ϵ-Lexicase selection: a probabilistic and multi-objective analysis of lexicase selection in continuous domains

March 8, 2017

Abstract

Lexicase selection is a parent selection method that considers test cases separately, rather than in aggregate, when performing parent selection. As opposed to previous work that has demonstrated the ability of lexicase selection to solve difficult problems, the goal of this paper is to develop the theoretical underpinnings that explain its performance. To this end, we derive an analytical formula that gives the expected probabilities of selection under lexicase selection, given a population and its behavior. In addition, we expand upon the relation of lexicase selection to many-objective optimization methods to show the effect of lexicase, which is to select individuals on the boundaries of Pareto fronts in high-dimensional space. We show analytically why lexicase selection performs more poorly for certain sizes of population and training cases, and why it has been shown to perform more poorly in continuous error spaces. To addres this last concern, we introduce ϵ-lexicase selection, which modifies the pass condition defined in lexicase selection to allow near-elite individuals to pass cases, thereby improving selection performance. We show that ϵ-lexicase outperforms several diversity-maintenance strategies for problems from three continuous-valued domains: regression, dynamical systems, and program synthesis.

1 INTRODUCTION

2 Preliminaries

We denote the population of N programs as $\mathcal{N} = \{n_i\}_{i=1}^{N}$, the training cases on which they are asses as $\mathcal{T} = \{t_i = (y_i, \mathbf{x}_i)\}_{i=1}^{T}$, and the errors they produce as $\mathbf{e}_t(n) = (y_t - \hat{y}_t(n))^2$.

$\mathcal{T} = \{(y_t, \mathbf{x}_t)\}_{t=1}^{N}$, using e.g. the mean absolute error (MAE), which is quantified for individual program $i \in P$ as:

$$f(i, \mathcal{T}) = \frac{1}{N} \sum_{t \in \mathcal{T}} |y_t - \hat{y}_t(i, \mathbf{x}_t)| \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^D$ represents the variables or features, the target output is y and $\hat{y}(i, \mathbf{x})$ is the program's output. As a result of the aggregation of the absolute error vector $e(i) = |y - \hat{y}(i, \mathbf{x})|$ in Eq. (1), the relationship of $\hat{y}$ to y is represented crudely when choosing models to propagate.

3 Lexicase Selection

Lexicase selection is a parent selection technique based on lexicographic ordering of test (i.e. fitness) cases. Each parent selection event proceeds as follows:

1. The entire population is added to the selection pool.
2. The fitness cases are shuffled.
3. Individuals in the pool with a fitness worse than the best fitness on this case among the pool are removed.
4. If more than one individual remains in the pool, the first case is removed and 3 is repeated with the next case. If only one individual remains, it is the chosen parent. If no more fitness cases are left, a parent is chosen randomly from the remaining individuals.

The lexicase selection algorithm for a single selection event is presented below:

Lexicase Selection

`GetLexicaseParent`$(\mathcal{N}, \mathcal{T})$:	
$T' \leftarrow$ `shuffle`$(\mathcal{T})$	training cases
$S \leftarrow \mathcal{N}$	initial selection pool is the population
while $\lvert T' \rvert > 0$ and $\lvert \mathcal{S} \rvert > 1$:	main loop
`case` $\leftarrow$ random choice from $\mathcal{T}'$	choose a random case
`elite` $\leftarrow$ best fitness in $\mathcal{S}$ on `case`	determine elite fitness
$\mathcal{S} \leftarrow n \in \mathcal{S}$ if fitness$(n) =$ `elite`	reduce selection pool to elites
$\mathcal{T}' \leftarrow \mathcal{T}' -$ `case`	remove top case
return random choice from $\mathcal{S}$	return parent

4 ϵ-Lexicase Selection

Static ϵ-Lexicase Selection

`Get_`ϵ`-Lexicase_Parent`$(\mathcal{N}, \mathcal{T})$:	
$T' \leftarrow$ `shuffle`$(\mathcal{T})$	training cases
$S \leftarrow \mathcal{N}$	initial selection pool is the population
$\epsilon \leftarrow MAD(\text{fitness}(\mathcal{N}))$ for $t \in \mathcal{T}$	get ϵ for each case
fitness$(n) \leftarrow \mathbb{1}(e_t(n) \leq e_t^*(n) + \epsilon_t)$ for $t \in \mathcal{T}$ and $n \in \mathcal{N}$	convert fitness using within-ϵ pass condition
while $\lvert T' \rvert > 0$ and $\lvert \mathcal{S} \rvert > 1$:	main loop
`case` $\leftarrow$ random choice from $\mathcal{T}'$	consider the top case
`elite` $\leftarrow$ best fitness in $\mathcal{S}$ on `case`	determine elite fitness
$\mathcal{S} \leftarrow n \in \mathcal{S}$ if fitness$(n) \leq$ `elite`	reduce selection pool to elites
$\mathcal{T}' \leftarrow \mathcal{T}' -$ `case`	remove top case
return random choice from $\mathcal{S}$	return parent

