Toward Learning Neural Network Encodings for Continuous Optimization Problems

Eric O. Scott
George Mason University
Fairfax, Virginia
escott8@gmu.edu

Kenneth A. De Jong
George Mason University
Fairfax, Virginia
kdejong@gmu.edu

ABSTRACT

To date, efforts to automatically configure problem representations for classes of optimization problems have yielded few practical results. We show that a recently proposed approach for training neural G-P maps for optimization problems yields maps that generalize poorly to translated problem instances. We propose that alternative neural architectures—especially ones that allow the number of control genes to be greater than the number of phenotypic traits—may provide a means of learning maps that are better able to generalize to new problem instances.

CCS CONCEPTS

• **Computing methodologies → Search methodologies**;

KEYWORDS

Representation Learning, Evolutionary Algorithms, Continuous Optimization

ACM Reference Format:
Eric O. Scott and Kenneth A. De Jong. 2018. Toward Learning Neural Network Encodings for Continuous Optimization Problems. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3205651.3205687

1 INTRODUCTION

Practitioners are increasingly relying on automated algorithm configuration tools as a means of selecting and tuning the components of evolutionary algorithms [1]. Such methods are especially useful in cases where we wish to solve a class of related problem instances. This makes it possible to use a training set of tasks to learn an effective configuration of various algorithm design decisions, and then to reuse the resulting algorithm on a potentially unbounded number of future tasks drawn from the same class. For this to be possible, the knowledge extracted from the target tasks must be able to generalize effectively to future tasks.

Solution representations offer an especially powerful means of representing information about problem structure. In evolutionary algorithms, a representation takes the form of a genotype-to-phenotype (G-P) map: reproductive operators operating on genotypes create variation in phenotype space, which selection then acts upon. Because of the complexity of decisions involved in defining novel, problem-specific G-P maps, however, automated methods for configuring them have seldom been proposed.

In this initial work, we begin a preliminary investigation into using neural-network-based representations as a means of representing and learning arbitrary G-P maps. Some mechanisms for neural representation learning have been proposed by Simões et al. and by Scott and Bassett [3, 4], but the practical challenges that prevent fine-grained representation learning from being useful in practice remain poorly understood to date.

Here we adapt a biological model that was introduced by Watson et al. [2, 5] to a continuous optimization context, and provide some initial analysis of how its ability to learn and generalize might be improved upon.

2 METHODS

To examine the effects of representation in isolation from the dynamics of population-based evolution, we restrict our attention to a (1 + 1)-style EA with additive gene-by-gene Gaussian mutation. Each individual then consists of a genotype-representation tuple $(\vec{g}, \mathbf{W})$. Before fitness evaluation, the initial genotype $\vec{g} \in \mathbb{R}^n$ is subjected to a developmental process defined by

$$\vec{p}_{i+1} = \vec{p}_i + \tau_1 \sigma(\mathbf{W}\vec{p}_i) - \tau_2 \vec{p}_i, \tag{1}$$

where τ_1, τ_2 are positive constants, $\mathbf{W} \in \mathbb{R}^{n \times n}$ is the individual's weight matrix, $\sigma(\cdot)$ is some nonlinear transform (we choose $\tanh(\cdot)$), and the initial phenotype vector $\vec{p}_0 = \vec{g}$. The process is carried out for a fixed number of steps (we choose 10), and an individual's fitness $f(\vec{p})$ is then defined over the adult phenotype, $\vec{p} = \vec{p}_{10}$. Note that this model requires that $\vec{g}$ and $\vec{p}$ have the same dimensionality.

Watson et al. have observed that when the elements of the control genes $\vec{g}$ are mutated at a significantly faster rate than the elements of the representation $\mathbf{W}$, then the model given by Equation 1—which closely resembles abstract models of genetic regulatory networks found in the biological literature—is able to learn to reproduce previously visited high-fitness phenotypic states in a way that is exactly analogous to Hebbian learning in neural networks [5].

Indeed, we find that, by repeatedly running the algorithm on a fixed continuous objective such as the Rastrigin function, this method is able to successfully learn a G-P map that considerably simplifies the structure of the search space (Figure 1). After 100 random restarts on the same objective of 1,000 evaluations each, an

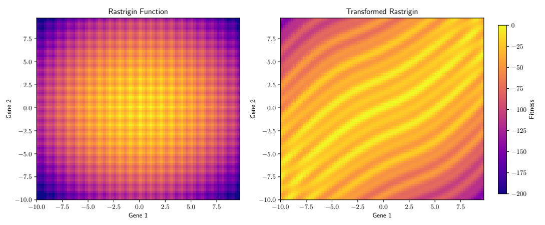

Figure 1: Example of a developmental encoding trained on a Rastrigin function centered at (0, 0). It is able to transform the original function into a landscape that is considerably easier to optimize.

