Exploring the Integration of Cellular Structures in Genetic Programming-Based Methods

Luigi Rovito[1(✉)], Lorenzo Bonin[1], Davide Farinati[2], Leonardo Vanneschi[2], Luca Manzoni[1], Andrea De Lorenzo[1], and Gloria Pietropolli[1]

[1] University of Trieste, 34127 Trieste, Italy
luigi.rovito@phd.units.it
[2] NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, 1070-312 Lisbon, Portugal

Abstract. The introduction of a Cellular Automata (CA)-like structure on the population of Evolutionary Algorithms (EAs) has been verified to be a method to improve solutions quality. However, the study of CA-like structures for Genetic Programming (GP) has been, so far, limited. In this work, we focus on the effect of introducing these structures on Geometric Semantic variants of GP, focusing on the well-known Geometric Semantic GP (GSGP) and its recently introduced variant SLIM-GSGP, which emphasizes producing smaller and more interpretable individuals. Here we provide guidance on how CA-like structures can impact the quality and size of the solutions for GSGP and SLIM-GSGP, giving a clear understanding of the trade-offs involved in applying these methods.

Keywords: Evolutionary Computation · Evolutionary Algorithms · Genetic Programming · Geometric Semantic Genetic Programming · Cellular Automata · Symbolic Regression

1 Introduction

Cellular Automata (CA) [19,59,69] is classical nature-inspired computing model consisting of a n-dimensional grid of cells, each in one of a finite set of possible states. Each cell is confined within a neighborhood, and at each discrete time step, cell states are all synchronously updated based on a uniform local rule.

The intrinsic properties of CAs make this model a promising substrate for evolution in Evolutionary Algorithms (EAs), as modeling the population as a toroidal structure restricts interactions to individuals within the same neighborhoods [3,4,68]. This structure can offer several advantages, depending on the

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-89991-1_8.

B. Xue et al. (Eds.): EuroGP 2025, LNCS 15609, pp. 120–138, 2025.
https://doi.org/10.1007/978-3-031-89991-1_8

algorithm. By constraining interactions, it helps prevent local-optima stagnation and encourages the population to explore diverse areas of the search space, often leading to the discovery of better solutions [3,10,28,52,58,66,90].

While combining CA with EAs [40,62] is a promising approach and despite Genetic Programming (GP) is an effective algorithm for Symbolic Regression (SR), its integration with spatial structures like CA is still underexplored. Existing studies on CA-inspired GP largely focus on problems other than SR [26,73,74] or on parallel implementations aimed primarily at improving computational performance [27]. Moreover, these works have not been evaluated using modern, well-established SR benchmarks [40,86]. In addition, these applications have not focused on semantic-based variations of GP, such as Geometric Semantic Genetic Programming (GSGP) [50,77], until very recently, when a cellular structure was introduced to slow the spread of dominant individuals and mitigate premature convergence in GSGP [10]. Despite this, the issue of exponential tree growth in GSGP remains. To counteract those issues, a promising approach is one of Semantic Learning Algorithm Based on Inflate and Deflate Mutation (SLIM) [78,81]. In this recent GSGP variant, a modified Geometric Semantic Mutation (GSM) operator is employed to generate smaller offspring.

Motivated by the findings in [10], we investigate integrating cellular structures within different GP methods. We begin with a comprehensive literature review that unifies prior work on GP methods and cellular structures (Sect. 2). Thereafter, we perform a comprehensive experimental campaign focusing on two key metrics: model performance, to assess quality and model size. Smaller models, as shown in [78,81], lead to more interpretable formulae and facilitate analysis with explainability techniques. Additionally, compact models are faster in computation, require less memory, and are easier to deploy and monitor in real-world applications.

In our experiments, we integrate cellular structures with GP, GSGP, and SLIM, chosen for their distinct strengths and limitations across the two metrics of interest. To the best of our knowledge, this is the first analysis to compare these popular GP methods in terms of accuracy and model size with the addition of cellular structures-known to influence evolutionary dynamics. Our goal is to provide a definitive understanding of how cellular toroidal grids impact both the population and outcomes of key GP algorithms and to clarify the trade-offs between accuracy and model size achievable by combining GP methods with CA-inspired structures.

2 Related Works

Previous research has explored the introduction of spatial structures in EAs [3,4,68]. This strategy limits interactions to smaller subsets of the population, called neighborhoods, typically organized in a grid. In Genetic Algorithms (GAs) [30], spatial structuring is implemented through a Cellular Genetic Algorithm (cGA) [3,4,68] with studies [3,68] analyzing how neighborhood configurations impact the performance. Alba et al. [3] introduced a dynamic cGA adjusting exploration-exploitation during evolution. For multi-objective optimization,

Murata et al. [52] developed C-MOGA, a GA with cellular structuring for local selection. Later, Nebro et al. [58] presented MOCell, inspired by traditional cGA. Mariot et al. [45] applied GA and GP to design orthogonal Latin squares using CAs. However, other EAs also used a cellular structure to manage the spread of solutions within the population [2,71,90]. Hence, it is generally possible to apply a cellular structure to the population regardless of the employed algorithm.

GP [36] is an EA that, by evolving computer programs, showed its flexibility and effectiveness in tackling several problems [1,9,21,33,38,39], especially because the learning process and the solution representation potentially enable the discovery of interpretable models [12,25,32,42,49,54–56,83,84]. Folino et al. [27] introduced a scalable parallel implementation of GP using a cellular structure, outperforming both traditional GP and the island model [46], where evolutionary runs occur on separate population subsets. They also proposed a parallel cellular-based implementation of GP to address classification problems [26]. Later, Takac proposed a cellular-based GP method for classification [74] as well as data mining [73]. Additionally, studies by [85] and [23] demonstrated that combining spatial population structure with local elitist replacement effectively reduces bloat (unnecessary growth in tree size) in GP, while maintaining performance.

To preserve biodiversity, speciation can be applied by dividing the population into distinct niches of individuals with comparable structural traits, forcing crossover only between individuals of the same niche. Della Cioppa et al. [22] introduced an adaptive species discovery strategy to address the limitations of traditional niching methods, which often require prior knowledge of the fitness landscape. Building on the NEAT algorithm [72], Trujillo et al. [76] applied speciation to control program growth in GP through neat-GP, which promotes complexity only when necessary. Juarez et al. [34] improved neat-GP by incorporating a local search operator. Cussat et al. [20] introduced a network distance metric to speciate populations, promoting diversity and maintaining smaller individuals. Martins et al. [48] combined GAs with speciation and grid pattern recognition to reduce investment risks and boost profits. Wickman et al. [87] applied speciation in Reinforcement Learning (RL) to evolve diverse policies, while Pietropolli et al. [66] proposed using substrates with barriers to slow genetic spread and enhance diversity.

Several studies have examined the effects of selection pressure and sampling strategies both in vanilla EAs [29,88] and cellular-based EAs [28,70]. In cellular-based EAs, spatial structure plays a critical role in controlling takeover time, i.e., the number of generations needed for a dominant individual to spread across the population. The higher the takeover time, the higher the diversity [28].

GSGP [50,77], which outperformed GP in various tasks [17,18,40,50,62, 79,80,82], is affected by several drawbacks mainly regarding the presence of local optima where evolution may stagnate and the premature convergence towards dominating individuals (low takeover time) [63]. To this end, Bonin et al. [10] implemented Cellular Geometric Semantic Genetic Programming (cGSGP), which imposes a cellular toroidal structure to the GSGP population,

showing that this variant can outperform GSGP and improve the diversity in the early stage of the evolution by increasing the takeover time.

