classical and semantic genetic operators.

The paper is organized as follows: Sect. 2 presents an overview on previous attempts to apply GP to panel datasets. Section 3 introduces the problems used in the experimental investigation. Section 4 describes the use of GSOs in VE_GP. Section 5 presents the experimental setting and proposes an analysis of the obtained results. Finally, Sect. 6 draws the conclusions of the investigation.

2 Panel Datasets in GP: Literature Review

When dealing with panel datasets, researchers have employed a common strategy to avoid the use of the standard representation of panel data, the aggregation of vectorial information into a summarizing scalar value. However, such approach usually results in a loss of information. In [13], ECG signals recorded from different patients are substituted by important signal characteristics, such as the mean, the energy etc. This is the typical "collapse approach" where instead of letting the data reveal the most important series characteristics, these are *a priori* fixed before the evolutionary process. Again, in [20], the signals measured to be the predictors or the target are pre-processed before feeding the

GP algorithm. In [14], instead, GP is used to predict glucose values of diabetic people based on insulin values and food intakes without a pre-processing of time series variables. However, the panel data regression problem is transformed into 4 simple regression problems by fixing 4 time values of glucose as the target. As stated by the authors, the main drawback of this approach is the impossibility of predicting a continuous time series of glucose.

Some works have already explored the idea of using vectors to represent time series as terminals. In [15] the authors designed a vector-based GP to discover signal processing algorithm by means of evolution. As well as in VE_GP, they introduced functions to combine scalars and vectors, and advance signal processing functions specific for vectors. In [5] again vectors and vectorial functions are included in the primitive set. Even if VE_GP is strongly influenced by these contributions, the evolution of time windows to capture the most informative signal behaviours is totally new in the field.

3 Problem Description and the Datasets

3.1 Mosquito Abundance (P_Mosq)

The surveillance plan established in Italy in 2008 aimed at quantifying mosquito abundance in order to predict the emergence and the spread of West Nile virus. Predictive models of mosquitoes dynamics were therefore a valuable tool to fulfil the goal. For this reason, modelling techniques with the objective of forecasting mosquito abundance based on environmental factors were explored [6,12]. In this article, we use the dataset produced by the Casale Monferrato Agreement for mosquitoes control from 2002 to 2006 in the context of the Piedmont surveillance program, already used in [6,12]. Mosquitoes were weekly collected from 36 CO_2-baited traps from May to September with a total of 20 collections per year for each trap.

We consider the same scalar predictive variables selected by [6] and reported in Table 3.

Table 3. Scalar mosquitoes predictors.

Variable	Description
ELEV	Elevation of the sampling location
DISTU	Distance of the sampling location from the nearest urban area
DISTR	Distance of the sampling location from the nearest rice field
DISTW	Distance of the sampling location from the nearest woodland
RICEA	Area of the nearest rice field

Regarding the time series predictors considered by [6], we introduce some novelties according to the results of [12]. First, we discard the variable *SIN*.

In fact, according to [12], *SIN*, which is a sinusoidal curve with a phase of 1 year included as a suggester of mosquitoes seasonality, is too frequently used in all the evolved models. This fact prevents the discovery of how environmental variables interact to determine the peaks in abundance of mosquitoes. However, the function *SIN* plays a key role in exploiting the knowledge on the time order of the observations, since it gives a "score" to each collection according to the day in which it took place. As suggested in [12], the use of VE_GP makes it possible to avoid *SIN* without losing the time order information. Thus, our results will also contribute to confirm the advantages of the vector representation proposed by VE_GP. Second, to highlight the benefit of evolving temporal windows, we remove for VE_GP the prior aggregations of the time series predictors considered by [6]. Therefore, VE_GP handles daily values of land surface temperatures, normalized difference vegetation index and rainfalls which are the environmental time series predictor selected by [6]. On the contrary, ST_GP keeps the same variables aggregated as in [6]. Table 4 describes the time series variables for the two approaches of GP.

Table 4. Time series mosquitoes predictors.

