The Effect of Distinct Geometric Semantic Crossover Operators in Regression Problems

Julio Albinati[1], Gisele L. Pappa[1], Fernando E. B. Otero[2], and Luiz Otávio V. B. Oliveira[13]

[1] Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
{jalbinati, glpappa, luizvbo}@dcc.ufmg.br
[2] University of Kent, Chatham Maritime, Kent, UK
F.E.B.Otero@kent.ac.uk
[3] Instituto Federal do Sul de Minas Gerais, Poços de Caldas, Brazil

Abstract. This paper investigates the impact of geometric semantic crossover operators in a wide range of symbolic regression problems. First, it analyses the impact of using Manhattan and Euclidean distance geometric semantic crossovers in the learning process. Then, it proposes two strategies to numerically optimize the crossover mask based on mathematical properties of these operators, instead of simply generating them randomly. An experimental analysis comparing geometric semantic crossovers using Euclidean and Manhattan distances and the proposed strategies is performed in a test bed of twenty datasets. The results show that the use of different distance functions in the semantic geometric crossover has little impact on the test error, and that our optimized crossover masks yield slightly better results. For SGP practitioners, we suggest the use of the semantic crossover based on the Euclidean distance, as it achieved similar results to those obtained by more complex operators.

Keywords: semantic genetic programming, crossover, crossover mask optimization

1 Introduction

The development of methods that take the semantics of the solutions being evolved into account is a trend in the genetic programming community, with special attention given to methods based on geometric semantic crossover operators [12, 8]. The main reason for researchers interest in geometric semantic crossover is that, by manipulating directly the semantics of solutions, it behaves in a much more controlled way since the semantic impact of operators can be easily bounded. It also has interesting properties regarding control of overfitting, establishing upper bounds in test error. Furthermore, the fitness landscapes induced by semantic operators are usually much simpler than regular landscapes, making optimization easier.

This paper is particularly interested in the impact of geometric semantic crossover operators into symbolic regression problems. Given a set of inputs I

and their respectively expected outputs O, the semantics of a function f being evolved can be indirectly assessed by a quality measure, such as the error rate, which calculates the differences between the expected outputs O and the obtained outputs $O' = f(I)$. As different functions can map to the same value of error, we can say this error measure performs a kind of syntactic–semantic (or genotype-phenotype) mapping.

Considering that the semantics of a solution can be represented by its output vector O', different ways of measuring the semantic distance between two functions have been proposed. In the case of symbolic regression, as the outputs generated return real values, the Manhattan and Euclidean distances are appropriate functions for measuring error. These distance metrics can then be used to measure the semantic distance between pairs of individuals.

Geometric semantic operators were defined to work into different spaces of functions defined by the distance metrics previously described. In this way, the geometric crossover is a function of the semantics of their parents. Given two real-valued functions f_1 and f_2, a geometric semantic crossover returns a third real-valued function f_3 representing the convex combination of the parents. The offspring is obtained by multiplying f_1 and f_2 by a *crossover mask*, which can be represented by a constant, in the case of the Euclidean distance or a function (e.g. logistic [13]), in the case of the Manhattan distance.

This paper investigates the impact of using Manhattan and Euclidean distance on geometric semantic crossovers in the learning process and proposes two strategies to numerically optimize the crossover mask based on mathematical properties of these operators, instead of simply generating them randomly. We present an experimental analysis in which the different distance metric crossovers and the proposed strategies are compared on a test bed composed of twenty datasets with distinct properties from both real and synthetic domains.

The remainder of this paper is organised as follows. Section 2 presents an overview of the methods that incorporate semantic awareness into GP. Section 3 introduces the two new optimization strategies applied to the crossover masks, followed by the experimental analysis in the test bed in Section 4. Finally, conclusions and perspectives of future work are presented in Section 5.

2 Related Work

The study of programs or individuals semantics in GP has been developed mostly in the last five years [12]. In [12], the authors divide semantic-aware techniques into three groups: *diversity methods*, *indirect semantic methods* and *direct semantic methods*.