Even though these efforts mitigate premature convergence, GSGP still suffers from exponential growth of the trees across the generations. Motivated by the large research literature regarding the enhancement of this technique [14,24,53,57,64,65] and the availability of efficient implementations [15,75], many research works tried to control bloating and reduce the trees exponential size [15,16,24,35,47,64,79]. However, these methods still rely on Geometric Semantic Operators (GSOs) that produce offspring that are larger than their parents [7].

Recently, Vanneschi et al. [78] presented the SLIM, a variant of GSGP that employs a specialized GSM that produces offspring smaller than their parents without affecting performances, thus revolutionizing the way GSGP and its solutions are created and employed.

The variety of GP and GSGP variants and the existence of CA-inspired methods makes it challenging to understand the impact of combining these techniques. In this paper, we explore the integration of a cellular toroidal structure over the population of the main GP algorithms, i.e., GSGP and SLIM. We take these tree GP algorithms (on their standard version for a clear comparison) as they have different strengths and limitations, and we perform an analysis on different metrics to provide, once for all, a clear understanding of their trade-off levels along with the impact of the toroidal grid.

3 Methods

In this section, we briefly describe the two variants [50,81] of GP [36,41] we will investigate alongside vanilla GP. We also describe the cellular-based selection strategy adopted in [10].

3.1 Geometric Semantic Genetic Programming

GP individuals are typically represented as computer programs, visualized as trees, that map inputs to outputs. Given an individual P and inputs $X = \{x_1,\ x_2,\ \ldots,\ x_n\}$, the predictions $P(X) = \{P(x_1),\ P(x_2),\ \ldots,\ P(x_n)\}$ define its semantics. Traditional GP operators modify the syntactic structure, but their effect on semantics is unpredictable, often complicating the search process [60].

Moraglio et al. [50] proposed replacing traditional operators with Geometric Semantic Crossover (GSC) and GSM, which reflects structural changes in an individual's semantics, creating a unimodal fitness landscape. Recent studies [13] found that using only GSM performs similarly or better than combining both operators.

In GSM, given a parent function $T : \mathbb{R}^d \rightarrow \mathbb{R}$, mutation is defined as $T_m = T + ms \cdot (T_{R1} - T_{R2})$, where ms is the mutation step, and T_{R1} and T_{R2} are random programs with a sigmoid function to restrict values to $[0, 1]$. This mutation produces offspring whose semantics lie within a sphere of radius

ms around T. While GSO creates a unimodal fitness landscape, it also increases individual size, causing rapid solution growth and making GSGP a black-box model [15].

3.2 Semantic Learning Algorithm Based on Inflate and Deflate Mutation

Recently, Vanneschi et al. [81] introduced a *deflate* GSM that generates smaller offspring. The definition and validity of Deflate Geometric Semantic Mutation (DGSM) are based on two key observations: first, GSM can be rewritten as $\text{GSM}(T) = T + ms \cdot (T_{R1} - T_{R2}) = T - ms \cdot (T_{R2} - T_{R1})$; second, the random trees T_{R1} and T_{R2} are interchangeable, as they are independently sampled from the same distribution. Thus, the GSM can be written as $\text{GSM}(T) = T - ms \cdot (T_{R1} - T_{R2})$. For instance, let T be an individual to which GSM is applied three consecutive times. We obtain:

$$T_M = \text{GSM}^3(T) = T + ms \cdot (T_{R1} - T_{R2}) + ms \cdot (T_{R3} - T_{R4}) + ms \cdot (T_{R5} - T_{R6})$$

If we apply GSM again by using subtraction, we obtain:

$$T_{M'} = T + ms \cdot (T_{R1} - T_{R2}) + ms \cdot (T_{R3} - T_{R4}) + ms \cdot (T_{R5} - T_{R6}) - ms \cdot (T_{R7} - T_{R8})$$

We recall that reusing random trees is widely adopted [79]. For example, we could use T_{R3} and T_{R4} instead of T_{R7} and T_{R8}:

$$T' = T + ms \cdot (T_{R1} - T_{R2}) \cancel{+ ms \cdot (T_{R3} - T_{R4})} + ms \cdot (T_{R5} - T_{R6}) \cancel{- ms \cdot (T_{R3} - T_{R4})}$$

After this simplification, the individual has a smaller genotype. This alternating inflate and deflate approach forms the basis of SLIM. A hyper-parameter controls the likelihood of applying deflate (if zero, SLIM resembles GSGP)

3.3 New Ways of Defining a Geometric Semantic Mutation

In recent works [7,81], the authors introduced three methods to define mutation values within $[-ms, ms]$ and two strategies for perturbing an individual. Given that T_R, T_{R1}, and T_{R2} are random programs with arbitrary output and let S be the sigmoid function, the functions are: (i) $\text{SIG2} = ms \cdot (S(T_{R1}) - S(T_{R2}))$; (ii) $\text{ABS} = ms \cdot (1 - \frac{2}{1+|T_R|})$; (iii) $\text{SIG1} = ms \cdot (2 \cdot S(T_R) - 1)$.

To introduce minor perturbations, function values can either be added/subtracted (SLIM^+) or scaled by factors around one (SLIM^*) w.r.t the parent tree.

3.4 Cellular Methods

In CA-inspired algorithms, a spatial structure is imposed. As in [10], individuals are arranged according to a toroidal grid, with each individual belonging to a specific cell, randomly chosen at initialization, which belongs to a specific neighborhood.

The neighborhood of radius r for a given individual is defined by a hypercube with side $2r + 1$, corresponding to the Moore neighborhood centered on that individual's cell. Each individual belongs to its own neighborhood. The toroidal configuration ensures all cells have complete neighborhoods. The n-dimensional toroidal grid with radius $r \in \mathbb{N}$ is denoted as $\mathcal{T}_r^n$.

We employ this structure along with the selection strategy in [10]. Especially, given an individual T_i in a cell i in the grid, and its neighborhood $\mathcal{N}_i$, T_i is replaced by a new tree derived from others in $\mathcal{N}_i$. Selection is limited to each cell neighborhood: for each cell i, the two individuals chosen for crossover are selected from within the same neighborhood $\mathcal{N}_i$. If no crossover is performed, T_i is replaced with another (eventually mutated) individual in $\mathcal{N}_i$.

This method (TRS_p) is a tournament-based selection within local neighborhoods, where the selection pressure is defined by a value $p \in (0, 1]$. A subset $\mathcal{S}$ of $\mathcal{N}_i$ is selected by iterating across the elements of $\mathcal{N}_i$ and inserting them into $\mathcal{S}$ with probability p. Then, a rank-based selection is applied on $\mathcal{S}$: if no crossover is performed, then the new individual in cell i will be the best one in $\mathcal{S}$, while, if crossover is performed, then it will be between the two best individuals in $\mathcal{S}$. Finally, the new individual is eventually mutated.

4 Experimental Methodology

We set a population size of 100 that is evolved for 1000 generations with elitism enforced and tournament selection with a tournament size of 4 in case of standard methods ($\mathcal{T}^0$). We use a 10×10 bi-dimensional toroidal grid in the case of cellular methods. We test cellular methods with $p = 1$ and radius between 2 ($\mathcal{T}_2^2$) and 3 ($\mathcal{T}_3^2$).