Variable	ST_GP description	VE_GP description
TWEEK [2]	The average land surface temperature 8–15 days prior to trapping	Daily value of land surface temperature
NDVI [2]	16-days average of normalized difference vegetation index	Daily value of normalized difference vegetation index
RAIN [1]	Cumulative rainfall 10–17 days prior to trapping registered by the nearest weather station	Daily rainfall registered by the nearest weather station

The variables involved as predictors are therefore the time series *NDVI*, *TWEEK* and *RAIN* (with different definitions depending on the GP approach) plus the scalar variables *ELEV*, *DISTU*, *DISTR*, *DISTW* and *RICEA*. The target is the number of mosquitoes collected *Mosq*. We have two different dataset representation:

- ST_GP with classical and geometric semantic operators: the dataset is a matrix of 3600 rows and 9 columns. Columns one to eight indicate a predictor while the rightmost is the target; each row corresponds to a day of collection.
- VE_GP with classical and geometric semantic operators: the dataset is a matrix of 180 rows and 9 columns. Columns one to eight indicate a predictor while the rightmost is the target; time series variables (*NDVI*, *TWEEK* and *RAIN*) are represented as vectors of length 173 since they contain daily values from April 1st (37 days before the first collection of the year) to September

20^{th} (the last collection day of the year). The target *Mosq* is instead a vector of length 20 representing the 20 collections per year from each trap. Each row corresponds to the collections from a trap during a year.

3.2 Ventilation Flow (P_Physio)

This task was proposed by the Centre of Preventive Medicine and Sport - SUISM - University Structure of Hygiene and Sport Sciences of Turin. The goal is to predict ventilation flow during outdoor activities based on physiological variables in order to monitor the intake of air pollution. We use the dataset employed in [3], which consists of static physiological data and time series variables such as heart rate and ventilation, recorded every 10 s from people running on a treadmill. We must point out that each person ran as long as he/she could, thus the heart rate and ventilation series have different lengths among people. The predictors are therefore the gender (*SEX*), the age (*AGE*), the body mass index (*BMI*) and the heart rate (*HR*). We use the acronym *VE* to indicate the target which is the ventilation. The dataset representation changes again according to the GP approach:

- ST_GP with classical and geometric semantic operators: the dataset is a matrix of 3600 rows and 5 columns. Columns one to four indicate a predictor while the rightmost is the target; each row corresponds to a recording instant of the heart rate from a person.
- VE_GP with classical and geometric semantic operators: the dataset is a matrix of 262 rows and 5 columns. Columns one to four indicate a predictor while the rightmost is the target; *HR* and *VE* are represented as vectors of variable length depending on the running time of the person. Each row corresponds to a person.

4 Methodology

4.1 Vectorial Genetic Programming

Vectorial genetic programming (VE-GP) is a recently developed approach of GP to properly deal with time series as predictors or targets. VE-GP allows vectors as terminals, providing a suitable representation for all time series. Besides the simple adjustments needed to cope with this new terminal structure, VE-GP includes other innovations to fully exploit vector representation. Here we describe the main novelties of this approach that we are going to use to carry out the experiments. Further details can be found in [4].

Primitive Set. In GP the primitive set consists of functions and terminals combined to build the individuals. In VE-GP, vectors join the classical scalar terminals and new functions are included as possible primitives. To avoid inconsistencies, vectors of length 1 and scalars are considered the same terminal form.

Functions of Arity 1. Aggregate functions are included in the primitive set in order to capture the behaviour of a vector. These functions group together multiple values to return a single summary value. Two main versions of aggregation functions are available in VE-GP: standard and cumulative. While standard aggregation functions collapse the whole vector into a single value, cumulative aggregation functions collapse only a portion of the vector. To clarify, let $v = [v_1, \ldots, v_n]$ be a vector terminal and Cf be the cumulative version of the aggregation function f; then $Cf(v) = [w_1, \ldots, w_n]$ where $w_j = f([v_1, \ldots, v_{j-1}, v_j])$ for each $j = 1, \ldots, n$. Standard aggregation functions are meant for problems where the recording time of predictors and target time series is different, cumulative aggregation functions are meant instead for problems where the predictors and the target time series are simultaneous. Both these versions have their parametric form that apply the aggregation function only to a window of the vector. In case of standard aggregation functions the window defined by the parameters can slide all the vector, while for cumulative aggregation functions the window slides only backwards. To explain, let p and q be two integer numbers where $p < q$; then $f_{p,q}(v) = f([v_p, \ldots, v_q])$. Let p and q, instead, be two integer numbers where $p > q$; then $Cf_{p,q}(v) = f([z_1, \ldots, z_n])$ where $z_j = f([v_{j-p}, \ldots, v_{j-p+q}])$ for each $j = 1, \ldots, n$. In both cases, if the window extends to not existing elements of the vector they are simply not included in the calculation. For a detailed explanation of these functions joined with numerical examples see Section 3, Table 3 of [4].

