Artificial selection methods from evolutionary computing show promise for directed evolution of microbes

Alexander Lalejini [1,2], Emily Dolson [3,4], Anya E. Vostinar [5], Luis Zaman [1,2]

Corresponding: Alexander Lalejini (`lalejini@umich.edu`)

1. Department of Ecology and Evolutionary Biology, University of Michigan, Ann Arbor, MI, USA
2. Center for the Study of Complex Systems, University of Michigan, Ann Arbor, MI, USA
3. Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA
4. Program in Ecology, Evolution, and Behavior, Michigan State University, East Lansing, MI, USA
5. Computer Science Department, Carleton College, Northfield, MN, USA

Abstract.

Directed microbial evolution harnesses evolutionary processes in the laboratory to construct microorganisms with enhanced or novel functional traits. Attempting to direct evolutionary processes for applied goals is fundamental to evolutionary computation, which harnesses the principles of Darwinian evolution as a general purpose search engine for solutions to challenging computational problems. Despite their overlapping approaches, artificial selection methods from evolutionary computing are not commonly applied to living systems in the laboratory. In this work, we ask if parent selection algorithms—procedures for choosing promising progenitors—from evolutionary computation might be useful for directing the evolution of microbial populations when selecting for multiple functional traits. To do so, we introduce an agent-based model of directed microbial evolution, which we used to evaluate how well three selection algorithms from evolutionary computing (tournament selection, lexicase selection, and non-dominated elite selection) performed relative to methods commonly used in the laboratory (elite and top-10% selection). We found that multi-objective selection techniques from evolutionary computing (lexicase and non-dominated elite) generally outperformed the commonly used directed evolution approaches when selecting for multiple traits of interest. Our results motivate ongoing work transferring these multi-objective selection procedures into the laboratory. Additionally, our findings suggest that more sophisticated artificial selection methods from evolutionary computation should also be evaluated for use in directed microbial evolution.

Keywords: directed evolution, artificial selection, evolutionary computing, selection schemes, digital organisms, agent-based modeling

1 Introduction

Directed evolution harnesses laboratory artificial selection to generate biomolecules or organisms with desirable functional traits (Arnold, 1998; Sánchez et al., 2021). The scale and specificity of artificial selection has been revolutionized by a deeper understanding of evolutionary and molecular biology in combination with technological innovations in sequencing, data processing, laboratory techniques, and culturing devices. These advances have cultivated growing interest in directing the evolution of whole microbial communities with functions that can be harnessed in medical, biotech, and agricultural domains (Sánchez et al., 2021).

Of course, attempting to direct evolutionary processes for applied goals has not been limited to biological systems. Evolutionary computing harnesses the principles of Darwinian evolution as a general-purpose search engine to find solutions to challenging computational and engineering problems (Fogel, 2000). As in evolutionary computing, directed evolution in the laboratory begins with a library—or population—of variants (*e.g.*, communities, genomes, or molecules). Variants are scored based on a phenotypic trait (or set of traits) of interest, and the variants with the "best" traits are chosen to produce the next generation. Such approaches to picking progenitors are known as elitist selection algorithms in evolutionary computing (Bäck et al., 1997). Evolutionary computing research has shown that these elitist approaches to artificial selection can be sub-optimal in complex search spaces. On their own, elitist selection schemes fail to maintain diversity, which can lead to premature convergence (Hernandez, Lalejini, & Ofria, 2021; Lehman & Stanley, 2011a), and they lack mechanisms to balance multiple objectives. Artificial selection routines (*i.e.*, parent selection algorithms or selection schemes) are intensely studied in evolutionary computing, and many *in silico* selection techniques have been developed that improve the quality and diversity of evolved solutions (*e.g.,* Goings et al., 2012; Goldberg, Richardson, et al., 1987; Hornby, 2006; Lehman & Stanley, 2011b; Mouret & Clune, 2015;

Spector, 2012).

Given their success, we expect that artificial selection methods developed for evolutionary computing will improve the efficacy of directed microbial evolution in the laboratory, especially when simultaneously selecting for more than one trait (a common goal in evolutionary computation). However, directed microbial evolution differs from evolutionary computing in ways that may inhibit our ability to predict which techniques are most appropriate to apply in the laboratory. For example, candidate solutions in evolutionary computing are evaluated individually, resulting in high-resolution genotypic and phenotypic information that can be used for selecting parents, which are then copied, recombined, and mutated to produce offspring. In directed microbial evolution, individual-level evaluation is often intractable at the scale required for directed evolution; as such, evaluation often occurs at the population-level, and the highest performing populations are partitioned (instead of copied) to create "offspring" populations. Moreover, when traits of interest do not benefit individuals' reproductive success, population-level artificial selection may work against individual-level selection, which increases the difficulty of steering evolution.

Here, we ask if artificial selection techniques developed for evolutionary computing might be useful for directing the evolution of microbial populations when selecting for multiple traits of interest: both for enhancing multiple traits in a single microbial strain and for producing a set diverse strains that specialize on different traits. To do so, we developed an agent-based model of directed evolution wherein we evolve populations of self-replicating computer programs that perform computation that contributes either to the phenotype of the individual or the phenotype of the population. Using our model, we evaluated how well three selection techniques from evolutionary computing (tournament, lexicase, and non-dominated elite selection) performed in a setting that mimics directed evolution on functions measurable at the population-level. Overall, we found that multi-objective selection techniques (lexicase and non-dominated elite selection) generally outperformed the selection schemes commonly applied to directed microbial evolution (elite and top-10%). In particular, our findings suggest that lexicase selection is a good candidate technique to transfer into the laboratory, especially when aiming to evolve a diverse set of specialist microbial populations. Additionally, we found population-level artificial selection can improve directed evolution outcomes even when functional traits of interest can be tied to individual-level reproductive success.

These findings lay the foundation for strengthened communication between the evolutionary computing and directed evolution communities. The evolution of biological organisms (both natural and artificial) inspired the origination of evolutionary computation, and insights from evolutionary biology are regularly applied to evolutionary computing. As evolutionary computation has immense potential as a system for studying how to control laboratory evolution, these communities are positioned to form a virtuous cycle where insights from evolutionary computing are then applied back to directing the evolution of biological organisms. With this work, we seek to strengthen this feedback loop.

2 Directed evolution

Humans have harnessed evolution for millennia, applying artificial selection (knowingly and unknowingly) to domesticate a variety of animals, plants, and microorganisms (Cobb et al., 2013; Driscoll et al., 2009; Hill & Caballero, 1992; Libkind et al., 2011). More recently, a deeper understanding of evolution, genetics, and molecular biology in combination with technological advances have extended the use of artificial selection beyond domestication and conventional selective breeding. For example, artificial selection has been applied to biomolecules (Beaudry & Joyce, 1992; Chen & Arnold, 1993; Esvelt et al., 2011), genetic circuits (Yokobayashi et al., 2002), microoganisms (Ratcliff et al., 2012), viruses (Burrowes et al., 2019; Maheshri et al., 2006), and whole microbial communities (Goodnight, 1990; Sánchez et al., 2021; Swenson et al., 2000). In this work, we focus on directed microbial evolution.

One approach to artificial selection is to configure organisms' environment such that desirable traits are linked to growth or survival (referred to as "selection-based methods" (Wang et al., 2021)). In some sense, these selection-based methods passively harness artificial selection, as individuals with novel or enhanced functions of interest will tend to outcompete other conspecifics without requiring intervention beyond initial environmental manipulations. In combination with continuous culture devices, this approach to directing evolution can be used to achieve high throughput microbial directed evolution, "automatically" evaluating many variants without manual analysis (DeBenedictis et al., 2021; Toprak et al., 2012; Wang et al., 2021). For example, to study mechanisms of antibiotic resistance, researchers have employed morbidostats that continuously monitor the growth of evolving microbial populations and dynamically adjust antibiotic concentrations to maintain constant selection on further resistance (Toprak et al., 2012). However, linking desirable traits to organism survival can be challenging, requiring substantial knowledge about the organisms and the functions of interest.

Similar to conventional evolutionary algorithms, "screening-based methods" of directed evolution assess each variant individually and choose the most promising to propagate (Wang et al., 2021). Overall, screening-based methods are more versatile than selection-based methods because traits that are desirable can be directly discerned. However, screening requires more manual intervention and thus limits throughput. In addition to their generality, screening-based methods also allow practitioners to more easily balance the relative importance of multiple objectives, such as yield, seed size, drought tolerance, *et cetera* in plant breeding (Bruce et al., 2019; Cooper et al., 2014).