Diversity methods were the first proposed, aiming to preserve or reinject diversity throughout evolution. Although methods aiming at GP diversity are not new, they usually considered only syntactic diversity. In [4], in turn, the authors studied the impact of semantic diversity during population initialization, showing that greater diversity leads to improved results. Indirect semantic methods, on the other hand, use regular GP operator, but only accept indi-

viduals if they respect some semantic-related criteria, such as their semantic difference to their parents [3] or to a geometric (semantically intermediate) individual of their parents [6]. Different versions of methods based on this approach were subsequently proposed [10, 9], and although they led to improved results over traditional crossover operators, they are *trial-and-error* techniques, without any guarantees that solutions respecting the criteria established will be actually generated.

In contrast to indirect methods, direct semantic methods use operators specifically designed to operate in a semantic level, and are our subject of interest. In [8], the authors proposed geometric semantic crossover and mutation operators for three domains: boolean, categorical and real-valued. The geometric semantic crossover operator follows Definition 1, which is essentially a convex combination of two previously generated solutions. The function that combines the solutions, $c(\mathbf{x})$, is the *crossover mask*.

Definition 1 *Let $\mathcal{F}$ be the set of functions mapping instances to real numbers and $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}) \in \mathcal{F}$ be two previously generated solutions. Then $XO : \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{F}$ is called a semantic geometric crossover (for real-valued functions) if $XO(f_1(\boldsymbol{x}), f_2(\boldsymbol{x})) = c(\boldsymbol{x}) \cdot f_1(\boldsymbol{x}) + (1 - c(\boldsymbol{x})) \cdot f_2(\boldsymbol{x})$, where $c(\boldsymbol{x})$ outputs values in the interval $[0, 1]$.*

Note that for the crossover operator, if $c(\mathbf{x}) = \beta$ for all $\mathbf{x}$, then its geometric properties will be related to the Euclidean distance in the semantic search space. However, if $c(\mathbf{x})$ is allowed to output distinct values for distinct instances, then its geometric properties will be related to the Manhattan distance.

Since the crossover operator performs a convex combination, a solution $f_3(\mathbf{x})$ generated by this operator will be semantically intermediate of its parents $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$: $dist(f_1, f_2) = dist(f_1, f_3) + dist(f_3, f_2)$, where $dist$ may be the Euclidean or Manhattan distance in the semantic search space depending on the choice of crossover mask. For the Euclidean distance, this fact lead to the property that the error committed by $f_3(\mathbf{x})$ is upper bounded by the error of the worst of its parents. More interestingly, this property holds for both training and test sets, thus being useful for controlling overfitting [8, 11].

Successive applications of the geometric semantic crossover, however, may lead to an exponential growth in the size of solutions, as pointed out in [8]. In fact, the number of nodes of a tree T_3 obtained as the crossover of two other trees, T_1 and T_2, is greater than the number of nodes of T_1 and T_2 altogether. Although it is possible to simplify the functions represented by such trees, this would lead to a large computational effort. In [5], the authors proposed an efficient way of dealing with this problem by avoiding replicating subtrees that are part of more than a solution. They also suggested the usage of a sigmoid (logistic) function $c(\mathbf{x}) = (1 + e^{-r(\mathbf{x})})^{-1}$ ($r(\mathbf{x})$ being a randomly generated function), which correctly outputs values in the required interval.

3 Optimized Semantic Crossover Operators

This section proposes two new versions of the semantic crossover operator showed in Definition 1. These new versions were generated by optimizing the crossover masks instead of randomly generating them.

3.1 Optimized Convex Geometric Semantic Crossover Operator

The first proposed operator was generated by finding the value of the crossover mask that leads to the minimum training error. As we show in this section, this new operator has an interesting property: it is *non-degenerative*, strengthening the convex property regarding training error. While the convex property states that the error of the function being generated will never be larger than the worst of its parents, we can now state that the error of the function being generated will never be larger than the best of its parents when considering training error, as showed below.

Suppose that the crossover mask $c(\mathbf{x})$ is a single constant, i.e, $c(\mathbf{x}) = \beta$ for all $\mathbf{x}$ (and the crossover is based on the Euclidean distance in the semantic space). Let $f_1(\mathbf{x})$, $f_2(\mathbf{x})$ be two previously generated solutions, and $f_3(\mathbf{x}) = XO(f_1(\mathbf{x}), f_2(\mathbf{x}))$. Then, the sum of squared errors (SSE) of f_3 can be expressed in terms of β:

$$SSE(\beta) = \sum_{i=1}^{n} [y_i - \beta \cdot f_1(\mathbf{x}_i) - (1 - \beta) \cdot f_2(\mathbf{x}_i)]^2 \tag{1}$$

where training data is represented as sequence of pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$.