Functions of Arity 2. Regarding functions of arity 2, they are simply extended in order to manage the new vector inputs. In particular, when the inputs of a function are two vectors of length greater than 1, the shortest is completed with the null element of the function up to the length of the longest before applying the function itself. Differently, when a scalar and a vector of length greater than 1 are the inputs, the scalar is initially replicated up to the length of the other vector input. This different input preparation highlights the static nature of scalars: since scalars are constant over time we can not "complete" a scalar with the value representing missing values (the null element); we know that its value for every time instant is always the same, thus we have to "complete" the scalar with its value. To clarify, see Section 3 of [4] where Table 4 reports all the aggregation functions of arity 2 available in VE_GP, with an example of application.

Initialization. VE-GP proposes a new initialization strategy in order not to misuse the aggregation functions added to the primitive set.

- n_1 individuals, during the generation with one of the classical techniques [19], are forced to apply aggregation functions only to vectorial variables;
- n_2 individuals are generated with one of the classical techniques [19] and checked in their output. If individual t returns scalars for each observation, a vectorial terminal X and an arity 2 function F are randomly selected and the

individual t is replaced with the following individual, using post-fix notation, *(F t X)*;
- the remaining n_3 individuals are generated with one of the classical techniques [19].

Parameter Mutation. The genetic operator of parameter mutation (PM) is developed in VE_GP in order to let the evolution find the most informative windows of time. PM simply looks in an individual for parametric functions, randomly selects one of them and randomly changes one of its parameters. Depending on the kind of function, standard or cumulative, the parameter is mutated without violating the rule of superiority, i.e. $p < q$ for standard functions and $p > q$ for cumulative functions.

Fitness Evaluation. Some individuals may output a scalar for each observation, when the target instead is supposed to be a vector. In order to evaluate their fitness, each scalar is preliminary replicated up to the length of the corresponding target vector. Moreover, these individuals are penalized by multiplying their fitness for a huge constant (panel data prediction problems belong to the area of minimization problem). The wrong size of their output suggests, in fact, that they unlikely to be good predictive models.

4.2 Geometric Semantic Operators

Geometric semantic operators (GSOs) are genetic operators recently introduced for GP [18] to replace the traditional syntax-based crossover and mutation. The term *semantic* in GP community indicates the vector of outputs an individual produce on the training instances. Thus, any GP individual can be identified as a point (its semantic) in a multidimensional space (dimension equal to the number of observations) called *semantic space*. While traditional crossover and mutation manipulate individuals only considering their syntax, GSOs define transformation on the syntax of individuals that correspond to the genetic algorithms operators of geometric crossover and ball mutation in the semantic space. Geometric crossover generates an offspring that stand on the segment joining the parents; ball mutation is a weak perturbation of the coordinates of an individual. We report the definition of the GSOs as given in [18] for individuals with real domain and considering Euclidean distance as the fitness function, since these are the operators we are going to use in the experimental phase.

Geometric semantic crossover (GSXO) returns, as the offspring of the parents $T_1, T_2 : \mathbb{R}^n \to \mathbb{R}$, the individual:

$$T_{X0} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2)$$

where T_R is a random number in $[0, 1]$. *Geometric semantic mutation* (GSM) transforms the individual $T : \mathbb{R}^n \to \mathbb{R}$ according to the expression:

$$T_M = T + m_s \cdot (T_{R1} - T_{R2})$$

where T_{R1} and T_{R2} are random real individuals with codomain in $[0, 1]$ and m_s is the mutation step. We refer to [18] for a proof of the fact that GSXO corresponds to geometric crossover in the semantic space, while GSM corresponds to ball mutation in the semantic space. The main advantage of these semantic operators is that they induce a unimodal fitness landscape, thus an error surface characterized by the absence of locally suboptimal solution, on every supervised learning problems. This property should enhance GP evolvability on all these problems. The main drawback that afflicts GSOs is that the size of the offsprings is larger than the one of their parent(s). To overcome this problem we use the implementation of GSOs proposed by [22] and the strategy of elitist replacement suggested in [9].

Unfortunately it is not possible to define the semantic equivalent of parameter mutation as described in Sect. 4.1. To clarify, let us assume that this operator exists, we call it geometric semantic parameter mutation (GSPM). GSPM has to change one of the parameter of a parametric aggregation function determining, as a result, a weak perturbation of the semantic of T, the individual containing the parametric aggregation function. However, modifying the window in which the aggregation function is applied means considering different values of the time series observed, thus the perturbation of the semantic of T depends on the semantic of T itself. Surely this fact is in contrast with the hypothesis of weak perturbation.

