A Framework for Search and Application Agnostic Interactive Optimization

Joshua Powers, Sarah Pell, Josh Bongard
University of Vermont

Abstract

In domains where measures of utility for automatically-designed artefacts (or agents performing subjective tasks) are difficult or impossible to mathematically describe (such as 'be interesting'), human interactive search algorithms are an attractive alternative. However, despite notable achievements, they are still designed around a specific search method, resulting in a lack of problem generality: applying a new search algorithm requires an excessive amount of redesign such that an altogether new interactive method is formed in the process. This leads to missed opportunities for human interactive methods to utilize the power of state of the art optimization algorithms. Here, we introduce for the first time a framework for human interactive optimization that is agnostic to both the search method and the application problem. Using 13 different search methods on 24 fitness functions commonly found in evolutionary algorithm benchmarks, we show that our approach works on the majority of tested applications: many of the search methods, provided with access to the fitness functions, performed no better than our framework, which employs surrogate human participants who act as less informed and erroneous representations of the fitness function. In this way, our framework for interactive optimization provides a scalable solution by facilitating the integration of numerous types of current state of the art or future search algorithms. Future work will involve generalizing this method to admit multi-objective optimization methods and validation with human participants.

Introduction

Some of the most crucial challenges facing machine learning and robotics are generality, computational cost, perverse instantiation [12], and value alignment [22, 18, 4, 10, 30]. In particular, as machine learning and robotics move beyond toy problems to real-world applications, it becomes increasingly difficult to formally define a perverse instantiation resistant objective function. For example, consider virtual agents trained to jump using an objective function that maximizes the height of the robot. Search methods may discover control policies that exploit inaccuracies in the physics engine employed, causing the simulation to become unstable and jettison robot body parts to great heights, satisfying the objective function. In such a toy problem, it is straightforward to alter the objective function to resist such forms of perverse instantiation. However, if the same objective function were employed by physical robots operating in complex, real-world environments, there are a near infinitude of ways to perversely instantiate height maximization: the robot could move such as to be hit by a moving vehicle or climb the side of a building, neither of which would be considered valid instances of jumping.

Hardening objective functions against perverse instantiation is even more challenging when there is a need to balance many antagonistic objective behavioral features, such as speed, accuracy, and efficiency against subjective ones, such as safety [10, 30] and lack of bias [27]).

Historically, the solution to many of the issues facing the optimization of subjective tasks, or tasks that are particularly susceptible to perverse instantiation, has been interactive evolutionary algorithms [14, 16, 19, 20, 21, 33]. Other non-evolutionary approaches for tasks that lack objective functions include inverse reinforcement learning via human demonstration. However, such approaches assume that demonstration is possible, which is not always true. Body-centric demonstration [36, 25] only works if the robot is sufficiently anthropomorphic. And, demonstration by remote control, such as apprenticeship learning [1], only works if, for example, the robot is mechanically simple with few mechanical degrees of freedom and if fast-moving machines provide sufficiently rapid feedback to the human operator.

Despite the feats state of the art interactive evolutionary algorithms have accomplished, many are designed for — and, therefore, only applicable to — the specific artefact of optimization [15, 8, 11, 29, 13]. In a similar vein, these solutions are also still limited by the considerable time required for the process of development, whether that be time spent adapting a standard search method to work in an interactive manner or deriving completely novel implementations inspired by standard search methods [35]. These methods have also demonstrated that human error and user fatigue [28] are a common problem that prevents them from being used at a scale beyond the scope of the that research.

Here we introduce for the first time a framework that is

agnostic to both the search method used and the function to be optimized. In this sense, the framework is modular, as it allows a search method to be easily incorporated by converting human responses into fitness values (without access to the fitness function) using a transformation that requires a minimal number of human responses to optimize the fitness function (in cases where the fitness function is known but hidden from the human participants.) Thus, our framework addresses many of the aforementioned issues with interactive optimization and provides an advantage over other approaches by allowing one to easily utilize state of the art search methods in an interactive manner without any change to the expected performance.

Methods

This section details both the implementation of the framework by which search agnostic optimization is achieved as well as the details relating to how the effectiveness of this implementation was measured.

Algorithmic Implementation

As described in Fig. 1, our interactive framework relies on a surrogate evaluator that mediates the presentation of candidate solutions from the search method to the human evaluators and determines a suitable value to return to the search method based on human responses. To test our method, we expose it to known fitness functions and determine how close the method can come to optimizing that function with a limited computational budget and human evaluators. Moreover, it is assumed that the surrogate evaluator does not have access to this function and the human evaluators return less informed or erroneous fitness values, as explained below.