Since $SSE(\beta)$ is continuous, we can calculate the derivative of Equation 1 and equals it to zero, finding

$$\beta^* = \frac{\sum_{i=1}^{n} [y_i - f_2(\mathbf{x}_i)][f_2(\mathbf{x}_i) - f_1(\mathbf{x}_i)]}{\sum_{i=1}^{n} [f_1(\mathbf{x}_i) - f_2(\mathbf{x}_2)]^2} \tag{2}$$

such that it minimizes the error of $f_3(\mathbf{x})$, as shown in Proposition 1. Note that calculating the optimized coefficient can be done in $O(n)$, the same time required for computing the semantic of the offspring. Therefore, the optimization of coefficients does not change the asymptotic complexity of SGP.

However, β^* as computed in Equation 2 may not fall in the $[0, 1]$ interval. This will happen whenever any convex combination of f_1 and f_2 is worse than f_1 and f_2. To enforce the interval constraint, we use a distinct value β^{**} such that

$$\beta^{**} = max(min(1, \beta^*), 0) \tag{3}$$

Proposition 1 *The argument β^{**} as expressed in Equation 3 minimizes the error function (Equation 1) while respecting the interval constraint.*

Proof. Since $\lim_{\beta \to c} SSE(\beta) = SSE(c)$, $\forall c \in \mathbb{R}$, $SSE(\beta)$ is a continuous function in $\mathbb{R}$ and we can compute its derivate with relation to β.

$$\frac{\delta SSE}{\delta \beta} = 2 \cdot \sum_{i=1}^{n} [y_i - f_2(\mathbf{x}_i)][f_2(\mathbf{x}_i) - f_1(\mathbf{x}_i)] + \beta \cdot [f_2(\mathbf{x}_i) - f_1(\mathbf{x}_i)]^2$$

By making the derivative equals to zero, we find a (local) minimum or maximum point.

$$\beta^* = \frac{\sum_{i=1}^{n} [y_i - f_2(\mathbf{x}_i)][f_1(\mathbf{x}_i) - f_2(\mathbf{x}_i)]}{\sum_{i=1}^{n} [f_2(\mathbf{x}_i) - f_1(\mathbf{x}_i)]^2}$$

We now need to show that β^* is a minimization point. We compute the second derivative of $SSE(\beta)$ with relation to β.

$$\begin{aligned} \frac{\delta^2 SSE(\beta)}{\delta \beta^2} &= \sum_{i=1}^{n} [f_2(\mathbf{x}_i) - f_1(\mathbf{x}_i)]^2 \\ &\geq 0 \end{aligned}$$

Since the second derivative obtained is always non-negative, we prove that β^* is a minimization point and $SSE(\beta)$ is convex. Suppose now that $\beta^* > 1$. Then, $\beta^{**} = 1$ and it minimizes $SSE(\beta)$ while being in the interval $[0, 1]$, since $SSE(\beta)$ is convex and β^{**} is the closest point to β^* in the interval. An analogue reasoning implies that if $\beta^* < 0$, $\beta^{**} = 0$ minimizes the error function while being in the required interval. Finally, if $0 \leq \beta^* \leq 1$, then $\beta^{**} = \beta^*$ and is also a minimization point in the required interval. Thus, we prove that β^{**} minimizes $SSE(\beta)$ while respecting the interval constraint.

As β is optimized in the closed interval $[0, 1]$, if $0 < \beta^{**} < 1$, then $SSE(\beta^{**}) \leq SSE(1)$ and $SSE(\beta^{**}) \leq SSE(0)$. Otherwise, β^{**} would be 1 or 0 and the best of the two functions used in crossover would be simply replicated (see Fig. 1). This shows that the error of the function being generated will never be larger than the best of its parents. Regarding test error, the geometric property is still valid, since this modified version is still a convex combinations of functions.