Semi-dynamic ϵ-Lexicase Selection

`Get_`ϵ`-Lexicase_Parent`$(\mathcal{N}, \mathcal{T})$:

$T' \leftarrow$ `shuffle`$(\mathcal{T})$	training cases
$S \leftarrow \mathcal{N}$	initial selection pool is the population
$\epsilon \leftarrow MAD(\text{fitness}(\mathcal{N}))$ for $t \in \mathcal{T}$	get ϵ for each case
while $\lvert T' \rvert > 0$ and $\lvert \mathcal{S} \rvert > 1$:	main loop
`case` $\leftarrow$ random choice from $\mathcal{T}'$	consider the top case
`elite` $\leftarrow$ best fitness in $\mathcal{S}$ on `case`	determine elite fitness
$\mathcal{S} \leftarrow n \in \mathcal{S}$ if fitness$(n) \leq$ `elite`$+\epsilon_{\texttt{case}}$	reduce selection pool to elites
$\mathcal{T}' \leftarrow \mathcal{T}' -$ `case`	remove top case
return random choice from $\mathcal{S}$	return parent

Dynamic ϵ-Lexicase Selection

`Get_`ϵ`-Lexicase_Parent`$(\mathcal{N}, \mathcal{T})$:

$T' \leftarrow$ `shuffle`$(\mathcal{T})$	training cases
$S \leftarrow \mathcal{N}$	initial selection pool is the population
while $\lvert T' \rvert > 0$ and $\lvert \mathcal{S} \rvert > 1$:	main loop
`case` $\leftarrow$ random choice from $\mathcal{T}'$	consider the top case
`elite` $\leftarrow$ best fitness in $\mathcal{S}$ on `case`	determine elite fitness
$\epsilon \leftarrow MAD(\text{fitness}(\mathcal{S}))$ on `case`	determine ϵ for this case
$\mathcal{S} \leftarrow n \in \mathcal{S}$ if fitness$(n) \leq$ `elite`$+\epsilon_{\texttt{case}}$	reduce selection pool to elites
$\mathcal{T}' \leftarrow \mathcal{T}' -$ `case`	remove top case
return random choice from $\mathcal{S}$	return parent

5 Expected Probabilities of Selection

What is the probability of an individual being selected, given its performance in a given population on a set of training cases?

To put it into words, the probability of n being selected is the probability that a case n passes (a member of $\mathcal{K}_n$) is selected and:

1. no more cases remain and n is selected among the set of individuals that pass the selected case; or
2. n is the only individual that passes the case; or
3. n is selected via the selection of another case that n passes (repeating the process).

Formally, let $P_{sel}(n|\mathcal{N}, \mathcal{T})$ be the probability of n being selected in a population $\mathcal{N}$ with training cases $\mathcal{T}$. Let $\mathcal{K}_n(\mathcal{T}, \mathcal{N}) = \{k_i\}_{i=1}^{K} \subseteq \mathcal{T}$ be the training cases from $\mathcal{T}$ for which individual n is elite among $\mathcal{N}$. We will use $\mathcal{K}_n$ for brevity. Then the probability of selection under lexicase can be represented as a piece-wise recursive function:

$$P_{sel}(n|\mathcal{N}, \mathcal{T}) = \begin{cases} 1 & : \quad |\mathcal{T}| > 0, |\mathcal{N}| = 1; \\ 1/|\mathcal{N}| & : \quad |\mathcal{T}| = 0; \\ \frac{1}{|\mathcal{T}|} \sum_{k_s \in \mathcal{K}_n} P_{sel}\left(n|\mathcal{N}(m|k_s \in K_m), \mathcal{T}(t|t \neq k_s)\right) & : \quad \text{otherwise} \end{cases} \tag{2}$$

The first two elements of P_{sel} follow from the lexicase algorithm: if there is one individual in $\mathcal{N}$, then it is selected; otherwise if there no more cases in in $\mathcal{T}$, then n has a probability of selection split among the individuals in $\mathcal{N}$, i.e., $1/|\mathcal{N}|$. If neither of these conditions are met, the remaining probability of selection is $1/|\mathcal{T}|$ times the summation of P_{sel} over n's elite cases. Each case for which n is elite among the current pool (represented by $k_s \in K_n$) has a probability of $1/|\mathcal{T}|$ of being selected. For each potential selection k_s, the probability of n being selected as a result of this case being chosen is dependent on the number of individuals that are also elite on these cases, represented by $\mathcal{N}(m|k_s \in K_m(\mathcal{T}))$, and the cases that are left to be traversed, represented by $\mathcal{T}(t|t \neq k_s)$.

Eqn. 2 also describes the probability of selection under ϵ-lexicase selection, with the condition that *elitism* on a case is defined as being within ϵ of the best error on that case, where the best error is defined among the whole population (statically) or among the current selection pool (semi-dynamic and dynamic). The definition of ϵ differs according to whether static, semi-dynamic, or dynamic lexicase are being used as well.

