An Investigation of Geometric Semantic GP with Linear Scaling

Giorgia Nadizar

Department of Mathematics and Geosciences, University of Trieste, Trieste, Italy, https://orcid.org/0000-0002-3535-9748

Fraser Garrow

Heriot-Watt University, Edinburgh, United Kingdom, https://orcid.org/0000-0001-5527-3952

Berfin Sakallioglu

NOVA Information Management School, Universidade Nova de Lisboa, Lisbon, Portugal, https://orcid.org/0009-0009-7835-6481

Lorenzo Canonne

Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France, https://orcid.org/0009-0009-8869-8071

Sara Silva

Department of Informatics, LASIGE, Lisbon, Portugal; Faculty of Sciences, University of Lisbon, Lisbon, Portugal; https://orcid.org/0000-0001-8223-4799

Leonardo Vanneschi

NOVA Information Management School, Universidade Nova de Lisboa, Lisbon, Portugal, https://orcid.org/0000-0003-4732-3328

This is the author accepted manuscript version of the paper published by ACM

Nadizar, G., Garrow, F., Sakallioglu, B., Canonne, L., Silva, S., & Vanneschi, L. (2023). An Investigation of Geometric Semantic GP with Linear Scaling. In GECCO'23: Proceedings of the 2023 Genetic and Evolutionary Computation Conference (pp. 1165-1174). Association for Computing Machinery (ACM). https://doi.org/10.1145/3583131.3590418

GECCO '23, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0119-1/23/07. . . $15.00
https://doi.org/10.1145/3583131.3590418

An Investigation of Geometric Semantic GP with Linear Scaling

Giorgia Nadizar
Department of Mathematics and Geosciences, University of Trieste
Trieste, Italy
giorgia.nadizar@phd.units.it

Fraser Garrow
Heriot-Watt University
Edinburgh, United Kingdom
fg28@hw.ac.uk

Berfin Sakallioglu
NOVA Information Management School, Universidade Nova de Lisboa, Portugal
bsakallioglu@novaims.unl.pt

Lorenzo Canonne
Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL
F-59000 Lille, France
lorenzo.canonne@inria.fr

Sara Silva
LASIGE, Department of Informatics Faculty of Sciences, University of Lisbon, Portugal
sara@fc.ul.pt

Leonardo Vanneschi
NOVA Information Management School, Universidade Nova de Lisboa, Portugal
lvanneschi@novaims.unl.pt

ABSTRACT

Geometric semantic genetic programming (GSGP) and linear scaling (LS) have both, independently, shown the ability to outperform standard genetic programming (GP) for symbolic regression. GSGP uses geometric semantic genetic operators, different from the standard ones, without altering the fitness, while LS modifies the fitness without altering the genetic operators. So far, these two methods have already been joined together in only one practical application. However, to the best of our knowledge, a methodological study on the pros and cons of integrating these two methods has never been performed. In this paper, we present a study of GSGP-LS, a system that integrates GSGP and LS. The results, obtained on five hand-tailored benchmarks and six real-life problems, indicate that GSGP-LS outperforms GSGP in the majority of the cases, confirming the expected benefit of this integration. However, for some particularly hard datasets, GSGP-LS overfits training data, being outperformed by GSGP on unseen data. Additional experiments using standard GP, with and without LS, confirm this trend also when standard crossover and mutation are employed. This contradicts the idea that LS is always beneficial for GP, warning the practitioners about its risk of overfitting in some specific cases.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by regression**; **Genetic programming**.

KEYWORDS

Symbolic Regression, Geometric Semantic Genetic Programming, Linear Scaling, Genetic Programming

GECCO '23, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0119-1/23/07…$15.00
https://doi.org/10.1145/3583131.3590418

ACM Reference Format:
Giorgia Nadizar, Fraser Garrow, Berfin Sakallioglu, Lorenzo Canonne, Sara Silva, and Leonardo Vanneschi. 2023. An Investigation of Geometric Semantic GP with Linear Scaling. In *Genetic and Evolutionary Computation Conference (GECCO '23), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583131.3590418

1 INTRODUCTION

In the last decades, the use of Genetic Programming (GP) [22] to tackle symbolic regression problems has gained popularity [3, 19, 31], possibly because of some qualities of GP, such as its ability to deal with problems where little or no information is known about the data, its ability to evolve models that do not have a previously fixed mathematical shape, and its ability to perform automatic feature selection while learning the model. GP is traditionally employed to tackle symbolic regression problems using well-known loss measures, such as the root mean square error (RMSE), to quantify the fitness of the evolving solutions. Even though this approach is still very popular, it has a drawback: some solutions may receive a bad fitness value, promising though they might be. This is the case, for instance, for solutions that have a similar shape to the one of the target function, but with different slope and/or location in the Cartesian space. Linear scaling (LS) was thus introduced by Keijzer [20] to tackle this issue and improve the performance of GP on symbolic regression. LS modifies the fitness function, rescaling each individual by using their slope and intercept, two constants that can be easily calculated with a cost that is linear in the size of the training set. In this way, the burden of searching for these two constants is removed from the evolution, leaving GP with the only task of searching for functions whose shape is most similar to that of the target function. Since its introduction, the benefit of LS was demonstrated on many theoretical benchmark functions [20] and real-life applications [2, 35, 38, 45]. These studies indicate that LS does not only improve standard GP on training data but can also bestow on GP a better generalisation ability, often outperforming standard GP also on unseen data. However, Costelloe and Ryan [9] pointed out that LS may not always improve GP's generalisation ability.

Ten years after the introduction of LS, Moraglio et al. [28] introduced Geometric Semantic GP (GSGP), a different variant of GP. GSGP uses specific genetic operators, called Geometric Semantic Operators (GSOs), instead of the traditional crossover and mutation

of GP. Although acting directly on the syntax of the GP individuals, GSOs have an indirect known effect on their semantics and have the important property of inducing a unimodal error surface for any supervised learning problem [28]. Several results were presented revealing the ability of GSGP to effectively fit training data [7, 28]. At the same time, GSGP was also shown to limit overfitting, often outperforming standard GP on unseen data for several real-life symbolic regression problems [44].

Given that LS works by redefining the fitness and GSGP works by redefining the genetic operators, which are in general two independent parts of the GP algorithm, it is natural to imagine a system that joins these two methods, possibly capturing the advantages of both GSGP and LS. Following this idea, in 2015 Vanneschi et al. [42] combined GSGP and LS, achieving outstanding results on a challenging application based on AIS (Automatic Identification System) for the prediction of the positions of vessels at sea. The success of that system in that particular application domain, together with the previous achievements of GSGP and LS used in isolation, may induce researchers to think that the integration of GSGP and LS is always beneficial. However, Costelloe's and Ryan's observations made on standard GP [9] sound like an important warning and call for a methodological study aimed at investigating the pros and cons of integrating GSGP and LS. In this paper, we present such a study, investigating a system that, similarly to what was done in [42], explores the search space using GSOs, guided by the LS fitness function. We call it GSGP-LS.

The remainder of this paper is organized as follows. In Section 2 we briefly review previous works relevant to this study, while in Section 3 we describe GSGP and LS. In Section 4 we describe our experimental setup, first presenting the used test problems and then discussing the parameter settings. Then, in Section 5 we present and comment on the obtained results. Finally, in Section 6 we conclude the work and propose ideas for future research.