3.2 Optimized Non-Convex Geometric Semantic Crossover Operator

The convex geometric semantic crossover operator can be very constrained: the fact that it can only generate solutions semantically intermediate of the functions being used for crossover implies that the performance can be strongly determined by the initial population. In order to build a more flexible operator, we propose the following non-convex crossover operator, based on linear combinations (not

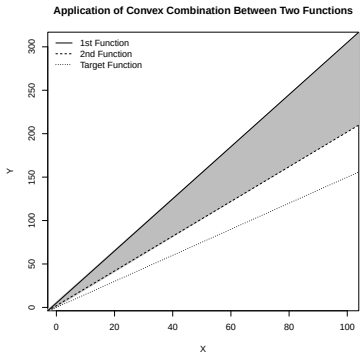

Fig. 1. Application of the convex semantic geometric crossover over functions $Y = 3X + 5$ and $Y = 2X + 2$, and the target function $Y = 1.5X$. The gray area represents possible convex combination of the two functions. Note, however, that the target function is outside of the gray area, meaning that any convex combination is worse (or equal) than the second function.

necessarily convex). However, this means increasing the risk of overfitting, as we now do not have any guarantees regarding test error.

$$XO(f_1(\mathbf{x}_i), f_2(\mathbf{x}_i)) = \beta_1 \cdot f_1(\mathbf{x}_i) + \beta_2 \cdot f_2(\mathbf{x}_i) \tag{4}$$

Again, we can express the error of a function generated through Equation 4 in terms of β_1 and β_2.

$$SSE(\beta_1, \beta_2) = \sum_{i=1}^{n} [y_i - \beta_1 \cdot f_1(\mathbf{x}_i) - \beta_2 \cdot f_2(\mathbf{x}_i)]^2 \tag{5}$$

Since $SSE(\beta_1, \beta_2)$ is continuous in $\mathbb{R}^2$, we can use the same strategy presented in the previous subsection to find β_1^* and β_2^* that minimizes Equation 5. Let F be a n-by-2 matrix where $F_{ij} = f_j(\mathbf{x}_i)$ and Y be a column vector of length n containing the target values of each training instance. Then

$$\begin{pmatrix} \beta_1^* \\ \beta_2^* \end{pmatrix} = (F^t F)^{-1} F^t Y \tag{6}$$

Note that similar approaches have been already proposed. In [2], the authors propose to linearly combine *subexpressions* of programs to re-interpret their semantics. In this work, however, we propose to apply a linear combination of two *distinct* programs.

Proposition 2 *The arguments β_1^* and β_2^* as expressed in Equation 6 minimize the error function (Equation 5).*

Proof. Since $\lim_{(\beta_1,\beta_2)\to(c_1,c_2)} SSE(\beta_1,\beta_2) = SSE(c_1,c_2)$ for an arbitrary pair $(c_1,c_2) \in \mathbb{R}^2$, $SSE(\beta_1,\beta_2)$ is continuous in $\mathbb{R}^2$ and we can compute its derivative with relation to β_1 and β_2.

$$\frac{\delta SSE}{\delta \beta_1} = \sum_{i=1}^{n} -2[y_i \cdot f_1(\mathbf{x}_i) - \beta_1 \cdot f_1(\mathbf{x}_i)^2 - \beta_2 \cdot f_1(\mathbf{x}_i) \cdot f_2(\mathbf{x}_i)]$$
$$\frac{\delta SSE}{\delta \beta_2} = \sum_{i=1}^{n} -2[y_i \cdot f_2(\mathbf{x}_i) - \beta_1 \cdot f_1(\mathbf{x}_i) \cdot f_2(\mathbf{x}_i) - \beta_2 \cdot f_2(\mathbf{x}_i)^2]$$

Letting F be a n-by-2 matrix where $F_{ij} = f_j(\mathbf{x}_i)$ and Y be a column vector of length n containing the target values for each training instance. By making the derivatives above equal to zero, we arrive in the following matrix formulation:

$$(F^t F)\begin{pmatrix}\beta_1^* \\ \beta_2^*\end{pmatrix} = F^t Y \Rightarrow \begin{pmatrix}\beta_1^* \\ \beta_2^*\end{pmatrix} = (F^t F)^{-1} F^t Y$$