Intuitions according to edge cases According to Eqn. 2, when fitness values across the population are unique, selection probability is $\frac{1}{|\mathcal{T}|}\sum_{k_s \in \mathcal{K}_n(\mathcal{T})} 1 = \frac{|\mathcal{K}_n|}{|\mathcal{T}|}\ \forall\ n$, since filtering the population according to any case for which n is elite will result in n being selected. Conversely, if the population semantics are completely homogeneous such that every individual is elite on every case, the selection will be uniformly random, giving the selection probability $\frac{1}{N}\ \forall\ n$. In fact, this holds true for individual cases: a case t will have no impact on the probability of selection of n if $t \notin K_n\ \forall\ n, t\ \in \mathcal{N}, \mathcal{T}$. This is clear when viewed from the lexicase algorithm: any case that every individual passes provides no selective pressure.

Complexity Can we analytically calculate the probability of selection an individual with less complexity than executing the lexicase selection algorithm? It appears not. Eqn. 2 has a worst-case complexity of $O(T^N)$ when all individuals are elite on $\mathcal{T}$, which discourages its use as a selection method. The lexicase selection algorithm samples the expected probabilities of each individual by recursing on random orders of cases in $\mathcal{T}$, considering one at a time rather than branching to consider all combinations of cases that could result in selection for each individual in question. This gives lexicase selection a complexity of $O(TN)$ for selecting a single parent, and therefore a complexity of $O(TN^2)$ per generation.

5.1 Effect of N and T

What does the an analysis of the probability of selection for lexicase selection tell us about how lexicase behaves for cases in which $|\mathcal{N}| << |\mathcal{T}|$? $|\mathcal{T}| << |\mathcal{N}|$?

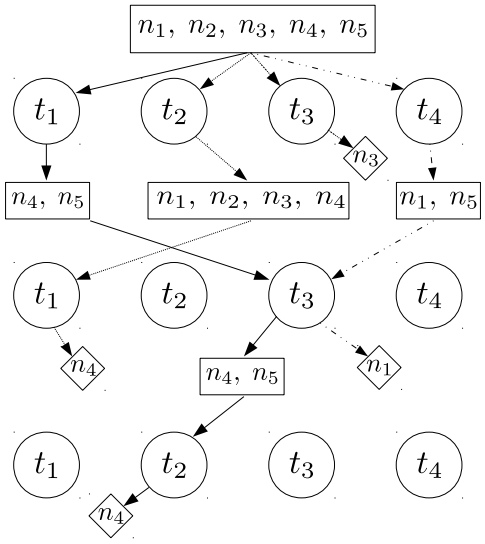

Figure 1: A graphical representation of 3 example parent selections using lexicase selection on the population in Table 1. The bold, dashed and dot-dashed lines indicate different selection paths through the test cases in circles. The boxes indicate the selection pool at each step in the process.

The sampling of P_{sel} done by lexicase is tied to the population size because lexicase selection conducts N depth-first searches of the case orderings to choose N parents. This implies that the value of N determines the fidelity with which P_{sel} is approximated via the sampling. Smaller populations will therefore produce poorer approximations of P_{sel}.

The effectiveness of lexicase selection has also been tied to the number of fitness cases, T. When T is small, there are very few ways in which individuals can be selected. For example, if $T = 2$, an individual must be elite on one of these two cases to be selected. For continuous errors in which few individuals are elite, this means that only two individuals are likely to produce all of the children for the subsequent generation.

5.2 Effect of different population structures

1. compare probability of lexicase selection to probability of selection with tournament / roulette

2. compare population structures: maintain correlation structure and vary population size/ number of test cases

3. see how many iterations of lexicase selection are required to converge on the probabilities of selection for the example problem in Table 1

4. population where there is one individual that sucks at everything but is good at other things - how does it compare to tournament selection probabilities?

5. do not assume that N rounds of lexicase selection are conducted!

Probabilities under tournament selection We compare the probability of selection under lexicase selection to that using tournament selection with an indentical population and fitness structure. To do so we must first formulate the probability of selection for an individual undergoing tournament selection with r-size tournaments. Consider that the mean absolute error is used to aggregate the fitness cases. Then the fitness ranks of $\mathcal{N}$ can be calculated, with lower rank indicating better fitness. Let S_i be the individuals in $\mathcal{N}$ with a fitness rank of i, and let Q be the number of unique fitness ranks. Then Xie et. al. [4] showed that the probability of selecting an individual with rank j in a single tournament is

$$P_t = \frac{1}{|S_j|}\left(\left(\frac{\sum_{i=j}^{Q}|S_i|}{N}\right)^r - \left(\frac{\sum_{i=j+1}^{Q}|S_i|}{N}\right)^r\right) \tag{3}$$

In Table 1, the selection probabilities for the example population are shown according to tournament selection.