In this work, we investigate screening-based methods of directed microbial evolution, as many insights and techniques from evolutionary computation are directly applicable. When directing microbial evolution, screening is applied at the population (or community) level (Sánchez et al., 2021; Xie & Shou, 2021). During each cycle of directed microbial evolution, newly founded populations grow over a maturation period in which members of each population reproduce, mutate, and evolve. Next, populations are assessed, and promising populations are chosen as “parental populations” that will be partitioned into the next generation of “offspring populations”.

Screening-based artificial selection methods are analogous to parent selection algorithms or *selection schemes* in evolutionary computing. We know from evolutionary computing research that the most effective selection scheme depends on a range of factors, including the number of objectives (*e.g.*, single- versus multi-objective), the form and complexity of the search space (*e.g.*, smooth versus rugged), and the practitioner’s goal (*e.g.*, generating a single solution versus many different solutions). Conventionally, however, screening-based methods of directing microbial evolution choose the overall “best” performing populations to propagate (*e.g.*, the single best population or the top 10% (Xie et al., 2019)).

To the best of our knowledge, the more sophisticated methods of choosing progenitors from evolutionary computing have not been applied to directed evolution of microbes. However, artificial selection techniques from evolutionary computing have been applied in a range of other biological applications. For example, multi-objective evolutionary algorithms have been applied to DNA sequence design (Chaves-González, 2015; Shin et al., 2005); however, these applications are treated as computational optimization problems. A range of selection schemes from evolutionary computing have also been proposed for both biomolecule engineering (Currin et al., 2015; Handl et al., 2007) and agricultural selective breeding (especially for scenarios where genetic data can be exploited) (Ramasubramanian & Beavis, 2021). For example, using an NK landscape model, O’Hagan et al. evaluated the potential of elite selection, tournament selection, fitness sharing, and two rule-based learning selection schemes for selective breeding applications (O’Hagan et al., 2012). Inspired by genetic algorithms, island model approaches (Tanese, 1989) have been proposed for improving plant and animal breeding programs (Ramasubramanian & Beavis, 2021; Yabe et al., 2016), and Akdemir et al. applied multi-objective selection algorithms like non-dominated selection to plant and animal breeding (Akdemir et al., 2019). In each of these applications, however, artificial selection acted as screens on individuals and not whole populations; therefore, our work focuses on screening at the population-level in order to test the applicability of evolutionary computing selection algorithms as general-purpose screening methods for directed microbial evolution.

3 Digital Directed Evolution

Conducting directed evolution experiments in the laboratory can be slow and labor intensive, making it difficult to evaluate and tune new approaches to artificial selection *in vitro*. We could draw directly from evolutionary computing results when transferring techniques into the laboratory, but the extent to which these results would predict the efficacy (or appropriate parameterization) of a given algorithm in a laboratory setting is unclear. To address this, we developed an agent-based model of directed evolution of microbes for evaluating which techniques from evolutionary computing might be most applicable in the laboratory.

Figure 1 overviews our model of laboratory directed microbial evolution. Our model contains a population of populations (*i.e.*, a “metapopulation”). Each population comprises digital organisms (self-replicating computer programs) that compete for space in a well-mixed virtual environment. Both the digital organisms and their virtual environment are inspired by those of the Avida Digital Evolution Platform (Ofria et al., 2009), which is a well-established study system for *in silico* evolution experiments (*e.g.,* A. Lalejini et al., 2021; Lenski et al., 1999; Lenski et al., 2003; Zaman et al., 2014) and is a closer analog to microbial evolution than conventional evolutionary computing systems. However, we note that our model's implementation is fully independent of Avida, as the Avida software

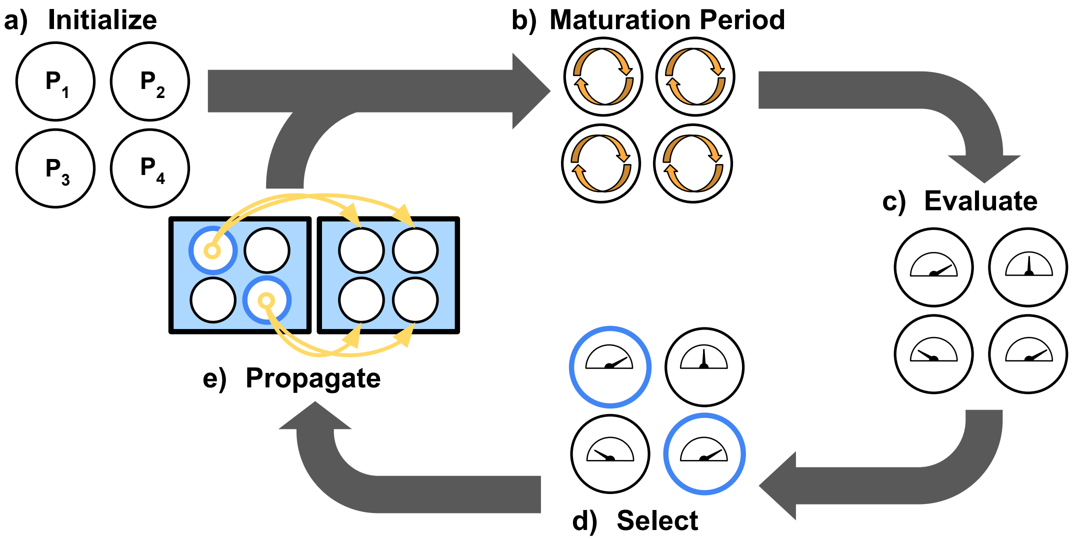

Figure 1: Overview of our model of directed microbial evolution. In (a), we found each of N populations with a single digital organism. Next (b), we give each population a maturation period during which organisms reproduce, mutate, and evolve. After maturation, (c) we evaluate each population based on population-level characteristics, and (d) we select populations (repeats allowed) to partition into N "offspring" populations (e).

platform does not allow for us to model laboratory setups of directed microbial evolution (as described in the previous section).

In our model, we initialize each population with a digital organism capable only of self-replication (Figure 1a). After initialization, directed evolution proceeds in cycles. During a cycle, we allow each population to evolve for a "maturation period" that comprises a fixed number of time steps (Figure 1b). We then evaluate each population's performance on a set of objectives (Figure 1c), and we select performant populations as "parental" populations (Figure 1d). To create an "offspring" population (Figure 1e), we use a random sample of digital organisms from the chosen parental population; in this work, we used 1% of the maximum population size.

3.1 Digital Organisms

Each digital organism is defined by a sequence of program instructions (its genome) and a set of virtual hardware components used to interpret and express those instructions. The virtual hardware and genetic representation used in this work extends that of (E. Dolson et al., 2019; Hernandez, Lalejini, & Dolson, 2021). The virtual hardware includes the following components: an instruction pointer indicating the position in the genome currently being executed, sixteen registers for performing computations, sixteen memory stacks, input and output buffers, "scopes" that facilitate modular code execution, and machinery to facilitate self-copying. For brevity, we refer readers to supplemental material for a more detailed description of these virtual hardware components (A. Lalejini et al., 2022).

Digital organisms express their genomes sequentially unless the execution of one instruction changes which instruction should be executed next (*e.g.*, "if" instructions). The instruction set is Turing Complete and syntactically robust such that any ordering of instructions is valid (though not necessarily useful). The instruction set includes operators for basic math, flow control (*e.g.*, conditional logic and looping), designating and triggering code modules, input, output, and self-replication. Each instruction contains three arguments, which may modify the effect of the instruction, often specifying memory locations or fixed values. We further document the instruction set in our supplemental material.

Digital organisms reproduce asexually by copying their genome instruction-by-instruction and then executing a `divide` instruction. However, copying is imperfect and can result in single-instruction and single-argument substitution mutations. We configured copy operations to err at an expected rate of one instruction per 100 copied and one argument per 200 copied. Genomes were fixed at a length of 100 instructions. When an organism replicates, its offspring is placed in a random position in the population, replacing any previous occupant. We limited the maximum population size to 1,000 organisms. As such, improvements to the rate of self-replication are advantageous in the competition for space within a population.

During evolution, organism replication can be improved two ways: by improving genome efficiency or by increasing the rate of genome expression ("metabolic rate"). An organ-

Function	# Inputs	Benefit
ECHO	1	Individual
NAND	2	Individual
NOT	1	Population
ORNOT	2	Population
AND	2	Population
OR	2	Population
ANDNOT	2	Population
NOR	2	Population
XOR	2	Population
EQU	2	Population
$2A$	1	Individual
A^2	1	Population
A^3	1	Population
$A + B$	2	Population
$A \times B$	2	Population
$A - B$	2	Population
$A^2 + B^2$	2	Population
$A^3 + B^3$	2	Population
$A^2 - B^2$	2	Population
$A^3 - B^3$	2	Population
$\frac{A+B}{2}$	2	Population

Table 1: Computational functions that conferred individual-level or population-level benefits. The particular functions were chosen to be used in our model based on those used in the Avida system (Bryson et al., 2021). In all experiments, we included two versions of ECHO (each for different input values), resulting in 22 possible functions that organisms could perform.

ism's metabolic rate determines the speed at which it executes its genome. Digital organisms can improve their metabolic rate by performing designated functions (referred to as individual-level functions), including some Boolean logic functions and simple mathematical expressions (Table 1). Organisms can perform functions by executing the `input` instruction to get numeric values from the environment, performing computations on those values, and executing an `output` instruction with the results. When an organism produces output, we check to see if the output completes any of the designated functions (Table 1); if so, the organism's metabolic rate is adjusted accordingly. We guarantee that the set of inputs received by an organism result in a unique output for each designated function. Organisms benefit from performing each function only once, preventing multiple rewards for repeating a single function result. In this work, we configured each function that confers an individual-level benefit to double an organism's metabolic rate, which doubles the rate the organism can copy itself.

