OPEN ACCESS Freely available online

PLoS COMPUTATIONAL BIOLOGY

# Natural Selection Fails to Optimize Mutation Rates for Long-Term Adaptation on Rugged Fitness Landscapes

**Jeff Clune[1,2]*, Dusan Misevic[3], Charles Ofria[1], Richard E. Lenski[4], Santiago F. Elena[2,5], Rafael Sanjuán[2,6]**

**1** Department of Computer Science and Engineering, Michigan State University, East Lansing, Michigan, United States of America, **2** Instituto de Biología Molecular y Celular de Plantas, Consejo Superior de Investigaciones Cientificas, Universidad Politecnica de València, València, Spain, **3** Institute of Integrative Biology, ETH Zurich, Zurich, Switzerland, **4** Department of Microbiology and Molecular Genetics, Michigan State University, East Lansing, Michigan, United States of America, **5** The Santa Fe Institute, Santa Fe, New Mexico, United States of America, **6** Institut Cavanilles de Biodiversitat i Biologia Evolutiva, Universitat de València, València, Spain

## Abstract

The rate of mutation is central to evolution. Mutations are required for adaptation, yet most mutations with phenotypic effects are deleterious. As a consequence, the mutation rate that maximizes adaptation will be some intermediate value. Here, we used digital organisms to investigate the ability of natural selection to adjust and optimize mutation rates. We assessed the optimal mutation rate by empirically determining what mutation rate produced the highest rate of adaptation. Then, we allowed mutation rates to evolve, and we evaluated the proximity to the optimum. Although we chose conditions favorable for mutation rate optimization, the evolved rates were invariably far below the optimum across a wide range of experimental parameter settings. We hypothesized that the reason that mutation rates evolved to be suboptimal was the ruggedness of fitness landscapes. To test this hypothesis, we created a simplified landscape without any fitness valleys and found that, in such conditions, populations evolved near-optimal mutation rates. In contrast, when fitness valleys were added to this simple landscape, the ability of evolving populations to find the optimal mutation rate was lost. We conclude that rugged fitness landscapes can prevent the evolution of mutation rates that are optimal for long-term adaptation. This finding has important implications for applied evolutionary research in both biological and computational realms.

**Citation:** Clune J, Misevic D, Ofria C, Lenski RE, Elena SF, et al. (2008) Natural Selection Fails to Optimize Mutation Rates for Long-Term Adaptation on Rugged Fitness Landscapes. PLoS Comput Biol 4(9): e1000187. doi:10.1371/journal.pcbi.1000187

**Editor:** Laurence D. Hurst, University of Bath, United Kingdom

**Received** April 21, 2008; **Accepted** August 18, 2008; **Published** September 26, 2008



**Funding:** This work was supported, in part, by the Defense Advanced Research Projects Agency "Fun Bio" Program, National Science Foundation grant CCF-0643952, and the Cambridge Templeton Consortium. Work in València was supported by grant BFU2006-14819-C02-01/BMC and the Ramón y Cajal program from the Spanish MEC.

**Competing Interests:** The authors have declared that no competing interests exist.

* E-mail: jclune@msu.edu

## Introduction

Mutation is the ultimate source of genetic variation, and thus the rate at which spontaneous mutations appear is a fundamental evolutionary parameter. The mechanisms of DNA replication and repair are themselves genetically encoded and variable [1–5], making mutation rates potential targets of evolutionary optimization. Two opposing forces contribute to the evolution of mutation rates. On the one hand, most mutations with phenotypic effects are deleterious, producing a genetic load that favors organisms with low mutation rates; on the other hand, beneficial mutations are necessary for adaptation. Given this trade-off between genetic load and adaptation, there should exist an intermediate mutation rate—hereafter referred to as the 'optimal' rate, or $U_{opt}$—that balances these forces and maximizes adaptation over the long-term [6–9]. It is important, however, to note that these two forces operate at different timescales. The costs of genetic load are continuously paid in the short-term, whereas the payoffs of adaptation come in the long-term [6–8,10–12].

Experiments have shown that genotypes with increased mutation rates can be favored by selection if they face novel or changing environments [1,13–21]. Similarly, recent work with RNA viruses has shown that certain high-fidelity genotypes have diminished fitness and virulence in mice [22,23], which might reflect their restricted ability to create the genetic variability needed to escape from immune surveillance. However, another recent study with an RNA virus failed to observe a positive association between mutation rate and the rate of adaptation to a novel environment [24]. Despite their importance, these studies suffer from some unavoidable limitations. For example, it is unknown whether the observed mutation rates are the product of evolutionary optimization or, alternatively, if they are far from their optimal values. Also, it is often difficult to assess whether experimental observations reflect evolutionary equilibria or transient states.