6 Multi-objective Interpretation of Lexicase Selection

Objectives and training cases are fundamentally different entities: objectives define the goals of the task being learned, whereas cases are the units by which progress towards those objectives is measured. By this criteria, lexicase selection and multi-objective optimization have historically been differentiated [2], although there is clearly a "multi-objective" interpretation of the behavior of lexicase selection with respect to the training cases. If we assume for a moment that we treat individual fitness cases as objectives to solve, we can consider most learning tasks to be high-dimensional many-objective optimization problems. At this scale, the most popular multi-objective methods (e.g. NSGA-II and SPEA-2) tend to perform poorly, a behavior that has been explained in literature [3, 1]. Farina and Amato [1] point out two short-comings of these multi-objective methods when many objectives are considered:

> the Pareto definition of optimality in a multi-criteria decision making problem can be unsatisfactory due to essentially two reasons: the number of improved or equal objective values is not taken into account, the (normalized) size of improvements is not taken into account

Lexicase selection is guarantees the return of individuals that are on the Pareto front with respect to the fitness cases. This is a necessary but not sufficient condition. In fact, lexicase selection only selects those individuals in the "corners" of the Pareto front, meaning they are on the front *and* elite, globally, with respect to at least one fitness case. Put another way, no individual can be selected via lexicase selection unless it is elite with respect to at least one objective among the entire population, regardless of its performance on other objectives.

Interestingly, the worst-case complexity of NSGA-II is the best-case complexity for lexicase selection. Add notes from discourse

Here we define Pareto dominance relations with respect to the training cases.

Definition 1: n_1 *dominates* n_2, i.e., $n_1 \prec n_2$, if $e_j(n_1) \leq e_j(n_2)\ \forall j \in \{1, \ldots, T\}$ and $\exists j \in \{1, \ldots, T\}$ for which $e_j(n_1) < e_j(n_2)$.

Definition 2: The *Pareto front* of $\mathcal{N}$ is the subset of $\mathcal{N}$ that is non-dominated with respect to $\mathcal{T}$; i.e., $n \in \mathcal{N}$ is on the Pareto front if $n \not\prec m\ \forall\ m \in \mathcal{N}$.

Definition 3: $n \in \mathcal{N}$ is a *Pareto front boundary* if n is on the Pareto front of $\mathcal{N}$ and $\exists j \in \{1, \ldots, T\}$ for which $e_j(n_1) \leq e_j(m)\ \forall\ m \in \mathcal{N}$.

With these definitions in mind, we show that individuals selected by lexicase are on the Pareto front boundaries.

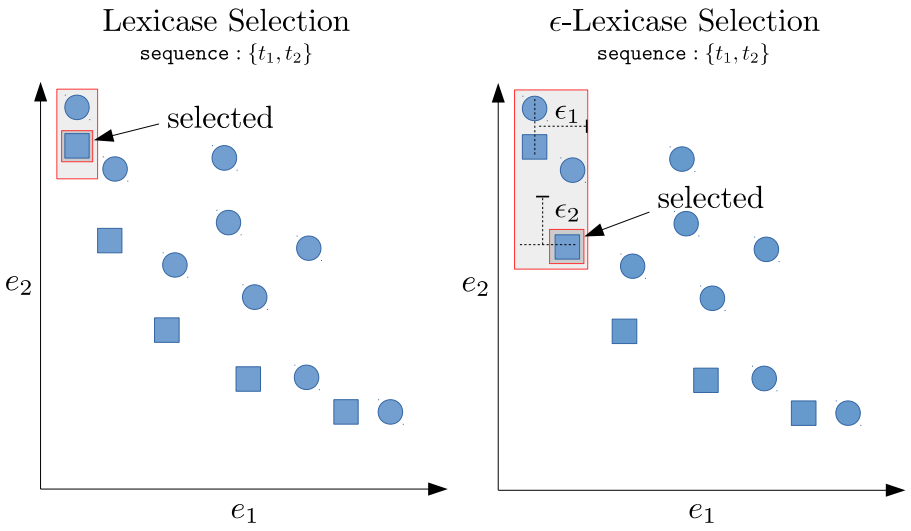

Figure 2: An illstration of the performance of lexicase selection in a scenario involving two cases. Each point represents and individual in the population. The squares are individuals on the Pareto front. In each case, the selection ordering shown is $\{t_1, t_2\}$.

Figure 3: An illstration of the performance of ϵ-lexicase selection in a scenario involving two cases. Each point represents and individual in the population. The squares are individuals on the Pareto front. In each case, the selection ordering shown is $\{t_1, t_2\}$.

Theorem 1: If individuals from a population $\mathcal{N}$ are selected by lexicase selection, those individuals are located at Pareto front boundaries of $\mathcal{N}$ with respect to $\mathcal{T}$.

Proof: First, we prove by contradiction that individuals selected by lexicase are on the Pareto front. We then prove these individuals to be Pareto front boundaries.