5 Experiments

5.1 Experimental Settings

We have adopted the Matlab implementations of ST_GP and VE_GP based on GPLab toolbox [21]. We have extended both implementations in order to include GSOs. The methods involved in the experiments are therefore ST_GP, VE_GP, ST_GP with GSOs (GSGP) and VE_GP with GSOs (GSVEGP). With each technique we have performed a total of 50 runs on both P_Mosq and P_Physio. Here we lay out the design of the experiments conducted on both problems. For the remainder of this paper, the training set is the portion of the dataset used to feed the algorithm in order to make it learn, while the test set is the remaining portion of the dataset which consists of unseen data used to validate the trained model performance.

P_Mosq. In each experimental run we have considered the same partition of training and test sets that follows the natural order of years: collections from 2002 to 2005 were used as the training set, while collections of 2006 formed the test set.

Fitness was calculated as the Root Mean Square Error (RMSE) between the output and the target. In case of vector based GP (VE_GP and GSVEGP) the output are the predictions of mosquito abundance over 173 days (April 1^{st}–September 20^{th}), thus for the evaluation of fitness we have considered as the

actual output the predictions corresponding to the collection days. Since the output of trees built by VE_GP and GSVEGP is supposed to be a vector, for these latter algorithms we have calculated the RMSE vertically disbanding both output and target; in this way the measures of fitness were ensured to be comparable among all the techniques.

All the runs used population of 100 individuals and the evolution stopped after 50 generations. ST_GP and GSGP initialized populations using the Ramped Half-and-Half (RHH) method [16] with a maximum initial depth equal to 6, while VE_GP and GSVEGP initialized populations using the process proposed in [4] based on RHH with maximum initial depth again equal to 6. The functions set for ST_GP and GSGP contained the four binary arithmetic operators $+, -, \times$ and / protected as in [16]. The days of mosquitoes collection are the same across years and traps, thus it is reasonable to look for common informative windows of time among all the observations. For this reason, the functions set for VE_GP contained the binary operators `VSUMW`, `V_W`, `VprW`, `VdivW` plus the parametric cumulative aggregation functions $\texttt{C\_max}_{p,q}$, $\texttt{C\_min}_{p,q}$, $\texttt{C\_mean}_{p,q}$, $\texttt{C\_sum}_{p,q}$. GSVEGP can not handle parametric functions, thus its functions set consisted of the binary operators `VSUMW`, `V_W`, `VprW`, `VdivW` plus the cumulative aggregation functions `C_max`, `C_min`, `C_mean`, `C_sum`. All the functions of the vectorial approaches are defined in [4]. The terminal sets contained the 8 variables as described in Sect. 3.1 plus random constants r between 0 and 1 generated in run time when building individuals. To select parents we used a tournament selection involving 4 individuals. To create new individuals, ST_GP used standard crossover and subtree mutation [16] with probabilities equal to 0.9 and 0.1 respectively. Besides crossover and mutation, VE_GP used parameter mutation with probabilities respectively 0.5, 0.1 and 0.4. The semantic algorithms of GSGP and GSVEGP, instead, used GSXO and GSM with probabilities respectively 0.1, 0.9 and 0.7, 0.3; the mutation steps were respectively 1 and 0.01. The different probabilities and mutation rates depends on a preliminary experimental study performed to find the best parameter setting. Survival of individuals was elitist for ST_GP and VE_GP, while we used the elitist replacement [9] for GSGP and GSVEGP. Maximum tree depth was fixed at 17 for ST_GP and VE_GP while no depth limit have been imposed in GSGP and GSVEGP.

P_Physio. Differently from the previous problem, a distinct partition of the training and test sets has been considered in each run. In particular, 70% of the data instances were randomly selected at the beginning of each run as training set, while the remaining 30% were used as the test set.

Fitness was calculated as the RMSE between the output and the target. In case of vector based GPs we followed the procedure described above to guarantee comparable measures.