Trees are initialized with ramped half-and-half [36,41,67], with initial maximum depth set to 6. The function set is composed by $+, -, \times, \div^*$ where * indicates that the division is protected (if the denominator is zero, then one is returned). The terminal set is composed of the variables of the problem. No ephemeral constants are explicitly added to the terminal set, as in [78,81].

GP performs sub-tree crossover with probability 0.8 and sub-tree mutation with probability 0.2, GSGP only performs GSM with probability 1.0. In SLIM, mutation occurs with a probability of 1.0, as the standard implementation does not include crossover, with inflate and deflate probabilities set to 0.3 and 0.7, respectively, following the original paper [78]. The mutation step ms is uniformly sampled at random in $[0, 1]$ for each mutation event in both GSGP and SLIM.

Regarding the specific SLIM algorithms, we test $\text{SLIM}^+_{\text{ABS}}$, $\text{SLIM}^+_{\text{SIG1}}$, $\text{SLIM}^+_{\text{SIG2}}$, $\text{SLIM}^*_{\text{ABS}}$, $\text{SLIM}^*_{\text{SIG1}}$, and $\text{SLIM}^*_{\text{SIG2}}$, which are, currently, all the SLIM versions available.

We adopt six datasets that are commonly used in GP tasks [14,79,82,86]: Airfoil (ARF) [11] with 1502 records and 5 variables, Concrete (CNC) [17] with 1029 records and 8 variables, Slump (SLM) [89] with 102 records and 9 variables, Yacht (YCH) [61] with 307 records and 6 variables, Parkinson (PRK) [18] with 5875 records and 18 variables, and QSAR Aquatic Toxicity (QSR) [8] with

546 records and 8 variables. Each dataset is randomly partitioned in 30 train-test splits with ratio 7:3. For each method, dataset, and combination of hyper-parameters, we perform 30 repetitions (one for each split).

In our experiments, we analyze three key measures: (i) the Root Mean Squared Error (RMSE) [6], which evaluates the qualitative performance of the models; (ii) the size ($\log_{10}(\ell)$) of the models (the 10-base logarithm of the number of nodes Nodes), which serves as a proxy for the interpretability, as outlined in [78], and both memory occupation and computational complexity; (iii) the Global Moran's I (I) [43,51], which is a spatial autocorrelation measure [5] that represents a measure of diversity for neighborhood-based populations.

I assesses the similarity (or dissimilarity) of values located in neighboring locations. Provided that an individual is represented by its semantics, we have:

$$I = \frac{M}{W} \frac{\sum_{i=1}^{M} \sum_{j=1}^{M} \mathbf{w}_{ij}(\mathbf{x}^i - \bar{\mathbf{x}})(\mathbf{x}^j - \bar{\mathbf{x}})}{\sum_{i=1}^{M} (\mathbf{x}^i - \bar{\mathbf{x}})^2}$$

where M is the population size, $\mathbf{w}_{ij} \in \mathbb{R}$ is the value located at the i-th row and the j-th column of the matrix $\mathbf{w} \in \mathbb{R}^{M \times M}$, $\mathbf{w}_{ii} = 0$ for $1 \leq i \leq M$, $W = \sum_{i=1}^{M} \sum_{j=1}^{M} \mathbf{w}_{ij}$, $\mathbf{x}^i$ is the i-th individual in the population as a semantic vector, and $\bar{\mathbf{x}}$ is the mean of all the semantic vectors of the individuals in the population. In our case, $\mathbf{w}_{ij} = 1$ if $i \neq j$ and the j-th individual belongs to the neighborhood of the i-th individual (in a non-cellular method the entire population is a single neighborhood), otherwise $\mathbf{w}_{ij} = 0$.

I ranges in the interval $[-1, 1]$, where values close to 1 indicate positive spatial autocorrelation (similar semantic content is spatially clustered in the population) and values close to -1 indicate negative autocorrelation. Values close to 0 depict a spatial pattern not different from a random one. On the other hand, a strictly positive value of I indicates that each neighborhood tends to be a cluster containing similar individuals that are, at the same time, dissimilar from individuals in other neighborhoods, meaning that the population is spatially arranged in different search space areas and, hence, diversity is better enforced by the toroidal grid.

For the sake of simplicity, we use "*algorithm*" to refer to the different GP variants (GSGP and SLIM), while we use "*method*" to refer to the different variants of cellular structure, including the non-cellular standard structure of the non-cellular baseline. Our Python code `cslim` is publicly available online.[1]

5 Results and Discussion

In this section, we present the results of our experimental campaign and discuss the main findings.

[1] https://github.com/lurovi/cslim.

5.1 Statistical Analysis and Convergence Rates

Table 1. Table with the median of RMSE on the test set and $\log_{10}(\ell)$ of the best individual for all the different methods.

		RMSE						$\log_{10}(\ell)$					
		ARF	CNC	SLM	YCH	PRK	QSR	ARF	CNC	SLM	YCH	PRK	QSR
GP	$\mathcal{T}^0$	**22.67**	**9.54**	**5.28**	**2.80***	**10.41***	**1.45**	3.04	2.90	**2.55**	2.70	2.65	2.83
	$\mathcal{T}_2^2$	25.86	10.27*	5.56	3.55	10.72	1.47	2.91	2.73*	2.56	2.50	**2.28***	2.39*
	$\mathcal{T}_3^2$	25.18	11.93	6.33	4.08	10.67	1.50	**2.78**	**2.66***	2.58	**2.47**	2.45*	**2.33***
GSGP	$\mathcal{T}^0$	15.20	7.59	3.88	4.84	9.92	1.34	**4.26***	**4.08***	**4.03***	**4.19***	**3.89***	**3.88***
	$\mathcal{T}_2^2$	11.30*	6.96*	**3.85**	3.97*	9.75*	**1.27***	4.32	4.13	4.06	4.24	3.94	4.03
	$\mathcal{T}_3^2$	**10.15***	**6.83***	3.97	**3.74***	**9.69***	1.29*	4.33	4.15	4.08	4.26	3.96	4.06
$\text{SLIM}^+_{\text{SIG2}}$	$\mathcal{T}^0$	19.74	9.14	4.59	7.34	10.22	**1.27**	**3.94***	**3.75***	**3.63***	**3.79***	**3.60***	**3.13***
	$\mathcal{T}_2^2$	15.53*	7.66*	4.31	6.22*	9.98*	1.28	4.24	4.02	3.90	4.11	3.91	3.57
	$\mathcal{T}_3^2$	**14.53***	**7.58***	**4.22**	**5.90***	**9.93***	1.28	4.30	4.09	3.96	4.19	3.99	3.69
$\text{SLIM}^*_{\text{SIG2}}$	$\mathcal{T}^0$	20.72	14.09	**6.76**	2.71	10.05	1.36	**2.90***	**2.99***	**2.83***	**3.10***	**2.92***	**2.70***
	$\mathcal{T}_2^2$	14.24*	**12.35**	8.15	**1.84***	9.86*	**1.34**	3.55	3.52	3.47	3.52	3.48	3.17
	$\mathcal{T}_3^2$	**13.96***	13.73	7.00	2.13*	**9.84***	1.36	3.70	3.62	3.59	3.69	3.62	3.35
$\text{SLIM}^+_{\text{SIG1}}$	$\mathcal{T}^0$	17.38	8.69	4.61	6.82	10.22	1.32	**3.76***	**3.51***	**3.34***	**3.57***	**3.43***	**2.73***
	$\mathcal{T}_2^2$	14.57*	7.72*	**4.21**	5.78*	10.10*	**1.30**	4.04	3.78	3.62	3.89	3.77	3.27
	$\mathcal{T}_3^2$	**13.41***	**7.61***	4.31	**5.40***	**10.06***	1.31	4.12	3.87	3.73	3.97	3.86	3.42
$\text{SLIM}^*_{\text{SIG1}}$	$\mathcal{T}^0$	21.82	13.47	7.61	2.63	10.41	1.43	**2.84***	**2.60***	**2.47***	**2.76***	**2.90***	**2.42***
	$\mathcal{T}_2^2$	13.26*	**10.74**	6.01	2.15*	**10.25**	**1.40**	3.58	3.16	3.04	3.19	3.50	2.95
	$\mathcal{T}_3^2$	**12.54***	13.24	**5.96***	**2.05***	10.30	1.41	3.80	3.36	3.26	3.36	3.64	3.14
$\text{SLIM}^+_{\text{ABS}}$	$\mathcal{T}^0$	19.25	9.32	4.96	6.37	10.19	1.32	**3.74***	**3.58***	**3.46***	**3.68***	**3.29***	**2.74***
	$\mathcal{T}_2^2$	15.75*	8.13*	**3.86***	5.23*	10.08*	**1.29**	4.05	3.83	3.70	3.93	3.55	3.18
	$\mathcal{T}_3^2$	**14.94***	**7.91***	3.87*	**4.89***	**10.04***	**1.29**	4.13	3.91	3.78	4.00	3.62	3.30
$\text{SLIM}^*_{\text{ABS}}$	$\mathcal{T}^0$	28.78	**9.84**	6.28	2.62	10.40	1.43	**2.39***	**2.53***	**2.37***	**2.81***	**2.29***	**2.35***
	$\mathcal{T}_2^2$	23.72*	11.76	6.61	**2.17***	10.35	**1.38**	2.96	2.89	2.78	3.09	2.74	2.85
	$\mathcal{T}_3^2$	**22.73***	19.37	**5.98**	2.21*	**10.34**	1.40	3.17	3.01	2.93	3.20	2.90	3.02