First: Let $n_1, n_2 \in \mathcal{N}$ be individuals in a population selected by lexicase selection. Suppose $n_1 \prec n_2$. Then $e_j(n_1) \leq e_j(n_2)\ \forall j \in \{1, \ldots, N\}$ and $\exists j \in \{1, \ldots, T\}$ for which $e_j(n_1) < e_j(n_2)$. Therefore n_1 is selected for every case that n_2 is selected, and $\exists t \in \mathcal{T}$ for which n_2 is removed from selection due to n_1. Hence, n_2 cannot be selected by lexicase selection and the supposition is false. Therefore n_1 and n_2 must be on the Pareto front.

Second: let $n \in \mathcal{N}$ be an individual in a population selected by lexicase selection. Then by definition of the algorithm, $\exists j \in \{1, \ldots, T\}$ for which $e_j(n) \leq e_j(m)\ \forall\ m \in \mathcal{N}$. Therefore according to Definition 3, n is a Pareto front boundary of $\mathcal{N}$.

We can extend Theorem 1 to ϵ-lexicase selection for conditions in which ϵ is pre-defined for each fitness case, i.e. in static and dynamic cases, but not when ϵ is recalculated for each selection pool. The theorem is extended by defining the dominance relation with respect to ϵ, as follows:

Definition: n_1 ϵ-dominates n_2, i.e., $n_1 \prec_\epsilon n_2$, if $e_j(n_1) + \epsilon_j \leq e_j(n_2)\ \forall j \in \{1, \ldots, T\}$ and $\exists j \in \{1, \ldots, T\}$ for which $e_j(n_1) + \epsilon_j < e_j(n_2)$, $\epsilon_j > 0$

In this case, the analogous condition holds for individuals selected with ϵ-lexicase selection.

Theorem 2: If ϵ is defined according to Eqn. **??**, and if individuals are selected from a population $\mathcal{N}$ by ϵ-lexicase selection, then those individuals are non-ϵ-dominated in $\mathcal{N}$ with respect to the training cases $\mathcal{T}$.

Proof: Let $n_1, n_2 \in \mathcal{N}$ be individuals in a population selected by ϵ-lexicase selection. Suppose $n_1 \prec_\epsilon n_2$. Then n_1 is selected for every case that n_2 is selected, and $\exists t \in \mathcal{T}$ in every selection event for which n_2 is removed from selection due to n_1. Therefore n_2 cannot be selected by ϵ-lexicase selection, the supposition is false, and the theorem is true.

7 Illustrative Example

Example As an example of calculating absolute probabilities, we consider the illustrative problem from the original lexicase selection paper [**?**], shown in Table 1. Using Eqn. 2, the probabilities for each individual can be calculated as follows:

Table 1: Example population from original lexicase paper (Spector 2013).

Program	Test				$\mathcal{K}(\mathcal{T})$	MAE	P_{sel}	P_t
	t_1	t_2	t_3	t_4				
n_1	2	2	4	2	$\{t_2, t_4\}$	2.5	0.25	0.28
n_2	1	2	4	3	$\{t_2\}$	2.5	0.00	0.28
n_3	2	2	3	4	$\{t_2, t_3\}$	2.75	0.33	0.12
n_4	0	2	5	5	$\{t_1, t_2\}$	3.0	0.208	0.04
n_5	0	3	5	2	$\{t_1, t_4\}$	2.5	0.208	0.28

Table 2: Example population with test case performances and selection probabilities according to the different algorithms.

$\mathcal{N}$	Cases					Mean	Probability of Selection				
	t_1	t_2	t_3	t_4	t_5		tourn	lex	ϵ lex static	ϵ lex semi	ϵ lex dyn
n_1	0.0	1.1	2.2	3.0	5.0	2.26	0.111	0.200	0.000	0.067	0.033
n_2	0.1	1.2	2.0	2.0	6.0	2.26	0.111	0.000	0.150	0.117	0.200
n_3	0.2	1.0	2.1	1.0	7.0	2.26	0.111	0.000	0.150	0.117	0.117
n_4	1.0	2.1	0.2	0.0	8.0	2.26	0.111	0.200	0.300	0.200	0.167
n_5	1.1	2.2	0.0	4.0	4.0	2.26	0.111	0.200	0.000	0.050	0.050
n_6	1.2	2.0	0.1	5.0	3.0	2.26	0.111	0.000	0.000	0.050	0.033
n_7	2.0	0.1	1.2	6.0	2.0	2.26	0.111	0.000	0.133	0.133	0.133
n_8	2.1	0.2	1.0	7.0	1.0	2.26	0.111	0.000	0.133	0.133	0.217
n_9	2.2	0.0	1.1	8.0	0.0	2.26	0.111	0.400	0.133	0.133	0.050
ϵ	0.9	0.9	0.9	2.0	2.0						

$$
\begin{aligned}
P_{sel}(n_1) &= 1/4 * (1/3 * (1) + 1/3 * (1 + 1)) = 0.25 \\
P_{sel}(n_2) &= 1/4 * (0) = 0 \\
P_{sel}(n_3) &= 1/4 * (1/3 * (1 + 1/2 * (1)) + 1/3 * (1)) = 0.20833 \\
P_{sel}(n_4) &= 1/4 * (1/3 * (1/2 * (1) + 1) + 1/3 * (1)) = 0.20833
\end{aligned}
$$

Illlustrative Example 2 A more complex population semantic structure is presented in Table 2 featuring floating point errors. In this case, the population semantics are completely unique, although they result in the same mean error across the training cases, as shown in the "Mean" column. As a result, tournament selection picks uniformly from among these individuals.