2 PREVIOUS AND RELATED WORK

Although similar ideas to LS had already been proposed for GP before Keijzer's contribution [20], the previous works involving multiple linear regression were considered costly and increased the likeliness of overfitting, since they introduced extra parameters and limitations to the system [16–18, 32]. Conversely, Keijzer's work showed a dramatic improvement in the performance of GP for symbolic regression by applying LS to the error measure [20], at a limited computational cost. In his first contribution, Keijzer demonstrated the benefits of LS on several synthetic test functions. Shortly after, he published another article, where he gave theoretical corroboration to the success of LS [21].

After Keijzer's contribution, LS has been used in several benchmark problems and real-life applications. Here, we discuss some of these works. For instance, Archetti et al. [2], reported using LS with GP to improve the performance on several regression tasks related to the area of drug discovery. A few years later, the same authors also successfully applied LS with GP on another problem from the medical field, consisting in predicting the effect of an anticancer therapy on a specific cohort of patients [1]. In the same year, Raja et al. [35] also combined LS with GP for applications in the telecommunication area and concluded that the system that used LS outperformed the system that did not use it.

A general trend has also been to integrate LS in GP systems that also contain other novel methods. For instance, Pennachin et al. [33] used affine arithmetic to improve both the performance and the robustness of GP for symbolic regression, and they also performed LS of outputs before fitness evaluation. The presented results indicate that the proposed system reduces the number of fitness evaluations needed during training and improves generalisation of GP, reducing overfitting. Similarly, Azad and Ryan [4] integrated LS and a method to maintain diversity in a GP system aimed at exploring lifetime learning. A few years later, Virgolin et al. [45] applied LS to a GP-based algorithm, called GP-GOMEA, on a symbolic regression problem from the area of oncology. Later, in another work where several other real-world datasets were employed [46], the same authors confirmed the power of LS, successfully integrating LS in a semantic backpropagation-based GP system. Recently, Ruberto et al. [38] tackled dynamic target problems by integrating LS with a GP system, using the hinge-loss functions to evolve a set of discriminant functions for multi-class classification. The authors reported on the advantage of the version that uses LS. Later, these results were confirmed and extended, providing an upper bound to the error in dynamic symbolic regression [37, 38] and classification [39].

Even though LS has been applied to GP several times, so far in the literature it is possible to find only one contribution in which LS has been integrated with GSGP: in 2015, [42] applied LS to GSGP for tackling an application in the maritime awareness domain. The objective of that work was to predict the position of vessels at sea, based on information related to the vessels' past positions in a specific time interval, using AIS data. The proposed system was compared to two different GP variants and three non-evolutionary machine learning methods, outperforming all of them. A study aimed at investigating GSGP with LS for other problems, from different application areas, possibly pointing out the pros and cons of the system, is currently still missing, and this paper aims to fill this gap. In Section 4.1, where we describe the test problems used in this work, we also point out the numerous differences between those test problems and the one studied in [42].

Despite the several successes on real-life applications, Costelloe and Ryan [9] remarked that several methods that improve GP's training performance, including LS, may not improve GP's generalisation ability, as well. This consideration is important for us since it partially reflects some of the findings of this work.

3 BACKGROUND

3.1 Geometric Semantic Genetic Programming

Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ be the set of input data (training instances, observations or fitness cases) of a symbolic regression problem, and $\mathbf{t} = [t_1, t_2, ..., t_n]$ the vector of the respective expected (scalar) output or target values (in other words, for each $i = 1, 2, ..., n$, t_i is the expected output corresponding to input $\mathbf{x}_i$). A GP individual (or program) P can be seen as a function that, for each input vector $\mathbf{x}_i$ returns the scalar value $P(\mathbf{x}_i)$. Following [28], we call *semantics* of P the vector $s_P = [P(\mathbf{x}_1), P(\mathbf{x}_2), ..., P(\mathbf{x}_n)]$. This vector can be represented as a point in an n-dimensional space, that we call *semantic space*. Note that the target vector $\mathbf{t}$ itself is a point in

the semantic space. As explained above, GSGP is a variant of GP where the standard crossover and mutation are replaced by GSOs. The objective of GSOs is to define modifications on the syntax of GP individuals that have a precise effect on their semantics. In particular, geometric semantic crossover (GSC) generates one offspring whose semantics stands in the line joining the semantics of the two parents in the semantic space, while geometric semantic mutation (GSM), by mutating an individual i, allows us to obtain another individual j such that the semantics of j stands inside a ball of a given predetermined radius centered in the semantics of i.

One of the reasons why GSOs became popular is because GSOs induce a unimodal error surface (on training data) for any supervised learning problem where fitness is calculated using an error measure between outputs and targets. In other words, when using GSOs the error surface on training data is guaranteed to not have any locally optimal solution. This property holds, for instance, for any regression or classification problem, independently of how big and how complex data are. A detailed explanation of the reason why the error surface is unimodal and why this is important can be found in [40]. The definitions of the GSOs are, as given in [28], respectively[1]:

Geometric Semantic Crossover (GSC). Given two parent functions $T_1, T_2 : \mathbb{R}^n \rightarrow \mathbb{R}$, GSC returns the function $T_{XO} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2)$, where T_R is a random function whose output values range in the interval $[0, 1]$.

Geometric Semantic Mutation (GSM). Given a parent function $T : \mathbb{R}^n \rightarrow \mathbb{R}$, GSM with mutation step ms returns the function $T_M = T + ms \cdot (T_{R1} - T_{R2})$, where T_{R1} and T_{R2} are random functions.

The reason why GSM uses two random trees T_{R1} and T_{R2} is that the amount of modification caused by GSM must be centred in zero. In other words, a random expression is needed that has the same probability of being positive or negative. As pointed out in [29], any isotropic Gaussian random function centred in zero can, in principle, be replaced with the term $(T_{R1} - T_{R2})$ in the definition of GSM. Even though this is not in the original definition of GSM, later contributions [12, 40, 44] have clearly shown that limiting the codomain of T_{R1} and T_{R2} in a predefined interval (for instance $[0, 1]$, as it is done for T_R in GSC) helps improve the generalisation ability of GSGP. For this reason, as in several previous works [7, 40], also in this paper we constrain the outputs of T_R, T_{R1}, and T_{R2} by wrapping them in a logistic function. As reported in [28, 40], GSOs have the drawback of generating larger offspring than the parents, and this entails a rapid growth of the size of the individuals in the population. To counteract this problem, in [7, 24, 27] implementations of GSOs were proposed, that make GSGP not only usable in practice but also significantly faster than standard GP. This is possible through a smart representation of GP individuals that allows us to not store their genotypes during the evolution. The implementation presented in [5] also employs the same idea, and it is the one used here.

[1]Only the definitions of the GSOs for symbolic regression problems are given here since they are the only ones used in this work. For the definition of GSOs for other domains, we refer the reader to [28].