This method proceeds as such: first a search method $M_i \in M$ sends a candidate solution s_0 to the surrogate evaluator. The surrogate stores this solution at the head of a binary tree with value 0 and returns that value to the search method. The second solution, s_1, is sent from the search method to the surrogate and the surrogate presents the human evaluator with a two-alternative forced choice task: it queries an available human with the question, "is $f_j(s_1) > f_j(s_0)$?" where $f_j \in F$ is the function to be optimized. If the human replies "yes," the surrogate places s_1 to the right of the head node with the value 1, if "no," then to the left with value -1; the surrogate then returns the decided value to the search method. This process continues for subsequent candidates sent from the surrogate evaluator.

In this process, if a node is already occupied, the follow-up two-alternative forced choice task, "is $f_j(s_{n+1}) > f_j(s_{\text{current}})$," is sent to a human evaluator by the surrogate until the solution can be placed in an empty node location. The general method for computing the value of a solution is as follows: if a solution is placed to the right of a leaf node that is also the right most node, its value is one plus the value of its current parent node. Similarly, if a solution is placed to the left of a leaf node that is also the left most node, its value is the value of its current parent node minus one. In all other cases, the value of a new node is the average of its parent and grandparent nodes. When this process concludes, the best candidate solution, s_{best}, will be at the right most node in the tree.

The tree built by the surrogate evaluator can be thought of as a binary tree with a structure described by relative value of presented solutions to each other. With each new solution sent to the surrogate, at most $log(n)$ comparisons will be presented to human evaluators, where n is the depth of the tree. This is the minimum amount of presentations necessary to determine the accurate relative value, thus minimizing human involvement and fatigue as much as possible. To obtain good performance, this tree should be balanced, for which there are many known strategies [2, 6, 7, 26, 32, 3, 23]. We also note that the question presented to the human evaluators via the $>$ symbol is equivalent to asking if a solution, s, is better than another solution at $f \in F$, where F is considered to be a set of functions that contain subjective elements or characteristics that are difficult to compute or measure.

As is illustrated in Fig. 1, this creates a framework that is agnostic to both the search method $M \in M$ and the function to be optimized $f \in F$, thus, these components can be changed without any negative affect on the underlying framework.

Measuring Performance

To validate whether the method produces "good" solutions under our framework, we compare its performance to that of the same method when optimized in a non-interactive manner. To do this, we use non-subjective, standard fitness functions that do not require an interactive component nor need to simulate human responses. We use the 24 standard functions from the Comparing Continuous Optimisers Benchmark (COCO) [17] and we use 13 different black box optimizations methods [1]. See the appendix for a complete methods and functions list.

We run each search method on every function 120 times each with a different random seed to get baseline performance for each method on every function. The methods all run until they have proposed 5000 solutions $(s_{n=5000})$. The performance is then calculated as the average value of the best solution found $(f(s_{\text{best}})$ over the 120 runs. The various hyperparameters of each method, such as population sizes, selection criteria, and reproduction operators are not reported here for brevity. Their details can be found in the repository mentioned in the appendix.

We then perform another 120 evolutionary trials for each search method and fitness function pair, allowing each 5000 solution evaluations, but insert our surrogate evaluator. Since the fitness of each solution is an easily computed