Our initial analysis evaluates the impact of incorporating a cellular structure on RMSE and $\log_{10}(\ell)$, with results detailed in Table 1. The term $\mathcal{T}^0$ refers to the non-cellular standard version, while $\mathcal{T}_2^2$ and $\mathcal{T}_3^2$ indicate, respectively, cellular methods with radius 2 and 3. Statistical comparisons are conducted using the Wilcoxon-Mann-Whitney test [44] ($\alpha = 0.05$), with p-values adjusted by the Holm-Bonferroni correction [31] (these tests are performed after a preliminary Kruskal-Wallis test [37]). For each dataset and algorithm, the statistically superior method among $\mathcal{T}^0$, $\mathcal{T}_2^2$, and $\mathcal{T}_3^2$ is marked with a blue asterisk (*), and the

one with lowest median value is indicated in bold. Additionally, cellular methods that outperform their non-cellular baseline are marked with a black asterisk (*****). We further analyze performance and size by presenting the median test fitness evolution of the best individual over 30 runs in Fig. 1 and the median size evolution of the best individual in Fig. 2.[2]

Table 1 shows that cellular methods generally achieve lower errors for GSGP and SLIM$^+$, but tend to increase solution size. Figure 1 confirms that this improved performance persists throughout the evolutionary process. Exceptions include SLM and QSR, where cellular and non-cellular methods yield similar results, even though cellular methods converge faster. Another exception is YCH, where SLIM* outperforms SLIM$^+$. Regarding SLIM*, cellular methods produce lower errors less frequently than for SLIM$^+$. Nevertheless, for both GSGP and SLIM variants, they are never statistically significantly worse than their non-cellular counterparts. This property is a first important take-home message: embedding GSGP and SLIM in a cellular substrate never leads to statistically worse results.

SLIM* variants often produce smaller solutions, frequently even smaller than those from GP, aligning with findings in [78,81]. Overall, cellular methods tend to increase tree size, except in GP, where they result in smaller solutions. In particular, the longer the radius, the greater the size increment (which often correlates with more accurate models). In Fig. 2, we show the trend of the size of the best-so-far individual during the evolution, aggregating values from all datasets and repetitions.[3] The plot confirms that the trend of the size we analyzed in Table 1 holds for each generation.

5.2 Diversity

Following the intuition of [10], we employ I to analyze diversity in cellular-based methods. In Fig. 3, we track the trend of I computed on the whole population across the generations, aggregating values from all datasets and repetitions.[4] The plot shows that for GSGP and SLIM$^+$ the cellular structure effectively preserves diversity during the first half of the evolution. This suggests that the grid preserves the spatial organization of the population, slowing the spread of dominant solutions by confining them to their local neighborhoods. However, this trend holds in the first phase of the evolution, then dominant solutions spread across all the neighborhoods, and eventually I converges to zero.

Building on the previous analysis, we conclude that cellular methods can outperform their non-cellular counterparts when I-based semantic diversity is

[2] The showed lineplots depict the median of the examined measure across all repetitions, with shaded area denoting the interquartile range.

[3] Preliminary analysis indicated similar trends across datasets, so $\log_{10}(\ell)$ values are aggregated. The trend of $\log_{10}(\ell)$ for each dataset separately is shown in the supplementary materials.

[4] Preliminary analysis indicated similar trends across datasets, so I values are aggregated. The trend of I for each dataset separately is shown in the supplementary materials.

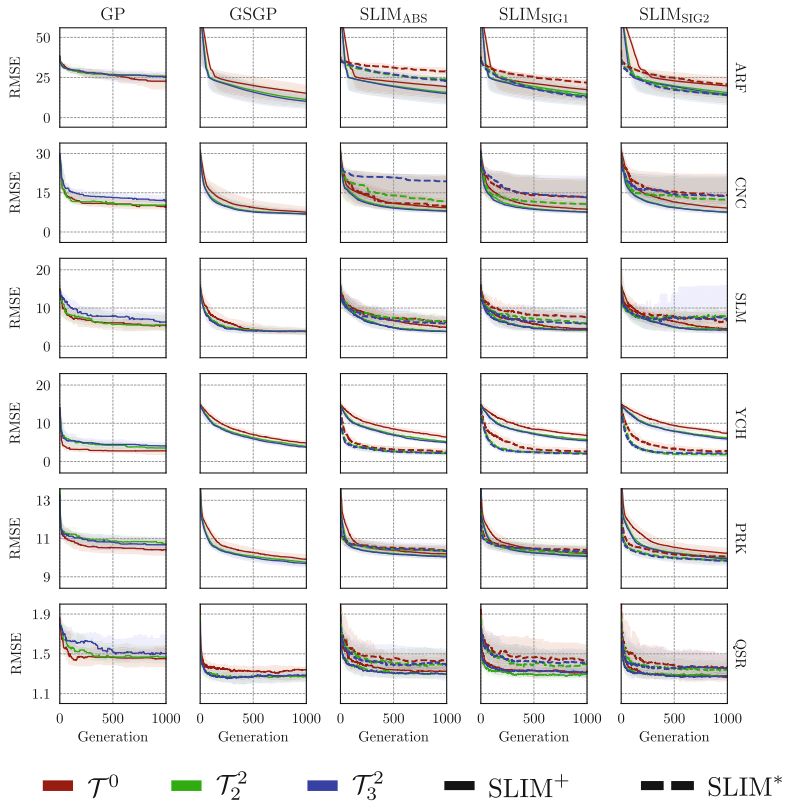

Fig. 1. Evolution of test fitness for the best individual across 30 runs.

preserved during part of the optimization process. On the other hand, cellular algorithms where I remains near zero (indicating a random spatial pattern in the population) generally fail to achieve better solutions, despite the cellular structuring. Notably, GP shows a constant trend for I across all methods, indicating that, even with a cellular structure, the spatial disposition of individuals resembles a random pattern.