3.2 Linear Scaling

LS [20] is a method that was introduced to facilitate the task of GP of searching for the best function matching a set of known data. It consists in calculating the slope and intercept of the formula coded by a GP individual. Let $P(\mathbf{x}_i)$ be the output of a GP individual P on the i-th observation of the training set. Using the same notation as in Section 3.1, a linear regression on the target values $\mathbf{t}$ can be performed using the equations:

$$b = \frac{\sum_{i=1}^{n} \left[(t_i - \overline{t}) \left(P(\mathbf{x}_i) - \overline{P} \right) \right]}{\sum_{i=1}^{n} \left(P(\mathbf{x}_i) - \overline{P} \right)^2}, \qquad a = \overline{t} - b\,\overline{P} \tag{1}$$

where n is the number of training observations (fitness cases) and $\overline{P}$ and $\overline{t}$ denote the average output and the average target value respectively. Values b and a respectively calculate the slope and intercept of the set of outputs $P(\mathbf{x}_i)$, such that the sum of the squared errors between $\mathbf{t}$ and $a + bP$ is minimised. After this, any error measure can be calculated on the scaled formula $a + bP$, for instance the RMSE. If a is different from 0 and b is different from 1, the procedure outlined above is guaranteed to reduce the RMSE for any formula P [20]. The cost of calculating the slope and intercept is linear in the size of the training set. By efficiently calculating the slope and intercept for each individual, the burden of searching for these two constants is removed from the evolution. GP is then free to search for the expression whose shape is most similar to that of the target function.

4 EXPERIMENTAL SETUP

We investigate the effectiveness of LS when combined with GSGP. To do so, we compare the performance of GSGP, with and without LS, on five hand-tailored symbolic regression benchmarks and six real-life regression datasets. In order to give a more complete interpretation of some of the obtained results, we perform the same experiments also for standard GP (the motivation for this further study will be clearer when the GSGP results will have been discussed, at the end of Section 5.1). The four configurations that we have studied are denoted by GSGP-LS, GSGP, GP-LS and GP, respectively. We implement each configuration using the General Purpose Optimisation Library (GPOL) [5], a publicly available software platform that integrates numerous computational intelligence algorithms, including several variants of GP, such as standard GP and GSGP. We have included LS by extending the library. This section describes the experimental study carried out: in Section 4.1 we overview the considered test problems and in Section 4.2 we describe the parameter settings.

4.1 Test Problems

The five theoretical benchmarks that we have studied were taken directly from the paper that introduced LS [20]. They are:

- $f_5(x) = x^3 \exp^{-x} \cos(x) \sin(x)(\sin^2(x) * \cos(x) - 1)$
- $f_6(x, y, z) = \dfrac{30xz}{(x - 10)y^2}$

- $f_7(x) = \sum_{i}^{x} 1/i$
- $f_8(x) = \log x$
- $f_9(x) = \sqrt{x}$

Besides being a good scientific practice to test a method (in this study, LS) on the same case studies that were used when it was introduced, motivations for choosing these benchmarks are the same as in [20]. Namely, "many of the problems above mix trigonometry with polynomials, or make the problems in other ways highly nonlinear". Also, it is relevant to point out that, as stated in [20], "being of low dimensionality does not make the problems easy". Exactly as in [20], we have used these benchmarks with the training and test intervals reported in Table 1. The six real-world regression problems that we have employed, and that have often been used as case studies for GP experiments, are:

- *Boston Housing* [15]: a dataset provided by the Statistical Library and maintained by Carnegie Mellon University. The purpose is to forecast housing prices using data such as air pollution, criminality, pupil-teacher ratio, etc.
- *Concrete Compressive Strength* [48]: a dataset aimed at predicting the strength of concrete depending on the age, mixture and other features of the ingredients.
- *Parkinson Total UPDRS* [23]: composed of a range of biomedical voice measurements and other features of Parkinson's disease patients. The aim is to predict the clinician's Parkinson's disease symptom score on the UPDRS scale.
- *Bioavailability* [2]: consists in forecasting the human oral bioavailability of a set of drug compounds, based on a set of molecular descriptors.
- *LD50* [2]: is also a problem in the field of pharmacokinetics. Its purpose is to predict the median lethal dose of a molecular compound, which is one of the most used measures to assess the toxicity of drugs.
- *PPB* [2]: is another dataset from the field of pharmacokinetics. Its aim is to predict the percentage of the initial drug dose which binds plasma proteins.

Table 2 reports the number of instances and attributes for each one of these datasets. Among these datasets, the Bioavailability one was criticised in [10], partially because of a lack of preprocessing, since it includes features that contain no information as well as contradictory relationships between the dependent and independent variables. However, according to many authors who have used this dataset, these characteristics are interesting and should be integrated in a reasonable benchmark suite, because they allow us to test the ability of our algorithms to deal with the difficulties and ambiguities that are typical of real-world data. It is not our objective to discuss what characteristics a good benchmark suite should possess (the interested reader is referred to [25, 26, 30, 47] for such a discussion). We simply observe that the Bioavailability dataset, as well as the PPB and LD50 datasets, have been used in several previous GP studies, clearly indicating a trend for overfitting to emerge [7, 12, 44]. We thus use these three datasets as a sort of stress-test cases to assess the generalisation ability of GSGP-LS, compared to the other studied algorithms.

4.2 Parameter Settings

The objective of this work is to compare the studied GP variants. Our focus is not on obtaining the best possible results on the considered test problems. For this reason, instead of optimizing the hyperparameters, which would probably lead us to the use of different parameters for each problem, we have preferred to use a relatively standard parameter setting, taken as much as possible from the literature. This allowed us to use the same setting across all the test cases. Table 3 reports the employed parameters for each configuration. GSGP is known for working well with smaller population sizes and a larger number of generations than GP [6]. As such, for GSGP we use a population of 100 individuals, while standard GP uses 500 individuals. For a fair comparison, we allow the same number of fitness evaluations for both GSGP and GP, therefore running GSGP for 500 generations and standard GP for 100 generations. The populations are initialised with the Ramped Half-and-Half method [22], with maximum initial tree depth of 6, and allowed a maximum depth of 17 during the run [22], with no depth limit for GSGP [28]. We employ the same function and terminal sets for both algorithms, with the four basic arithmetic operators and no random constants, as in [12, 40, 44]. In the function set, $\div_p$ refers to the protected division function that returns 1.0 if the denominator is less than 0.001. Both algorithms use tournament selection and elitism of size 1 (best individual copied unchanged into the next generation). The different tournament sizes represent the same proportion of individuals (2 % of the population) and therefore the same selective pressure for both algorithms. Regarding the genetic operators, for GSGP we use the GSOs described in Section 3.1, while for standard GP we use standard subtree crossover and mutation [22]. The genetic operator probabilities follow the general guidelines for each method, without any particular tuning. GSGP uses a logistic wrapper on all random trees, as described in Section 3.1. As suggested in [44], the mutation step (*ms* parameter in the definition of GSM in Section 3.1) is a random number between 0 and 1, that is generated independently of the previous ones at each mutation event—note that the value of *ms* could also be optimized via gradient descent, as in [34]. Each algorithm uses the same parameters with and without LS.

For each of the studied test problems, we performed 30 independent runs for each configuration. Concerning the training-test partitioning, for the theoretical benchmarks we have used the intervals reported in Table 1, which are the same as in [20]. For the real-life problems, at each run, we have selected at random, with uniform probability, 70 % of the observations to form the training set, while the remaining 30 % were used as test set. It is important to point out that, in the same run, all the configurations used the same partitions, and the partitions change (because they are randomly generated each time) from one run to the other. The results reported in the next section are the medians and interquartile range, computed over the performed 30 runs, of the fitness on the training and on the test set of the individual with the best fitness on the training set at each generation.