As mentioned earlier, with unique populations, lexicase selection is proportional to the number of cases for which an individual is elite. This leads lexicase selection to pick from among the 4 individuals that are elite on cases, i.e. n_1 (t_1), n_4 (t_4), n_5 ($t3$), and n_9 (t_2,t_5), with respective probabilities 0.2, 0.2, 0.2, and 0.4.

Due to its strict definition of elitism, lexicase selection does not account for the fact other individuals are very close to being elite on these cases as well. The ϵ-lexicase variants address this as noted by the smoother distribution of selection probabilities among this population. Focusing first on static ϵ-lexicase selection,

it is worth illustrating the pass conditions for the population by applying the ϵ threshold, yielding the following semantics:

	t_1	t_2	t_3	t_4	t_5
n_1	0	1	1	1	1
n_2	0	1	1	0	1
n_3	0	1	1	0	1
n_4	1	1	0	0	1
n_5	1	1	0	1	1
n_6	1	1	0	1	1
n_7	1	0	1	1	0
n_8	1	0	1	1	0
n_9	1	0	1	1	0

The selection probabilities for static ϵ lexicase selection are equivalent to the selection probabilities of lexicase selection on this converted semantic matrix. Despite elitism on case t_1, n_1 is not selected since n_2 and n_3 are ϵ-elite on this case as well as t_4. Consider n_4, which has a higher probability of selection under static ϵ-lex than lexicase selection. This is due to it being ϵ-elite on a unique combination of cases: t_3 and t_4. Lastly n_8 is selected in equal proportions to n_7 and n_6 since all are within ϵ of the elite error on the same cases.

Semi-dynamic ϵ lexicase selection allows for all 9 individuals to be selected with varying proportions that are similar to those derived for static ϵ lexicase selection. Selection probabilities for n_1 illustrate the differences in the static and semi-dynamic variants: n_1 has a chance for selection in the semi-dynamic case because when t_1 is selected as the first case, n_1 is within ϵ of the best case errors *among the pool*, i.e. $\{n_1, n_2, n_3\}$, for any subsequent order of cases. The probability of selection for n_5 and n_6 follow the same pattern.

Dynamic ϵ-lexicase selection produces the most differentiated selection pressure for this example. Consider individual n_8 which is the most likely to be selected for this example. It is selected more often than n_7 or n_9 due to the adaptations to ϵ as the selection pool is winnowed. For example, n_8 is selected by case sequence $\{t_2, t_1, t_3\}$, for which the selection pool takes the following form after each case: $\{n_7, n_8, n_9\}, \{n_7, n_8\}, \{n_8\}$. Under the other semi-dynamic ϵ lexicase selection, n_7 and n_8 would not be removed by these cases due to the fixed nature of ϵ for those cases. The probabilities of selection for the population in Table 2 is shown graphically in Figure 7.

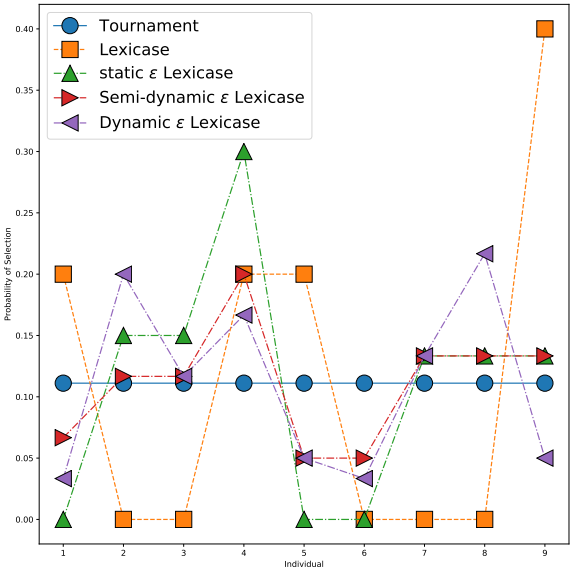

Figure 4: A graphical representation of the probabilities of selection for the population in Table 2 according to different selection methods.

Table 3: GP settings.

Setting	Value
Population size	1000
Crossover / mutation	60/40%
Program length limits	[3, 50]
ERC range	[-1,1]
Generation limit	1000
Trials	50
Terminal Set	{$\mathbf{x}$, ERC, $+$, $-$, $*$, /, sin, cos, exp, log}
Elitism	keep best

Table 4: Regresion problems used for method comparisons.