To analyze spatial distribution and diversity in detail, we use heatmaps to visualize training fitness across individuals. In Fig. 4, we present the heatmaps for a single repetition of ARF for different algorithms and methods.[5]

These plots confirm that for GP and its cellular variant, the toroidal grid induces low intra-neighborhood similarity in training fitness. Individuals are randomly distributed, with good and poor solutions often placed close together, lacking spatially coherent patterns. For GSGP and its cellular variant, dominant individuals gradually spread through the population over generations. In

[5] Despite we show a single example, we provide additional heatmaps for the other datasets and for SLIM$^*_{SIG1}$ and SLIM$^+_{SIG1}$ in the supplementary materials.

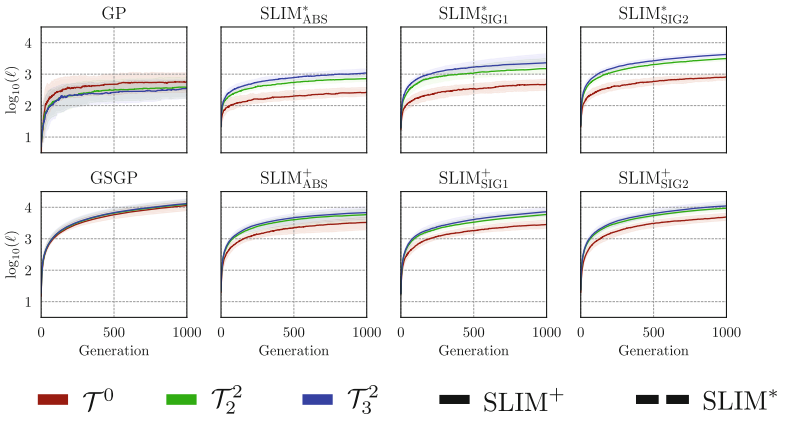

Fig. 2. Evolution of the size of the best individual across 30 runs. Each line represents the median across all datasets and repetitions for each algorithm.

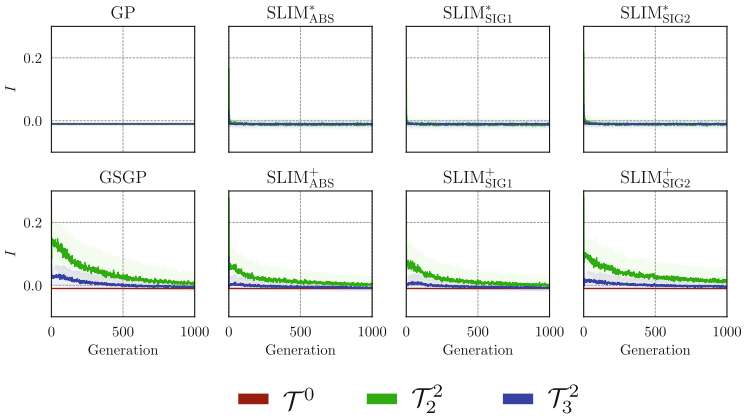

Fig. 3. Trend of the Global Moran's I calculated on the population of each algorithm and method. Each line represents the median across all datasets and repetitions for each algorithm.

the early generations of cGSGP, a distinct spatial clustering emerges, forcing the optimization to focus on different (good and bad) areas of the search space and slowing interactions with dominant solutions. This clustering enables the cellular variant to construct a solution with a lower error than that found by GSGP by the end of the process.

The behavior observed in Fig. 3 and Fig. 4 can be explained by examining how algorithms transform individuals to balance exploration and exploitation. Diversity, according to I, is better preserved in GSGP and SLIM$^+$, as these algorithms generate new solutions by adding components to existing individuals. In contrast, SLIM* generates new solutions by continuously multiplying individuals

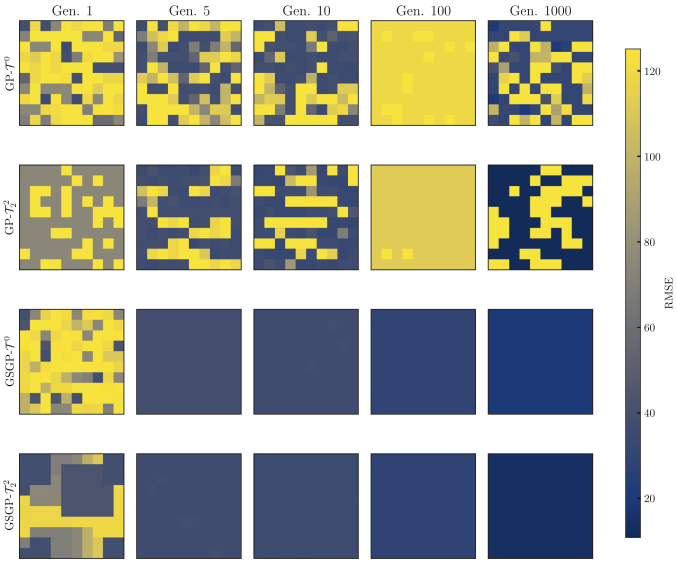

Fig. 4. Heatmaps of the individuals training fitness in a random repetition of GP and GSGP (and their cellular variants with radius 2) with ARF dataset.

with new components. This approach risks quickly invalidating solutions, as the size of the new component-whether very small or very large-can have a significant impact. Similarly, crossover in GP can drastically alter the behavior of individuals by swapping code segments between parents, leading to significant changes in structure. To sum up, disruptive operations quickly erase spatial patterns in the population, whereas conservative operations build on previous populations, thus preserving diversity and spatial patterns, which in cellular structures are often correlated since a toroidal grid may force interactions between individuals to happen only within neighborhoods and preserve diversity among different population areas. However, this holds if individuals within a neighborhood do not change too much when mutation is applied to them, otherwise, random patterns can be quickly formed despite the enforcement of a spatial arrangement.

5.3 Algorithms Pareto Front

We summarize our main findings and provide a global comparison among all the algorithms by representing them in Pareto fronts highlighting the trade-offs between model error and model size (Fig. 5). In the following plot, we aggregate the results from all datasets and repetitions for each algorithm and method.[6]

This figure indicates that cGSGP achieves the best performance, while $\text{SLIM}^*_{\text{ABS}}$ produces the smallest solutions (even smaller than GP itself). As

[6] Supplementary materials contain a version of this plot for each dataset separately.