[1] `github.com/robertfeldt/BlackBoxOptim.jl`

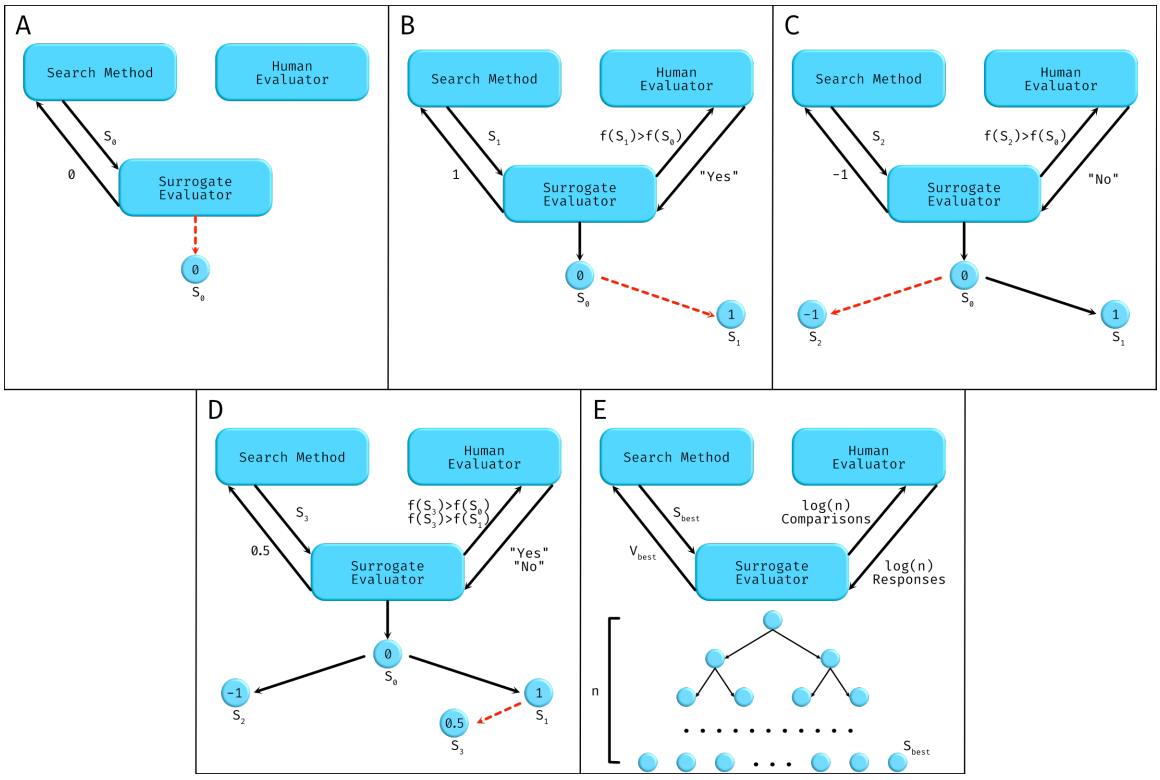

Figure 1: **Overview of search agnostic interactive evolution.** **(A)** A search method, M, sends an initial solution, s_0, to the surrogate evaluator and becomes the head of a binary tree with value 0, which is returned to the search method. **(B)** The second solution, s_1, is sent to the surrogate with the query, "is s_1 better than s_0 at f¿" The human replies "yes," which results in the surrogate placing s_1 to the right of the head node with the value 1 and returns it. **(C)** We perform this sequence of operations again with s_2, which is eventually added to the left of the head node with value -1. **(D)** We repeat the process for s_3, but now make two queries to the human evaluators. The first query would place s_2 to the right of the head node where it encounters the s_1 node, triggering the second query which places it to the left; the value assigned is the average of the parent and grandparent nodes (0.5) which is returned. **(E)** This process continues such that at each step, the surrogate makes a max number of $log(n)$ queries to available humans to determine the value of a solution s. At the end, the best solution, s_{best}, will be stored in the rightmost node of the binary tree.

and non-subjective function, we simulate human responses to the two-alternative forced choice comparisons mathematically. When a query is sent to a simulated human, it computes the boolean value $f(s_{\text{current}}) > f(s_{n+1})$. If true, a "yes" is returned, otherwise, "false" is returned. We again average the results of all of the values $f(s_{\text{best}})$ across the 120 runs. We then run a Welch's t-test comparing the baseline results to those under the interactive framework. Before determining significance, we adjust this value for multiple comparisons by multiplying by 2496 (24 methods x 13 functions x 8 levels of simulated human accuracy, described below). The results of this are reported in Fig. 2. To display the results we create a table (Fig. 3) where we color each cell green (using our framework performed significantly better, $p < 0.05$), blue (no significance in performance difference when using the framework), or red (using our framework performed significantly worse, $p < 0.05$) to show how performance compared with and without the surrogate. We consider our framework successful in all cases where the cell is not colored red.

As described, simulating human participants for this comparison is as simple as running the mathematical comparison presented by the surrogate to the human, which can be done in this case since f is known and returns a numerical value. However, to attempt simulation of human error, we ran multiple versions of the framework, differing the accuracy of the comparison. This is done by adding a probability of performing a boolean "not" on the simulated human comparison before returning a response to the surrogate. We used 8 different levels of accuracy for the simulated human comparisons: $100\%, 95\%, 90\%, 85\%, 80\%, 75\%, 70\%, 50\%$, where the 50% comparison creates a completely random comparison scheme.