Problem	Dimension	Samples
Airfoil	5	1503
Concrete	8	1030
Crime	127	1993
ENC	8	768
ENH	8	768
Housing	14	506
Tower	25	3135
UBall5D	5	6024
Yacht	6	309

Table 5: Dynamical systems used for method comparisons. The systems were simulated four times using initial conditions within stable basins of attraction. γ indicates zero-mean, unit-variance Gaussian noise.

Problem	Equations	Initial Conditions	Samples
Bacterial Respiration	$\dot{x} = 20 - x - \frac{x \cdot y}{1+0.5 \cdot x^2}$, $\dot{y} = 10 - \frac{x \cdot y}{1+0.5 \cdot x^2}$	$x_0 = 5 \pm \gamma$, $y_0 = 10 \pm 0.1\gamma$	400
Bar Magnets	$\dot{\theta} = 0.5 \cdot \sin(\theta - \phi) - \sin(\theta)$, $\dot{\phi} = 0.5 \cdot \sin(\phi - \theta) - \sin(\phi)$	$\theta_0 \in [-2\pi, 2\pi]$, $\phi_0 \in [-2\pi, 2\pi]$	400
Glider	$\dot{v} = -0.05 \cdot v^2 - sin(\theta)$, $\dot{\theta} = v - \cos(\theta)/v$	$v_0 = 5 \pm \gamma$, $\theta_0 = 10 \pm 0.1\gamma$	400
Lotka-Volterra interspecies dynamics	$\dot{x} = 3 \cdot x - 2 \cdot x \cdot y - x^2$, $\dot{y} = 2 \cdot y - x \cdot y - y^2$	$x_0 = [1, 4, 8, 3]$, $y_0 = [3, 1, 2, 3]$	400
Predator Prey	$\dot{x} = x \cdot \left(4 - x - \frac{y}{1+x}\right)$, $\dot{y} = 2 \cdot y - x \cdot y - y^2$	$x_0 = 5 \pm \gamma$, $y_0 = 10 \pm 0.1\gamma$	400
Shear Flow	$\dot{\theta} = \cot(\phi) \cdot cos(\theta)$, $\dot{\phi} = \left(\cos^2(\phi) + 0.1 \cdot \sin^2(\phi)\right) \cdot sin(\theta)$	$\theta_0 \in [-\pi, \pi]$, $\phi_0 \in [-\pi/2, \pi/2]$	400
van der Pol oscillator	$\dot{x} = 10 \cdot \left(y - \frac{1}{3} \cdot (x^3 - x)\right)$, $\dot{y} = -\frac{1}{10} \cdot x$	$x, y \in [0, 1]$	400

Program Synthesis

Problem	Input	Output	Training Cases	Test Cases
Number IO	integer in [-100,100], float in [-100.0, 100.0]	printed float	25	1000
Wallis PI	integer in [1, 200]	float	150	50
Vector Average	vector of float of length [1,50] with each float in [-1000.0, 1000.0]	float	100	1000