3.2 Population-level Evaluation

In addition to individual-level functions, organisms can perform 18 different population-level functions (Table 1). Unless stated otherwise, performing a population-level function does not improve an organism's metabolic rate. Instead, population-level functions are used for population-level evaluation and selection, just as we might screen for the production of different biomolecules in laboratory populations. We assigned each population a score for each population-level function based on the number of organisms that performed that function during the population's maturation period. The use of these scores for selecting progenitors varied by selection scheme (as described in Section 4.1).

While population-level functions benefit a population's chance to propagate, they do not benefit an individual organism's immediate reproductive success: time spent computing population-level functions is time not spent on performing individual-level functions or self-replicating. Such conflicts between group-level and individual-level fitness are well-established in evolving systems (Simon et al., 2013; Waibel et al., 2009), and are indeed a problem recognized for screening-based methods of artificial selection that must be applied at the population-level (Brenner et al., 2008; Escalante et al., 2015).

4 Methods

Using our model of laboratory directed evolution, we investigated if selection schemes from evolutionary computing might be useful for directed evolution of microbes. Specifically, we compared two selection schemes used in directed evolution (elite and top-10% selection) with three other methods used in evolutionary computing (tournament, lexicase, and non-dominated elite selection). Additionally, we ran two controls that ignored population-level performance.

We conducted three independent experiments. First, we evaluated the relative performance of parent selection algorithms in a conventional evolutionary computing context, which established baseline expectations for subsequent experiments using our model of laboratory directed evolution. Next, we compared parent selection algorithms using our model of laboratory directed evolution in two contexts. In the first context, we did not link population-level functions (Table 1) to organism survival to evaluate how well each parent selection algorithm performs as a screening-based method of artificial selection. In the second context, we tested whether any of the selection schemes still improve overall directed evolution outcomes even when organism survival *is* aligned with population-level functions.

4.1 Selection Schemes

4.1.1 Elite and top-10% selection

Elite and top-10% selection are special cases of truncation selection (Mühlenbein & Schlierkamp-Voosen, 1993) or (μ, λ) evolutionary strategies (Bäck et al., 1991) wherein candidates are ranked and the most performant are chosen as progenitors. We implement these selection methods as they are used in laboratory directed evolution (Xie & Shou, 2021; Xie et al., 2019). Here, both elite and top-10% selection rank populations according to their aggregate performance on all population-level functions. Elite selection chooses the single best performing population to generate the next metapopulation, and top-10% chooses the best 10% (rounded up to the nearest whole number) as parental populations.

4.1.2 Tournament selection

Tournament selection is one of the most common parent selection methods in evolutionary computing. To select a parental population, T populations are randomly chosen from the metapopulation to form a tournament ($T = 4$ in this work). The population with the highest aggregate performance on all population-level functions wins the tournament and is chosen as a parent. We run N tournaments to choose the parental populations for each of N offspring populations.

4.1.3 Lexicase selection

Unlike the previously described selection schemes, lexicase selection does not aggregate measures of performance across population-level functions (*i.e.*, objectives) to choose parental populations. Instead, lexicase selection considers performance on each population-level function independently. For each parent selection event, all members of the metapopulation are initially considered candidates for selection. To select a parental population, the set of population-level functions are shuffled and considered in sequence. Each function (in shuffled order) is then used to sequentially filter candidates, removing all but the best candidates from further consideration. This process continues until only one candidate remains to be selected or until all functions have been considered; if more than one candidate remains, one is selected at random.

Lexicase selection was originally proposed for test-based genetic programming problems (Helmuth et al., 2015; Spector, 2012), but has since produced promising results in a variety of domains (Aenugu & Spector, 2019; La Cava et al., 2016; Metevier et al., 2019; Moore & Stanton, 2017). By randomly permuting the objectives for each parent selection, lexicase selection maintains diversity (E. L. Dolson et al., 2018; Helmuth et al., 2016), which improves search space exploration (Hernandez, Lalejini, & Ofria, 2021) and overall problem-solving success (Helmuth & Spector, 2015; Hernandez, Lalejini, & Dolson, 2021). In particular, lexicase selection focuses on maintaining specialists (Helmuth et al., 2019).

4.1.4 Non-dominated elite selection

Non-dominated elite selection is a simple multi-objective selection algorithm that chooses all populations that are not Pareto dominated by another population (Zitzler, 1999). A candidate, c_a, Pareto dominates another candidate, c_b, if the following two conditions are met: (1) c_a performs no worse than c_b on all population-level functions, and (2) c_a has strictly better performance than c_b on at least one population-level function. After identifying all non-dominated populations, these populations are selected with replacement to found each offspring population.

Pareto domination is a fundamental component in many successful evolutionary multi-objective optimization (EMOO) algorithms (Deb et al., 2002; Fonseca & Fleming, 1995; Horn et al., 1994; Zitzler, 1999). In general, EMOO algorithms aim to produce the set of solutions with optimal trade-offs of the objective set. Most EMOO algorithms have more sophisticated routines for parent selection than non-dominated elite selection (*e.g.*, use of external archives or crowding metrics). We opted to use non-dominated elite selection for its simplicity, but future work will explore more EMOO selection schemes.

4.1.5 Selection controls

We used random and no selection as controls. Random selection chooses a random set of populations (with replacement) to serve as parental populations. "No selection" chooses all populations in the metapopulation as sources for founding the next generation of populations; that is, each population is chosen to produce one offspring population. Both controls apply no selection pressure for performing population-level functions.

4.2 Experimental design

4.2.1 Establishing baseline problem-solving expectations in an evolutionary computing context

First, we evaluated the relative performance of parent selection algorithms in a conventional evolutionary computing context (linear genetic programming (Brameier & Banzhaf, 2007)), in which we evolved programs to compute the functions in Table 1. This control experiment allowed us to verify that the genetic representation used by digital organisms (Section 3.1) is sufficient for evolving each of the computational functions used in subsequent experiments. Additionally, the relative performances of each algorithm establishes an expectation for how each parent selection algorithm might perform in our model of laboratory directed evolution.

For each of the seven selection schemes described in Section 4.1, we evolved 50 replicate populations of 1,000 programs. We chose to evolve populations for 55,000 generations to approximate the number of digital organism generations that elapsed in our directed evolution experiments (based on exploratory runs). We used the same genetic representation as described in Section 3.1; however, we excluded self-replication instructions from the instruction set, as we did not require programs to copy themselves during this experiment.

Each program was evaluated independently to determine its phenotype. To evaluate a program, we executed it for 200 time steps, and we tracked its inputs and outputs to determine which of the functions in Table 1 it performed (if any). For the purpose of selection, we treated each of the 22 possible functions as a pass-fail task. Lexicase and non-dominated elite selection considered each task separately to choose parents, while elite, top-10%, and tournament selection used the number of task passes as fitness values for choosing parents. Chosen parents reproduced asexually, and we applied mutations to offspring of the same types and frequencies as in our model of laboratory directed evolution (Section 3.1).

At the end of each run, we identified the program that performed the most tasks, and we compared these values across treatments. We considered a run to be successful if it produced a program capable of performing all 22 tasks during evaluation.

4.2.2 Applying parent selection algorithms in a digital directed evolution context

Next, we evaluated each selection scheme's performance in our model of laboratory directed evolution. For each selection scheme, we ran 50 independent replicates of directed evolution for 2,000 cycles of population maturation, screening, and propagation (as shown in Figure 1). During each cycle, we gave populations a maturation period of 200 updates[1] (approximately 25 to 35 generations). Within each replicate, the metapopulation comprised 96 populations (following the number of samples held by a standard microtiter plate used in laboratory experiments), each with a maximum carrying capacity of 1,000 digital organisms. During a population's maturation period, we measured the number of organisms that performed each of the 18 population-level functions (Table 1) as the population's "phenotype" for evaluation. We selected populations to propagate according to the treatment-specific selection scheme, and propagated chosen parental populations as

[1] One update is the amount of time required for the average organism in a population to execute 30 instructions.

described in Section 3.