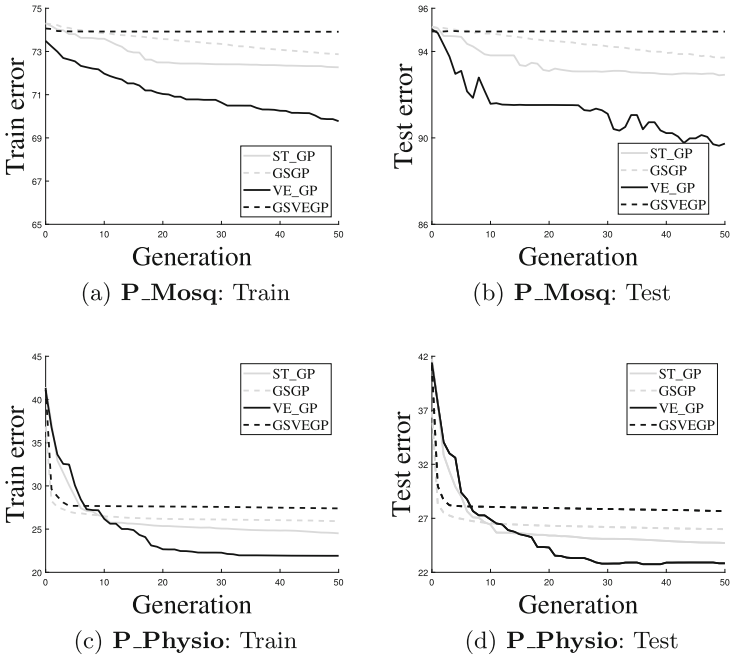
All the runs used population of 100 individuals and the evolution stopped after 50 generations. ST_GP and GSGP initialized populations using the RHH with a maximum initial depth equal to 6, while VE_GP and GSVEGP initialize populations using the process proposed in [4] based on RHH with maximum

initial depth again equal to 6. The functions set for ST_GP and GSGP contained the four binary arithmetic operators $+, -, \times$ and / protected as in [16]. The subjects of the trial ran on the trade mill as long as they could, thus the HR and VE series have different lengths among the people. Looking for a common informative window of time across all the people may weaken the learning phase. In fact, some time windows may be more adequate for long time series compared to shorter ones, causing a loss of generalization ability. For this reason, the functions set for both VE_GP and GSVEGP contained the binary operators `VSUMW`, `V_W`, `VprW`, `VdivW` plus the cumulative aggregation functions $\texttt{C\_min}_{p,q}$ and $\texttt{C\_mean}_{p,q}$ as in [3]. All the terminal sets contained the 4 variables as described in Sect. 3.2 plus random constants r between 0 and 1 generated in runtime when building individuals. To select parents we used a tournament selection involving 4 individuals. To create new individuals, ST_GP used standard crossover and subtree mutation [16] with probabilities equal to 0.9 and 0.1 respectively. Besides crossover and mutation, VE_GP used parameter mutation with probabilities respectively 0.5, 0.1 and 0.4. The semantic algorithms of GSGP and GSVEGP, instead, used GSXO and GSM with probabilities respectively 0.3, 0.7 and 0.5, 0.5; the mutation steps were respectively 1 and 0.1. Also in this case, the different probabilities and mutation rates depends on a preliminary experimental study performed to find the best parameter setting. Survival of individuals was elitist for ST_GP and VE_GP, while we used the elitist replacement [9] for GSGP and GSVEGP. Maximum tree depth was fixed at 17 for ST_GP and VE_GP while no depth limit have been imposed in GSGP and GSVEGP.

5.2 Experimental Results

In this section, we report the results that we have obtained in terms of training and test RMSE. In particular, at each generation we stored the value of RMSE on the training and on the test set of the best individual in the population, i.e. the one with the smallest RMSE on the training data. The curves report the median over the 50 runs of all these values collected at each generation. The median was preferred over the mean due to its robustness to outliers which are common in stochastic methods. Figure 1 reports the training and test errors for P_Mosq and P_Physio.

These plots clearly show that VE_GP in both problems is the fastest in learning, with perspective of further improvement going on with generations, at least for the P_Mosq problem. Moreover, the fast decreasing of the test error confirms that VE_GP is learning with generalization ability. On the contrary, both GSGP and GSVEGP exhibit a slow and almost static (GSVEGP in particular) learning phase. We claim that the main reason behind this fact is the huge size of the semantic space. Considering in fact GSVEGP, in P_Physio problem the semantic space has dimension $(\text{length}(p_1) \times \cdots \times \text{length}(p_{183}))$ where 183 is the number of people in the training set (70% of data instances) and $\text{length}(p_i)$ is the length of the time series recorded for person p_i; in P_Mosq the size is still huge, being $(20)^{144}$ where 20 is the number of mosquitoes collections over a year and 144 is the number of collections in the training set (36 traps $\times$ 4 year).