These limitations can be overcome using evolution with digital organisms owing to the speed and ease of data collection. Digital organisms are self-replicating computer programs that inhabit a virtual world where they reproduce, mutate, compete for resources, and evolve according to the same fundamental processes as biological organisms [25]. Here, we use digital organisms to study the ability of natural selection to adjust the mutation rate. We first validate the existence of an optimal mutation rate by extensively exploring a range of mutation rates and observing which rate maximizes adaptation over the long-term. Then we allow mutation rates to evolve under natural selection and assess whether the optimal rate is reached. Even in conditions highly favorable for mutation rate optimization, mutation rates systematically evolve that are far below the optimum, showing that natural selection fails to optimize mutation

## Author Summary

Natural selection is shortsighted and therefore does not necessarily drive populations toward improved long-term performance. Some traits may evolve because they provide immediate gains, even though they are less successful in the long run than some alternatives. Here, we use digital organisms to analyze the ability of evolving populations to optimize their mutation rate, a fundamental evolutionary parameter. We show that when the mutation rate is constrained to be high, populations adapt considerably faster over the long term than when the mutation rate is allowed to evolve. By varying the fitness landscape, we show that natural selection tends to reduce the mutation rate on rugged landscapes (but not on smooth ones) so as to avoid the production of harmful mutations, even though this short-term benefit limits adaptation over the long term.


rates. We propose a novel hypothesis for these results based on the topology of the underlying fitness landscape, and we then proceed to experimentally test it.

## Results

### Selection Fails To Find the Optimal Mutation Rate

We studied the evolution of mutation rates using the Avida digital evolution platform [25–34]. To test empirically whether there was an intermediate, optimal rate of mutation that maximized adaptation, we performed a series of evolution experiments. In each experiment, a genetically homogenous population was placed in a novel environment where it evolved for 150,000 updates (~15,000 generations) at a constant mutation rate (see Methods). We explored 15 different mutation rates spanning six orders of magnitude ($10^{-5}$ to 10 mutations per genome per generation). The final fitness values confirmed that there was an optimal mutation rate at an intermediate value, with $U_{opt} \approx 4.641$ (Figure 1). An analysis of the temporal dynamics of these experiments showed that this rate yielded the highest fitness from about generation 230 onward. Interestingly, for the very earliest time points (before generation 50), the lowest mutation rate ($10^{-5}$) produced the highest fitness values, whereas for generations 50–230 a mutation rate of 2.2 gave the highest fitness values.

To assess whether evolution would produce organisms with mutation rates near the long-term $U_{opt}$, we ran additional experiments in which mutation rates were allowed to change (see Methods), starting from rates either below ($10^{-3}$) or above (10) the optimum. Strikingly, mutation rates evolved to levels far below the long-term $U_{opt}$, regardless of the starting value (Figure 1). In light of our observation that the optimum rate can change over time, one might hypothesize that the typical mutation rate of an evolving population had actually followed a near-optimal trajectory throughout its evolution, but that the final mutation rate is not a good indicator of the ability to optimize the mutation rate. However, this explanation can be ruled out because the final average fitness of the populations whose mutation rates could change was significantly lower than the fitness levels of the populations that evolved at a constant $U_{opt}$. The log-transformed final fitness values for treatments with changing mutation rates were $4.61 \pm 0.70$ and $1.23 \pm 0.15$ (mean $\pm 1$ s.e.m.) for the populations starting at high and low initial rates, respectively. Both of these values are significantly lower than the $14.45 \pm 0.64$ obtained for populations evolved at $U_{opt}$ (Mann-Whitney tests, both $P < 0.001$). The fitness advantage for $U_{opt}$ is also clear for nearly all intermediate time points (Figure 2A). While populations starting below $U_{opt}$ did experience a transient increase in their mutation rates (Figure 2B), the mutation rates still stayed more than two orders of magnitude below $U_{opt}$. For populations starting above $U_{opt}$, the results were particularly striking because selection pushed the populations through the optimal rate on their way to an evidently very suboptimal rate (Figures 1 and 2B).

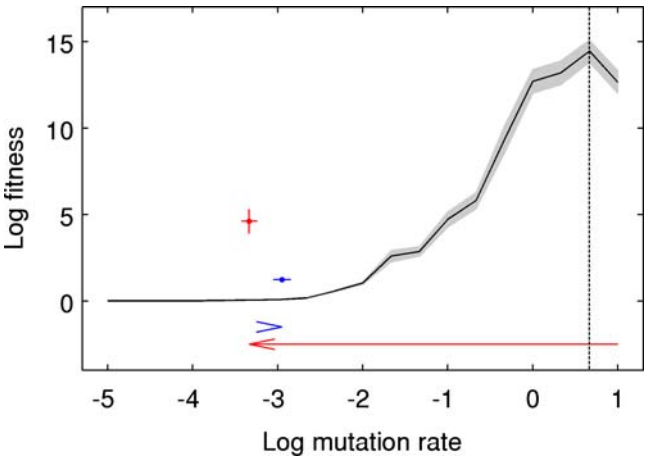


**Figure 1. Evolution of suboptimal mutation rates on a complex fitness landscape.** Fitness is shown as a function of the genomic mutation rate. The solid line shows mean fitness of the final population, itself averaged over 50 runs, for 15 different static mutation rates ($U = 10^{-5}$, $10^{-4}$ and from $10^{-3}$ to 10 at 1/3 $\log_{10}$ intervals). The shaded area represents $\pm 1$ s.e.m. The optimal mutation rate—the rate that maximized final fitness—was $U_{opt} \approx 4.641$ (vertical dashed line). The two colored points show the mean fitness and mutation rate of the final population, averaged over 50 runs, in experiments where mutation rates freely evolved with starting values of either 10 (red) or $10^{-3}$ (blue) (error bars represent $\pm 1$ s.e.m). Evolved mutation rates and fitness values were both orders of magnitude lower than those observed in the experiment with $U_{opt}$.
doi:10.1371/journal.pcbi.1000187.g001

