EDDA-V2 – An Improvement of the Evolutionary Demes Despeciation Algorithm

Illya Bakurov[1], Leonardo Vanneschi[1], Mauro Castelli[1(✉)], and Francesco Fontanella[2]

[1] NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, 1070-312 Lisbon, Portugal
{ibakurov,lvanneschi,mcastelli}@novaims.unl.pt
[2] Dipartimento di Ingegneria Elettrica e dell'Informazione (DIEI), Università di Cassino e del Lazio Meridionale, Cassino, FR, Italy
fontanella@unicas.it

Abstract. For any population-based algorithm, the initialization of the population is a very important step. In Genetic Programming (GP), in particular, initialization is known to play a crucial role - traditionally, a wide variety of trees of various sizes and shapes are desirable. In this paper, we propose an advancement of a previously conceived Evolutionary Demes Despeciation Algorithm (EDDA), inspired by the biological phenomenon of demes despeciation. In the pioneer design of EDDA, the initial population is generated using the best individuals obtained from a set of independent subpopulations (demes), which are evolved for a few generations, by means of conceptually different evolutionary algorithms - some use standard syntax-based GP and others use a semantics-based GP system. The new technique we propose here (EDDA-V2), imposes more diverse evolutionary conditions - each deme evolves using a distinct random sample of training data instances and input features. Experimental results show that EDDA-V2 is a feasible initialization technique: populations converge towards solutions with comparable or even better generalization ability with respect to the ones initialized with EDDA, by using significantly reduced computational time.

Keywords: Initialization algorithm · Semantics · Despeciation

1 Introduction

Initialization of the population is the first step of Genetic Programming (GP). John Koza proposed three generative methods of the initial population - Grow, Full and Ramped Half-and-Half (RHH) [7]. All of them consist in constructing trees in an almost random fashion and vary only in the doctrine which guides the process. Since the RHH method is a mixture of both the Full and Grow methods, it allows the production of trees of various sizes and shapes and it was

A. Auger et al. (Eds.): PPSN 2018, LNCS 11101, pp. 185–196, 2018.
https://doi.org/10.1007/978-3-319-99253-2_15

frequently used in many applications. The emergence of new geometric semantic operators [8], which introduce semantic awareness into Genetic Programming (GP), strengthened the importance of semantics-awareness in GP. Semantics was considered as a fundamental factor for the success of the evolutionary search process, leading to the definition of initialization algorithms [2,3] that aimed at increasing semantic diversity in the initial GP population. These studies clearly showed the importance of semantics in this part of the evolutionary process. Other contributions had already recognized that an initial population characterized by a high diversity increases the effectiveness of GP, bestowing a wider exploration ability on the process [7,12]. With the aim of directly searching in the semantic space, Moraglio and colleagues introduced Geometric Semantic Genetic Programming (GSGP) [8], which rapidly raised an impressive interest in the GP community - in part, because of its interesting property of inducing a fitness landscape characterized by the absence of locally suboptimal solutions for any problem consisting in matching sets of input data into known targets [13]. In GSGP, standard crossover and mutation variation operators are replaced with so-called geometric semantic operators, that have precise effects on the semantics of the trees, from now on, called individuals. After the emergence of Geometric Semantic Operators (GSOs), a conceptually distinct sub-field aiming at investigating the properties of GSGP was born inside the GP community. As a result, new techniques were proposed to favor the search process of GSGP, making it more efficient. In the context of the initialization process, Pawlak and Krawiec introduced semantic geometric initialization [10] and Oliveira and colleagues introduced the concept of dispersion of solutions, for increasing the effectiveness of geometric semantic crossover [9]. Following this research track, in 2017, we proposed a new initialization method which mimics the evolution of demes, followed by despeciation, called Evolutionary Demes Despeciation Algorithm [14]. In summary, our idea consisted in seeding the initial population of N individuals with good quality individuals that have been evolved, for few a generations, in N independent subpopulations (demes), by means of conceptually different evolutionary algorithms. In this system, $n\%$ of demes use standard GP and the remaining $(100-n)\%$ use GSGP. After evolving one deme, the best individual is extracted to seed the initial population in the Main Evolutionary Process (MEP). The work presented in this paper improves the previously proposed initialization method, EDDA, by including even more adverse evolutionary conditions in each deme. In summary, in our advancement of the EDDA method, which we will from now on call, EDDA-V2, every deme evolves using a distinct random sample of training data instances and input features.