Table 1: Intervals used as training and test set for the hand-tailored benchmarks used in this work (taken from [20]). Intervals are expressed using the notation *[start:step:stop]*.

Benchmark	Training Set	Test Set	Note
f_5	$x, z = \text{rnd}(-1, 1)$, $y = \text{rnd}(1, 2)$	$x, z = \text{rnd}(-1, 1)$, $y = \text{rnd}(1, 2)$	Train: 1000 cases, Test: 10000 cases
f_6	$x = [1 : 1 : 50]$	$x = [1 : 1 : 120]$	Extrapolation
f_7	$x = [1 : 1 : 100]$	$x = [1 : 0.1 : 100]$	Interpolation
f_8	$x = [0 : 1 : 100]$	$x = [1 : 0.1 : 100]$	Interpolation
f_9	$x = [0 : 1 : 100]$	$x = [1 : 0.1 : 100]$	Interpolation

Table 2: Number of instances and attributes of the datasets.

Problems	Instances	Attributes
Boston	506	14
Concrete	1030	9
Parkinson	5875	20
Bioavailability	260	247
LD50	234	627
PPB	131	627

Table 3: Parameter settings used in the experiments.

Parameter	GSGP	GP
Generations	500	100
Population size	100	500
Initialisation	Ramped	Ramped
Max. init. depth	6	6
Max. depth	∞	17
Function set	$\{+, -, \times, \div_p\}$	$\{+, -, \times, \div_p\}$
Terminal set	{ *features* }	{ *features* }
Selection	Tournament	Tournament
Tournament size	2	10
Elites	1	1
Genetic operators	GSOs	Standard
Crossover probability	0.3	0.7
Mutation probability	0.7	0.3
Mutation step	$\mathcal{U}(0, 1]$	N.A.
Random tree wrapper	Logistic	N.A.

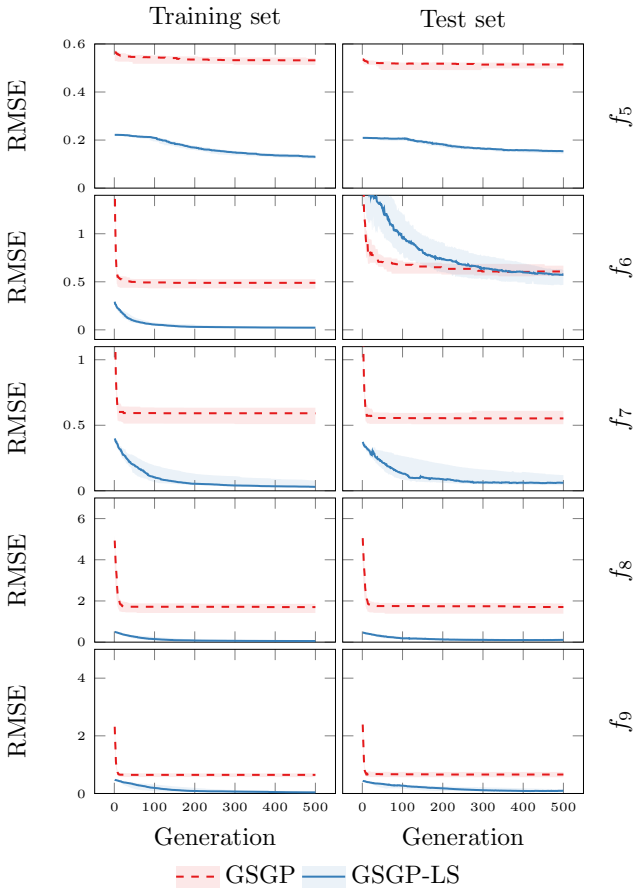

Figure 1: Comparison between GSGP and GSGP-LS on Keijzer's theoretical benchmarks. Evolution of median fitness and interquartile range (in 30 independent runs) of the best individual on the training and test sets.

5 RESULTS AND DISCUSSION

5.1 GSGP vs GSGP-LS

Figure 1 reports the evolution of training and test fitness for GSGP and GSGP-LS on the theoretical benchmarks. From these plots, we can observe that GSGP-LS outperforms GSGP on the training set for all the case studies. Also, GSGP-LS outperforms GSGP on the test set for all the problems, except for function f_6, on which the performance of GSGP-LS and the one of GSGP are comparable. To assess the statistical significance of these results, we performed a Mann-Whitney U test with significance level $\alpha = 0.05$, for both training and test sets, for each problem, at each generation, with the null hypothesis that the distribution of the RMSE of the best individual originated from the 30 runs is the same for GSGP and GSGP-LS. The p-values resulting from the comparison of the two algorithms on the training set are not reported here to save space. However, the differences between GSGP and GSGP-LS on the training set are always statistically significant. The evolution of the p-value resulting from the comparison of the two algorithms on the test set is reported in Figure 2. We also show the significance threshold in each plot (red horizontal line at 0.05) and we clip the y-axis at 0.2 to ease the visual examination of the results. These plots clearly

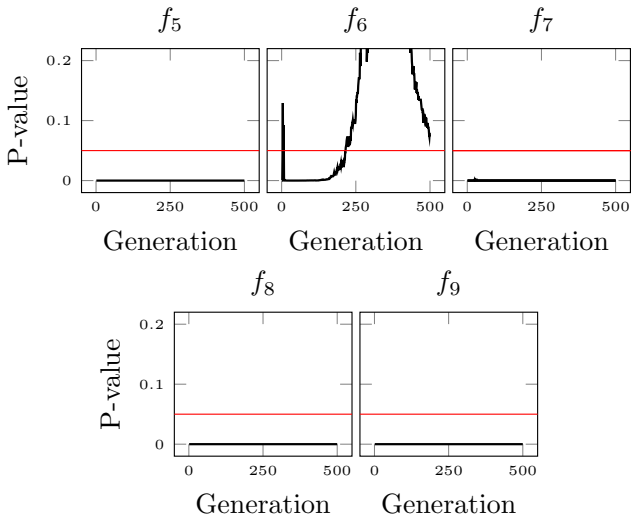

Figure 2: Evolution of p-values of the Mann-Whitney U test from the comparison between GSGP and GSGP-LS on the test data of Keijzer's benchmarks.

confirm that the difference between GSGP-LS and GSGP is statistically significant also on the test set for all the studied theoretical benchmarks, except for function f_6, on which the p-value is larger than the threshold, except for the initial phase of the run.