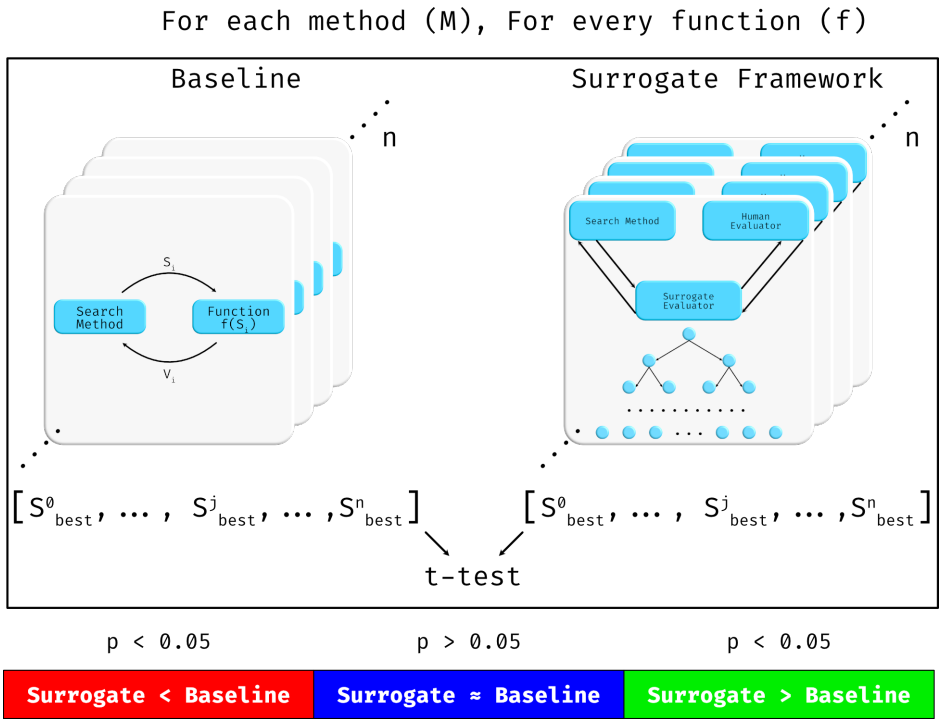

Figure 2: **How performance is measured.** To measure performance, we compare (Welsh's t-test) the fitness of the best solution found with our interactive framework with the best solution found by the non-interactive baseline on 24 search methods over 13 computable functions for $n = 120$ runs. We adjust this value for multiple comparison by multiplying by the number of comparisons using the resulting p-value to color the cells of figure 3, which shows all the results of the various comparisons.

Results

The results are detailed graphically in Fig. 3. In this figure we find that when the simulated human gives 100% accurate feedback, our surrogate evaluator not only performs just as well as traditional optimization, but actually significantly better in almost all cases. The only exceptions to this level of performance occur in regard to method m_{12}, in which case it performs significantly worse in 12 out of the 24 functions. We also found that as the simulated human accuracy decreases, the performance of optimization with the surrogate falls dramatically such that even at 95% accuracy, in 34% of the method and function pairs, it performs significantly worse and only 2 pairs perform significantly better. This decreases gradually with each step in accuracy until reaching 50% accuracy, or essentially random selection; about half of the pairs perform significantly better and half significantly worse.

The results suggest that, given accurate comparisons, the surrogate strategy creates a successful framework for interactive optimization. The results persist across the functions and search methods, suggesting that our surrogate evaluator is indeed agnostic to both search method and application problem. Method 12 was Simultaneous Perturbation Stochastic Approximation (SPSA) [34], which uses a unique form of gradient approximation that, in itself, could be considered a type of surrogate for standard gradient approximation. Thus, when coupling that surrogate with our surrogate, there appears to be a loss of performance on some of the functions.

In Fig. 4, rather than compare the absolute performance of the baseline against the surrogate, we compute the relative performances of the methods with each other by ranking their performance for each function. This tests whether a search method retains its performance relative to other search methods when the surrogate is used. Here we would ideally like both images to be exactly the same, implying that relative performance was consistently maintained. This means that if one method outperforms another on a traditionally optimized function, it will still outperform that method when used with our framework for interactive optimization. Given the results over these functions and methods, this holds true $\approx 87\%$ of the time (computed as $100\times(1-\frac{c}{13*24})$, where c is the count of cells that differ between a treatment and the baseline) in the case of the 100% accurate human, thus indicating a high likelihood of maintaining this condition in novel problem applications. This number again decreases with lowered human accuracy, however, when visually comparing the baseline to our framework, a general pattern of relative performance is retained.