Therefore, we only need to show that (β_1^*, β_2^*) consists of a minimization (and not maximization) point. For that, we will need second order derivatives of SSE.

$$\frac{\delta^2 SSE}{\delta \beta_1^2} = 2\sum_{i=1}^{n} f_1(\mathbf{x}_i)^2$$
$$\frac{\delta^2 SSE}{\delta \beta_2^2} = 2\sum_{i=1}^{n} f_2(\mathbf{x}_i)^2$$
$$\frac{\delta^2 SSE}{\delta \beta_1 \delta \beta_2} = 2\sum_{i=1}^{n} f_1(\mathbf{x}_i) \cdot f_2(\mathbf{x}_i)$$

Since

$$\begin{aligned} D &= \frac{\delta^2 SSE}{\delta \beta_1^2}(\beta_1^*, \beta_2^*) \cdot \frac{\delta^2 SSE}{\delta \beta_2^2}(\beta_1^*, \beta_2^*) - [\frac{\delta^2 SSE}{\delta \beta_1 \delta \beta_2}(\beta_1^*, \beta_2^*)]^2 \\ &= 4\sum_{i=1}^{n} f_1(\mathbf{x}_i)^2 \sum_{i=1}^{n} f_2(\mathbf{x}_i)^2 - 4(\sum_{i=1}^{n} f_1(\mathbf{x}_i) \cdot f_2(\mathbf{x}_i))^2 \\ &> 0 \end{aligned}$$

we can conclude that (β_1^*, β_2^*) is indeed a minimization point.

The operator proposed in this section is also non-degenerative regarding training error, since we are optimizing parameters over a set that includes $(\beta_1 = 1, \beta_2 = 0)$ and $(\beta_1 = 0, \beta_2 = 1)$.

4 Experimental Results

The experiments reported in this section were performed to evaluate the role of geometric semantic crossover on a large set of datasets with distinct properties.

The first experiment (reported in Section 4.1) was designed to show that, different from traditional crossover operators, semantic geometric operators have nothing to do with a macro mutation [1], as they guarantee their offspring will be semantically intermediate to its parents, and they also outperform strictly mutation-based methods. The second experiment, showed in Section 4.2, compares variations of convex semantic crossover operators using different distances and optimized coefficients.

For all experiments, we used 20 datasets with distinct properties. Eight of these dataset are synthetic and were recommended in [7], the others being real-world datasets. For each real-world dataset, we did a 5-fold cross-validation with 10 replications, making 50 replications. For the synthetic ones (except *keijzer-6* and *keijzer-7*), we generated 5 samples and, for each sample, applied the algorithms 10 times, again making 50 replications. For *keijzer-6* and *keijzer-7*, the test set is fixed, so we simply replicated the executions 50 times.

For all methods, a preliminary parameter study was performed, and we defined the population size equal to 1,000 individuals, evolved for 2,000 generations to ensure convergence. The operator set included basic arithmetic operations: addition, subtraction, multiplication and protected division. The terminal set included the variables of the problem and constant values in the interval $[-1, 1]$. The tournament size was defined as 10. Finally, both probabilities of crossover and mutation were defined as 0.5.

It is important to point out that, in all Semantic Genetic Programming (SGP) versions,the semantic mutation operator used was implemented as in [5]. This is because this mutation operator presented better results in preliminary tests than the mutation operator proposed in [8]. We believe this difference is due to the fact that the semantic impact of the latter is still unbounded, which is not true for the mutation operator used in this work. The mutation step required by the mutation operator was defined as 10% of the standard deviation of the training data. For each algorithm, the following variations of semantic crossover were tested:

- **SGXE:** Euclidean-based geometric semantic crossover with random crossover mask;
- **SGXM:** Manhattan-based geometric semantic crossover with random crossover mask;
- **SGXE-C:** Optimized convex Euclidean-based geometric semantic crossover operator (as in Equation 3);
- **SGXE-L:** Optimized non-convex Euclidean-based geometric semantic crossover operator (as in Equation 6);
- **SGP-Mut:** SGP with crossover rate equal to 0.

All statistical tests considered a confidence level of 95%. Whenever multiple tests were necessary, a Bonferroni correction was applied to assure that the required confidence level was maintained.