The finding that mutation rates evolved to be suboptimal was robust to diverse and substantial changes in the experimental conditions. First, we tested whether our results depended on the particular ancestral organism used. In the original experiments, the ancestor was a default, hand-coded organism. To assess whether this condition substantively influenced our results, we let a population founded by this organism adapt for 50,000 updates to an environment without any rewarded functions, using $U = 4.641$. The most abundant genotype at the end of this preliminary run was then used as the ancestor in repetitions of our original experiments. Second, we modified the complexity of the environment by varying the number of rewarded functions. Third, we tested the effect of environmental fluctuations by introducing periodic changes in the set of rewarded functions. In some of these experiments the non-rewarded functions were neutral, and in others performing these functions reduced fitness. The rate at which environmental fluctuations occurred was also varied. Fourth, we experimented with different implementations of how mutation rates could themselves change over time. In the original experiments, each organism's mutation rate had a constant probability $\Pi$ of changing every generation, and the magnitude of any resulting change was controlled by a dispersion parameter $\sigma$, with $\Pi = 0.5$ and $\sigma = 0.1$. We conducted additional experiments in which we lowered $\Pi$, raised $\sigma$, or both by orders of magnitude. We also explored a configuration where increases in the mutation rate were more likely than decreases, as may happen in biological

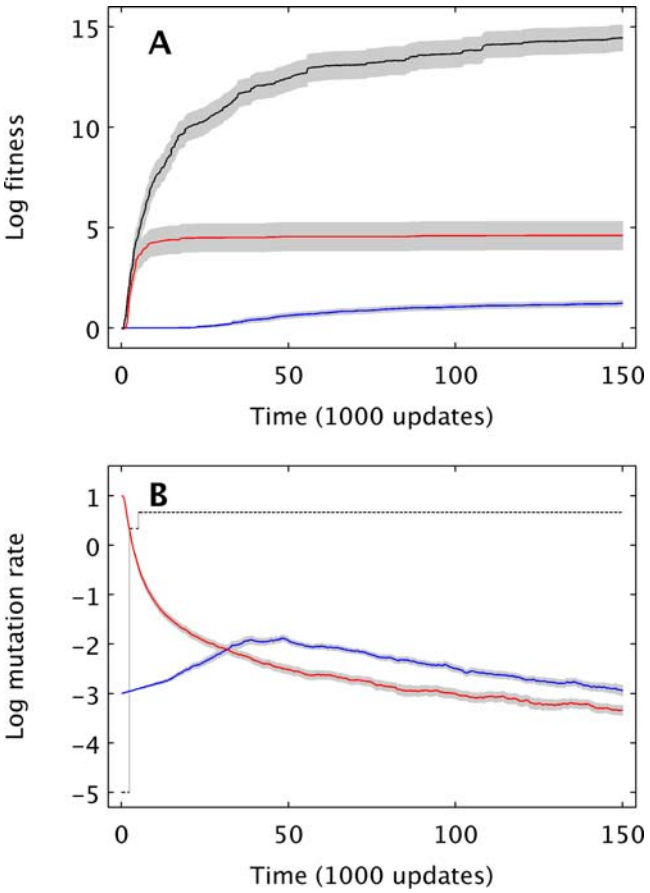


**Figure 2. Evolutionary trajectories for fitness and mutation rate on a complex fitness landscape.** (A) Evolution of average log-fitness±1 s.e.m. for treatments with the mutation rate fixed at $U_{opt}=4.641$ (black) and for treatments with variable mutation rates starting at either 10 (red) or $10^{-3}$ (blue). (B) Evolution of average log genomic mutation rate±1 s.e.m. for treatments with variable mutation rates starting at either 10 (red) or $10^{-3}$ (blue). The black line indicates the mutation rate that had produced the highest average fitness for that time point.
doi:10.1371/journal.pcbi.1000187.g002

systems where it is more likely for mutations to harm than to improve an existing DNA repair pathway. Finally, we let the mutation rate apply reflexively to itself, such that high-fidelity genotypes rarely changed their mutation rates whereas low-fidelity genotypes did so frequently. In all of these additional experiments, mutation rates evolved to suboptimal levels (data not shown). We conclude, therefore, that selection fails to optimize mutation rates for long-term adaptation in a broad range of experimental conditions.