Overall, the results for our framework are rather interest-

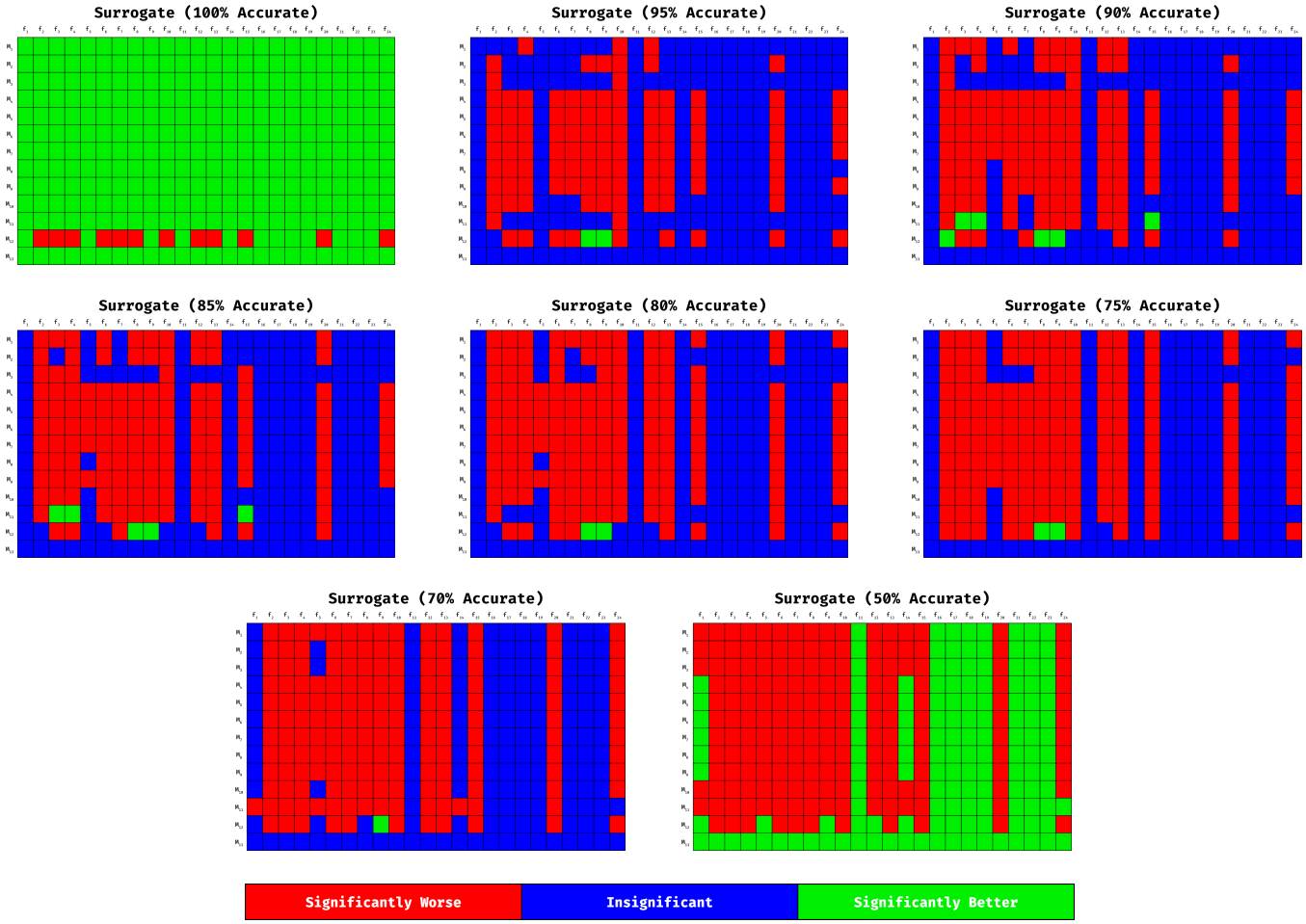

Figure 3: **Comparison of baseline and surrogate performance.** We compare 8 experimental setups with varying accuracy for our simulated human responses against the baseline performance for all of the 13 methods on each of the 24 functions. The comparisons are made using a Welch's t-test over 120 independent trials. We adjust the resulting p-value for multiple comparisons multiplying by 2496 (24 methods x 13 functions x 8 experimental setups). **Green**: The surrogate performed significantly better ($p < 0.05$) than the baseline on that method function pair. **Red**: The surrogate performed significantly worse ($p < 0.05$) than the baseline on that method function pair. **Blue**: The surrogate and baseline performance weren't significantly different. To show that the surrogate performance is at least as good as the baseline on a method function pair, the cell must be green or blue.

ing and promising. The performance gains found in the surrogate in the 100% accurate case might be accounted for by its method of comparison and value assignment, which transforms the fitness landscape space presented to the optimization method. This might make it easier to search by making it less likely to become trapped in local minima.

Discussion

In this paper, we have presented for the first time a framework for interactive evolution that is agnostic to both the optimization algorithm and the application domain. We have demonstrated in our results that the framework performed as well as the baseline when accurate simulated human comparisons were used and that the relative performance (rank) of the various search methods was often maintained across treatments. However, we acknowledge that in some cases, search methods under performed, as we saw when using SPSA optimization. It is not clear why our method had particular difficulties accommodating SPSA, though we hypothesize this may be due to the offsetting of gradients used in this method which could cause it to either over or undershoot when proposing new candidate solutions. From this demonstration, it could be expected that other search methods are indigestible by our framework. In this way, other potential methods that would not work well could be other stochastic gradient based methods. This also holds true for the application problems, as shown per our results. Regardless of this specific flaw, the rest of the methods worked well,