Figure 3 reports the evolution of training and test fitness for GSGP and GSGP-LS on the real-life problems. Also in this case, GSGP-LS consistently outperforms GSGP on the training set for all considered problems. Concerning the results on the test set, instead, we notice that although GSGP-LS outperforms GSGP on three of the considered problems, it suffers from overfitting issues on the Bioavailability, LD50, and PPB datasets (in the figure, these three cases are separated from the previous three by means of a horizontal dashed line). Analogously to the case of the theoretical benchmarks, also for the real-life problems we do not report the p-values of the Mann-Whitley U test on the training set. However, those p-values confirm that all the differences between GSGP-LS and GSGP are statistically significant on the training set for each one of the studied real-life problems. Figure 4 reports the evolution of the p-values of the Mann-Whitley U test for the real-life problems on the test set. Considering the results on the test set for the real-life problems, at first glance, we can clearly divide those results into two groups: (1) the first, including the Boston, Concrete, and Parkinson datasets, for which GSGP-LS is always significantly outperforming GSGP (p-values of the order of 10^{-15}), (2) and the second, involving the remaining datasets, for which the results are more controversial. From these last three plots, we notice a common trend: the p-value, in the beginning, is smaller than the threshold; it initially grows, and then decays, eventually crossing the significance level for LD50 and PPB, and approaching it for Bioavailability. This, together with the plots of Figure 3, tells us that there are three distinct phases during the evolution: (1) initially, GSGP-LS significantly outperforms GSGP, (2) then, for some generations, the two methods have comparable performance, and (3) in the end, GSGP-LS becomes (significantly)

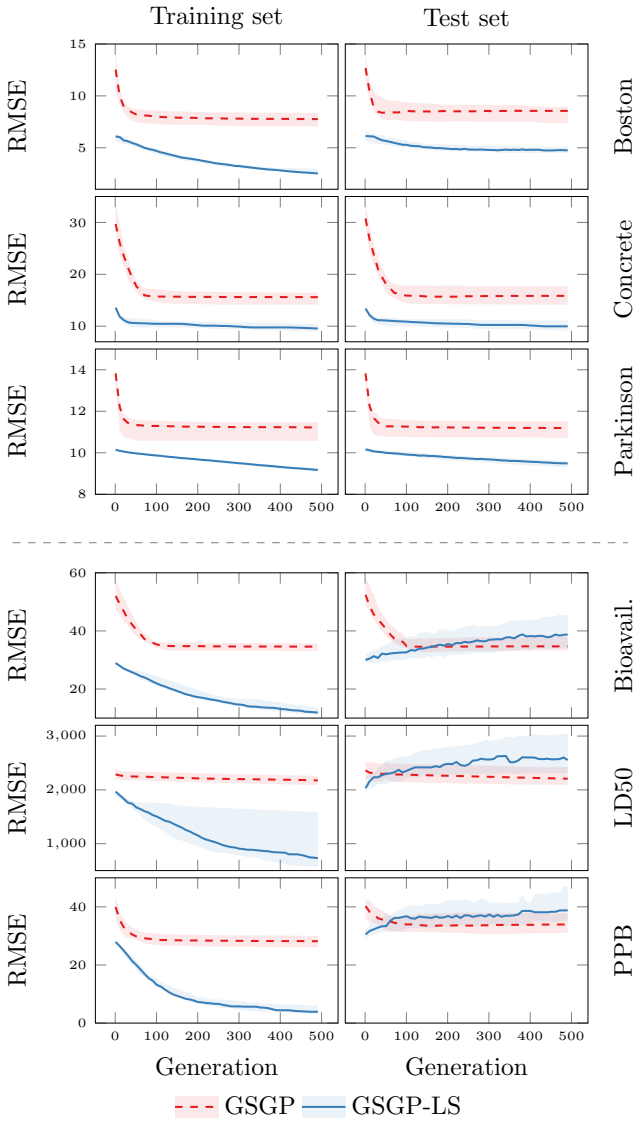

Figure 3: Comparison between GSGP and GSGP-LS on the real-life problems. Evolution of median fitness and interquartile range (in 30 independent runs) of the best individual on the training and test sets. Easier problems are on top (above the horizontal dashed line) and harder problems are on the bottom (below the dashed line).

worse due to overfitting. However, the fact that the GSGP-LS error curves on the test set for these three problems are increasing since the very first generations should also be noticed. We extend our analysis and discussion on the matter in Section 5.2.

Last, it is interesting to notice that, for both training and test sets, the initial RMSE of GSGP-LS is already lower than that at the end of evolution for GSGP. On the training set, this outcome was expected, given the known benefits of LS on the initial population [20]. In addition, although GSGP plateaus fairly soon during evolution,

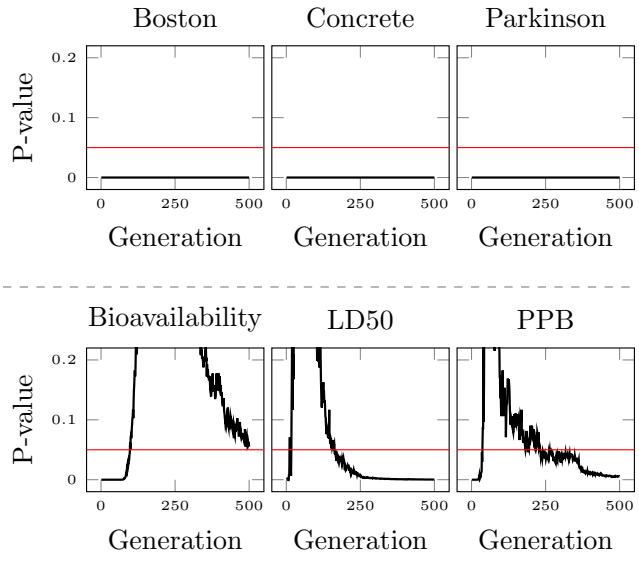

Figure 4: Evolution of p-values of the Mann-Whitney U test from the comparison between GSGP and GSGP-LS on the test data of the real-life problems, divided by difficulty.

the performance of GSGP-LS steadily improves across generations on the training set (which sometimes leads to overfitting). This is probably caused by the fact that GSGP has to look for constants a and b of Equation (1), while GSGP-LS already has those constants calculated. This gives GSGP-LS a greater degree of freedom in the search for a function with optimal shape, which clearly leads to an overall better fit of the training data.

5.2 Discussion on Overfitting

Even though the results obtained with GSGP-LS appear generally promising, we have encountered some overfitting issues. Since GSGP has demonstrated its ability to control overfitting [43, 44], it is natural to wonder if the introduction of LS specifically disrupts the benefits of GSGP or if LS is in general more prone to overfitting on some datasets. To answer this question, we repeated our experimental evaluation with standard GP compared to GP with LS (GP-LS). We report these results in Figure 5 for the theoretical benchmark problems and in Figure 6 for the real-life problems. We can observe that (1) GP is itself more prone to overfitting than GSGP, and (2) LS worsens the overfitting issues on the Bioavailability, LD50 and PPB datasets. Thus, we can conclude that LS can induce or worsen overfitting on some problems, regardless of the evolutionary algorithm on which it is applied, be it standard GP or GSGP.