Table 1. Median RMSEs (and IQR) obtained after 2,000 generations for each dataset, considering 50 replications.

Dataset	SGP-Mut		SGXM	
	median	IQR	median	IQR
airfoil	2.28	0.16	2.65	0.91
bioavailability	33.09	3.39	30.63	4.48
concrete	5.61	0.80	4.92	0.50
cpu	37.22	9.82	30.09	12.42
energyCooling	1.34	0.18	1.19	0.16
energyHeating	0.82	0.14	0.63	0.16
forestfires	59.87	40.37	52.55	46.07
keijzer-5	0.31	0.07	0.08	0.17
keijzer-6	0.44	0.33	0.30	0.40
keijzer-7	0.05	0.02	0.03	0.10
korns-1	207.74	47.61	106.61	138.02
korns-2	476.71	60.66	687.22	2914.61
korns-12	1.13	0.11	1.02	0.01
ppb	32.38	6.27	29.22	4.87
tower	25.46	0.71	19.13	1.01
vlad-1	0.66	2.57	2.00	5.61
vlad-4	0.40	0.15	0.21	1.77
wine-red	0.65	0.07	0.59	0.05
wine-white	0.72	0.02	0.67	0.01
yacht	1.51	0.39	1.31	0.47

4.1 Measuring the Impact of Geometric Semantic Crossover

This section compares SGP-Mut and SGXM, the most recent geometric semantic crossover operator proposed in literature. The reason for this comparison is that, as already stated, the (semantic) fitness landscape induced by the fitness function, the distance function and the set of solutions is quite simple: since the fitness function is actually the distance function, we have that the landscape is unimodal. This may indicate that methods based on local decisions are sufficient for achieving good solutions, and hence crossover might have similar effects to mutation.

Tables 1 shows the median results of Root Mean Squared Error (RMSE) followed by the Interquartile Range (IQR) obtained by both configurations on the 20 datasets used as benchmarks. The results leave no doubts that SGXM performs better than SGP-Mut. In half of the datasets, SGXM was statistically better than SGP-Mut, while being statistically worse in only two.

These results indicate that the geometric semantic crossover operator in SGP is indeed beneficial and has a different effect from using mutation-only based methods, despite the simplicity of the fitness landscape. The poor results obtained by the mutation operator might be explained by the fact that the operator has access only to the *observed* fitness landscape (training set), and not the complete one.

4.2 Comparing Different Distance Functions and Crossover Masks

Given that geometric semantic crossover is indeed necessary, we now turn our attention to the impact of the crossover operator distance function, as well as the performance of the two operators proposed here, which work by optimizing

Table 2. Median training RMSEs (and IQR) obtained after 2,000 generations for each dataset, considering 50 replications.

Dataset	SGXE		SGXM		SGXE-C		SGXE-L	
	median	IQR	median	IQR	median	IQR	median	IQR
airfoil	1.82	0.21	1.89	0.46	1.68	0.16	1.66	0.14
bioavailability	5.12	1.17	4.67	1.44	4.77	1.43	6.12	1.01
concrete	3.19	0.14	2.61	0.15	2.96	0.15	4.26	0.17
cpu	3.49	0.61	2.03	0.38	1.80	0.38	7.65	1.57
energyCooling	0.89	0.05	0.70	0.06	0.81	0.05	1.14	0.05
energyHeating	0.42	0.03	0.36	0.06	0.37	0.04	0.65	0.08
forestfires	21.80	3.22	14.43	2.31	18.86	2.98	22.12	4.25
keijzer-5	0.03	0.01	0.03	0.01	0.03	0.01	0.05	0.01
keijzer-6	0.01	0.00	0.00	0.00	0.00	0.00	0.01	0.00
keijzer-7	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
korns-1	86.67	17.98	78.26	18.67	82.79	26.64	7.27	10.62
korns-2	301.20	437.22	131.13	140.62	228.16	320.80	220.05	268.34
korns-12	0.93	0.01	0.87	0.01	0.91	0.01	0.95	0.01
ppb	0.25	0.03	0.09	0.02	0.08	0.02	1.25	0.17
tower	17.45	0.29	16.58	0.57	16.93	0.29	21.20	0.84
vlad-1	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00
vlad-4	0.03	0.00	0.02	0.00	0.03	0.00	0.05	0.00
wine-red	0.36	0.01	0.29	0.01	0.33	0.01	0.38	0.01
wine-white	0.57	0.00	0.53	0.00	0.55	0.00	0.60	0.01
yacht	0.48	0.07	0.38	0.05	0.39	0.07	0.78	0.17

the crossover masks. Therefore, this section compares SGXE, SGXM, SGXE-C and SGXE-L using the same 20 datasets listed in the last section.