## Selection Favors Suboptimal Mutation Rates Because They Are Advantageous in the Short Term

A possible explanation for why mutation rates evolved to be much lower than $U_{opt}$ is that selection favored those genotypes that minimized the short-term fitness costs caused by deleterious mutations. This explanation is supported by the observations that, during the earliest generations of the evolution experiments, the lowest mutation rate yielded the highest fitness values. To test whether short-term selection would favor low mutation rates, we performed competition experiments between two kinds of organisms, designated A and B. These organisms were identical except for their mutation rate, which was set to $U_{opt}$ for A and 0 for B; neither mutation rate was allowed to change during the competition. All competitions were conducted with the same environmental configurations as in the main experiments. In all of 50 runs, B drove A extinct in fewer than 40 generations. Competitions were also performed using $U=1.0$ and $U=2.154$ for B in order to address whether selection would also favor less extreme reductions in mutation rate. In both treatments, B drove A extinct in all 50 trials in fewer than 800 generations. These experiments confirm our hypothesis that natural selection was shortsighted and favored low mutation rates, even when such low rates precluded further adaptation.

## Whether an Optimal Mutation Rate Can Evolve Depends on the Ruggedness of the Fitness Landscape

We conclude from the results presented thus far that the failure of the evolving populations to achieve or even maintain the mutation rates that maximize long-term adaptation reflect the conflict between the short-term cost of deleterious mutations and the long-term potential for adaptive evolution. We further hypothesize that the resolution of this tension may depend on the topology of the fitness landscape on which evolution occurs. In a rugged fitness landscape, where there are multiple peaks separated by maladaptive valleys [35,36], populations at a local optimum must traverse regions of low fitness in the short-term in order to reach higher-fitness solutions in the long-term. This conflict leads us to hypothesize that the inability of natural selection to optimize mutation rates may depend on the ruggedness of the fitness landscape. The ideal test of this hypothesis requires comparing the evolution of mutation rates on fitness landscapes with and without fitness valleys. This test cannot be performed using the standard Avida setup, owing to the presence of extensive genetic interactions that make the fitness landscape complex and rugged [23]. We therefore modified Avida to allow simple, explicit, user-defined fitness functions that allowed us to manipulate the ruggedness of the fitness landscape (Methods, Figure 3). Adaptation occurs so fast when using these simple configurations that we also had to make the environment fluctuate between two 'seasons' in order to ensure a continual opportunity for beneficial mutations. These fluctuations mean that genotypes that are more fit in one season are less fit in the other (Figure 3).

A quantitative investigation of mutation rates spanning orders of magnitude revealed, once again, that intermediate mutation rates were optimal over the long-term (Figure 3). We then allowed mutation rates to evolve starting at a genomic mutation rate either below ($10^{-5}$) or above (1) the long-term optimum. Near-optimal values were efficiently selected in those landscapes without a fitness valley or with a narrow valley (Figure 3, rows 1 and 2). However, as the width of the valley grew, mutation rates evolved to be orders of magnitude lower than $U_{opt}$ (Figure 3, rows 3 and 4). Fitness values were again used to judge the optimality of mutation rates. With no valleys or with narrow valleys, the average fitness in populations with variable mutation rates was slightly above that of populations with a constant rate of $U_{opt}$ (Figure 3, rows 1 and 2, Mann-Whitney test, $P<0.001$ in both cases), which indicates a small benefit of adjusting mutation rates during evolution [37]. In stark contrast, for wider valleys, the average fitness in populations with variable mutation rates was far below that of populations with a constant rate of $U_{opt}$ (Figure 3, , rows 3 and 4 Mann-Whitney test, $P<0.001$ in both cases), confirming that the evolved mutation rates were suboptimal on the these rugged landscapes.

These results show that there exists a conflict between short-term and long-term evolutionary strategies on rugged landscapes. In the short-term, low mutation rates are favored because they