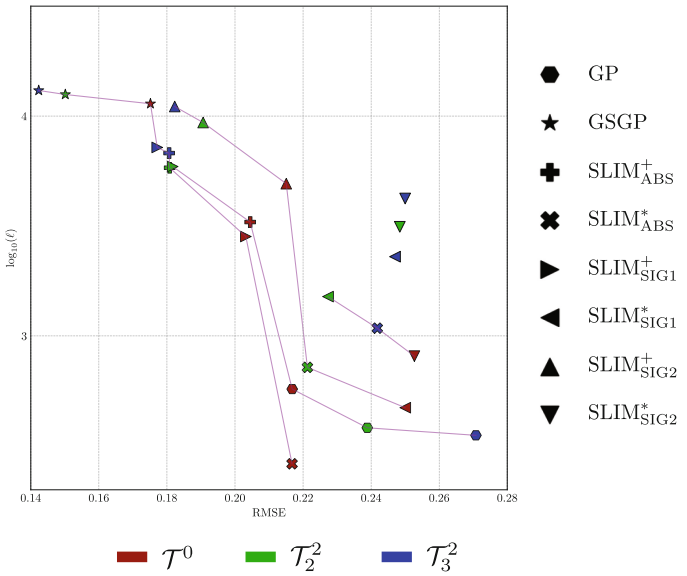

Fig. 5. Pareto fronts of the tested methods. Each Pareto front represents a set of non-dominated solutions w.r.t. to both error and size. Each method is identified by the median across all datasets and repetitions. RMSE is scaled in $[0, 1]$ by using the maximum RMSE discovered among the best solutions from all the experiments.

noted in the previous analysis, GP is the only algorithm where the cellular variant yields smaller solutions than the non-cellular one, though this comes with a slight reduction in performance quality. In general, SLIM* leads to smaller solutions while SLIM^{+} leads to more accurate models, especially when a cellular structure is employed. This overall outlines the possible trade-offs between error and size when choosing a GP algorithm for a SR problem. cGSGP and SLIM* offer the best performance and interpretability, respectively, while the cellular SLIM variants fall in-between, providing varying levels of trade-off between these two quality criteria. In some variants, cellular structure may lead to worst models as regards both accuracy and size, as in SLIM$^{*}_{ABS}$, perhaps because of the disruptive effect of the multiplication combined with mutation. A general rule of thumb can be that cellular methods are worth employing with algorithms that do not contain disruptive operations when creating new solutions, i.e., genetic operators lead to offspring that are similar to their parents.

6 Conclusion and Future Work

In this paper, we explore the effects of the integration of CA-inspired spatial structures on popular GP methods that are characterized by different strengths and limitations as regards both the qualitative performance of the solutions and their size. Taking inspiration from [10], we apply a cellular-based toroidal grid

to the population of three GP algorithms, namely, standard tree-based GP [36], GSGP [50], and SLIM [78,81], the latter being a recent and innovative variant of GSGP in which the problem of exponential tree growth is addressed by using genetic operators that lead to smaller offspring. In particular, with our study, we assess the effects of cellular structures on SLIM for the first time. Our comparative analysis highlights the different levels of trade-off that these methods exhibit both when executed in their standard version and when integrated with a cellular-based spatial structure. Especially, cellular helps in boosting the qualitative performance of GSGP and SLIM^{+} at the cost of increasing the model size. On the other hand, SLIM* provides the smallest models, with cellular that occasionally improves its performance, even though it increases the size, but not as much as cellular SLIM^{+}. We additionally assess the diversity preservation mechanisms examined via Global Moran's I, in which we see that diversity is better preserved in the early stage of the evolution when cellular is applied to GSGP and SLIM^{+} (the algorithms that benefit more from a toroidal grid as regards model accuracy). As a take-home message, we hypothesize that cellular can be beneficial for algorithms that do not perform mutations that can quickly alter the individuals' behavior. Future research works should focus on enlarging the spectrum of the algorithms tested with cellular to other variants of GP that account for the syntactical structure of the individuals to evaluate the impact of the toroidal grid also on particular methods that do not directly explore the semantic space.

Acknowledgements. This work was supported by national funds through FCT (Fundação para a Ciência e a Tecnologia), under the project - UIDB/04152/2020 - Centro de Investigação em Gestão de Informação (MagIC)/NOVA IMS (https://doi.org/10.54499/UIDB/04152/2020).

This research is partially supported by the PRIN 2022 PNRR project "Cellular Automata Synthesis for Cryptography Applications (CASCA)" (P2022MPFRT) financed by the European Union - Next Generation EU.

References

1. Ahvanooey, M.T., Li, Q., Wu, M., Wang, S.: A survey of genetic programming and its applications. KSII Trans. Internet Inf. Syst. (TIIS) **13**(4), 1765–1794 (2019)
2. Al-Betar, M.A., Khader, A.T., Awadallah, M.A., Alawan, M.H., Zaqaibeh, B.: Cellular harmony search for optimization problems. J. Appl. Math. **2013** (2013)
3. Alba, E., Dorronsoro, B.: The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. IEEE Trans. Evol. Comput. **9**(2), 126–142 (2005)
4. Alba, E., Dorronsoro, B.: Introduction to Cellular Genetic Algorithms, pp. 3–20. Springer, Boston (2008). https://doi.org/10.1007/978-0-387-77610-1_1
5. Anselin, L., Rey, S.: Properties of tests for spatial dependence in linear regression models. Geogr. Anal. **23**(2), 112–131 (1991)
6. Armstrong, J.S., Collopy, F.: Error measures for generalizing about forecasting methods: empirical comparisons. Int. J. Forecast. **8**(1), 69–80 (1992)

7. Bakurov, I., et al.: Geometric semantic genetic programming with normalized and standardized random programs. Genet. Program. Evolvable Mach. **25**(1) (2024). https://doi.org/10.1007/s10710-024-09479-1
8. Ballabio, D., Cassotti, M., Consonni, V., Todeschini, R.: QSAR aquatic toxicity. UCI Mach. Learn. Repository (2014). https://doi.org/10.24432/C5SG7H
9. Banzhaf, W., Koza, J., Ryan, C., Spector, L., Jacob, C.: Genetic programming. IEEE Intell. Syst. Appl. **15**(3), 74–84 (2000). https://doi.org/10.1109/5254.846288
10. Bonin, L., Rovito, L., De Lorenzo, A., Manzoni, L.: Cellular geometric semantic genetic programming. Genet. Program Evolvable Mach. **25**(1), 8 (2024)
11. Brooks, T.F., Pope, D.S., Marcolini, M.A.: Airfoil self-noise and prediction. Technical report (1989)
12. Brotto Rebuli, K., Giacobini, M., Silva, S., Vanneschi, L.: A comparison of structural complexity metrics for explainable genetic programming. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, pp. 539–542 (2023)
13. Castelli, M., Manzoni, L., Gonçalves, I., Vanneschi, L., Trujillo, L., Silva, S.: An analysis of geometric semantic crossover: a computational geometry approach. pp. 201–208 (2016). https://doi.org/10.5220/0006056402010208
14. Castelli, M., Manzoni, L., Vanneschi, L., Silva, S., Popovič, A.: Self-tuning geometric semantic genetic programming. Genet. Program Evolvable Mach. **17**, 55–74 (2016)
15. Castelli, M., Silva, S., Vanneschi, L.: A c++ framework for geometric semantic genetic programming. Genet. Program Evolvable Mach. **16**, 73–81 (2015)
16. Castelli, M., Vanneschi, L., Popovič, A.: Controlling individuals growth in semantic genetic programming through elitist replacement. Comput. Intell. Neurosci. **2016**, 42–42 (2016)
17. Castelli, M., Vanneschi, L., Silva, S.: Prediction of high performance concrete strength using genetic programming with geometric semantic genetic operators. Expert Syst. Appl. **40**(17), 6856–6862 (2013)
18. Castelli, M., Vanneschi, L., Silva, S.: Prediction of the unified Parkinson's disease rating scale assessment using a genetic programming system with geometric semantic genetic operators. Expert Syst. Appl. **41**(10), 4608–4616 (2014)
19. Codd, E.F.: Cellular Automata. Academic Press, Cambridge (1968)
20. Cussat-Blanc, S., Harrington, K., Pollack, J.: Gene regulatory network evolution through augmenting topologies. IEEE Trans. Evol. Comput. **19**(6), 823–837 (2015)
21. Dabhi, V.K., Chaudhary, S.: Empirical modeling using genetic programming: a survey of issues and approaches. Nat. Comput. **14**, 303–330 (2015)
22. Della Cioppa, A., Marcelli, A., Napoli, P.: Speciation in evolutionary algorithms: adaptive species discovery. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, pp. 1053–1060 (2011)
23. Dick, G., Whigham, P.A.: Controlling bloat through parsimonious elitist replacement and spatial structure. In: European Conference on Genetic Programming, pp. 13–24. Springer (2013)
24. Farinati, D., Bakurov, I., Vanneschi, L.: A study of dynamic populations in geometric semantic genetic programming. Inf. Sci. **648**, 119513 (2023). https://doi.org/10.1016/j.ins.2023.119513, https://www.sciencedirect.com/science/article/pii/S0020025523010988
25. Ferreira, L.A., Guimarães, F.G., Silva, R.: Applying genetic programming to improve interpretability in machine learning models. In: 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2020). https://doi.org/10.1109/CEC48606.2020.9185620