Table 2 shows the final training error obtained by each operator on each dataset, while Table 4 shows the number of datasets where the operator positioned in the line beats the operator positioned in the column in the training set, according to Wilcoxon test. As expected, in most datasets, SGXM and SGXE-C are statistically better than SGXE. SGXE-L was consistently worse than SGXE-C (and all other operators) despite considering a larger set of possible combinations. Another interesting point is that SGXM is better than SGXE-C in 11 datasets and worse in 4 datasets, leading to the conclusion that the optimization is easier when considering the Manhattan distance than when considering the Euclidean distance, despite optimized coefficients.

Tables 3 and 4 show the same information as the two previous tables, now considering test error. From these tables, we observe that SGXE-L is by far the worst operator: it lost in 15 datasets and won in only 2. We attribute these results to overfitting: for instance, on the *cpu* dataset, the training error achieved by SGXE-L was 7.65, while the test error was 105.85. As expected, the removal of the convex property increased risk of overfitting, as we eliminated any bounds on the test error.

On the test set, SGXE, SGXM and SGXE-C achieved similar results. These results indicate that both SGXE-C and SGXM may lead to overfitting, since the good results obtained in the training set were not replicated in the test set. In fact, this situation is even worse for SGXM, which obtained results slightly worse than SGXE-C in the test set despite winning in the majority of datasets when considering training error.

Table 3. Median test RMSEs (and IQR) obtained after 2000 generations for each datasets, considering 50 replications.

Dataset	SGXE		SGXM		SGXE-C		SGXE-L	
	median	IQR	median	IQR	median	IQR	median	IQR
airfoil	2.28	0.29	2.65	0.91	2.21	0.27	2.17	0.31
bioavailability	31.06	3.85	30.63	4.48	31.49	4.48	38.06	12.00
concrete	4.82	0.44	4.92	0.50	4.68	0.50	6.19	0.91
cpu	28.95	11.95	30.09	12.42	28.04	15.74	136.51	128.12
energyCooling	1.21	0.14	1.19	0.16	1.20	0.12	1.76	1.12
energyHeating	0.59	0.09	0.63	0.16	0.55	0.07	1.51	2.06
forestfires	51.55	46.68	52.55	46.07	53.00	45.81	105.85	39.83
keijzer-5	0.07	0.19	0.08	0.17	0.09	0.20	0.19	0.35
keijzer-6	0.61	0.39	0.30	0.40	0.50	0.56	0.41	0.25
keijzer-7	0.03	0.03	0.03	0.10	0.02	0.03	8.56	15.89
korns-1	104.13	160.15	106.61	138.02	102.00	147.86	8.75	47.95
korns-2	702.62	3124.15	687.22	2914.61	930.37	3031.87	2263.33	6511.75
korns-12	1.03	0.01	1.02	0.01	1.04	0.01	1.02	0.01
ppb	29.21	5.49	29.38	5.67	30.15	4.97	44.64	54.36
tower	19.36	0.71	19.13	1.01	19.29	0.89	24.36	2.45
vlad-1	1.42	3.76	2.00	5.61	2.91	7.09	6.61	27.17
vlad-4	0.09	0.43	0.21	1.77	0.34	0.60	1.85	5.49
wine-red	0.60	0.05	0.59	0.05	0.60	0.06	0.65	0.06
wine-white	0.68	0.01	0.67	0.01	0.68	0.01	0.71	0.02
yacht	1.15	0.25	1.31	0.47	1.16	0.34	2.01	0.99

Therefore, we conclude that the distance function and the use of optimized coefficients reduce training error drastically, but does not have the same impact when considering test error. We also observe that the flexibility gained by using linear combinations instead of convex combinations in the crossover operator was not worth the loss of the convex property, which exhibited interesting results regarding control of overfitting. Based on these results, we suggest the use of SGXE, as it achieved similar results of more complex operators.