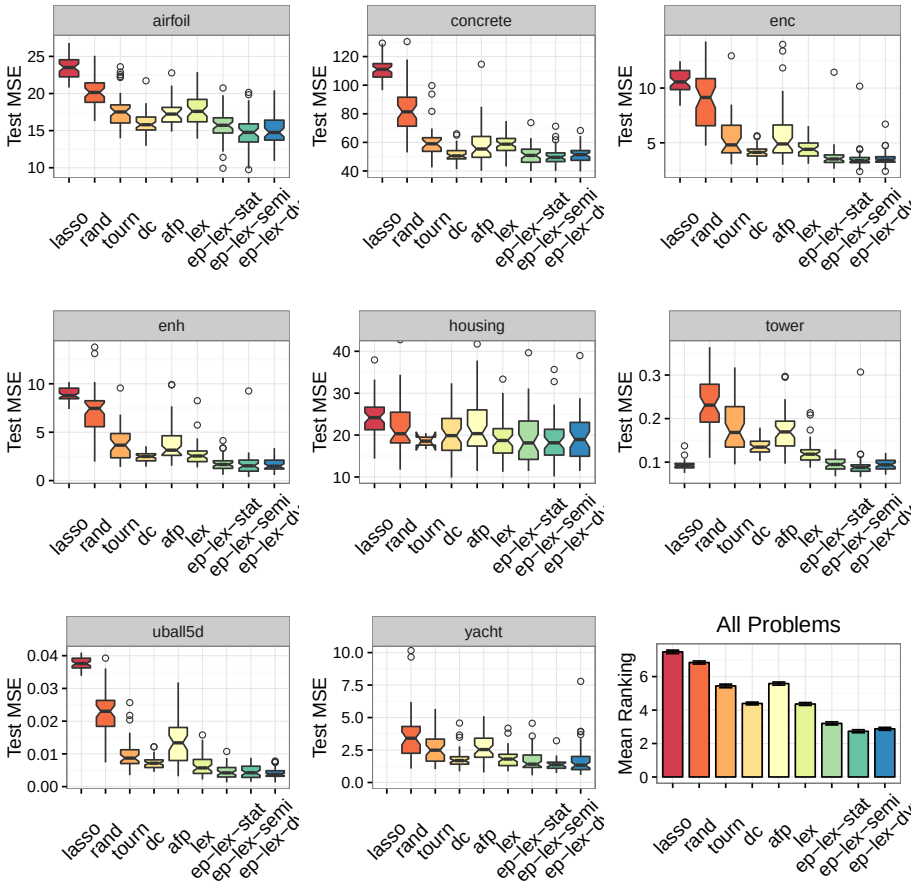

Figure 5: Boxplots of the mean squared error on the test set for 50 randomized trials of each algorithm on the regression benchmark datasets.

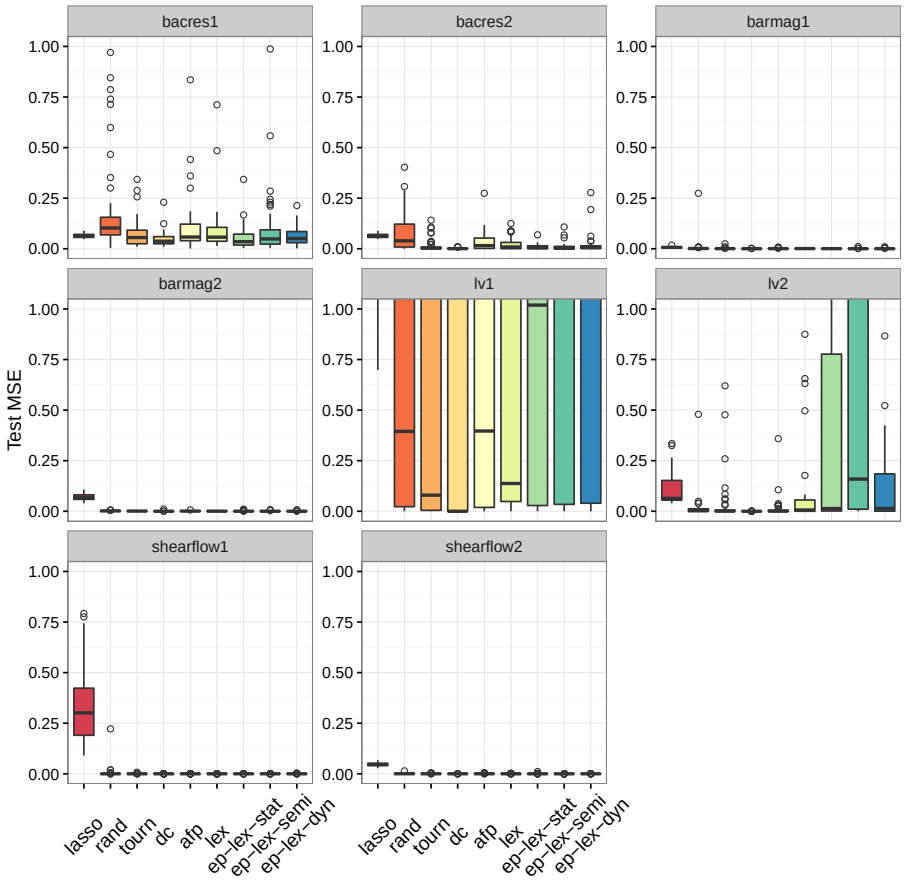

Figure 6: Boxplots of the mean squared error on the test set for 50 randomized trials of each algorithm on the regression benchmark datasets.

8 Related Work

9 Experimental Analysis

9.1 Effect of population structure on probabilities of selection

9.2 Results

10 Discussion

11 Conclusions

12 Acknowledgments

References

[1] M. Farina and P. Amato. On the optimal solution definition for many-criteria optimization problems. In *Fuzzy Information Processing Society, 2002. Proceedings. NAFIPS. 2002 Annual Meeting of the North American*, pages 233–238. IEEE, 2002.

[2] T. Helmuth. General Program Synthesis from Examples Using Genetic Programming with Parent Selection Based on Random Lexicographic Orderings of Test Cases. *Doctoral Dissertations May 2014 - current*, Jan. 2015.

[3] T. Wagner, N. Beume, and B. Naujoks. Pareto-, Aggregation-, and Indicator-Based Methods in Many-Objective Optimization. In *Evolutionary Multi-Criterion Optimization*, pages 742–756. Springer, Berlin, Heidelberg, Mar. 2007. DOI: 10.1007/978-3-540-70928-2_56.

[4] H. Xie, M. Zhang, and P. Andreae. Another investigation on tournament selection: modelling and visualisation. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1468–1475. ACM, 2007.