Going back to the results of Figures 3 and 4, we can also gain additional insights on possible ways to overcome the overfitting issue. Since the test error starts with a reasonable value and only significantly increases after some generations, as per definition of overfitting, the most trivial solution would be to stop the evolution earlier. However, deciding a proper stopping condition to tackle the problem is not straightforward. The simplest solution in that

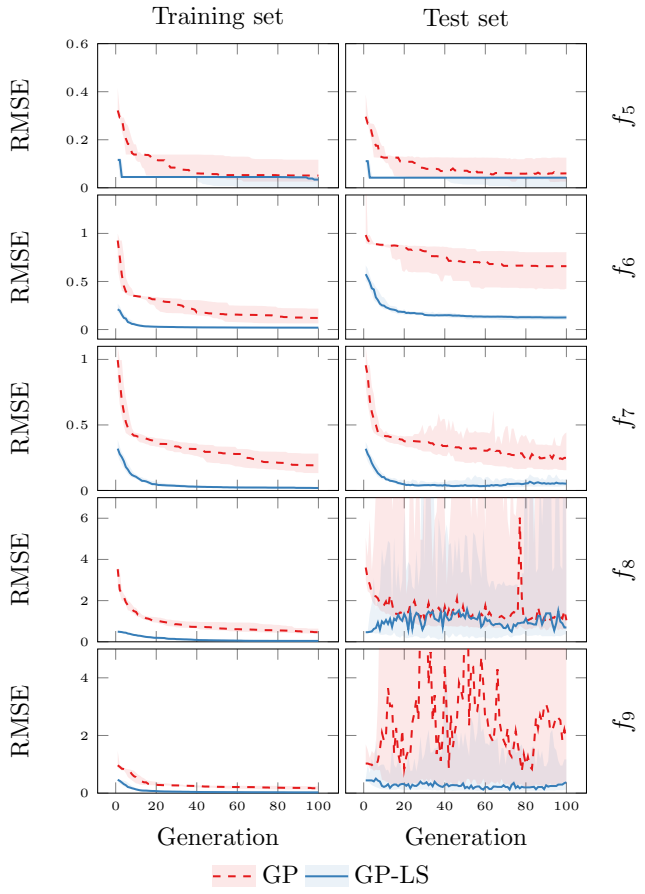

Figure 5: Comparison between GP and GP-LS on Keijzer's theoretical benchmarks. Evolution of median fitness and interquartile range (in 30 independent runs) of the best individual on the training and test sets.

sense would be to introduce a validation set to simulate the error on unseen data along evolution, to detect immediately if the solution is starting to overfit. However, this strategy is only viable with large enough datasets, since it implies splitting the dataset into more parts, hence reducing the number of observations in each. In fact, all three datasets where we observed overfitting suffer from a lack of instances, thus becoming not ideal candidates for this solution. A more sophisticated early stopping criterion, which seems more suitable for the datasets at hand, has been proposed in [13] to leverage the semantic information available in GSGP for deciding when to end the evolutionary optimization. Notably, such a criterion would also yield the positive side effect of improving the overall computation efficiency of the process.

However, both the curves of the test set fitness of GSGP-LS (as already noticed at the end of the previous section) and GP-LS are increasing since the very beginning of the run on Bioavailability, LD50 and PPB. This induces us to think that, although worth investigating, early stopping may not be enough to generate reliable

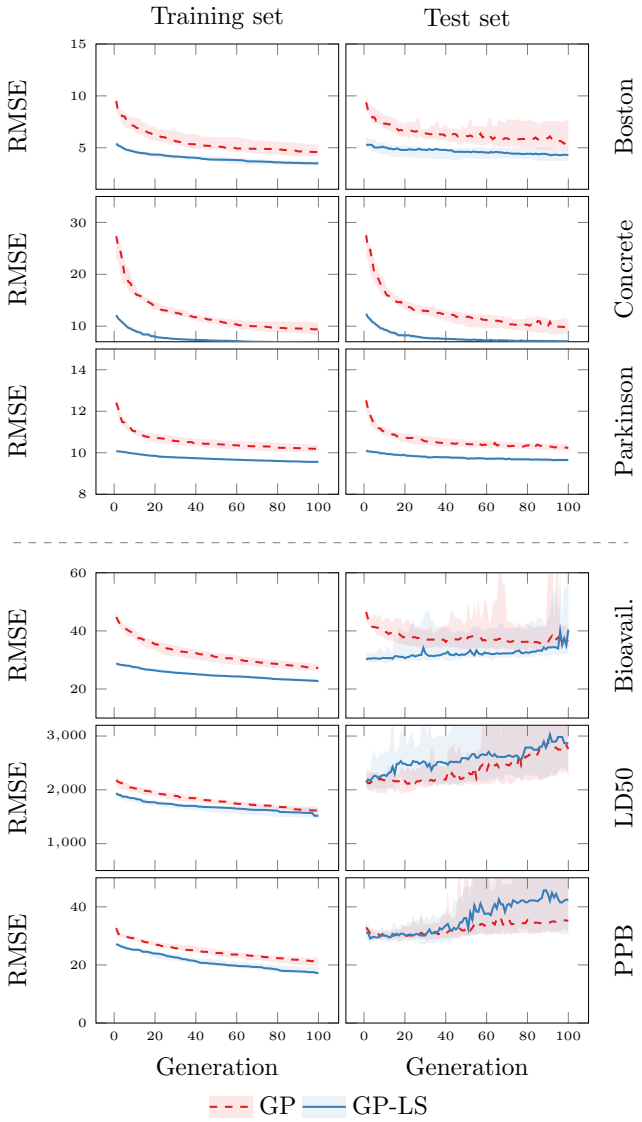

Figure 6: Comparison between GP and GP-LS. Evolution of median fitness and interquartile range (in 30 independent runs) of the best individual on the training and test sets. We divide the datasets into two difficulty groups.

models in those cases. For this reason, in the second part of Section 6, we propose other strategies that should be explored in the future to limit overfitting.

6 CONCLUSIONS AND FUTURE WORK

We explored the effects of augmenting Geometric Semantic Genetic Programming (GSGP) with Linear Scaling (LS). In particular, encouraged by the success that was obtained in [42], we aimed at investigating the combination of the beneficial traits of these two methods, which can both outperform standard Genetic Programming (GP), by improving the genetic operators – for GSGP – and the fitness – for LS. Our analysis involved a thorough experimental evaluation on the task of symbolic regression for five theoretical benchmarks and six real-life problems of various difficulties. We compared GSGP against GSGP with LS (GSGP-LS), both in terms of efficiency, i.e., how fast evolution is able to achieve the desired goal, and in terms of generalisation, i.e., how well the induced model is able to generalise to unseen data. Our results show significant improvements over standard GSGP for most cases on both training and test sets, and also highlight that LS speeds up the evolutionary search w.r.t. GSGP. However, we have observed that the integration with LS makes GSGP more prone to overfitting when the addressed problem is characterized by particularly difficult data. Yet, we have noticed a similar trend for standard GP with LS on the same datasets, which led us to conclude that LS might induce overfitting in some specifically hard cases. Nonetheless, from the behaviour of LS during the evolution, we concluded that we could stop the search process earlier to achieve comparable or better results than without LS, for both GP and GSGP, with the desirable side effect of saving computation time. For GSGP, we consider the approach of early stopping based on semantic neighbourhood presented in [13] particularly promising.

Besides early stopping, other methods have recently demonstrated their effectiveness in controlling overfitting. For instance, one may imagine developing a system in which LS is turned on and off dynamically during the evolution. A similar approach has been recently presented for suitable use of local search inside GSGP [8]. In that contribution, GSGP was enhanced with local search at the beginning of the run, but the local search was later disabled to reduce overfitting. A similar idea may let us exploit the advantages of LS in the initial generations, and later the evolution could continue using GSGP without LS to control overfitting. Other methods that have been defined to control overfitting for GSGP and GP are the dynamic interleaving of training instances [11] and soft target regularization [41]. These methods look promising for GSGP-LS. Last but not least, the use of an explicit feature selection in a preprocessing phase [14], to integrate the implicit feature selection already performed by GP during the learning, like for instance the approach proposed in [36], should be investigated.