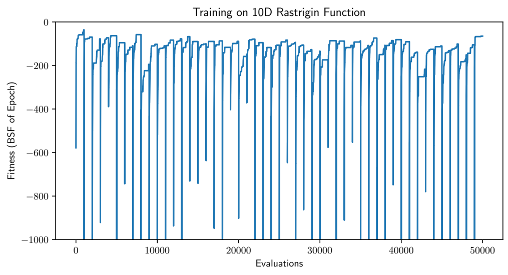

Figure 2: Fitness trajectory on a sequence of randomly-translated 10D Rastrigin functions. Evolution is unable to find a mapping that generalizes to new translations of the objective function.

initial weight matrix $\mathbf{W} = \mathbf{0}$ adapts to effectively "zoom in" on the region containing the global optimum, making it easier to find from arbitrary starting points. In order for a representation-learning scheme to be useful in practice, however, our learned encoding needs to be able to generalize to novel problem instances, whose local optima may be located in a different region of the search space. We hypothesize that the architecture given by Equation 1 will struggle to learn a mapping that remains useful across randomly translated instances of the Rastrigin function.

3 RESULTS

We applied the algorithm to a sequence of 10-D Rastrigin function instances, each of which undergoes a random translation in phenotype space, but which are otherwise identical. Evolution proceeds on each task for 1,000 evaluations, then the genotype $\vec{g}$ (but not the weights) are re-initialized, and evolution begins on the next task in the sequence. Figure 2 shows a fitness trajectory for this sequence of learning trials. The algorithm never finds the global optimum (fitness value of zero), and its ability to solve arbitrarily translated Rastrigin functions does not improve over time.

This makes sense if we consider that the Hebbian learning procedure produces a representation that memorizes how to reach previously-seen high-fitness regions. Under random translation, past performance in one region of the landscape is a poor predictor of future success. Thus, if we apply the mapping that was trained on the 2-D Rastrigin in Figure 1 to translated test instances of the same problem, the representation ends up overfitting, zooming in on low- and medium-fitness regions of the space, rendering the global optimum entirely unreachable (Figure 3).

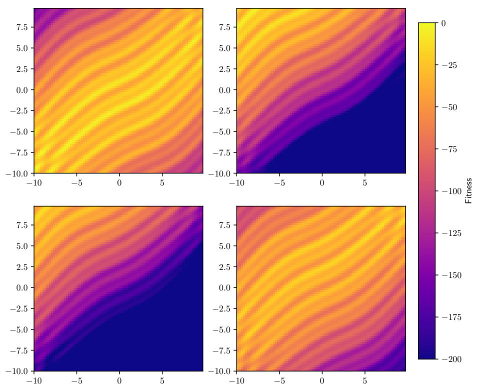

Figure 3: The successful encoding from Figure 1 fails to generalize to translated instances: high-fitness phenotypes are not only difficult to find, but often impossible to reach with the trained encoding.

4 NEXT STEPS

The developmental encoding provied by Equation 1 is provocative for its ability to learn in an online way from repeated EA trials, without suffering from the overhead of a meta-evolutionary approach, and for its ability to use Hebbian learning to store and recall multiple phenotypic patterns. To be useful in practice, however, it seems that it will be necessary to improve upon this model's ability to generalize to modified problem instances—such as translations and rotations. It seems to us that these limitations arise from the requirement that $\vec{g}, \vec{p}$ have the same dimensionality. We hypothesize that neural architectures with a greater number of genes may mitigate the issue posed by translations. Further insights from the neural network literature—such as adding bias units, higher-dimensional hidden layers, or altering the form of $\sigma(\cdot)$—also promise to improve the expressive power of the neural representations, which may enhance learning and generalization.

REFERENCES

[1] Holger H Hoos, Frank Neumann, and Heike Trautmann. 2017. Automated Algorithm Selection and Configuration (Dagstuhl Seminar 16412). In *Dagstuhl Reports*, Vol. 6. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[2] Loizos Kounios, Jeff Clune, Kostas Kouvaris, Günter P Wagner, Mihaela Pavlicev, Daniel M Weinreich, and Richard A Watson. 2016. Resolving the paradox of evolvability with learning theory: How evolution learns to improve evolvability on rugged fitness landscapes. *arXiv preprint arXiv:1612.05955* (2016).

[3] Eric O Scott and Jeffrey K Bassett. 2015. Learning genetic representations for classes of real-valued optimization problems. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, 1075–1082.

[4] Luís F Simões, Dario Izzo, Evert Haasdijk, and Agoston Endre Eiben. 2014. Self-adaptive genotype-phenotype maps: neural networks as a meta-representation. In *International Conference on Parallel Problem Solving from Nature*. Springer, 110–119.

[5] Richard A Watson, Günter P Wagner, Mihaela Pavlicev, Daniel M Weinreich, and Rob Mills. 2014. The evolution of phenotypic correlations and "developmental memory". *Evolution* 68, 4 (2014), 1124–1138.