This document is organized as follows: In Sect. 2 we recall basic concepts related to GSGP. Section 3 describes the previous and new EDDA variants, showing their differences. Section 4 presents the experimental study. Section 5 discusses the experimental results. Finally, Sect. 6 concludes the work summarizing its contribution.

2 Geometric Semantic Genetic Programming

The term semantics, in the GP community, refers to the vector of output values produced by evaluating an individual on a set of training instances [15]. Under this definition, a GP individual can be seen as a point in a multi-dimensional semantic space, where the number of dimensions is equal to the number of fitness cases. In standard GP, variation operators produce an offspring by making syntactic manipulation of the parent trees, in the hope that such manipulation will result in a semantics which is closer to the target one. The term Geometric Semantic Genetic Programming (GSGP) designates a GP variant in which syntactic-based GP operators - crossover and mutation - are replaced with so-called Geometric Semantic Operators (GSOs). GSOs introduce semantic awareness in the search process and induce a unimodal fitness landscape in any supervised problem where the fitness function can be defined as a distance between a solution and the target. GSOs, introduced in [8], raised an impressive interest in the GP community [16] because of their attractive property of directly searching the space of underlying semantics of the programs. In this paper, we report the definition of the GSOs for real functions domains, because we used them in our experimental phase. For applications that consider other types of data, the reader is referred to [8].

Geometric semantic crossover (GSC) generates, as the unique offspring of parents $T_1, T_2 : \mathbb{R}^n \rightarrow \mathbb{R}$, the expression $T_{XO} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2)$, where T_R is a random real function whose output values range in the interval $[0, 1]$. Analogously, *geometric semantic mutation (GSM)* returns, as the result of the mutation of an individual $T : \mathbb{R}^n \rightarrow \mathbb{R}$, the expression $T_M = T + ms \cdot (T_{R1} - T_{R2})$, where T_{R1} and T_{R2} are random real functions with codomain in $[0, 1]$ and ms is a parameter called mutation step. In their work, Moraglio and colleagues show that GSOs create an offspring of significantly larger size with respect to standard GP operators. This makes the fitness evaluation unacceptably slow and considerably constrains practical usability of the GSGP system. To overcome this limitation a possible workaround was proposed in [5], with an efficient implementation of GSOs that makes them usable in practice. This is the implementation used in this work.

3 Evolutionary Demes Despeciation Algorithm

In this paper, we propose an advancement of a previously conceived initialization technique, Evolutionary Demes Despeciation Algorithm (EDDA) [14], inspired by the biological concepts of demes evolution and despeciation. In biology, demes are local populations, or subpopulations, of polytypic species that actively interbreed with one another and share distinct gene pools [17]. The term despeciation indicates the combination of demes of previously distinct species into a new population [11]. Albeit not so common in nature, despeciation is a well-known biological phenomenon, and in some cases, it leads to a fortification of the populations. The main idea of EDDA consists in seeding the initial population of N individuals with good quality individuals that have been evolved, for

a few generations, in N independent subpopulations (demes), on which distinct evolutionary conditions were imposed. Concretely, demes are evolved by means of conceptually different evolutionary algorithms - $n\%$ of demes are evolved by means of GSGP, while the remaining $(100 - n)\%$ use standard GP - and distinct - mostly randomly generated - parameter sets. The experimental results presented in [14] have shown the effectiveness of EDDA. In particular, in all the benchmark problems taken into account, the search process, whose population was initialized with EDDA, ended with solutions with higher, or at least comparable, generalization ability, but with significantly smaller size than the ones found by GSGP using the traditional RHH generative method to initialize the population. In the remaining part of the paper, we will refer to EDDA with the term EDDA-V1.