At the end of the experiment, we analyzed the population-level functions performed by populations in each replicate's metapopulation. First, we calculated each population's "task profile", which is a binary vector that describes which population-level functions are "covered" by the population (zeroes are assigned for functions that are not covered and ones for those that are covered). A function is considered covered if it is performed by at least 50 organisms (a threshold ensuring the performance was not one-off) during a given maturation period.

Next, we measured the "best population task coverage" and "metapopulation task coverage" for each replicate. Best population task coverage is measured as the number of functions covered by the population with the largest set of covered functions. Metapopulation task coverage is measured as the number of functions covered across the entire metapopulation (*i.e.*, the union of unique tasks covered by each population in the metapopulation).

We also measured the phenotypic diversity within each metapopulation. Specifically, we measured the number of different task profiles present in the metapopulation (*i.e.*, phenotypic richness), and we measured the "spread" of task profiles in the metapopulation. To measure a metapopulation's task profile spread, we calculated a centroid task profile as the average of all task profiles in the metapopulation, and then we calculated the average normalized cosine distance between each population's task profile and the centroid. A metapopulation's task spread summarizes how different the constituent populations' task profiles are from one another.

4.2.3 Evaluating whether selection schemes improve directed evolution outcomes when population-level functions are aligned with organism survival

Selection-based methods of artificial selection tie desired traits to organism survival, eliminating the need to apply screening-based methods to populations. We tested whether the addition of population-level selection improves directed evolution outcomes even when traits of interest (population-level functions) are selected for at the individual level (*i.e.*, tied to organism survival). To do so, we repeated our previously described directed evolution experiment (Section 4.2.2), except we configured all population-level functions to improve an organism's metabolic rate in addition to the individual-level functions. As such, all population-level functions were beneficial in all treatments, including the random and no selection controls. However, only treatments with non-control selection schemes applied artificial selection at the population-level.

4.3 Statistical Analyses

In general, we differentiated between sample distributions using non-parametric statistical tests. For each major analysis, we first performed a Kruskal-Wallis test (Kruskal & Wallis, 1952) to determine if there were significant differences in results across treatments (significance level $\alpha = 0.05$). If so, we applied a Wilcoxon rank-sum test (Wilcoxon, 1992) to distinguish between pairs of treatments, using a Bonferroni correction for multiple comparisons (Rice, 1989). Due to space limitations, we do not report all pairwise comparisons in our main results; however, all of our statistical results are included in our supplemental material.

4.4 Software and Data Availability

Our model of laboratory directed evolution is available on GitHub (see A. Lalejini et al., 2022) and is implemented using the Empirical scientific software library (Ofria et al., 2020).

We conducted all statistical analyses with R version 4.04 (R Core Team, 2021), using the following R packages for data analysis and visualization: tidyverse (Wickham et al., 2019), ggplot2 (Wickham, 2016), cowplot (Wilke, 2020), viridis (Garnier, 2018), and Color Brewer (Harrower & Brewer, 2003; Neuwirth, 2014). Our source code for experiments, analyses, and visualizations is publicly available on GitHub (see A. Lalejini et al., 2022). Additionally, our experiment data are publicly archived on the Open Science Framework (see A. M. Lalejini, 2022).

5 Results and Discussion

5.1 Baseline problem-solving expectations in an evolutionary computing context

First, we established baseline performance expectations for the selection schemes in a conventional genetic programming context to validate the solvability of the individual- and population-level functions used in our digital directed evolution experiments. Two selection schemes produced successful replicates, where success is defined as evolving a program capable of performing all 22 functions: elite (1/50) and lexicase selection (47/50). No solutions evolved in any other treatment. Figure 2 depicts the number of functions performed by the best program from each replicate. All selection schemes outperformed the random and no selection controls. Differences between all pairs except random and no selection were statistically significant (Bonferroni-corrected Wilcoxon rank-sum, $p < 0.01$). Lexicase selection was the most performant followed by top-10%, elite, tournament, and non-dominated elite selection.

These data confirm that our genetic representation allows for the evolution of each computational function used in our model of laboratory directed evolution. Moreover, these data establish some expectations for the relative performance of each selection scheme in

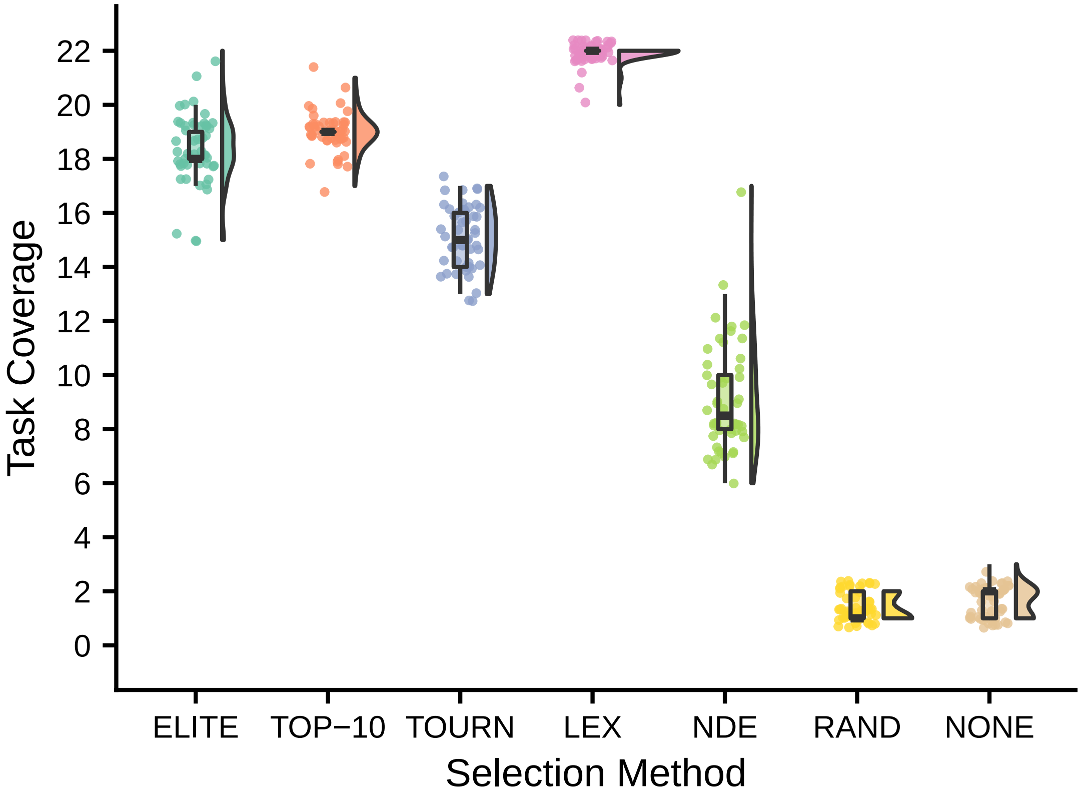

Figure 2: Task coverage of the best program (per replicate) evolved in an evolutionary computing context. Selection scheme abbreviations are as follows: TOURN is tournament, LEX is lexicase, NDE is non-dominated elite, RAND is random, and NONE is no selection. Differences among treatments were statistically significant (Kruskall-Wallis, $p < 10^{-4}$).

our directed evolution experiments. Lexicase selection's strong performance is consistent with previous work demonstrating its efficacy on program synthesis problems (Helmuth & Abdelhady, 2020; Helmuth & Spector, 2015). While initially surprised by non-dominated elite's poor performance (relative to other non-control selection schemes), we note that selection methods based on Pareto domination are rarely applied to pass-fail test-based genetic programming problems, and perhaps the course-grained function scores (0 or 1) hindered its capacity for problem-solving success.