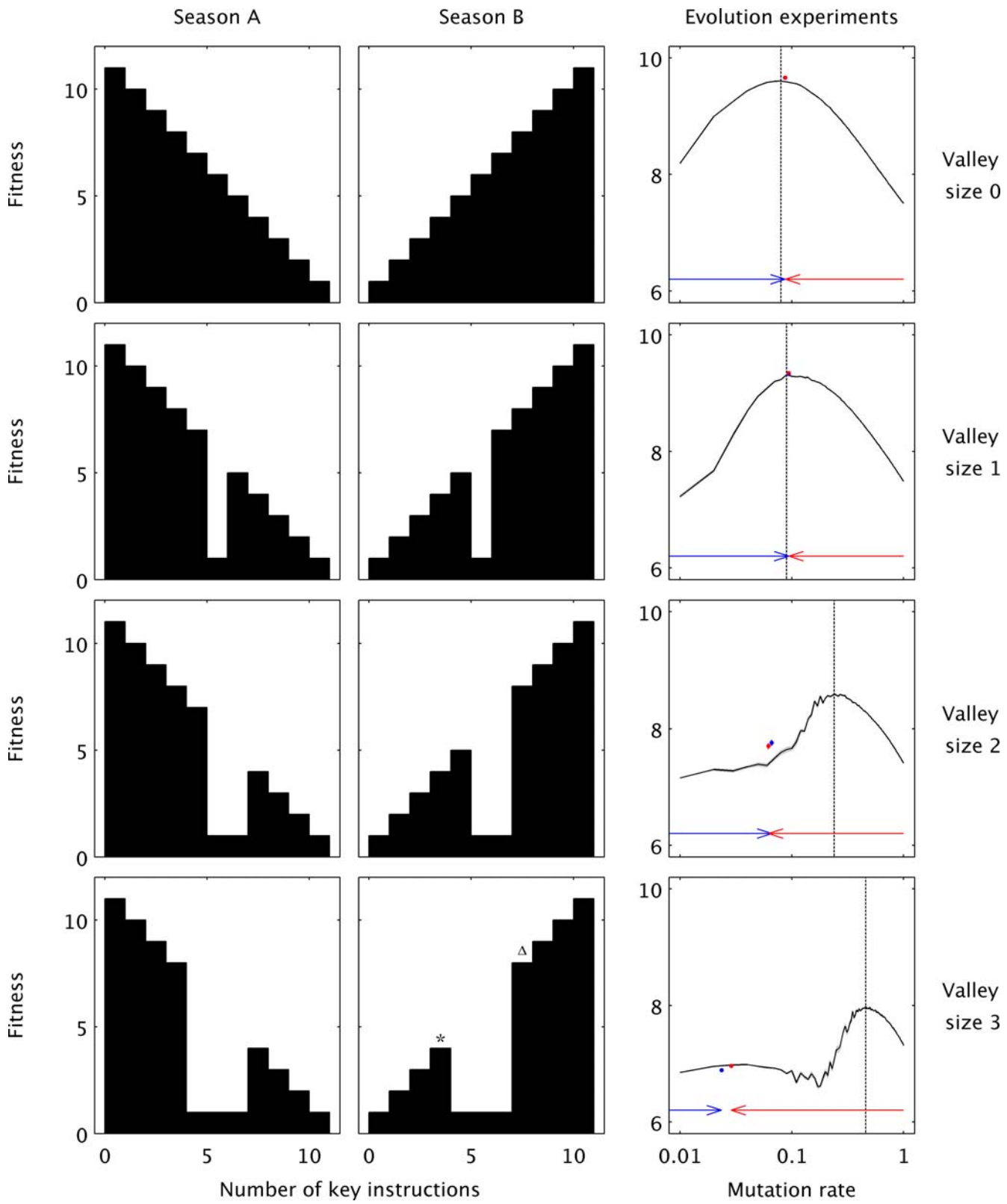


**Figure 3. Evolution of mutation rates on simple fitness landscapes with different ruggedness.** Here, fitness depended solely on the match between the environment and the number of a key instruction that organisms had in their genomes. In season A (left column) the key instruction was deleterious while it was beneficial in season B (center column). Rugged fitness landscapes with maladaptive valleys (rows 2–4) were introduced by setting the fitness of organisms with intermediate numbers of the key instruction to the minimum fitness level of one. The right-most column shows the results of evolution experiments under each of these selective regimes. Final fitness is shown as a function of genomic mutation rate for both static and dynamic mutation rates. The solid black line represents the average of the mean fitness across 10 runs for each of 100 different static mutation rates ranging from $U = 0.01$ to 1 in increments of 0.01. The two colored points represent the mean fitness and mutation rate,

both averaged over 50 runs where the mutation rate freely evolved, with initial rates of $U = 1$ (red) or $10^{-5}$ (blue). Mutation rate and fitness values were time-averaged over the last 10 of 50 environmental changes. Owing to very similar final values, despite the very large initial differences, the individual colored points are indistinguishable in the first two rows, and error bars are not visible. The arrows indicate where mutation rates began and ended, on average, for the dynamic-rate experiments. Although the optimal mutation rate increases as a function of valley size (note the right-shift in the dashed line from top to bottom), the evolved mutation rates in fact decrease as a function of valley size (note the left-shift of the blue and red points from top to bottom).
doi:10.1371/journal.pcbi.1000187.g003

reduce the load of deleterious mutations, whereas in the long-term, high rates are favored because they increase the chance of producing beneficial mutants. Whether the short-term interests dominate, allowing genotypes with suboptimal mutation rates to spread, should be a function of the expected waiting time until the discovery of a beneficial mutant. To test this prediction, we competed genotypes with either optimal or suboptimal rates in the explicit fitness landscape with a valley size of three (Figure 3). In one set of experiments, we placed all organisms of both types on the low local fitness peak (asterisk in Figure 3) and let them compete for 300 generations (the duration of one season in the previous experiments). We then repeated the same experiments except that one of the individuals with the long-term optimal mutation rate started on the other side of the valley (triangle in Figure 3), such that the waiting time for the production of a beneficial mutant was eliminated. A comparison between these two sets of competition experiments shows that the probability that a genotype with a mutation rate that is below the long-term optimum can invade declines significantly when the waiting time to discover beneficial mutants is artificially eliminated (Table 1). This result illustrates why wider valleys, which create longer waiting times for beneficial mutants, cause the evolution of suboptimal mutation rates.

The reader may also notice that the probability of invasion by the genotype with the suboptimal mutation rate was rather small in both sets of experiments (Table 1). This observation might seem, at first glance, to be at odds with the fact that mutation rates evolved over the long run to be extremely suboptimal (Figure 3, rows 3 and 4). This difference makes sense, however, for two interrelated reasons. First, each environmental change that follows the fixation of a mutation on one adaptive peak requires another waiting period for a beneficial mutation, which provides another opportunity for invasion by a genotype with a suboptimal mutation rate that reduces the mutational load. Second, any reductions in the mutation rate become self-reinforcing, as the lower mutation rates make it less likely to generate a beneficial mutant on a distant peak, which increases the expected waiting time for the generation of the next beneficial mutants, thereby increasing the opportunity for a genotype with an even lower mutation rate to invade.