The advancement we propose in this work, denominated as EDDA-V2, imposes even more adverse evolutionary conditions. Concretely, each deme is evolved using not only a different evolutionary algorithm and parameter set but also a distinct random sample of training data instances and input features. In the following pseudo-code, the distinctive algorithmic features of EDDA-V2 in comparison to EDDA-V1 are presented in bold.

EDDA-$n\%$ (evolving demes for m generations) ::

1. Create an empty population P of size N;
2. Repeat $N * (n/100)$ times:
   (a) Create an empty deme D;
   (b) **Create a sample s of the training dataset by randomly selecting i data instances and f input features**;
   (c) Randomly initialize this D using traditional initialization algorithm (RHH is used here);
   (d) **Using** s, evolve D, for m generations, by means of GSGP;
   (e) After finishing 2.c), select the best individual from D and store it in P;
3. Repeat $N * (1 - n/100)$ times:
   (a) Create an empty deme D;
   (b) **Create a sample s of the training dataset by randomly selecting i data instances and f input features**;
   (c) Randomly initialize this D using traditional initialization algorithm (RHH is used here);
   (d) **Using** s, evolve D, for m generations, by means of GP;
   (e) After finishing 3.c), select the best individual from D and store it in P;
4. Retrieve P and use it as the initial population of GSGP **with full training data set**.

Fig. 1. Pseudo-code of the EDDA-$n\%$ system, in which demes are left to evolve for m generations.

As one can see from the pseudo-code reported in Fig. 1, EDDA-V2 uses a different subset of the training instances in each deme, as well as different input features. We claim is that the presence of demes with different fitness cases

and input features should increase the diversity of the initial population, with individuals that are focused on different areas of the semantic space. As a final result of the search process, we would expect a model with an increased generalization ability with respect to GSGP initialized with EDDA-V1 and RHH. As a side effect, using only a percentage of the training instances and input features can be beneficial for reducing the computational effort when a vast amount of training data is available.

4 Experimental Study

4.1 Test Problems

To assess the suitability of EDDA-V2 as a technique for initializing a population, three real-life symbolic regression problems were considered. Two of them - Plasma Protein Binding level (PPB), and Toxicity (LD50) - are problems from the drug discovery area and their objective is to predict the value of a pharmacokinetic parameter, as a function of a set of molecular descriptors of potential new drugs. The third benchmark is the Energy problem, where the objective is to predict the energy consumption in particular geographic areas and on particular days, as a function of some observable features of those days, including meteorological data. Table 1 reports, for each one of these problems, the number of input features (variables) and data instances (observations) in the respective datasets. The table also reports a bibliographic reference for every benchmark, where a more detailed description of these datasets is available.

Table 1. Description of the benchmark problems. For each dataset, the number of features (independent variables) and the number of instances (observations) were reported.

Dataset	# Features	# Instances
Protein plasma binding level (PPB) [1]	626	131
Toxicity (LD50) [1]	626	234
Energy [6]	8	768

4.2 Experimental Settings

During the experimental study, we compared the performance of EDDA-V2 against EDDA-V1. The performance are evaluated by considering the quality of the solution obtained at the end of the evolutionary process considering populations initialized with EDDA-V1 and EDDA-V2. Additionally, to consolidate results of our previous work, we also included the GSGP evolutionary algorithm that uses the traditional RHH initialization algorithm. Table 2 reports, in the first column, the main parametrization used for every initialization algorithm (columns two, three and four). The first line in the table contains the number of generations after initialization.

Table 2. Parametrization used in every initialization algorithm.