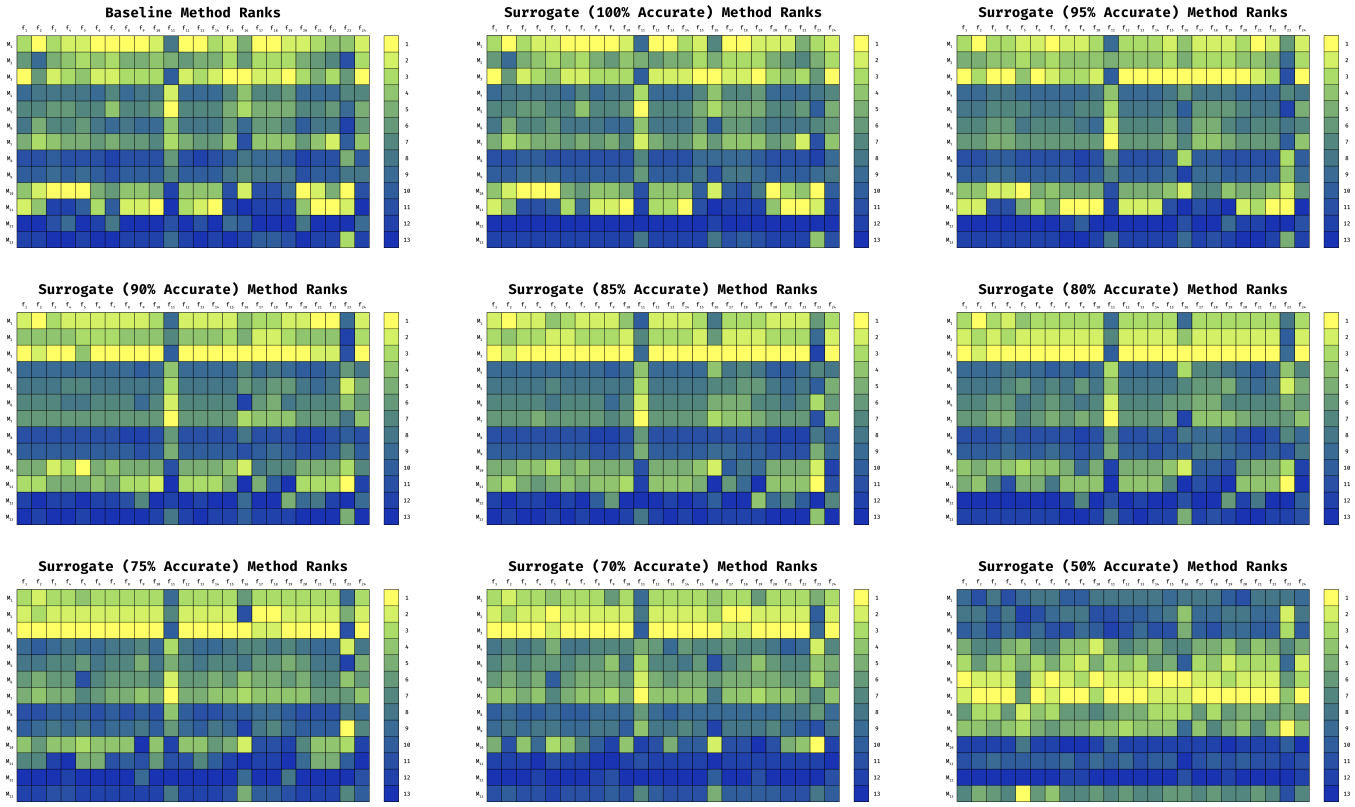

Figure 4: **Comparison of method rankings for the baseline and surrogate performance.** This figure shows the relative performance of the methods comparatively by ranking them against one another on each function. Every column shows the rank (out of 13) of a method on that function based on the average performance over the 120 runs. On the **top left** we show this ranking for the baseline performance. The other plots show the ranking for the surrogate with various levels of accuracy for the simulated human. Comparing them shows whether a search method retains its performance relative to other search methods when the surrogate is used. Ideally, all of these images would look exactly like the baseline image, however, here we can see a number of differences from about a 23% difference, in the best case (**top middle**) to about 92% difference (**bottom right**). With the best treatment this can be thought of as a 77% likelihood that a method that generally performs better than another will retain that performance on a novel problem application.

demonstrating that any new search methods not tested here have a compelling likelihood of being successful when utilizing the framework described in this paper.

It is important to mention that because none of the application functions tested were subjective (as is necessary to make comparisons with the baseline algorithms), it remains to be seen how well our framework would perform under real conditions of subjective optimization. We also acknowledge that these subjective factors remain untested in our treatment of simulated human responses. The viability of our framework decreased rapidly as simulated human accuracy decreased, but it is unclear how accurately humans would be able to measure the performance of subjective elements. Additionally, our method for simulating humans did not take into account that humans likely err less when the comparisons have drastically different performance. In these ways, our work was not able to capture how well or poorly our framework adapts to the complexities of human decision processes when making subjective comparisons.

Ultimately, this study has laid the groundwork, if not a promising starting point, for agnositc paradigms within the field of interactive evolution. Future work will focus on testing our framework with real humans. We anticipate that such testing would raise a number of challenges, such as detecting and resolving groups with differing opinions [8, 5]. A beneficial starting point for this work could therefore be testing on at least one problem where the measure of utility can be specified mathematically and can also utilize real human evaluations; the culmination of this framework that utilizes human input on one problem could then finally be used to compare its performance against the standard optimization tactic.