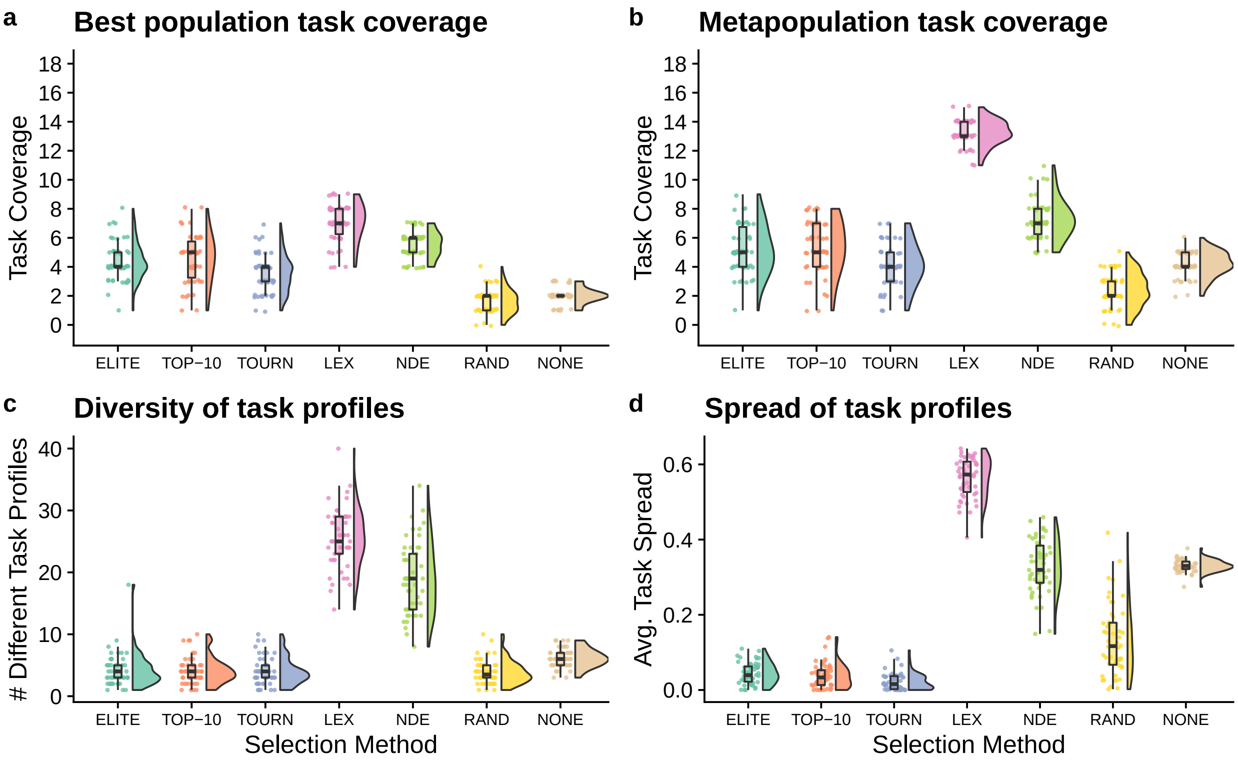

Figure 3: Digital directed evolution results. Differences among treatments were statistically significant for each panel (Kruskall-Wallis, $p < 10^{-4}$).

5.2 Lexicase and non-dominated elite selection show promise for directed evolution

Next, we compared selection scheme performance when modeling the directed evolution of digital organisms. Figures 3a and 3b show the best population and metapopulation task coverages, respectively. All selection schemes resulted in greater single-population task coverage than both random and no selection controls (Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-4}$). Metapopulation coverage under tournament selection was not significantly different than coverage under the no selection control, but all other selection schemes resulted in significantly better metapopulation coverage than both controls (Bonferroni-corrected Wilcoxon rank-sum, $p < 0.03$). Overall, lexicase and non-dominated elite selection scored the greatest population and metapopulation task coverage out of all selection schemes, and lexicase was the overall best selection scheme according to both metrics of performance.

While differences were significant on the best population-task coverage, they were not necessarily substantial. However, other measures had more substantial differences. Both multi-objective selection schemes—lexicase and non-dominated elite—had the greatest metapopulation task coverage (Figure 3b), and the greatest diversity of task profiles in the final metapopulations (Figure 3c; Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-4}$). Lexicase selection in particular also had the greatest task profile *spread* (Figure 3d; Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-4}$), which is consistent with previous results demonstrating that lexicase excels at maintaining diverse specialists (E. L. Dolson et al., 2018; Helmuth et al., 2016; Helmuth et al., 2019; Hernandez, Lalejini, & Ofria, 2021).

We hypothesized that lexicase and non-dominated elite selection's mechanisms for selecting different *types* of parental populations underpinned their improved performance over elite, top-10%, and tournament selection. This, however, is confounded by each selection scheme's varying capacity to select a greater number of different populations (regardless of differences in those selected). As such, we asked whether lexicase and non-dominated elite's success could be explained by a capacity to select a greater number of different parental populations. Elite selection selected exactly one population per cycle, top-10% selected 10, lexicase selected an average of 12, tournament selected an average of 50, and non-dominated elite selected an average of 83 different populations. Thus, we can rule out the number of populations selected per cycle as the sole explanation for lexicase selection's success; we argue that this, in combination with our diversity data, suggests that directed evolution practitioners should consider incorporating mechanisms for selecting phenotypically diverse parental populations into their artificial selection approaches.

These results are also informative when compared to our genetic programming control experiment (Figure 2). While results across these two contexts are not directly comparable, we argue that, taken together, our experiments suggest that steering evolution at the population-level is more challenging than steering at the individual-level. For example, across all treatments, no single population in our model of directed evolution performed all 18 population-level functions. Yet, after a similar number of organism-level generations ($\sim 55,000$), both elite and lexicase selection produced programs capable of all 22 functions in a genetic programming context; even after only 2,000 generations (the number of cycles in our directed evolution experiments), we found that conventional genetic programming produced more performant programs than those evolved under our model of laboratory directed evolution (supplemental material A. Lalejini et al., 2022). We also observed differences in the rank order of selection schemes between experiments. For example, non-dominated elite selection performed poorly in a genetic programming context relative to the other non-control selection schemes; however, non-dominated elite outperformed all selection schemes except lexicase selection in our model of laboratory directed evolution. On its own, non-dominated elite's difference in performance is not surprising, as non-dominated elite selection is not conventionally used for evolving computer programs where evaluation criteria are evaluated on a pass-fail basis. More broadly, however, we argue that this result highlights modeling as an important intermediate step when evaluating which techniques from evolutionary computing are likely to be effective in a laboratory setting.

5.3 Selection schemes improve outcomes even when organism survival can be tied to population-level functions

Next, we tested whether the addition of population-level screening improves directed evolution outcomes even when population-level functions can be tied to organism survival. Overall, each non-control selection scheme resulted in better single-population task coverage than either control treatment (Figure 4a; Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-4}$). We did not find significant differences in best population coverage among elite, top-10%, tournament, and non-dominated elite selection. In contrast to our previous

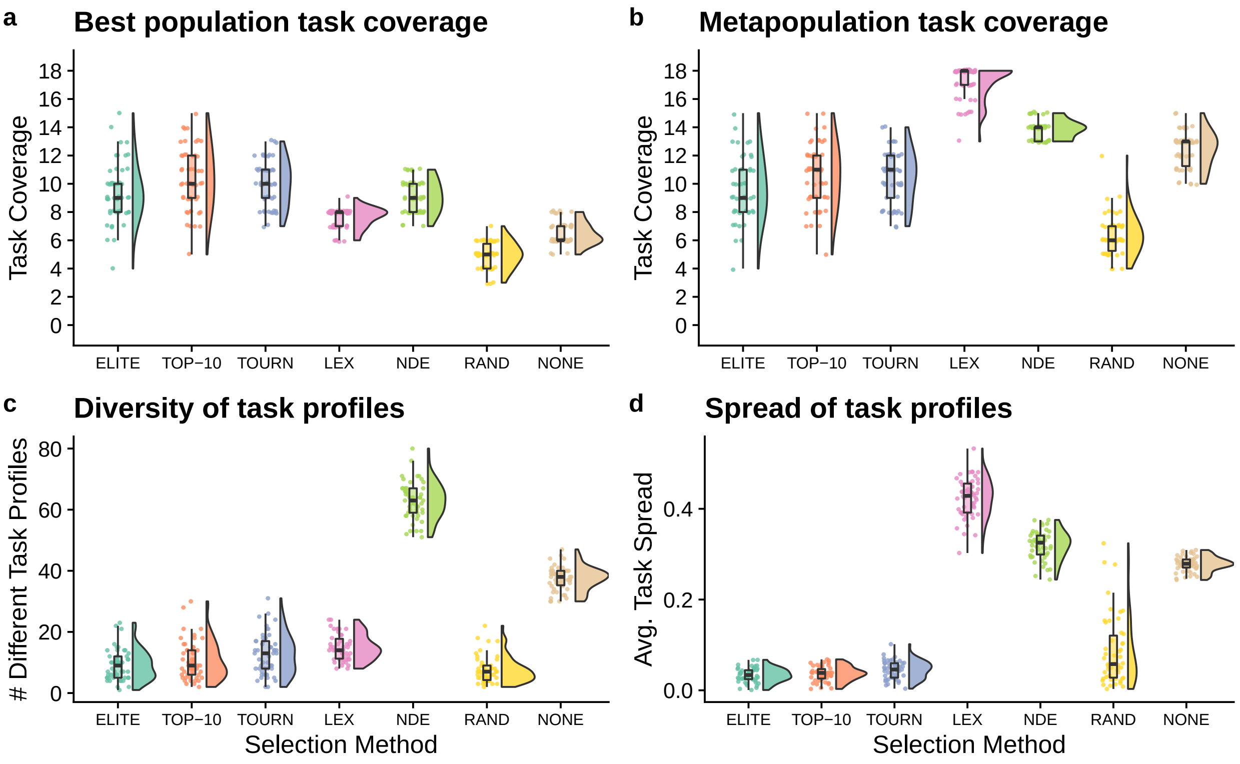

Figure 4: Digital directed evolution results when organism survival is tied to population-level functions. Differences among treatments were statistically significant for each panel (Kruskall-Wallis, $p < 10^{-4}$).

experiment, lexicase selection resulted in lower best population coverage than each other non-control selection scheme (Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-4}$).