	Parameters	RHH	EDDA.V1	EDDA.V2
1	# generations	2750	{2250, 1250}	{2250, 1250}
2	# generations/deme	-	{5, 15}	{5, 15}
3	# demes	-	100	100
4	% GSGP demes	-	{0, 25, 50, 75, 100}	{0, 25, 50, 75, 100}
5	% sampled instances	-	-	{25, 50, 75}
6	% sampled features	-	-	{25, 50, 75}

For each experiment, any considered initialization algorithm was used to create 100 initial individuals, later evolved by means of GSGP evolutionary process for a given number of generations. In order to ensure comparability of results, all the studied systems performed the same number of fitness evaluations - including, in particular, demes evolution in both EDDA variants. In our experiments, there are 275000 fitness evaluations per run, independently on initialization technique. Every deme, regardless of EDDA variant and parametrization, was initialized by means of the traditional RHH algorithm with 100 individuals, later evolved for some generations. Whenever the traditional RHH was used, tree initialization was performed with a maximum initial depth equal to 6 and no upper limit to the size of the individuals was imposed during the evolution. Depending on the number of iterations used for demes evolution in EDDA variants, the number of generations after despeciation may vary. For example, if, in a given experiment, demes are evolved for 5 generations, then the number of generations after despeciation will be 2250. Similarly, if demes are evolved for 15 generations, then the number of generations after despeciation will be 1250. For both previously mentioned cases, the number of fitness evaluations per run is 275000.

The function set we considered in our experiments was $\{+, -, *, /\}$, where / was protected as in [7]. Fitness was calculated as the Root Mean Squared Error (RMSE) between predicted and expected outputs. The terminal set contained the number of variables corresponding to the number of features in each dataset. Tournament selection of size 5 was used. Survival was elitist as it always copied the best individual into the next generation. As done in [4], the probability of applying GSC and GSM is dynamically adapted during the evolutionary process where the crossover rate is p and the mutation rate is $1 - p$. Following [16], the mutation step ms of GSM was randomly generated, with uniform probability in $[0, 1]$, at each mutation event.

For all the considered test problems, 30 independent runs of each studied system were executed. In each one of these runs, the data was split into a training and a test set, where the former contains 70% of the data samples selected randomly with uniform distribution, while the latter contains the remaining 30% of the observations. For each generation of every studied system, the best individual on the training set has been considered, and its fitness (RMSE) on

the training and test sets was stored. For simplicity, from now on, we will refer to the former as training error and to the latter as test error or unseen error.

5 Results

This section presents the results obtained in the experimental phase. In particular, the section aims at highlighting the differences in terms of performance between the two EDDA variants taken into account and GSGP. For each benchmark, we considered four different parameterizations of EDDA-V1 and EDDA-V2. We denote each parametrization by using the quadruple $\%GSGP\_MATURITY\_\%INSTANCES\_\%FEATURES$, where the first term corresponds to the percentage of individuals in each deme evolved by means of GSGP, the maturity (i.e., the number of generations the individuals are evolved), the percentage of instances in the dataset and, finally, the percentage of input