ACKNOWLEDGMENTS

This work was partially supported by FCT, Portugal, through funding of research units MagIC/NOVA IMS (UIDB/04152/2020) and LASIGE (UIDB/00408/2020 and UIDP/00408/2020). We also wish to thank the SPECIES Society and Anna Esparcia-Alcázar for organizing the SPECIES Summer School 2022, which brought us together and gave us the chance to start this collaboration.

REFERENCES

[1] Francesco Archetti, Ilaria Giordani, and Leonardo Vanneschi. 2010. Genetic programming for anticancer therapeutic response prediction using the NCI-60 dataset. *Computers & Operations Research* 37 (08 2010), 1395–1405. https://doi.org/10.1016/j.cor.2009.02.015

[2] Francesco Archetti, Stefano Lanzeni, Enza Messina, and Leonardo Vanneschi. 2007. Genetic programming for computational pharmacokinetics in drug discovery and development. *Genetic Programming and Evolvable Machines* 8, 4 (2007), 413–432.

[3] D.A. Augusto and H.J.C. Barbosa. 2000. Symbolic regression via genetic programming. In *Proceedings. Vol.1. Sixth Brazilian Symposium on Neural Networks.* 173–178. https://doi.org/10.1109/SBRN.2000.889734

[4] Raja Muhammad Atif Azad and Conor Ryan. 2014. A Simple Approach to Lifetime Learning in Genetic Programming-Based Symbolic Regression. *Evolutionary Computation* 22, 2 (06 2014), 287–317. https://doi.org/10.1162/EVCO_a_00111 arXiv:https://direct.mit.edu/evco/article-pdf/22/2/287/1509584/evco_a_00111.pdf

[5] Illya Bakurov, Marco Buzzelli, Mauro Castelli, Leonardo Vanneschi, and Raimondo Schettini. 2021. General Purpose Optimization Library (GPOL): A Flexible and Efficient Multi-Purpose Optimization Library in Python. *Applied Sciences* 11, 11 (2021). https://doi.org/10.3390/app11114774

[6] Mauro Castelli, Luca Manzoni, Sara Silva, Leonardo Vanneschi, and Aleš Popovič. 2017. The influence of population size in geometric semantic GP. *Swarm and Evolutionary Computation* 32 (2017), 110–120. https://doi.org/10.1016/j.swevo.2016.05.004

[7] Mauro Castelli, Sara Silva, and Leonardo Vanneschi. 2015. A C++ framework for geometric semantic genetic programming. *Genetic Programming and Evolvable Machines* 16, 1 (2015), 73–81.

[8] Mauro Castelli, Leonardo Trujillo, Leonardo Vanneschi, Sara Silva, Emigdio Z-Flores, and Pierrick Legrand. 2015. Geometric Semantic Genetic Programming with Local Search. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation* (Madrid, Spain) *(GECCO '15)*. Association for Computing Machinery, New York, NY, USA, 999–1006. https://doi.org/10.1145/2739480.2754795

[9] Dan Costelloe and Conor Ryan. 2009. On Improving Generalisation in Genetic Programming. In *Genetic Programming*, Leonardo Vanneschi, Steven Gustafson, Alberto Moraglio, Ivanoe De Falco, and Marc Ebner (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 61–72.

[10] Grant Dick, Aysha P Rimoni, and Peter A Whigham. 2015. A re-examination of the use of genetic programming on the oral bioavailability problem. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. 1015–1022.

[11] Ivo Gonçalves and Sara Silva. 2013. Balancing Learning and Overfitting in Genetic Programming with Interleaved Sampling of Training Data. In *Genetic Programming*, Krzysztof Krawiec, Alberto Moraglio, Ting Hu, A. Şima Etaner-Uyar, and Bin Hu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 73–84.

[12] Ivo Gonçalves, Sara Silva, and Carlos M. Fonseca. 2015. On the Generalization Ability of Geometric Semantic Genetic Programming. In *Genetic Programming*, Penousal Machado, Malcolm I. Heywood, James McDermott, Mauro Castelli, Pablo García-Sánchez, Paolo Burelli, Sebastian Risi, and Kevin Sim (Eds.). Springer International Publishing, Cham, 41–52.

[13] Ivo Gonçalves, Sara Silva, Carlos M Fonseca, and Mauro Castelli. 2017. Unsure when to stop? ask your semantic neighbors. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 929–936.

[14] Isabelle Guyon and André Elisseeff. 2003. An Introduction to Variable and Feature Selection. *J. Mach. Learn. Res.* 3, null (mar 2003), 1157–1182.

[15] David Harrison and Daniel L Rubinfeld. 1978. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management* 5, 1 (1978), 81–102. https://doi.org/10.1016/0095-0696(78)90006-2

[16] H.G. Hiden, M.J. Willis, M.T. Tham, P. Turner, and G.A. Montague. 1997. Nonlinear principal components analysis using genetic programming. In *Second International Conference On Genetic Algorithms In Engineering Systems: Innovations And Applications*. 302–307. https://doi.org/10.1049/cp:19971197

[17] Hitoshi Iba, Hugo de Garis, and Taisuke Sato. 1994. *Genetic Programming Using a Minimum Description Length Principle*. MIT Press, Cambridge, MA, USA, 265–284.

[18] H. Iba and N. Nikolaev. 2000. Genetic programming polynomial models of financial data series. In *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, Vol. 2. 1459–1466 vol.2. https://doi.org/10.1109/CEC.2000.870826

[19] Ilknur Icke and Joshua C. Bongard. 2013. Improving genetic programming based symbolic regression using deterministic machine learning. In *2013 IEEE Congress on Evolutionary Computation*. 1763–1770. https://doi.org/10.1109/CEC.2013.6557774

[20] M. Keijzer. 2003. Improving Symbolic Regression with interval arithmetic and linear scaling. In *Genetic Programming, Proceedings of the 6th European Conference, EuroGP 2003 (LNCS, Vol. 2610)*, C. Ryan *et al.* (Ed.). Springer, Berlin, Heidelberg, New York, Essex, 71–83.

[21] Maarten Keijzer. 2004. Scaled Symbolic Regression. *Genetic Programming and Evolvable Machines* 5, 3 (sep 2004), 259–269. https://doi.org/10.1023/B:GENP.0000030195.77571.f9

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

[23] Max A. Little, Patrick E. McSharry, Stephen J. Roberts, Declan AE Costello, and Irene M. Moroz. 2007. Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection. *BioMedical Engineering OnLine* 6, 1 (26 Jun 2007), 23. https://doi.org/10.1186/1475-925X-6-23

[24] Joao Francisco B. S. Martins, Luiz Otavio V. B. Oliveira, Luis F. Miranda, Felipe Casadei, and Gisele L. Pappa. 2018. Solving the Exponential Growth of Symbolic Regression Trees in Geometric Semantic Genetic Programming. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Kyoto, Japan) *(GECCO '18)*. ACM, New York, NY, USA, 1151–1158.