Lexicase selection, however, outperformed all other selection schemes on metapopulation task coverage (Figure 4b; Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-4}$), producing 30 metapopulations that cover all 18 population-level functions. In general, lexicase selection produced metapopulations containing distinct specialist populations, resulting in high metapopulation task coverage while each specialist population had low task coverage on its own. Indeed, while lexicase metapopulations did not necessarily comprise many different population task profiles (Figure 4c), the task profiles were very different from one another (Figure 4d).

Of our two control selection methods, we found that performing no selection was better than random selection for both single-population and metapopulation task coverage

(Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-4}$). In fact, performing no selection at all resulted in better metapopulation task coverage than elite, top-10%, and tournament selection (Bonferroni-corrected Wilcoxon rank-sum test, $p < 10^{-3}$). We hypothesize that this result is because elite, top-10%, and tournament selection converge to metapopulations with homogeneous task profiles, while performing no selection at all allows populations to diverge from one another.

6 Conclusion

In this work, we investigated whether the selection schemes from evolutionary computing might be useful for directing the evolution of microbial populations. To do so, we introduced an agent-based model of laboratory directed evolution. Overall, our results suggest that lexicase and non-dominated elite selection are promising techniques to transfer into the laboratory when selecting for multiple traits of interest, as both of these selection schemes resulted in improved outcomes relative to conventional directed evolution selection methods. In particular, we expect lexicase selection to be especially useful for evolving a set of microbial populations, each specializing on different population-level functions. We also found that the addition of screening-based methods of artificial selection can improve directed evolution outcomes in cases where organisms' reproductive success can be tied to traits of interest.

Our study has several important limitations that warrant future model development and experimentation. For example, we focused on modeling microbial populations that grow (and evolve) in a simple environment without complex ecological interactions. We plan to add ecological dynamics by incorporating features such as limited resources, waste byproducts, symbiotic interactions, and spatial structure. These extensions will allow us to model the directed evolution of complex microbial communities (*e.g.*, Sánchez et al., 2021; Xie & Shou, 2021), which is an emerging frontier in laboratory directed evolution.

In this study, we compared simple versions of each selection scheme. We plan to test more sophisticated selection schemes as we continue to transfer techniques developed for evolutionary computation into the laboratory. For example, non-dominated elite selection is one of the simplest methods that uses Pareto domination to choose parents; given its strong performance, we see more sophisticated multi-objective selection algorithms (*e.g.*, NSGA-II (Deb et al., 2002)) as particularly promising for laboratory directed evolution. Lexicase selection variants are also promising for laboratory directed evolution: epsilon lexicase (La Cava et al., 2016; Spector et al., 2018) might be useful when population-level characteristics are measured as real-valued quantities, and cohort lexicase selection (Hernandez et al., 2019) could reduce the amount of screening required to select parental populations. Beyond selection schemes, we also see quality diversity algorithms (Hagg, 2021) (*e.g.*, MAP-Elites (Mouret & Clune, 2015)) as promising techniques to transfer into the laboratory.

We see digital experiments like the ones reported here as a critical step for transferring techniques developed for evolutionary computing into the laboratory. Indeed, our results are currently informing the design of laboratory experiments that apply evolutionary computing techniques to the directed evolution of *E. coli*. Our model of directed microbial evolution provides a testbed for rigorously evaluating different artificial selection methods with different laboratory setups (*e.g.*, metapopulation size, maturation period, *etc.*) before embarking on costly or timing consuming laboratory experiments.

Acknowledgements

We thank the ZE^3 Laboratory for thoughtful discussions, feedback, and support. This research was supported through computational resources provided by the University of Michigan's Advanced Research Computing and by Michigan State University's Institute for Cyber-Enabled Research. Additionally, this research was supported by the National

Science Foundation (NSF) through support to LZ (DEB-1813069) and a sub-award to AEV (MCB-1750125).

Supplementary information

The supplemental material for this article, including all source code, is archived (see reference) and hosted on GitHub and can be found online at https://github.com/amlalejini/directed-digital-evolution (A. Lalejini et al., 2022). The datasets generated and analyzed for this study are archived on the Open Science Framework at https://osf.io/zn63x (A. M. Lalejini, 2022).

References

Aenugu, S., & Spector, L. (2019). Lexicase selection in learning classifier systems. *Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO '19*, 356–364. https://doi.org/10.1145/3321707.3321828

Akdemir, D., Beavis, W., Fritsche-Neto, R., Singh, A. K., & Isidro-Sánchez, J. (2019). Multi-objective optimized genomic breeding strategies for sustainable food improvement. *Heredity*, *122*(5), 672–683. https://doi.org/10.1038/s41437-018-0147-1

Arnold, F. H. (1998). Design by Directed Evolution. *Accounts of Chemical Research*, *31*(3), 125–131. https://doi.org/10.1021/ar960017f

Bäck, T., Fogel, D., & Michalewicz, Z. (Eds.). (1997). *Handbook of Evolutionary Computation* (1st ed.). https://doi.org/10.1201/9780367802486

Bäck, T., Hoffmeister, F., & Schwefel, H.-P. (1991). A Survey of Evolution Strategies. *Proceedings of the Fourth International Conference on Genetic Algorithms*, 2–9.

Beaudry, A. A., & Joyce, G. F. (1992). Directed Evolution of an RNA Enzyme. *Science*, *257*(5070), 635–641. https://doi.org/10.1126/science.1496376

Brameier, M. F., & Banzhaf, W. (2007). *Linear Genetic Programming*. Springer US. https://doi.org/10.1007/978-0-387-31030-5

Brenner, K., You, L., & Arnold, F. H. (2008). Engineering microbial consortia: A new frontier in synthetic biology. *Trends in Biotechnology*, *26*(9), 483–489. https://doi.org/10.1016/j.tibtech.2008.05.004

Bruce, R. W., Grainger, C. M., Ficht, A., Eskandari, M., & Rajcan, I. (2019). Trends in Soybean Trait Improvement over Generations of Selective Breeding. *Crop Science*, *59*(5), 1870–1879. https://doi.org/10.2135/cropsci2018.11.0664

Bryson, D., Baer, B., Vostinar, A., Ofria, C., Barrick, J., Zaman, L., Goldsby, H., cholden23, cse801a, Goings, S., Dolson, E., Rupp, M., Covert, A., vogelmi3-msu, Grace, dule123, Nahum, J., Blackwood, D., Wagner, A. P., ... jcfisher. (2021). *Avida digital evolution platform software* (Version 2.14.0). Zenodo. https://doi.org/10.5281/zenodo.5068026

Burrowes, B., Molineux, I., & Fralick, J. (2019). Directed in Vitro Evolution of Therapeutic Bacteriophages: The Appelmans Protocol. *Viruses*, *11*(3), 241. https://doi.org/10.3390/v11030241

Chaves-González, J. M. (2015). Hybrid multiobjective metaheuristics for the design of reliable DNA libraries. *Journal of Heuristics*, *21*(6), 751–788. https://doi.org/10.1007/s10732-015-9298-x

Chen, K., & Arnold, F. H. (1993). Tuning the activity of an enzyme for unusual environments: Sequential random mutagenesis of subtilisin E for catalysis in dimethylformamide. *Proceedings of the National Academy of Sciences*, *90*(12), 5618–5622. https://doi.org/10.1073/pnas.90.12.5618

Cobb, R. E., Chao, R., & Zhao, H. (2013). Directed evolution: Past, present, and future. *AIChE Journal*, *59*(5), 1432–1440. https://doi.org/10.1002/aic.13995

Cooper, M., Gho, C., Leafgren, R., Tang, T., & Messina, C. (2014). Breeding drought-tolerant maize hybrids for the US corn-belt: Discovery to product. *Journal of Experimental Botany*, *65*(21), 6191–6204. https://doi.org/10.1093/jxb/eru064

Currin, A., Swainston, N., Day, P. J., & Kell, D. B. (2015). Synthetic biology for the directed evolution of protein biocatalysts: Navigating sequence space intelligently. *Chemical Society Reviews*, *44*(5), 1172–1239. https://doi.org/10.1039/C4CS00351A