In addition, further work should also attempt to incorporate multi-objective optimization methods into this frame-

work, specifically in which one or more objectives admit to formalization (such as prediction accuracy or behavioral efficiency) while other objectives must be inferred from human subjective preferences (such as safety or aesthetics). The simplest, albeit not the only way, to further generalize our framework to accept multiobjective optimization may be to expand the surrogate evaluator from a binary tree into a forest, in which each tree is employed to estimate the value of each of the objectives.

Finally, to reinforce this framework, we hope to strengthen it by making it more robust to human error, and similarly, reduce the amount of human comparisons needed. This could perhaps be achieved by introducing further heuristics into the surrogate evaluator. It would also be useful to test not just the 'horizontal' scalability of this framework across search methods and application domains, but also its 'vertical' scalability. That is, how many human participants — with increasingly varying degrees of bias, error, and differing opinions — can usefully be employed to collectively discover useful solutions.

Acknowledgments

This work was supported by NSF EFRI award 1830870, and by the Vermont Space Grant Consortium under NASA Cooperative Agreement NNX15AP86H. Computational Resources were provided by The Vermont Advanced Computing Core (VACC). Thanks is also due to the beautiful wife of the first author whose continual support makes work like this possible.

References

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1.

Adel'son-Vel'skii, G. M. and Landis, E. M. (1962). An algorithm for organization of information. In *Doklady Akademii Nauk*, volume 146, pages 263–266. Russian Academy of Sciences.

Andersson, A. (1989). Improving partial rebuilding by using simple balance criteria. In *Workshop on Algorithms and Data Structures*, pages 393–402. Springer.

Arnold, T., Kasenberg, D., and Scheutz, M. (2017). Value alignment or misalignment–what will keep systems accountable? In *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*.

Awad, E., Dsouza, S., Kim, R., Schulz, J., Henrich, J., Shariff, A., Bonnefon, J.-F., and Rahwan, I. (2018). The moral machine experiment. *Nature*, 563(7729):59–64.

Bayer, R. (1972). Symmetric binary b-trees: Data structure and maintenance algorithms. *Acta informatica*, 1(4):290–306.

Bayer, R. and McCreight, E. (2002). Organization and maintenance of large ordered indexes. In *Software pioneers*, pages 245–262. Springer.

Bernatskiy, A., Hornby, G., and Bongard, J. (2014). Improving Robot Behavior Optimization by Combining User Preferences. pages 973–980.

Bongard, J. and Anetsberger, J. (2016). Robots can ground crowd-proposed symbols by forming theories of group mind. In *Proceedings of the Artificial Life Conference 2016 13*, pages 684–691. MIT Press.

Bonnefon, J.-F., Shariff, A., and Rahwan, I. (2016). The social dilemma of autonomous vehicles. *Science*, 352(6293):1573–1576.

Bontrager, P., Lin, W., Risi, S., and Togelius, J. (2017). Deep interactive evolutionary computation. In *Neural Information Processing Systems*.

Bostrom, N. (2014). *Superintelligence: Paths, dangers, strategies*. OUP Oxford.

Chevalier-Boisvert, M., Lahlou, S., Nguyen, T. H., Bahdanau, D., Willems, L., Bengio, Y., and Saharia, C. (2019). Babyai: A platform to study the sample efficiency of grounded language learning. *7th International Conference on Learning Representations, ICLR 2019*, pages 1–13.

Dawkins, R. and Others (1996). *The blind watchmaker: Why the evidence of evolution reveals a universe without design*. WW Norton & Company.

Deb, K. and Kumar, A. (2007). Interactive evolutionary multi-objective optimization and decision-making using reference direction method. *Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference*, pages 781–788.

Dozier, G., Carnahan, B., Seals, C., Kuntz, L. A., and Fu, S. G. (2005). An interactive distributed evolutionary algorithm (IDEA) for design. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 1:418–422.

Finck, S., Hansen, N., Ros, R., and Auger, A. (2010). Real-parameter black-box optimization benchmarking 2010: Presentation of the noisy functions.

Fisac, J. F., Gates, M. A., Hamrick, J. B., Liu, C., Hadfield-Menell, D., Palaniappan, M., Malik, D., Sastry, S. S., Griffiths, T. L., and Dragan, A. D. (2020). Pragmatic-pedagogic value alignment. In *Robotics Research*, pages 49–57. Springer.

Gomes, A., Antunes, C. H., and Martins, A. G. (2007). A multiple objective approach to direct load control using an interactive evolutionary algorithm. *IEEE Transactions on Power Systems*, 22(3):1004–1011.

Gong, D., Guo, G., Lu, L., and Ma, H. (2008). Adaptive interactive genetic algorithms with individual interval fitness. *Progress in Natural Science*, 18(3):359–365.

Gruau, F. and Quatramaran, K. (2018). Cellular Encoding for Interactive Evolutionary Robotics. *Advances in the Evolutionary Synthesis of Intelligent Agents*, pages 1–23.