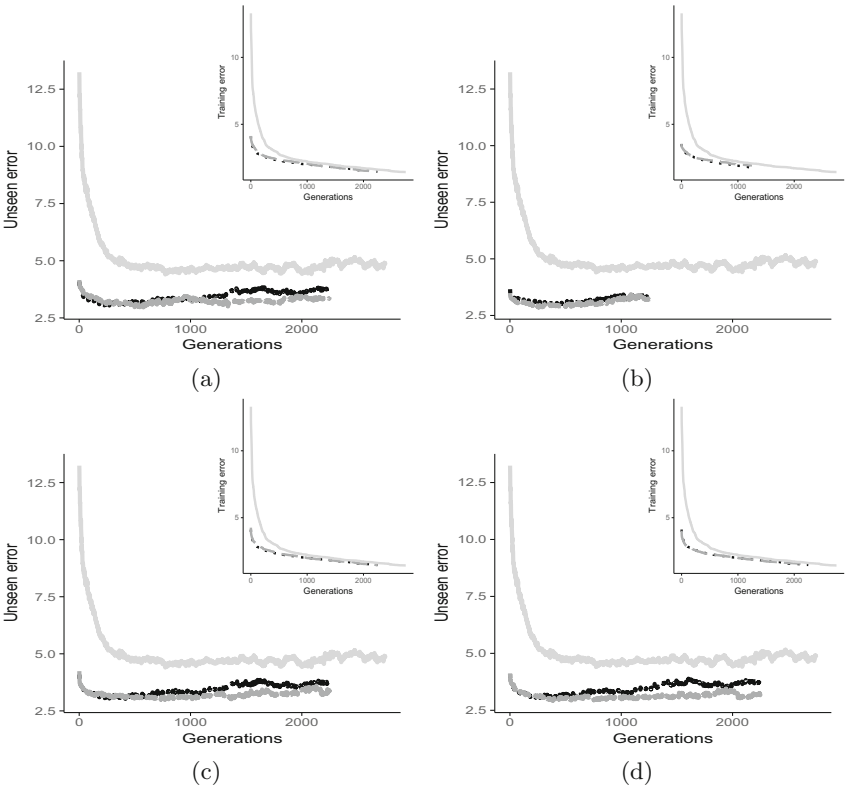

Fig. 2. Evolution of the (median) best fitness on the training (insets image) and test sets for the energy benchmark and the following parameterizations: (a) 50_5_25_25; (b) 50_15_50_50; (c) 50_5_75_75; (d) 50_5_50_50. The legend for all the plots is: ■ EDDA−V1 ■ EDDA−V2 ■ RHH

features considered. The results reported in this section consider values of these four parameters that were randomly selected from the values reported in Table 2. This allows analyzing the performance of EDDA-2 across different problems and parameterizations. Results of the experimental phase are reported from Figs. 2, 3 and 4. Each plot displays the generalization error (i.e., the fitness on unseen instances) and contains an inset showing the training fitness. Considering the training fitness, one can notice the same evolution of the fitness in all the considered benchmarks. EDDA-V1, in particular, is the best performer followed by EDDA-V2 and GSGP. Focusing on the two EDDA variants, these results were expected since EDDA-V1 is learning a model by using the whole training set and all the available features. On the other hand, EDDA-V2 is learning a model of the data considering a sample of the whole training set and, additionally, only a reduced number of features. Under this light, it is interesting to comment on the performance of EDDA-V2 and GSGP. The experimental results suggest that the

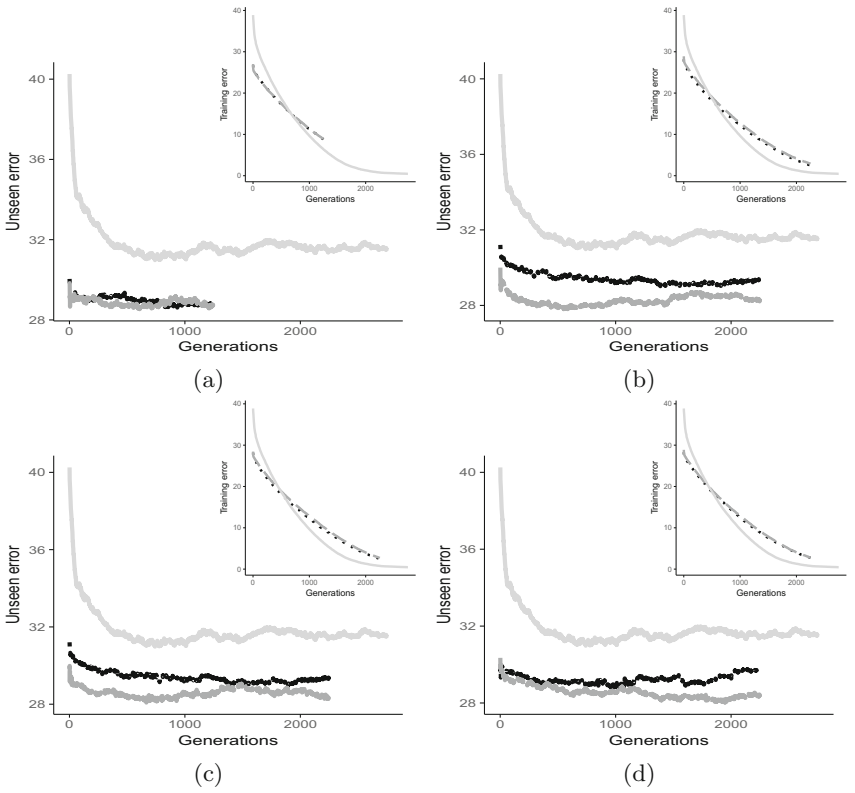

Fig. 3. Evolution of the (median) best fitness on the training (insets image) and test sets for the PPB benchmark and the following parametrizations: (a) 25_15_75_75; (b) 25_5_50_50; (c) 25_5_75_75; (d) 75_5_75_75. The legend for all the plots is: ■ EDDA−V1 ■ EDDA−V2 ■ RHH

usage of RHH for initializing the population results in poor performance when compared to EDDA-V2.