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. https://doi.org/10.1109/4235.996017

DeBenedictis, E. A., Chory, E. J., Gretton, D. W., Wang, B., Golas, S., & Esvelt, K. M. (2021). Systematic molecular evolution enables robust biomolecule discovery. *Nature Methods*. https://doi.org/10.1038/s41592-021-01348-4

Dolson, E., Lalejini, A., & Ofria, C. (2019). Exploring Genetic Programming Systems with MAP-Elites. In W. Banzhaf, L. Spector, & L. Sheneman (Eds.), *Genetic Programming Theory and Practice XVI* (pp. 1–16). Springer International Publishing. https://doi.org/10.1007/978-3-030-04735-1_1

Dolson, E. L., Banzhaf, W., & Ofria, C. (2018). *Ecological theory provides insights about evolutionary computation* (preprint). PeerJ Preprints. https://doi.org/10.7287/peerj.preprints.27315v1

Driscoll, C. A., Macdonald, D. W., & O'Brien, S. J. (2009). From wild animals to domestic pets, an evolutionary view of domestication. *Proceedings of the National Academy of Sciences*, *106*, 9971–9978. https://doi.org/10.1073/pnas.0901586106

Escalante, A. E., Rebolleda-GÃ³mez, M., BenÃtez, M., & Travisano, M. (2015). Ecological perspectives on synthetic biology: Insights from microbial population biology. *Frontiers in Microbiology*, *6*. https://doi.org/10.3389/fmicb.2015.00143

Esvelt, K. M., Carlson, J. C., & Liu, D. R. (2011). A system for the continuous directed evolution of biomolecules. *Nature*, *472*(7344), 499–503. https://doi.org/10.1038/nature09929

Fogel, D. (2000). What is evolutionary computation? *IEEE Spectrum*, *37*(2), 26–32. https://doi.org/10.1109/6.819926

Fonseca, C. M., & Fleming, P. J. (1995). An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, *3*(1), 1–16. https://doi.org/10.1162/evco.1995.3.1.1

Garnier, S. (2018). *Viridis: Default color maps from 'matplotlib'* [R package version 0.5.1]. https://CRAN.R-project.org/package=viridis

Goings, S., Goldsby, H., Cheng, B. H., & Ofria, C. (2012). An ecology-based evolutionary algorithm to evolve solutions to complex problems. *Artificial Life 13*, 171–177. https://doi.org/10.7551/978-0-262-31050-5-ch024

Goldberg, D. E., Richardson, J. et al. (1987). Genetic algorithms with sharing for multimodal function optimization. *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*, 41–49.

Goodnight, C. J. (1990). Experimental Studies of Community Evolution I: The Response to Selection at the Community Level. *Evolution*, *44*(6), 1614. https://doi.org/10.2307/2409341

Hagg, A. (2021). Phenotypic Niching Using Quality Diversity Algorithms [Series Title: Natural Computing Series]. In M. Preuss, M. G. Epitropakis, X. Li, & J. E. Fieldsend (Eds.), *Metaheuristics for Finding Multiple Solutions* (pp. 287–315). Springer International Publishing. https://doi.org/10.1007/978-3-030-79553-5_12

Handl, J., Kell, D. B., & Knowles, J. (2007). Multiobjective Optimization in Bioinformatics and Computational Biology. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *4*(2), 279–292. https://doi.org/10.1109/TCBB.2007.070203

Harrower, M., & Brewer, C. A. (2003). ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps. *The Cartographic Journal*, *40*(1), 27–37. https://doi.org/10.1179/000870403235002042

Helmuth, T., & Abdelhady, A. (2020). Benchmarking parent selection for program synthesis by genetic programming. *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, 237–238. https://doi.org/10.1145/3377929.3389987

Helmuth, T., McPhee, N. F., & Spector, L. (2016). Lexicase Selection for Program Synthesis: A Diversity Analysis [Series Title: Genetic and Evolutionary Computation]. In R. Riolo, W. Worzel, M. Kotanchek, & A. Kordon (Eds.), *Genetic Programming Theory and Practice XIII* (pp. 151–167). Springer International Publishing. https://doi.org/10.1007/978-3-319-34223-8_9

Helmuth, T., Pantridge, E., & Spector, L. (2019). Lexicase selection of specialists. *Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO '19*, 1030–1038. https://doi.org/10.1145/3321707.3321875

Helmuth, T., & Spector, L. (2015). General Program Synthesis Benchmark Suite. *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference - GECCO '15*, 1039–1046. https://doi.org/10.1145/2739480.2754769

Helmuth, T., Spector, L., & Matheson, J. (2015). Solving Uncompromising Problems With Lexicase Selection. *IEEE Transactions on Evolutionary Computation*, *19*(5), 630–643. https://doi.org/10.1109/TEVC.2014.2362729

Hernandez, J. G., Lalejini, A., & Dolson, E. (2021). What can phylogenetic metrics tell us about useful diversity in evolutionary algorithms? [arXiv: 2108.12586]. *arXiv:2108.12586 [cs, q-bio]*. Retrieved December 29, 2021, from http://arxiv.org/abs/2108.12586 Comment: 21 page, 7 figures. Presented Genetic Programming in Theory and Practice, 2021

Hernandez, J. G., Lalejini, A., Dolson, E., & Ofria, C. (2019). Random subsampling improves performance in lexicase selection. *Proceedings of the Genetic and Evolutionary Computation Conference Companion on - GECCO '19*, 2028–2031. https://doi.org/10.1145/3319619.3326900

Hernandez, J. G., Lalejini, A., & Ofria, C. (2021). An Exploration of Exploration: Measuring the ability of lexicase selection to find obscure pathways to optimality [arXiv: 2107.09760]. *arXiv:2107.09760 [cs]*. Retrieved December 30, 2021, from http://arxiv.org/abs/2107.09760

Comment: Changes to the axis labels and added funding sources to acknowledgments

Hill, W. G., & Caballero, A. (1992). Artificial Selection Experiments. *Annual Review of Ecology and Systematics*, *23*(1), 287–310. https://doi.org/10.1146/annurev.es.23.110192.001443

Horn, J., Nafpliotis, N., & Goldberg, D. (1994). A niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, 82–87. https://doi.org/10.1109/ICEC.1994.350037

Hornby, G. S. (2006). ALPS: The age-layered population structure for reducing the problem of premature convergence. *Proceedings of the 8th annual conference on Genetic and evolutionary computation - GECCO '06*, 815. https://doi.org/10.1145/1143997.1144142

Kruskal, W. H., & Wallis, W. A. (1952). Use of Ranks in One-Criterion Variance Analysis. *Journal of the American Statistical Association*, *47*(260), 583–621. https://doi.org/10.1080/01621459.1952.10483441

La Cava, W., Spector, L., & Danai, K. (2016). Epsilon-Lexicase Selection for Regression. *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 741–748. https://doi.org/10.1145/2908812.2908898

Lalejini, A., Dolson, E., Vostinar, A. E., & Zaman, L. (2022). *Supplemental material for this manuscript hosted on GitHub*. https://doi.org/10.5281/zenodo.6403135

Lalejini, A., Ferguson, A. J., Grant, N. A., & Ofria, C. (2021). Adaptive Phenotypic Plasticity Stabilizes Evolution in Fluctuating Environments. *Frontiers in Ecology and Evolution*, *9*, 715381. https://doi.org/10.3389/fevo.2021.715381

Lalejini, A. M. (2022). [Data] Selection schemes from evolutionary computing show promise for directed evolution of microbes. https://doi.org/10.17605/OSF.IO/ZN63X

Lehman, J., & Stanley, K. O. (2011a). Abandoning Objectives: Evolution Through the Search for Novelty Alone. *Evolutionary Computation*, *19*(2), 189–223. https://doi.org/10.1162/EVCO_a_00025

Lehman, J., & Stanley, K. O. (2011b). Evolving a diversity of virtual creatures through novelty search and local competition. *Proceedings of the 13th annual conference on Genetic and evolutionary computation - GECCO '11*, 211. https://doi.org/10.1145/2001576.2001606

Lenski, R. E., Ofria, C., Collier, T. C., & Adami, C. (1999). Genome complexity, robustness and genetic interactions in digital organisms. *Nature*, *400*(6745), 661–664. https://doi.org/10.1038/23245

Lenski, R. E., Ofria, C., Pennock, R. T., & Adami, C. (2003). The evolutionary origin of complex features. *Nature*, *423*(6936), 139–144. https://doi.org/10.1038/nature01568

Libkind, D., Hittinger, C. T., Valerio, E., Goncalves, C., Dover, J., Johnston, M., Goncalves, P., & Sampaio, J. P. (2011). Microbe domestication and the identification of the wild genetic stock of lager-brewing yeast. *Proceedings of the National Academy of Sciences*, *108*(35), 14539–14544. https://doi.org/10.1073/pnas.1105430108