Finally, we examined whether the frequency with which the mutation rate changes (in essence, the mutation rate in the pathway that encodes the mutation rate), which we call $\Pi$, affects the evolutionarily stable mutation rate. Our intuition was that lower values of $\Pi$ would make contests between lineages with different mutation rates less frequent, but that the long-term results of many such contests would remain the same. To test this prediction we again used the explicit landscape with a valley size of three. Even when $\Pi$ varied over four orders of magnitude, it did not affect the final mutation rate that was reached (Figure 4). Hence, the inability of selection to optimize the mutation rate for long-term adaptation depends on the topology of the fitness landscape, but not on the frequency with which the mutation rate itself changes.

## Discussion

We have shown that mutation rates evolve to near-optimal levels on extremely smooth fitness landscapes. However, if fitness landscapes are rugged, and the maladapted valleys between nearby fitness peaks are wide, then the scarcity of immediately accessible beneficial mutations tips the scale such that short-term selection favors mutation rates that are far below the optimum that would produce the fastest long-term adaptation. Moreover, this process is self-reinforcing because the lower the mutation rate, the less likely it becomes to produce a genotype on the other side of the fitness valley, thereby effectively widening the valley. The digital organisms in the standard Avida configuration used in our first set of experiments exhibit extensive and variable genetic interactions, making the fitness landscape rugged [23]. In those experiments, populations invariably evolved to have mutation rates that were far below the rate that would maximize their long-term fitness gains. We hypothesized that the ruggedness of the landscape was responsible for this inability to optimize their mutation rate for long-term adaptation. In order to test this hypothesis rigorously, we had to change the fitness landscape in Avida from one that is an emergent feature of complex interactions among many

**Table 1.** Outcomes of competitions between lineages with optimal ($U_{opt} = 0.24$) versus suboptimal ($U_{subopt}$) mutation rates in the explicit fitness landscape with a valley size of 3.

| | With Waiting Time | | | Without Waiting Time | | | |
|---|---|---|---|---|---|---|---|
| $U_{subopt}$ | $U_{opt}$ Fixed | $U_{subopt}$ Fixed | Neither Fixed | $U_{opt}$ Fixed | $U_{subopt}$ Fixed | Neither Fixed | *P* |
| 0 | 238 | 2 | 10 | 249 | 0 | 1 | 0.0082 |
| 0.06 | 149 | 24 | 77 | 242 | 0 | 8 | $<0.0001$ |
| 0.12 | 185 | 12 | 53 | 229 | 1 | 20 | $<0.0001$ |

A total of 250 runs were performed for each treatment shown below. The two lineages started with equal numbers in all cases. The entries show the number of times that each lineage was fixed (i.e., reached 100% of the total population) or that neither lineage was fixed within 300 generations. With waiting time: all individuals started at the lower fitness peak (asterisk in Figure 3). Without waiting time: one individual belonging to the lineage with $U_{opt}$ started on the other side of the fitness valley (triangle in Figure 3). Three different values of $U_{subopt}$ were examined. *P* values are based on $\chi^2$ tests (with 2 degrees of freedom) that measured the effect of waiting time.
doi:10.1371/journal.pcbi.1000187.t001

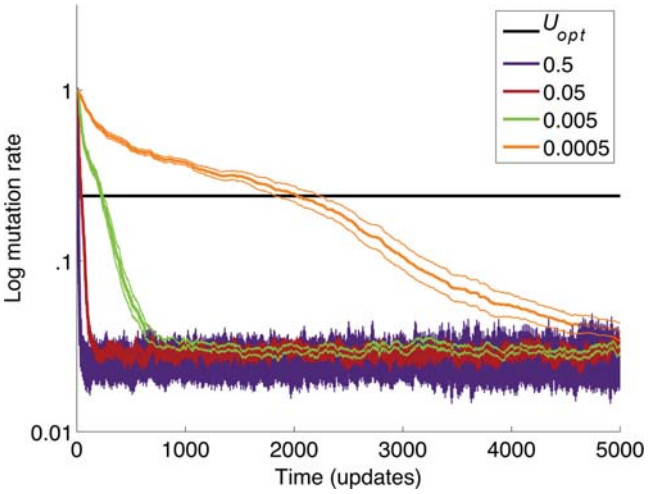


**Figure 4. Evolutionarily stable mutation rate does not depend on the frequency with which the mutation rate changes (Π).** The evolution of mutation rates in the explicit fitness landscape with a valley size of three is shown for several values of Π, as indicated by the colored key. Each curve shows the average of 20 runs; the adjacent bands represent±1 s.e.m. The value of $U_{opt}$ was determined in previous experiments (see text). The rate of approach toward the evolutionarily stable mutation rate depends on Π, but the equilibrium value itself does not.
doi:10.1371/journal.pcbi.1000187.g004

instructions to a much simpler surface that could be tuned to be either smooth or rugged. We found that evolving populations were indeed able to achieve mutation rates that maximized their rate of adaptation on smooth landscapes, whereas they became stuck at much lower mutation rates when the valleys between fitness peaks became too large, thus confirming our hypothesis. A growing body of experiments with viruses, bacteria, yeast, and higher eukaryotes shows that epistatic interactions are widespread and vary in their sign and intensity, implying that natural fitness landscapes are also often rugged [35,36,38]. Thus, our finding that rugged fitness landscapes can impede the optimization of mutation rates for long-term evolutionary adaptation is relevant to the natural world.