26. Folino, G., Pizzuti, C., Spezzano, G.: A cellular genetic programming approach to classification. In: GECCO, pp. 1015–1020 (1999)
27. Folino, G., Pizzuti, C., Spezzano, G.: A scalable cellular implementation of parallel genetic programming. IEEE Trans. Evol. Comput. **7**(1), 37–53 (2003)
28. Giacobini, M., Tomassini, M., Tettamanzi, A.G., Alba, E.: Selection intensity in cellular evolutionary algorithms for regular lattices. IEEE Trans. Evol. Comput. **9**(5), 489–505 (2005)
29. Goldberg, D.E., Deb, K.: A comparative analysis of selection schemes used in genetic algorithms. In: Foundations of Genetic Algorithms, vol. 1, pp. 69–93. Elsevier (1991)
30. Holland, J.H.: Genetic algorithms and the optimal allocation of trials. SIAM J. Comput. **2**(2), 88–105 (1973)
31. Holm, S.: A simple sequentially rejective multiple test procedure. Scand. J. Stat. 65–70 (1979)
32. Hu, T.: Genetic Programming for Interpretable and Explainable Machine Learning, pp. 81–90. Springer, Singapore (2023). https://doi.org/10.1007/978-981-19-8460-0_4
33. Huynh, Q., Singh, H., Ray, T., Oyama, A.: Improved genetic programming for symbolic regression: case studies on practical applications. In: 2022 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1135–1142 (2022). https://doi.org/10.1109/SSCI51031.2022.10022279
34. Juárez-Smith, P., Trujillo, L., García-Valdez, M., Fernández de Vega, F., Chávez, F.: Local search in speciation-based bloat control for genetic programming. Genet. Program Evolvable Mach. **20**(3), 351–384 (2019). https://doi.org/10.1007/s10710-019-09351-7
35. Koga, D., Ohnishi, K.: Non-generational geometric semantic genetic programming. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–7. IEEE (2021)
36. Koza, J.R.: Genetic programming as a means for programming computers by natural selection. Stat. Comput. **4**(2), 87–112 (1994)
37. Kruskal, W.H., Wallis, W.A.: Use of ranks in one-criterion variance analysis. J. Am. Stat. Assoc. **47**(260), 583–621 (1952)
38. La Cava, W., Danai, K., Spector, L.: Inference of compact nonlinear dynamic models by epigenetic local search. Eng. Appl. Artif. Intell. **55**, 292–306 (2016)
39. La Cava, W., Danai, K., Spector, L., Fleming, P., Wright, A., Lackner, M.: Automatic identification of wind turbine models using evolutionary multiobjective optimization. Renew. Energy **87**, 892–902 (2016)
40. La Cava, W., et al.: Contemporary symbolic regression methods and their relative performance. In: Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (2021)
41. Langdon, W.B., Poli, R., McPhee, N.F., Koza, J.R.: Genetic programming: an introduction and tutorial, with a survey of techniques and applications. In: Computational Intelligence: A Compendium, pp. 927–1028 (2008)
42. Lensen, A., Xue, B., Zhang, M.: Genetic programming for evolving a front of interpretable models for data visualization. IEEE Trans. Cybern. **51**(11), 5468–5482 (2021). https://doi.org/10.1109/TCYB.2020.2970198
43. Li, H., Calder, C.A., Cressie, N.: Beyond Moran's i: testing for spatial dependence based on the spatial autoregressive model. Geogr. Anal. **39**(4), 357–375 (2007)
44. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. **18**(1), 50–60 (1947)

45. Mariot, L., Picek, S., Jakobovic, D., Leporati, A.: Evolutionary algorithms for the design of orthogonal Latin squares based on cellular automata. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2017, pp. 306–313. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3071178.3071284
46. Martin, W.N.: Island (migration) models: evolutionary algorithms based on punctuated equilibria. In: Handbook of Evolutionary Computation (1997)
47. Martins, J.F.B., Oliveira, L.O.V., Miranda, L.F., Casadei, F., Pappa, G.L.: Solving the exponential growth of symbolic regression trees in geometric semantic genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1151–1158 (2018)
48. Martins, T.M., Neves, R.F.: Applying genetic algorithms with speciation for optimization of grid template pattern detection in financial markets. Expert Syst. Appl. **147**, 113191 (2020)
49. Mei, Y., Chen, Q., Lensen, A., Xue, B., Zhang, M.: Explainable artificial intelligence by genetic programming: a survey. IEEE Trans. Evol. Comput. **27**(3), 621–641 (2023). https://doi.org/10.1109/TEVC.2022.3225509
50. Moraglio, A., Krawiec, K., Johnson, C.G.: Geometric semantic genetic programming. In: Coello, 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
51. Moran, P.A.: Notes on continuous stochastic phenomena. Biometrika **37**(1/2), 17–23 (1950)
52. Murata, T., Gen, M.: Cellular genetic algorithm for multi-objective optimization. In: Proceedings of the 4th Asian Fuzzy System Symposium, pp. 538–542. Citeseer (2002)
53. Nadizar, G., Garrow, F., Sakallioglu, B., Canonne, L., Silva, S., Vanneschi, L.: An investigation of geometric semantic GP with linear scaling. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2023, pp. 1165–1174. Association for Computing Machinery, New York (2023). https://doi.org/10.1145/3583131.3590418
54. Nadizar, G., Medvet, E., Wilson, D.: Searching for a diversity of interpretable graph control policies. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2024, pp. 933–941. Association for Computing Machinery, New York (2024). https://doi.org/10.1145/3638529.3653987
55. Nadizar, G., Medvet, E., Wilson, D.G.: Naturally interpretable control policies via graph-based genetic programming. In: Giacobini, M., Xue, B., Manzoni, L. (eds.) Genetic Programming. LNCS, pp. 73–89. Springer Nature Switzerland, Cham (2024)
56. Nadizar, G., Rovito, L., De Lorenzo, A., Medvet, E., Virgolin, M.: An analysis of the ingredients for learning interpretable symbolic regression models with human-in-the-loop and genetic programming. ACM Trans. Evol. Learn. Optim. **4**(1) (2024). https://doi.org/10.1145/3643688
57. Nadizar, G., Sakallioglu, B., Garrow, F., Silva, S., Vanneschi, L.: Geometric semantic GP with linear scaling: Darwinian versus Lamarckian evolution. Genet. Program Evolvable Mach. **25**(2), 1–24 (2024)
58. Nebro, A.J., Durillo, J.J., Luna, F., Dorronsoro, B., Alba, E.: Mocell: a cellular genetic algorithm for multiobjective optimization. Int. J. Intell. Syst. **24**(7), 726–746 (2009)
59. Neumann, J.V.: Theory of self-reproducing automata. Math. Comput. **21**, 745 (1966)