To summarize, results show the superior performance of EDDA-V1 when training error is taken into account, but EDDA-V2 produces a final model with an error that is smaller than the one produced by GSGP. This is a notable result because it shows that the proposed initialization method can outperform GSGP initialized with ramped half and half by considering a lower number of training instances and features.

While results on the training set are important to understand the ability of EDDA-V2 to learn the model of the training data, it is even more important and interesting to evaluate its performance on unseen instances. Considering the plots reported from Figs. 2, 3 and 4, one can see that the EDDA-V2 actually produces good quality solutions that are able to generalize over unseen instances. In particular, EDDA-V2 presents a very nice behavior in the vast majority of

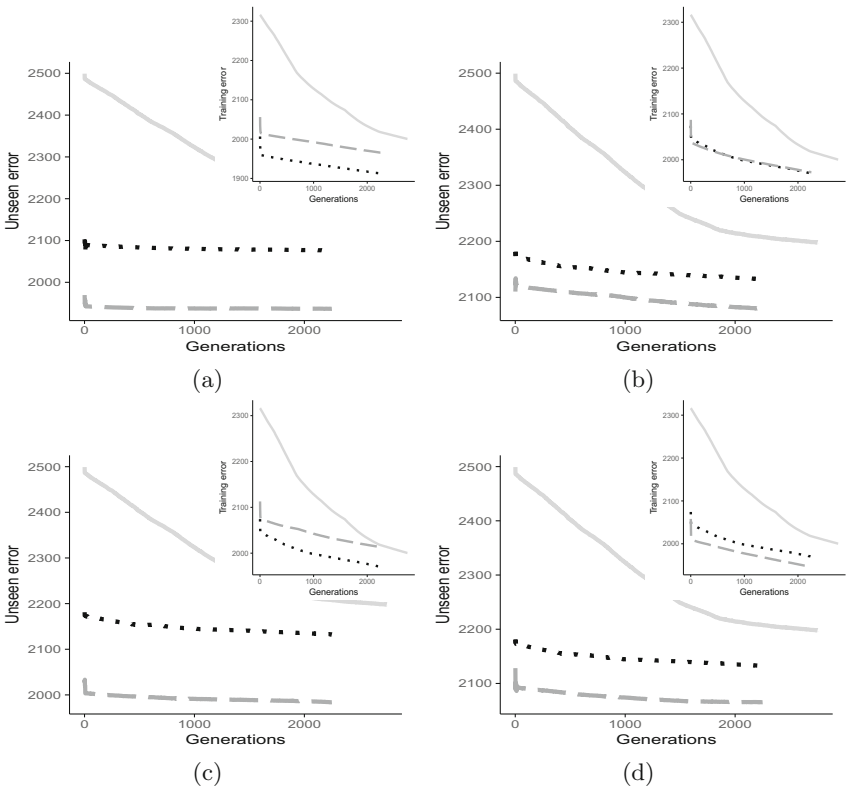

Fig. 4. Evolution of the (median) best fitness on the training (insets image) and test sets for the LD50 benchmark and the following parameterizations: (a) 25_5_75_75; (b) 10_5_25_25; 10_5_50_50(c); (d) 10_5_75_75. The legend for all the plots is: ■ EDDA−V1 ■ EDDA−V2 ■ RHH

the problems, showing better or comparable performance with respect to the other competitors without overfitting to the training data. Focusing on the other techniques, GSGP is the worst performer over all the considered benchmarks and parameterizations.