Our experiments were performed under conditions that were favorable for the optimization of mutation rates. First, the organisms reproduced asexually. Both theoretical [12,39,40] and experimental work [15] has shown that asexuality facilitates the evolution of elevated mutation rates, because sexual recombination breaks up the linkage between mutator alleles that increase mutation rates and the beneficial mutations that are generated by the mutators. Second, to ensure that beneficial mutations were always available, our experiments used either an environment with more rewarded functions than the organisms ever evolved during a run (standard configuration) or a changing environment (explicit landscapes configuration). Third, population sizes were large and strong directional selection was imposed, so that drift was only a minor force in our experiments. Smaller populations might traverse maladaptive valleys more easily, owing to increased drift. However, small populations would be less likely to generate the multiple simultaneous mutations that would allow them to leap across these valleys in a single generation. In populations much larger than those we tested, the probability of an adaptive leap involving multiple simultaneous mutations would increase, but selection should be more powerful in preventing a multi-generation transition across a valley via drift. The effect of population size on the optimal mutation rate, and on the evolution of suboptimal mutation rates, thus remains an interesting area for future investigation. Nevertheless, while the optimal mutation rate and the precise width of the valley that is necessary to cause the evolution of a suboptimal rate may depend on population size, we would not expect that dependency to undermine the general conclusion of this paper, namely, that on sufficiently rugged fitness landscapes, mutation rates will evolve to be suboptimal for long-term adaptation.

The inability of evolving populations to optimize their mutation rates for long-term adaptation, even with such favorable conditions, indicates that mutation rates will be suboptimal under a wide range of circumstances, at least when fitness landscapes are rugged and populations are far from a global fitness peak. While novel environments can promote increases in the mutation rate if many beneficial mutations become accessible [1,13–21,40], our work suggests that this rise will be temporary and, moreover, that even the elevated mutation rates may be suboptimal (Figure 2B). Also, given the difficulty of optimizing mutation rates that we have shown, it seems unlikely that stably high mutation rates, such as those for RNA viruses, are maintained primarily because of the rapid adaptive capacity they bestow, as has sometimes been argued [23,41]. Alternative explanations are needed. For example, the evolution of mutation rates is also influenced by the costs of replication fidelity [8,23], and recent work has suggested that this cost might explain the high mutation rates observed in RNA viruses [24,42]. We expect that a cost of replication fidelity, all else being equal, will increase the evolved mutation rate. However, we would not expect the resulting increase to cause the optimization of mutation rates in general, although in a few fortuitous situations the cost of fidelity might increase the evolved mutation rate by just enough to push it near the optimal rate.

Recent theoretical work by Gerrish et al. [43] has predicted that, contrary to our results, natural selection could favor a self-reinforcing increase in mutation rates in asexual populations. This process would continue even until a population suffered a mutational meltdown and went extinct, because a genotype with an increased mutation rate generates greater numbers of deleterious as well as beneficial mutations. Although not explicitly stated, the prediction of Gerrish et al. [43] of a run-away process toward higher mutation rates appears to assume a smooth fitness landscape. However, as we have shown here, the mutation rate typically evolves to a low value on a rugged fitness landscape, so that the runaway process explored by Gerrish et al. should not occur on such landscapes.

Beyond their implications for understanding nature, our findings are also relevant for applied fields that use evolution to improve the performance of biological and computational systems, from molecular and microbial engineering to robotics and evolutionary computation [44,45]. Researchers using evolution in computational fields have long sought to use natural selection to adjust mutation rates automatically and "on the fly", in such a way that would sustain and even optimize long-term adaptation [46–48]. These efforts were successful on simple "toy" problems [46], but became frustrated when applied to more complex problems because self-adaptive mutation rates generally evolved to suboptimal levels [47,48]. Our results suggest an explanation: the toy problems had smooth fitness landscapes, whereas the complex problems had rugged landscapes with wide valleys that favored evolutionary conservatism. Our findings also imply that high, fixed mutation rates will often outperform self-adaptive rates on more complex problems, although what the fixed rate should be will depend on the particular problem at hand.

In summary, natural selection is not universally effective at optimizing mutation rates for long-term adaptation; in fact, it is very poor in this respect for populations that evolve on complex

fitness landscapes. Also, our results caution against making generalizations based on analyses of simple fitness landscapes, whether one is studying natural systems or using evolution for engineering. As we have shown, the mere inclusion of fitness valleys—which are presumably common to the vast majority of fitness landscapes—can yield radically different conclusions from those based on smooth fitness landscapes.