60. Nguyen, Q.U.: Examining Semantic Diversity and Semantic Locality of Operators in Genetic Programming. Ph.D. thesis, University College Dublin, Ireland (2011). http://ncra.ucd.ie/papers/Thesis_Uy_Corrected.pdf
61. Ortigosa, I., Lopez, R., Garcia, J.: A neural networks approach to residuary resistance of sailing yachts prediction. In: Proceedings of the International Conference on Marine Engineering Marine, vol. 2007, p. 250 (2007)
62. Orzechowski, P., La Cava, W., Moore, J.H.: Where are we now? A large benchmark study of recent symbolic regression methods. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1183–1190 (2018)
63. Pawlak, T.P., Wieloch, B., Krawiec, K.: Review and comparative analysis of geometric semantic crossovers. Genet. Program Evolvable Mach. **16**, 351–386 (2015)
64. Pietropolli, G., Manzoni, L., Paoletti, A., Castelli, M.: Combining geometric semantic GP with gradient-descent optimization. In: Medvet, E., Pappa, G., Xue, B. (eds.) Genetic Programming, pp. 19–33. Springer, Cham (2022)
65. Pietropolli, G., Manzoni, L., Paoletti, A., Castelli, M.: On the hybridization of geometric semantic GP with gradient-based optimizers. Genet. Program Evolvable Mach. **24**(2), 16 (2023)
66. Pietropolli, G., Nichele, S., Medvet, E.: The role of the substrate in ca-based evolutionary algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2024, pp. 768–777. Association for Computing Machinery, New York (2024). https://doi.org/10.1145/3638529.3654112
67. Poli, R., Langdon, W.B., McPhee, N.F., Koza, J.R.: Genetic programming: an introductory tutorial and a survey of techniques and applications. Univ. Essex School Computer Science and Electronic Engineering Technical report No. CES-475, pp. 1–112 (2007)
68. Salto, C., Alba, E.: Cellular genetic algorithms: understanding the behavior of using neighborhoods. Appl. Artif. Intell. **33**(10), 863–880 (2019)
69. Sarkar, P.: A brief history of cellular automata. ACM Comput. Surv. (CSUR) **32**(1), 80–107 (2000)
70. Sarma, J., De Jong, K.: An analysis of the effects of neighborhood size and shape on local selection algorithms. In: Voigt, H.-M., Ebeling, W., Rechenberg, I., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 236–244. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-61723-X_988
71. Shi, Y., Liu, H., Gao, L., Zhang, G.: Cellular particle swarm optimization. Inf. Sci. **181**(20), 4460–4493 (2011)
72. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evol. Comput. **10**(2), 99–127 (2002)
73. Takac, A.: Application of cellular genetic programming in data mining. In: Proceedings of Conference Knowledge, Citeseer (2004)
74. Takac, A.: Cellular genetic programming algorithm applied to classification task. Neural Netw. World **14**, 435–452 (2004)
75. Trujillo, L., Contreras, J., Hernandez, D.E., Castelli, M., Tapia, J.J.: GSGP-CUDA-a CUDA framework for geometric semantic genetic programming. SoftwareX **18**, 101085 (2022)
76. Trujillo, L., Muñoz, L., Galván-López, E., Silva, S.: neat genetic programming: controlling bloat naturally. Inf. Sci. **333**, 21–43 (2016)
77. Vanneschi, L.: An introduction to geometric semantic genetic programming. In: NEO 2015: Results of the Numerical and Evolutionary Optimization Workshop NEO 2015 held at 23–25 Sep 2015 in Tijuana, Mexico, pp. 3–42. Springer (2016)

78. Vanneschi, L.: SLIM_GSGP: The non-bloating geometric semantic genetic programming. In: European Conference on Genetic Programming (Part of EvoStar), pp. 125–141. Springer (2024)
79. Vanneschi, L., Castelli, M., Manzoni, L., Silva, S.: A new implementation of geometric semantic gp and its application to problems in pharmacokinetics. In: Genetic Programming: 16th European Conference, EuroGP 2013, Vienna, Austria, 3–5 Apr 2013. Proceedings 16, pp. 205–216. Springer (2013)
80. Vanneschi, L., Castelli, M., Silva, S.: A survey of semantic methods in genetic programming. Genet. Program Evolvable Mach. **15**(2), 195–214 (2014). https://doi.org/10.1007/s10710-013-9210-0
81. Vanneschi, L., Farinati, D., Rasteiro, D., Rosenfeld, L., Pietropolli, G., Silva, S.: Exploring non-bloating geometric semantic genetic programming. In: Genetic Programming Theory and Practice. (to appear)
82. Vanneschi, L., Silva, S., Castelli, M., Manzoni, L.: Geometric semantic genetic programming for real life applications. In: Genetic Programming Theory and Practice xi, pp. 191–209 (2014)
83. Virgolin, M., Alderliesten, T., Witteveen, C., Bosman, P.A.: Improving model-based genetic programming for symbolic regression of small expressions. Evol. Comput. **29**(2), 211–237 (2021)
84. Virgolin, M., De Lorenzo, A., Randone, F., Medvet, E., Wahde, M.: Model learning with personalized interpretability estimation (ml-pie). In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO 2021, pp. 1355–1364. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3449726.3463166
85. Whigham, P.A., Dick, G.: Implicitly controlling bloat in genetic programming. IEEE Trans. Evol. Comput. **14**(2), 173–190 (2009)
86. White, D.R., McDermott, J., Castelli, M., Manzoni, L., Goldman, B.W., Kronberger, G., Jaśkowski, W., O'Reilly, U.M., Luke, S.: Better GP benchmarks: community survey results and proposals. Genet. Program Evolvable Mach. **14**, 3–29 (2013)
87. Wickman, R., Poudel, B., Villarreal, T.M., Zhang, X., Li, W.: Efficient quality-diversity optimization through diverse quality species. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, pp. 699–702 (2023)
88. Xie, H., Zhang, M.: Impacts of sampling strategies in tournament selection for genetic programming. Soft. Comput. **16**, 615–633 (2012)
89. Yeh, I.C.: Simulation of concrete slump using neural networks. Proc. Inst. Civ. Eng. Constr. Mater. **162**(1), 11–18 (2009)
90. Zhang, M., Tian, N., Palade, V., Ji, Z., Wang, Y.: Cellular artificial bee colony algorithm with gaussian distribution. Inf. Sci. **462**, 374–401 (2018)