To summarize the results of this first part of the experimental phase, it is possible to state that EDDA-V2 outperforms GSGP with respect to the training fitness by also producing models able to generalize over unseen instances. When EDDA-V2 is compared against EDDA-V1, it performs poorer on the training instances, but the generalization error is better with respect to the latter system.

To conclude the experimental phase, Fig. 5 shows the time (ms) needed to initialize and evolve demes with EDDA-V1 and EDDA-V2. As expected EDDA-V2 requires less computational time.

To assess the statistical significance of these results, a statistical validation was performed considering the results achieved with the EDDA variants. First of all, given that it is not possible to assume a normal distribution of the values obtained by running the different EDDA variants, we ran the Shapiro-Wilk test and we considered a value of $\alpha = 0.05$. The null hypothesis of this test is that the values are normally distributed. The result of the test suggests that the null hypothesis should be rejected. Hence, we used the Mann–Whitney U test for comparing the results returned EDDA-V2 against the ones produced by EDDA-V1 under the null hypotheses that the distributions are the same across repeated measures. Also, in this test a value of $\alpha = 0.05$ was used.

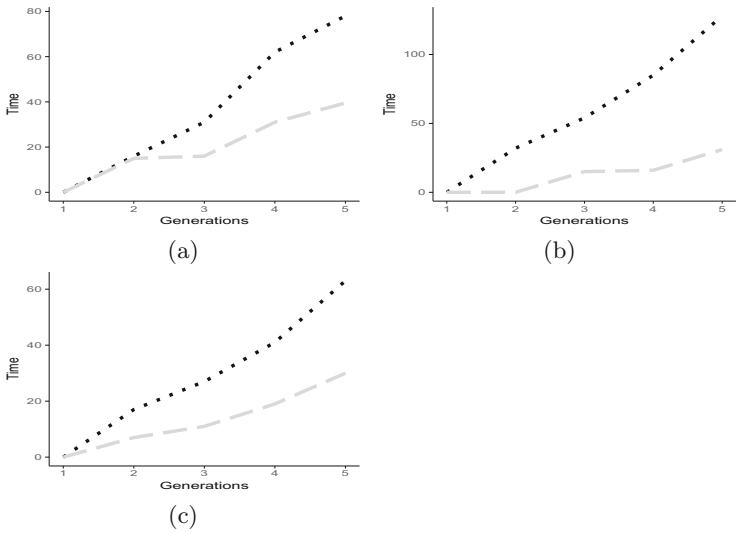

Fig. 5. Time needed to initialize and evolve a deme for 5 iterations, with a 50% of GSGP individuals, 50% of training instances, and 50% of features. Median calculated over all the demes and runs for (a) Energy, (b) PPB, and (c) LD50. The legend for all the plots is: ▪ EDDA−V1 ▬ EDDA−V2

Table 3 reports the p-values returned by the Mann–Whitney test, and **bold** is used to denote values suggesting that the null hypotheses should be rejected. Considering these results, it is interesting to note that with respect to the training error, EDDA-V2 and EDDA-V1 produced comparable results in the vast majority of the benchmarks and configurations taken into account. The same result applies to the test error, where it is important to highlight that, in each benchmark, there exists at least one parameters configuration that allows EDDA-V2 to outperform EDDA-V1.

Table 3. p-values returned by the Mann–Whitney U test. Test and training error achieved by populations initialized with EDDA-V2 and EDDA-V1 are compared. Values in the column parametrization correspond to the ones used in subplots (A), (B), (C), (D) of Figs. 2, 3, and 4. **Bold** is used to denote p-values suggesting that the null hypotheses should be rejected.