(a) **P_Mosq**: Train (b) **P_Mosq**: Test

(c) **P_Physio**: Train (d) **P_Physio**: Test

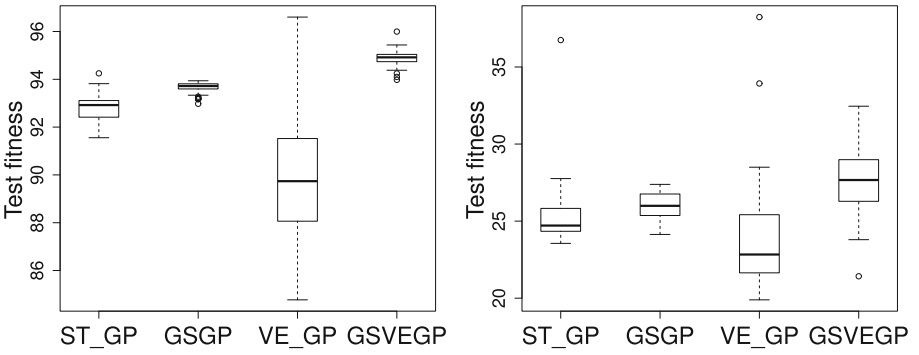
Fig. 1. ST_GP, GSGP, VE_GP and GSVEGP fitness evolution plots.

Since the goal of the paper is to understand how parametric functions influence the performance, we compared the RMSE on the test set of the models found out in the 50 runs by all the techniques. We consider as a model the best individual on the training set at the end of the evolution. Statistical significance of the null hypothesis of no difference among the methods was determined with pairwise Kruskal-Wallis non-parametric ANOVAs at $p = 0.05$. In both problems the resulting $p - value$ stated that there was a significant difference in performance among techniques, thus we performed multiple two-sample Wilcoxon signed rank tests to understand which method differs from the other. The significance level for each test depends on the Bonferroni correction. We report the values of the statistical tests in Table 5, as well as the boxplots of models test

Table 5. Results of comparison between techniques on P_Mosq and P_Physio. Significance level of Wilcoxon test after Bonferroni correction $p = 0.05/3 = 0.02$.

P_Mosq: Kruskal-Wallis ANOVA $p < 10^{-16}$		
VE_GP vs GSVEGP	VE_GP vs GSGP	VE_GP vs GP
$p < 10^{-16}$	$p = 5.7 \cdot 10^{-16}$	$p = 2.2 \cdot 10^{-14}$
P_Phisio: Kruskal-Wallis ANOVA $p < 10^{-16}$		
VE_GP vs GSVEGP	VE_GP vs GSGP	VE_GP vs GP
$p = 2.6 \cdot 10^{-10}$	$p = 2.1 \cdot 10^{-7}$	$p = 1.1 \cdot 10^{-5}$

fitness in Fig. 2. According to the statistical tests, VE_GP performance differs from all the other methods for both problems. Moreover, boxplots in Fig. 2 show that VE_GP is outperforming all the other techniques. This outcome confirms that VE_GP is the better GP approach when dealing with panel data, rather than classical GP approach.



(a) P_Mosq (b) P_Physio

Fig. 2. Test fitness boxplots of models found out by each technique. Figure (a) refers to P_Mosq, while figure (b) refers to P_Physio.

Regarding P_Mosq, the results confirm our intuition on the benefit of evolving time windows to discover the most informative ones without prior fixing them just by means of experts knowledge. Surely GSVEGP's slow learning is due to the semantic space dimension, but we claim that considering always all the data points of previous collections (classical cumulative functions) rather than an evolving windows over previous times may cause a loss in population diversity and thus be another reason of slow learning. In fact, in VE_GP we find individuals containing different aggregations that span different time series portion, while in GSVEGP we find surely individuals containing different aggregations, but they all span the same time series portion. To confirm this observation we report the median (over the 50 runs) diversity curves along generations for GSVEGP and VE_GP. We use as a subjective measure of diversity the standard deviation of the fitness values in the population at each generation. Figure 3 clearly shows that GSVEGP is unable to keep good diversity levels which is a key feature of a successful search process. We tried to give an explanation of other reasonable reasons responsible of this GSVEGP diversity drop. GSOs seems to quickly direct the diversified initial population towards the target; however, after the individuals have converged, the improvements are thinner and thinner and the weak perturbation of one of the components of one of the output time series results in a weak perturbation of the individual fitness. At a certain point, thus, GSOs seems to be less efficient due to the high dimension of the semantic space. In addition, the elitist replacement used to control individual growth [9], at that