5 Conclusions

This work performed an extensive evaluation of the effects of the use of different distance functions when defining the semantic distance between two symbolic regression functions. It also proposed two new versions of the traditional operators by optimizing the coefficients involved in the convex and linear combinations of solutions.

Experimental results indicated that the use of a Euclidean or Manhattan distance function for semantic geometric crossover has little impact on test error,

Table 4. Number of datasets where the operator presented in the line was statistically better than the operator presented in the column according to a Wilcoxon test considering training and test error.

	Training Error				Test Error			
	SGXE	SGXM	SGXE-C	SGXE-L	SGXE	SGXM	SGXE-C	SGXE-L
SGXE	0	0	0	14	0	2	1	15
SGXM	19	0	11	17	2	0	2	15
SGXE-C	18	4	0	16	1	6	0	15
SGXE-L	3	3	1	0	2	2	2	0

even when using our proposed versions with optimized coefficients. The use of linear combinations instead of convex combinations led to poor results, mainly attributed to the lack of any property regarding generalization. For SGP practitioners, we suggest the use of SGXE, as it achieved similar results to those obtained by more complex operators.

References

1. Peter J Angeline. Subtree crossover: Building block engine or macromutation. *Genetic Programming*, 97:9–17, 1997.
2. Ignacio Arnaldo, Krzysztof Krawiec, and Una-May O'Reilly. Multiple regression genetic programming. In *Proceedings of the 2014 conference on Genetic and evolutionary computation*, pages 879–886. ACM, 2014.
3. Lawrence Beadle and Colin G Johnson. Semantically driven crossover in genetic programming. In *IEEE World Congress on Computational Intelligence*, pages 111–116, 2008.
4. Lawrence Beadle and Colin G Johnson. Semantic analysis of program initialisation in genetic programming. *Genetic Programming and Evolvable Machines*, 10(3):307–337, 2009.
5. Mauro Castelli, Sara Silva, and Leonardo Vanneschi. A C++ framework for geometric semantic genetic programming. *Genetic Programming and Evolvable Machines*, pages 1–9, 2014.
6. Krzysztof Krawiec and Pawel Lichocki. Approximating geometric crossover in semantic space. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 987–994. ACM, 2009.
7. James McDermott, David R White, Sean Luke, Luca Manzoni, Mauro Castelli, Leonardo Vanneschi, Wojciech Jaskowski, Krzysztof Krawiec, Robin Harper, Kenneth De Jong, et al. Genetic programming needs better benchmarks. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference*, pages 791–798. ACM, 2012.
8. Alberto Moraglio, Krzysztof Krawiec, and Colin G Johnson. Geometric semantic genetic programming. In *Parallel Problem Solving from Nature-PPSN XII*, pages 21–31. Springer, 2012.
9. Nguyen Quang Uy, Nguyen Xuan Hoai, Michael ONeill, Robert I McKay, and Edgar Galván-López. Semantically-based crossover in genetic programming: application to real-valued symbolic regression. *Genetic Programming and Evolvable Machines*, 12(2):91–119, 2011.
10. Nguyen Quang Uy, Michael O'Neill, Nguyen Xuan Hoai, Bob Mckay, and Edgar Galván-López. Semantic similarity based crossover in gp: The case for real-valued function regression. In *Artifical Evolution*, pages 170–181. Springer, 2010.
11. Leonardo Vanneschi, Mauro Castelli, Luca Manzoni, and Sara Silva. *A new implementation of geometric semantic GP and its application to problems in pharmacokinetics*. Springer, 2013.
12. Leonardo Vanneschi, Mauro Castelli, and Sara Silva. A survey of semantic methods in genetic programming. *Genetic Programming and Evolvable Machines*, 15(2):195–214, 2014.
13. Leonardo Vanneschi, Sara Silva, Mauro Castelli, and Luca Manzoni. Geometric semantic genetic programming for real life applications. In *Genetic Programming Theory and Practice XI*, pages 191–209. Springer, 2014.