[25] James McDermott, Gabriel Kronberger, Patryk Orzechowski, Leonardo Vanneschi, Luca Manzoni, Roman Kalkreuth, and Mauro Castelli. 2022. Genetic Programming Benchmarks: Looking Back and Looking Forward. *SIGEVOlution* 15, 3, Article 1 (dec 2022), 19 pages. https://doi.org/10.1145/3578482.3578483

[26] James McDermott, David R. White, Sean Luke, Luca Manzoni, Mauro Castelli, Leonardo Vanneschi, Wojciech Jaskowski, Krzysztof Krawiec, Robin Harper, Kenneth De Jong, and Una-May O'Reilly. 2012. Genetic Programming Needs Better Benchmarks. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation* (Philadelphia, Pennsylvania, USA) *(GECCO '12)*. Association for Computing Machinery, New York, NY, USA, 791–798. https://doi.org/10.1145/2330163.2330273

[27] A. Moraglio. 2014. An efficient implementation of GSGP using higher-order functions and memoization. In *Semantic Methods in Genetic Programming, Workshop at Parallel Problem Solving from Nature* (Ljubljana, Slovenia).

[28] Alberto Moraglio, Krzysztof Krawiec, and ColinG. Johnson. 2012. Geometric Semantic Genetic Programming. In *Parallel Problem Solving from Nature - PPSN XII*, CarlosA.Coello Coello, Vincenzo Cutello, Kalyanmoy Deb, Stephanie Forrest, Giuseppe Nicosia, and Mario Pavone (Eds.). Lecture Notes in Computer Science, Vol. 7491. Springer Berlin Heidelberg, 21–31.

[29] Alberto Moraglio and Andrea Mambrini. 2013. Runtime Analysis of Mutation-Based Geometric Semantic Genetic Programming for Basis Functions Regression. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation* (Amsterdam, The Netherlands) *(GECCO '13)*. Association for Computing Machinery, New York, NY, USA, 989–996. https://doi.org/10.1145/2463372.2463492

[30] Miguel Nicolau, Alexandros Agapitos, Michael O'Neill, and Anthony Brabazon. 2015. Guidelines for defining benchmark problems in Genetic Programming. In *2015 IEEE Congress on Evolutionary Computation (CEC)*. 1152–1159. https://doi.org/10.1109/CEC.2015.7257019

[31] Miguel Nicolau and James McDermott. 2020. *Genetic Programming Symbolic Regression: What Is the Prior on the Prediction?* Springer International Publishing, Cham, 201–225. https://doi.org/10.1007/978-3-030-39958-0_11

[32] N.Y. Nikolaev and H. Iba. 2001. Regularization approach to inductive genetic programming. *IEEE Transactions on Evolutionary Computation* 5, 4 (2001), 359–375. https://doi.org/10.1109/4235.942530

[33] Cassio Pennachin, Moshe Looks, and João A. de Vasconcelos. 2010. Robust Symbolic Regression with Affine Arithmetic. In *Genetic and Evolutionary Computation COnference (GECCO)*.

[34] Gloria Pietropolli, Luca Manzoni, Alessia Paoletti, and Mauro Castelli. 2022. Combining geometric semantic gp with gradient-descent optimization. In *European Conference on Genetic Programming (Part of EvoStar)*. Springer, 19–33.

[35] Adil Raja, R. Muhammad Atif Azad, Colin Flanagan, and Conor Ryan. 2007. Real-Time, Non-Intrusive Evaluation of VoIP *(EuroGP'07)*. Springer-Verlag, Berlin, Heidelberg, 217–228.

[36] Nuno M. Rodrigues, João E. Batista, William La Cava, Leonardo Vanneschi, and Sara Silva. 2022. SLUG: Feature Selection Using Genetic Algorithms and Genetic Programming. In *Genetic Programming*, Eric Medvet, Gisele Pappa, and Bing Xue (Eds.). Springer International Publishing, Cham, 68–84.

[37] Stefano Ruberto, Valerio Terragni, and Jason Moore. 2021. A semantic genetic programming framework based on dynamic targets. *Genetic Programming and Evolvable Machines* 22 (12 2021), 1–31. https://doi.org/10.1007/s10710-021-09419-3

[38] Stefano Ruberto, Valerio Terragni, and Jason H. Moore. 2020. SGP-DT: Towards Effective Symbolic Regression with a Semantic GP Approach Based on Dynamic Targets. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion* (Cancún, Mexico) *(GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 25–26. https://doi.org/10.1145/3377929.3397486

[39] Stefano Ruberto, Valerio Terragni, and Jason H. Moore. 2021. Towards Effective GP Multi-Class Classification Based on Dynamic Targets. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Lille, France) *(GECCO '21)*. Association for Computing Machinery, New York, NY, USA, 812–821. https://doi.org/10.1145/3449639.3459324

[40] Leonardo Vanneschi. 2017. *An Introduction to Geometric Semantic Genetic Programming*. Springer International Publishing, Cham, 3–42.

[41] Leonardo Vanneschi and Mauro Castelli. 2021. Soft target and functional complexity reduction: A hybrid regularization method for genetic programming. *Expert Systems with Applications* 177 (2021), 114929. https://doi.org/10.1016/j.eswa.2021.114929

[42] Leonardo Vanneschi, Mauro Castelli, Ernesto Costa, Alessandro Re, Henrique Vaz, Victor Lobo, and Paulo Urbano. 2015. Improving Maritime Awareness with Semantic Genetic Programming and Linear Scaling: Prediction of Vessels Position Based on AIS Data. In *Applications of Evolutionary Computation*, Antonio M. Mora and Giovanni Squillero (Eds.). Springer International Publishing, Cham, 732–744.

[43] L Vanneschi, M Castelli, L Manzoni, and S Silva. 2013. A new implementation of geometric semantic GP applied to predicting pharmacokinetic parameters. In *Genetic Programming: 16th European Conference, EuroGP 2013, Vienna, Austria, April 3–5, 2013. Proceedings*, Vol. 7831. Springer Berlin, Germany, 205–216.

[44] Leonardo Vanneschi, Sara Silva, Mauro Castelli, and Luca Manzoni. 2014. *Geometric Semantic Genetic Programming for Real Life Applications*. Springer New

York, New York, NY, 191–209.

[45] Marco Virgolin, Tanja Alderliesten, Arjan Bel, Cees Witteveen, and Peter A. N. Bosman. 2018. Symbolic Regression and Feature Construction with GP-GOMEA Applied to Radiotherapy Dose Reconstruction of Childhood Cancer Survivors. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Kyoto, Japan) *(GECCO '18)*. Association for Computing Machinery, New York, NY, USA, 1395–1402. https://doi.org/10.1145/3205455.3205604

[46] Marco Virgolin, Tanja Alderliesten, and Peter A. N. Bosman. 2019. Linear Scaling with and within Semantic Backpropagation-Based Genetic Programming for Symbolic Regression. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Prague, Czech Republic) *(GECCO '19)*. Association for Computing Machinery, New York, NY, USA, 1084–1092. https://doi.org/10.1145/3321707.3321758

[47] John Woodward, Simon Martin, and Jerry Swan. 2014. Benchmarks That Matter for Genetic Programming. In *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation* (Vancouver, BC, Canada) *(GECCO Comp '14)*. Association for Computing Machinery, New York, NY, USA, 1397–1404. https://doi.org/10.1145/2598394.2609875

[48] I.-C. Yeh. 1998. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research* 28, 12 (1998), 1797–1808. https://doi.org/10.1016/S0008-8846(98)00165-3