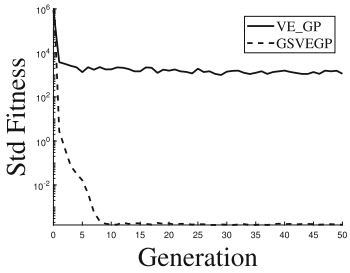


Fig. 3. Diversity evolution for VE_GP and GSVEGP on P_Mosq. Curves are plotted in logarithmic scale.

certain point, causes more frequently the replication of individuals instead of the offspring replacements. In fact, if the weak perturbations are not efficient (produce offsprings with bigger fitness) parents are preferred rather then their offsprings. All these behaviours are feasible reasons of GSVEGP loss of diversity.

Concerning P_Physio results, we expected GSVEGP to be the outperforming method, since parametric functions are not involved in any primitive set. However, statistical tests and the boxplots reveal that VE_GP is the method with the best performance. These results confirm that GSOs are not suitable to deal with time series variables, probably because of the high dimension of the semantic space induced.

6 Conclusions

This paper contains an investigation on the usefulness of evolving parametric aggregation functions for panel data forecasting. Aggregations of values may return informative features of predictors time series for the target, however aggregations on all historical times of predictors may not be needed to forecast the target series. The behaviour of the predictors over a window of time may in fact be more meaningful for the target. To clarify, let us consider the P_Mosq problem of predicting the abundance of mosquitoes during a year: mosquitoes collected at day t are more likely to be affected by the rainfalls over the week before t rather than on all the rainfalls of the days before t; mosquitoes growth in fact, lasts more or less one week, thus rainfalls over a week may cause the loss of eggs and thus adult mosquitoes at day t.

The recently developed vectorial genetic programming (VE_GP) includes in the functions set aggregation function depending on parameters to define time window in which to apply the function. The genetic operator of parameter mutation, moreover, gives the possibility to parameters to evolve in order to catch the most informative windows. The objective of the paper was therefore to highlight the benefits of tackling panel dataset forecasting using VE_GP. In particular, we compared VE_GP performance against VE_GP with geometric semantic operators (GSVEGP) on two problems. While the first one, P_Mosq,

demanded for parametric aggregation functions in the functions set, the second problem, P_Physio, did not require evolving time windows. We chose GSVEGP as a benchmark because although geometric semantic operators should improve the performance, the geometric semantic awareness does not allow parameter mutation as a genetic operator.

The main contribution of this work consisted in showing that parametric aggregation functions can further improve the performance when the dataset hypothesis allow for their inclusion (P_Mosq). Moreover we found out that considering all the history of time series may influence the maintenance of diversity in the evolving population. Surprisingly, however, results achieved on P_Physio revealed a weakness of GP algorithms with geometric semantic operators. We impute this result to the high dimension of the semantic space caused by time series variables that slow the learning process.

The outcomes paved the way for future works on the design of more efficient geometric semantic operators for problems involving time series. A wider result is, however, the highlight of VE_GP as a successful approach in panel data forecasting, making the GP community aware of its value.

Acknowledgments. This work was partially supported by FCT, Portugal through funding of LASIGE Research Unit (UID/CEC/00408/2019), and projects PREDICT (PTDC/CCI-CIF/29877/2017), BINDER (PTDC/CCI-INF/29168/2017), GADgET (DSAIPA/DS/0022/2018) and AICE (DSAIPA/DS/0113/2019).

References

1. Arpa Piemonte. http://www.arpa.piemonte.it
2. NASA MODIS Web. https://modis.gsfc.nasa.gov/
3. Azzali, I., Vanneschi, L., Bakurov, I., Silva, S., Ivaldi, M., Giacobini, M.: Towards the use of vector based GP to predict physiological time series. App. Soft Comput. (forthcoming)
4. Azzali, I., Vanneschi, L., Silva, S., Bakurov, I., Giacobini, M.: A vectorial approach to genetic programming. In: Sekanina, L., Hu, T., Lourenço, N., Richter, H., García-Sánchez, P. (eds.) EuroGP 2019. LNCS, vol. 11451, pp. 213–227. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-16670-0_14
5. Bartashevich, P., Bakurov, I., Mostaghim, S., Vanneschi, L.: Evolving PSO algorithm design in vector fields using geometric semantic GP. In: GECCO 2018: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 262–263 (2018). https://doi.org/10.1145/3205651.3205760
6. Bisanzio, D., et al.: Spatio-temporal patterns of distribution of West Nile virus vectors in eastern Piedmont Region, Italy. Parasites Vectors **4** (2011). https://doi.org/10.1186/1756-3305-4-230
7. Castelli, M., et al.: An efficient implementation of geometric semantic genetic programming for anticoagulation level prediction in pharmacogenetics. In: Correia, L., Reis, L.P., Cascalho, J. (eds.) EPIA 2013. LNCS (LNAI), vol. 8154, pp. 78–89. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40669-0_8
8. Castelli, M., Vanneschi, L., Felice, M.D.: Forecasting short-term electricty consumption using a semnatics-based genetic programming framework: the South Italy case. Energy Econ. **47**, 37–41 (2015). https://doi.org/10.1016/j.eneco.2014.10.009