Hadfield-Menell, D., Russell, S. J., Abbeel, P., and Dragan, A. (2016). Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*, pages 3909–3917.

Haeupler, B., Sen, S., and Tarjan, R. E. (2009). Rank-balanced trees. In *Workshop on Algorithms and Data Structures*, pages 351–362. Springer.

Hansen, N., Brockhoff, D., Mersmann, O., Tusar, T., Tusar, D., ElHara, O. A., Sampaio, P. R., Atamna, A., Varelas, K., Batu, U., Nguyen, D. M., Matzner, F., and Auger, A. (2019). COmparing Continuous Optimizers: numbbo/COCO on Github.

Laskey, M., Chuck, C., Lee, J., Mahler, J., Krishnan, S., Jamieson, K., Dragan, A., and Goldberg, K. (2017). Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 358–365. IEEE.

Lehman, T. J. and Carey, M. J. (1985). A study of index structures for main memory database management systems. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2019). A survey on bias and fairness in machine learning. *arXiv preprint arXiv:1908.09635*.

Pallez, D., Collard, P., Baccino, T., and Dumercy, L. (2007). Eye-tracking evolutionary algorithm to minimize user fatigue in iec applied to interactive one-max problem. In *Proceedings of the 9th annual conference companion on Genetic and evolutionary computation*, pages 2883–2886.

Pei, Y. and Takagi, H. (2018). Research progress survey on interactive evolutionary computation. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–14.

Rahwan, I., Cebrian, M., Obradovich, N., Bongard, J., Bonnefon, J.-F., Breazeal, C., Crandall, J. W., Christakis, N. A., Couzin, I. D., Jackson, M. O., et al. (2019). Machine behaviour. *Nature*, 568(7753):477–486.

Shah, N. B. and Wainwright, M. J. (2018). Simple, robust and optimal ranking from pairwise comparisons. *Journal of Machine Learning Research*, 18:1–38.

Sleator, D. D. and Tarjan, R. E. (1985). Self-adjusting binary search trees. *Journal of the ACM (JACM)*, 32(3):652–686.

Smith, J. R. (1991). Designing Biomorphs with an Interactive Genetic Algorithm. pages 535—-538.

Spall, J. C. (1998). Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on aerospace and electronic systems*, 34(3):817–823.

Thiele, L., Miettinen, K., Korhonen, P. J., and Molina, J. (2007). A Preference-based Interactive Evolutionary Algorithm for Multiobjective Optimization. (W-412).

Tsarouchi, P., Makris, S., and Chryssolouris, G. (2016). Human–robot interaction review and challenges on task planning and programming. *International Journal of Computer Integrated Manufacturing*, 29(8):916–931.

Appendix

All the search methods used came from the Julia BlackBoxOptim and their implementation can be found at `github.com/robertfeldt/BlackBoxOptim.jl`. The function used came from the COCO benchmark package [24] whose implementation can be found at `github.com/numbbo/coco` and were used with 40 dimensions. All code and data for this reasearch can be found at `github.com/jpp46/SAIE2020`.

Natural Evolution Strategies (NES)

- Separable NES (M_1)
- Exponential NES (M_2)
- Distance-weighted exponential NES (M_3)

Differential Evolution (DE)

- Adaptive DE/rand/1/bin (M_4)
- Adaptive DE/rand/1/bin with radius limited sampling (M_5)
- DE/rand/1/bin (M_6)
- DE/rand/1/bin with radius limited sampling (M_7)
- DE/rand/2/bin (M_8)
- DE/rand/2/bin with radius limited sampling (M_9)

Direct Search

- Compass coordinate generating set search (M_{10})
- Direct search through probabilistic descent (M_{11})

Stochastic and Random Search

- Simultaneous perturbation stochastic approximation (M_{12})
- Random search (M_{13})

Function List

- Sphere Function (f_1)
- Ellipsoidal Function (f_2)
- Rastrigin Function (f_3)
- Büche-Rastrigin Function (f_4)
- Linear Slope (f_5)
- Attractive Sector Function (f_6)
- Step Ellipsoidal Function (f_7)
- Rosenbrock Function, original (f_8)
- Rosenbrock Function, rotated (f_9)
- Ellipsoidal Function (f_{10})
- Discus Function (f_{11})
- Bent Cigar Function (f_{12})
- Sharp Ridge Function (f_{13})
- Different Powers Function (f_{14})

- Multi-modal Rastrigin Function (f_{15})
- Weierstrass Function (f_{16})
- Schaffers F7 Function (f_{17})
- Schaffers F7 Functions, moderately ill-conditioned (f_{18})
- Composite Griewank-Rosenbrock Function F8F2 (f_{19})
- Schwefel Function (f_{20})
- Gallagher's Gaussian 101-me Peaks Function (f_{21})
- Gallagher's Gaussian 21-hi Peaks Function (f_{22})
- Katsuura Function (f_{23})
- Lunacek bi-Rastrigin Function (f_{24})