Parametrization	Test			Training		
	Energy	PPB	LD50	Enrgy	PPB	Ld50
A	**0.048**	0.203	0.065	**0.043**	0.523	0.109
B	0.708	0.267	0.230	0.123	**0.035**	0.273
C	0.440	0.230	**0.016**	0.187	0.142	**0.031**
D	0.708	0.203	0.390	0.109	0.843	0.901

6 Conclusions

Population initialization plays a fundamental role in the success of GP. Different methods were developed and investigated in the EA literature, all of them pointing out the importance of maintaining diversity among the different individuals in order to avoid premature convergence. A recent contribution, called Evolutionary Demes Despeciation Algorithm (EDDA-V1 in this paper), introduced an initialization technique in GP inspired by the biological phenomenon of demes despeciation. The method seeds a population of N individuals with the best solutions obtained by the independent evolution of N different populations, or demes. EDDA-V1 has demonstrated its effectiveness in initializing a GSGP population when compared to the standard ramped half and half method. This paper extended the initialization technique by defining a new method, called EDDA-V2 that initializes a population by evolving different parallel demes and, in each deme, it uses a different subset of the training instances and a different subset of the input features. This ensures an increased level of diversity, by also reducing the time needed for the initialization step. Experimental results obtained over three benchmark problems demonstrated that populations initialized with EDDA-V2 and evolved by GSGP converge towards solutions with a comparable or better generalization ability with respect to the ones initialized with EDDA-V1 and the traditional ramped half and half technique.

References

1. Archetti, F., Lanzeni, S., Messina, E., Vanneschi, L.: Genetic programming for computational pharmacokinetics in drug discovery and development. Genet. Program. Evol. Mach. **8**(4), 413–432 (2007)
2. Beadle, L.C.J.: Semantic and structural analysis of genetic programming. Ph.D. thesis, University of Kent, Canterbury, July 2009
3. Beadle, L.C.J., Johnson, C.G.: Semantic analysis of program initialisation in genetic programming. Genet. Program. Evol. Mach. **10**(3), 307–337 (2009)
4. Castelli, M., Manzoni, L., Vanneschi, L., Silva, S., Popovič, A.: Self-tuning geometric semantic genetic programming. Genet. Program. Evol. Mach. **17**(1), 55–74 (2016)
5. Castelli, M., Silva, S., Vanneschi, L.: A C++ framework for geometric semantic genetic programming. Genet. Program. Evol. Mach. **16**(1), 73–81 (2015)
6. Castelli, M., Vanneschi, L., Felice, M.D.: Forecasting short-term electricity consumption using a semantics-based genetic programming framework: the south italy case. Energy Econ. **47**, 37–41 (2015)
7. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
8. 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
9. Oliveira, L.O.V., Otero, F.E., Pappa, G.L.: A dispersion operator for geometric semantic genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016, GECCO 2016, pp. 773–780. ACM (2016)
10. Pawlak, T.P., Wieloch, B., Krawiec, K.: Review and comparative analysis of geometric semantic crossovers. Genet. Program. Evol. Mach. **16**(3), 351–386 (2015)
11. Taylor, E.B., Boughman, J.W., Groenenboom, M., Sniatynski, M., Schluter, D., Gow, J.L.: Speciation in reverse: morphological and genetic evidence of the collapse of a three-spined stickleback (gasterosteus aculeatus) species pair. Mol. Ecol. **15**(2), 343–355 (2006)
12. Tomassini, M., Vanneschi, L., Collard, P., Clergue, M.: A study of fitness distance correlation as a difficulty measure in genetic programming. Evol. Comput. **13**(2), 213–239 (2005)
13. Vanneschi, L.: An introduction to geometric semantic genetic programming. In: Schütze, O., Trujillo, L., Legrand, P., Maldonado, Y. (eds.) NEO 2015. SCI, vol. 663, pp. 3–42. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-44003-3_1
14. Vanneschi, L., Bakurov, I., Castelli, M.: An initialization technique for geometric semantic GP based on demes evolution and despeciation. In: 2017 IEEE Congress on Evolutionary Computation, CEC 2017, Donostia, San Sebastián, Spain, 5–8 June 2017, pp. 113–120 (2017)
15. Vanneschi, L., Castelli, M., Silva, S.: A survey of semantic methods in genetic programming. Genet. Program. Evol. Mach. **15**(2), 195–214 (2014)
16. 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
17. Wilson, D.S.: Structured demes and the evolution of group-advantageous traits. Am. Nat. **111**(977), 157–185 (1977). https://doi.org/10.1086/283146