9. Castelli, M., Vanneschi, L., Popovic, A.: Controlling individuals growth in semantic genetic programming through elitist replacement. Comput. Intell. Neurosci. (2016). https://doi.org/10.1155/2016/8326760
10. Castelli, M., Vanneschi, L., Silva, S.: Prediction of the unified Parkinson's disease rating scale assessment using a genetic programming system with geometric semantic genetic operators. Expert Syst. Appl. **41**(10), 4608–4616 (2014). https://doi.org/10.1016/j.eswa.2014.01.018
11. Dermofal, D.: Time-series cross-sectional and panel data models. Spat. Anal. Soc. Sci. **32**, 141–157 (2015). https://doi.org/10.1017/CBO9781139051293.009
12. Gervasi, R., Azzali, I., Bisanzio, D., Mosca, A., Bertolotti, L., Giacobini, M.: A genetic programming approach to predict mosquitoes abundance. In: Sekanina, L., Hu, T., Lourenço, N., Richter, H., García-Sánchez, P. (eds.) EuroGP 2019. LNCS, vol. 11451, pp. 35–48. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-16670-0_3
13. Guo, H., Jack, L.B., Nandi, A.K.: Automated feature extraction using genetic programming for bearing condition monitoring. In: Proceedings of the 14th IEEE Signal Processing Society Workshop Machine Learning for Signal Processing, pp. 519–528 (2004). https://doi.org/10.1109/MLSP.2004.1423015
14. Hidalgo, J.I., Colmenar, J.M., Kronberger, G., Winkler, S.M., Garnica, O., Lanchares, J.: Data based prediction of blood glucose concentrations using evolutionary methods. J. Med. Syst. **41**(9), 1–20 (2017). https://doi.org/10.1007/s10916-017-0788-2
15. Holladay, K., Robbins, K.A.: Evolution of signal processing algorithm using vector based genetic programming. In: 15th International Conference on Digital Signal Processing, pp. 503–506 (2007). https://doi.org/10.1109/ICDSP.2007.4288629
16. Koza, J.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
17. Vanneschi, L., Silva, S., Castelli, M., Manzoni, L.: Geometric semantic genetic programming for real life applications. In: Riolo, R., Moore, J.H., Kotanchek, M. (eds.) Genetic Programming Theory and Practice XI. GEC, pp. 191–209. Springer, New York (2014). https://doi.org/10.1007/978-1-4939-0375-7_11
18. Moraglio, A., Krawiec, K., Johnson, C.G.: Geometric semantic genetic programming. In: Coello, C.A.C., Cutello, V., Deb, K., Forrest, S., Nicosia, G., Pavone, M. (eds.) PPSN 2012. LNCS, vol. 7491, pp. 21–31. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32937-1_3
19. Poli, R., Langdon, W.B., McPhee, N.F.: A Field Guide to Genetic Programming. Lulu Enterprises, UK Ltd. (2008)
20. Sannino, G., Falco, I.D., Pietro, G.D.: Non-invasive estimation of blood pressure through genetic programming - preliminary results. In: SmartMedDev 2015, pp. 241–249 (2015). https://doi.org/10.5220/0005318002410249
21. Silva, S., Almeida, J.: GPLAB a genetic programming toolbox for MATLAB (2007). http://gplab.sourceforge.net/index.html
22. Vanneschi, L., Castelli, M., Manzoni, L., Silva, S.: A new implementation of geometric semantic GP and its application to problems in pharmacokinetics. In: Krawiec, K., Moraglio, A., Hu, T., Etaner-Uyar, A.Ş., Hu, B. (eds.) EuroGP 2013. LNCS, vol. 7831, pp. 205–216. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37207-0_18