Maheshri, N., Koerber, J. T., Kaspar, B. K., & Schaffer, D. V. (2006). Directed evolution of adeno-associated virus yields enhanced gene delivery vectors. *Nature Biotechnology*, *24*(2), 198–204. https://doi.org/10.1038/nbt1182

Metevier, B., Saini, A. K., & Spector, L. (2019). Lexicase Selection Beyond Genetic Programming [Series Title: Genetic and Evolutionary Computation]. In W. Banzhaf, L. Spector, & L. Sheneman (Eds.), *Genetic Programming Theory and Practice XVI* (pp. 123–136). Springer International Publishing. https://doi.org/10.1007/978-3-030-04735-1_7

Moore, J. M., & Stanton, A. (2017). Lexicase selection outperforms previous strategies for incremental evolution of virtual creature controllers. *Proceedings of the 14th European Conference on Artificial Life ECAL 2017*, 290–297. https://doi.org/10.7551/ecal_a_050

Mouret, J.-B., & Clune, J. (2015). Illuminating search spaces by mapping elites [arXiv: 1504.04909]. *arXiv:1504.04909 [cs, q-bio]*. Retrieved November 23, 2021, from http://arxiv.org/abs/1504.04909

Mühlenbein, H., & Schlierkamp-Voosen, D. (1993). Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization. *Evolutionary Computation*, *1*(1), 25–49. https://doi.org/10.1162/evco.1993.1.1.25

Neuwirth, E. (2014). *Rcolorbrewer: Colorbrewer palettes* [R package version 1.1-2]. https://CRAN.R-project.org/package=RColorBrewer

Ofria, C., Bryson, D. M., & Wilke, C. O. (2009). Avida: A Software Platform for Research in Computational Evolutionary Biology. In M. Komosinski & A. Adamatzky (Eds.), *Artificial Life Models in Software* (pp. 3–35). Springer London. https://doi.org/10.1007/978-1-84882-285-6_1

Ofria, C., Moreno, M. A., Dolson, E., Lalejini, A., Rodriguez-Papa, S., Fenton, J., Perry, K., Jorgensen, S., Hoffman, R., Miller, R., Edwards, O. B., Stredwick, J., G, N. C., Clemons, R., Vostinar, A., Moreno, R., Schossau, J., Zaman, L., & Rainbow, D. (2020). Empirical: A scientific software library for research, education, and public engagement [doi: 10.5281/zenodo.4141943]. https://doi.org/10.5281/zenodo.4141943

O'Hagan, S., Knowles, J., & Kell, D. B. (2012). Exploiting Genomic Knowledge in Optimising Molecular Breeding Programmes: Algorithms from Evolutionary Computing (J. Bongard, Ed.). *PLoS ONE*, *7*(11), e48862. https://doi.org/10.1371/journal.pone.0048862

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Ramasubramanian, V., & Beavis, W. D. (2021). Strategies to Assure Optimal Trade-Offs Among Competing Objectives for the Genetic Improvement of Soybean. *Frontiers in Genetics*, *12*, 675500. https://doi.org/10.3389/fgene.2021.675500

Ratcliff, W. C., Denison, R. F., Borrello, M., & Travisano, M. (2012). Experimental evolution of multicellularity. *Proceedings of the National Academy of Sciences*, *109*(5), 1595–1600. https://doi.org/10.1073/pnas.1115323109

Rice, W. R. (1989). Analyzing Tables of Statistical Tests. *Evolution*, *43*(1), 223. https://doi.org/10.2307/2409177

Sánchez, Á., Vila, J. C., Chang, C.-Y., Diaz-Colunga, J., Estrela, S., & Rebolleda-Gomez, M. (2021). Directed Evolution of Microbial Communities. *Annual Review of Biophysics*, *50*(1), annurev–biophys–101220–072829. https://doi.org/10.1146/annurev-biophys-101220-072829

Shin, S.-Y., Lee, I.-H., Kim, D., & Zhang, B.-T. (2005). Multiobjective Evolutionary Optimization of DNA Sequences for Reliable DNA Computing. *IEEE Transactions on Evolutionary Computation*, *9*(2), 143–158. https://doi.org/10.1109/TEVC.2005.844166

Simon, B., Fletcher, J. A., & Doebeli, M. (2013). Towards a general theory of group selection. *Evolution*, *67*(6), 1561–1572. https://doi.org/10.1111/j.1558-5646.2012.01835.x

Spector, L. (2012). Assessment of problem modality by differential performance of lexicase selection in genetic programming: A preliminary report. *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion - GECCO Companion '12*, 401. https://doi.org/10.1145/2330784.2330846

Spector, L., Cava, W. L., Shanabrook, S., Helmuth, T., & Pantridge, E. (2018). Relaxations of Lexicase Parent Selection. In W. Banzhaf, R. S. Olson, W. Tozier, & R. Riolo (Eds.), *Genetic Programming Theory and Practice XV* (pp. 105–120). Springer International Publishing. https://doi.org/10.1007/978-3-319-90512-9_7

Swenson, W., Wilson, D. S., & Elias, R. (2000). Artificial ecosystem selection. *Proceedings of the National Academy of Sciences*, *97*(16), 9110–9114. https://doi.org/10.1073/pnas.150237597

Tanese, R. (1989). *Distributed genetic algorithms for function optimization* (PhD Thesis) [Publication Title: ProQuest Dissertations and Theses]. https://proxy.lib.umich.edu/login?url=https://www.proquest.com/dissertations-theses/distributed-genetic-algorithms-function/docview/303814520/se-2?accountid=14667

Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2021-10-04

Toprak, E., Veres, A., Michel, J.-B., Chait, R., Hartl, D. L., & Kishony, R. (2012). Evolutionary paths to antibiotic resistance under dynamically sustained drug selection. *Nature Genetics*, *44*(1), 101–105. https://doi.org/10.1038/ng.1034

Waibel, M., Keller, L., & Floreano, D. (2009). Genetic Team Composition and Level of Selection in the Evolution of Cooperation. *IEEE Transactions on Evolutionary Computation*, *13*(3), 648–660. https://doi.org/10.1109/TEVC.2008.2011741

Wang, Y., Xue, P., Cao, M., Yu, T., Lane, S. T., & Zhao, H. (2021). Directed Evolution: Methodologies and Applications. *Chemical Reviews*, acs.chemrev.1c00260. https://doi.org/10.1021/acs.chemrev.1c00260

Wickham, H. (2016). *Ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., ... Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Wilcoxon, F. (1992). Individual Comparisons by Ranking Methods [Series Title: Springer Series in Statistics]. In S. Kotz & N. L. Johnson (Eds.), *Breakthroughs in Statistics* (pp. 196–202). Springer New York. https://doi.org/10.1007/978-1-4612-4380-9_16

Wilke, C. O. (2020). *Cowplot: Streamlined plot theme and plot annotations for 'ggplot2'* [R package version 1.1.0]. https://CRAN.R-project.org/package=cowplot

Xie, L., & Shou, W. (2021). Steering ecological-evolutionary dynamics to improve artificial selection of microbial communities. *Nature Communications*, *12*(1), 6799. https://doi.org/10.1038/s41467-021-26647-4

Xie, L., Yuan, A. E., & Shou, W. (2019). Simulations reveal challenges to artificial community selection and possible strategies for success (M. L. Siegal, Ed.). *PLOS Biology*, *17*(6), e3000295. https://doi.org/10.1371/journal.pbio.3000295

Yabe, S., Yamasaki, M., Ebana, K., Hayashi, T., & Iwata, H. (2016). Island-Model Genomic Selection for Long-Term Genetic Improvement of Autogamous Crops (S.-H. Shiu, Ed.). *PLOS ONE*, *11*(4), e0153945. https://doi.org/10.1371/journal.pone.0153945

Yokobayashi, Y., Weiss, R., & Arnold, F. H. (2002). Directed evolution of a genetic circuit. *Proceedings of the National Academy of Sciences*, *99*(26), 16587–16591. https://doi.org/10.1073/pnas.252535999

Zaman, L., Meyer, J. R., Devangam, S., Bryson, D. M., Lenski, R. E., & Ofria, C. (2014). Coevolution Drives the Emergence of Complex Traits and Promotes Evolvability (L. Keller, Ed.). *PLoS Biology*, *12*(12), e1002023. https://doi.org/10.1371/journal.pbio.1002023

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: Methods and applications* (Doctoral dissertation) [Artwork Size: 122 S. Medium: application/pdf Pages: 122 S.]. ETH Zurich. https://doi.org/10.3929/ETHZ-A-003856832
Other Berichte aus der Informatik, 030Other ISBN:3-8265-6831-1