## Methods

### Experiment One: Standard Configuration

A general description of the Avida software can be found elsewhere [25]. Here, each experiment started with 3,600 identical digital organisms. Genome length was held constant at 100 instructions, with 26 possible instructions per site [27]. Reproduction was asexual. To replicate, an organism first had to copy its genome line by line by repeatedly executing the copy instruction; it then had to execute a divide instruction, which took the offspring and used it to replace a random organism from the population.

During replication, each genomic instruction could mutate to another with probability $\mu$, the genomic mutation rate being $U = 100 \times \mu$. All instructions were equally likely to result from any given mutation. The mutation rate was held constant in some experiments, while in others the rate could change by evolving over time. In treatments where the mutation rate could change, $\mu$ had a constant and high probability $\Pi$ of changing by a small amount during any replication cycle. The magnitude of any resulting change was obtained by drawing $\log_2(\mu_{\text{offspring}}/\mu_{\text{parent}})$ values from a Gaussian distribution $(0,\sigma^2)$. For the experiments in which mutation rates were more likely to increase than to decrease, we drew $\log_2(\mu_{\text{offspring}}/\mu_{\text{parent}})$ from a Gaussian $(b\sigma^2,\sigma^2)$, where $b$ controls the upward bias, and tested values such that mutation rates were up to $\sim$1.6 times more likely to increase than decrease (though seemingly small, this bias has a large cumulative effect over many generations).

Organisms died when another organism's offspring replaced them or when they executed 2,000 instructions without producing an offspring of their own. All experiments using the standard configuration lasted 150,000 updates. Updates are an arbitrary unit of time in Avida; they represent the time during which each organism, on average, executes 30 instructions [25]. In this configuration, an update corresponded to roughly 0.1 generations, although the precise generation time varied depending on the complexity of the evolved organisms' phenotypes.

Each organism's phenotype depended on the complex rules that governed how its genomic program was executed, and its fitness depended on the interaction between the resulting phenotype and its environment [25]. More specifically, each organism had a metabolic rate that affected how fast it executed instructions, which, in turn, affected its reproduction rate. The ancestral rate doubled with every rewarded logic function that an organism performed. The ancestral organisms could self-replicate but not perform any other function. The ability to perform logic functions evolved by mutation and selection during each run. An organism's fitness, therefore, represents its expected growth rate relative to others in the population and depended on both its replication efficiency and its ability to perform computations. All fitness values are expressed relative to the ancestor. In reporting fitness data, relative fitness values were first averaged over all organisms in a population, then $\log_{10}$ transformed, and finally averaged over all replicate populations (independent trials) in an experimental treatment.

To perform logic functions, organisms used inputs consisting of three randomly generated 32-bit strings, which they manipulated to produce an output. The manipulation of these numbers occurred as organisms moved them on and off stacks or between registers by executing instructions such as push, pop, add (combines the numbers in the two specified registers and places the result in a third), shift-r (bit shift right), and so on. A function was rewarded only if the input to output conversion conformed to one of the 77 canonical one-, two- or three-input logic operations. For example, the two-input EQU ('equals') function requires inputting two strings and outputting a third string that had a 1 for each of the 32 bits where both inputs had the same value and a 0 where they differed.

Avida runs are inherently stochastic with respect to mutation and death. Therefore, we performed 50 replicate runs for each treatment. Those replicates had identical initial conditions except for a random number seed. That seed affects the outcome of all subsequent stochastic events.

### Experiment Two: Explicit Landscapes Configuration

The standard and the explicit Avida configurations differed in the instruction set, the fitness calculation and the mode of replication. We modified Avida to mimic a two-allele, 10-locus bit-string model used in a previous study [49]. Genome length was always 10, while each "instruction" was either A or B; the ancestral genome was entirely A. Fitness depended only on the number of A or B instructions in an organism's genome, according to the seasonal scheme shown in Figure 3. Every 300 generations the environment fluctuated between the two seasons, and the experiments ran for 15,000 generations. We found empirically that fluctuating the environment more or less frequently than every 300 generations produced smaller fitness differences between the optimal fixed mutation rate and suboptimal mutation rates (data not shown). That high mutation rates are most fit at an intermediate rate of environmental change has been previously shown [49].

In the standard configuration, digital organisms had to copy their genomic instructions in order to replicate, and their fitness depended on their speed of replication as well as any rewards they obtained for performing computational functions. Under this alternative configuration, the organisms did not copy themselves, and only the number of A or B instructions mattered to their fitness. The rest of the setup, such as population size, was identical to the standard configuration.

### Software

All experiments were performed with the Avida software, which can be downloaded for free at http://devolab.cse.msu.edu/software/avida. Default settings were used unless otherwise indicated.

## Acknowledgments

We thank members of the Digital Evolution Lab at Michigan State University for valuable discussions of this work.

## Author Contributions

Conceived and designed the experiments: JC REL SFE RS. Performed the experiments: JC DM. Analyzed the data: JC DM CO REL SFE RS. Contributed reagents/materials/analysis tools: JC. Wrote the paper: JC RS. Initiated investigations into the subject: JC. Designed the initial experiments: JC SFE RS. Made necessary modifications to Avida: JC CO. Conducted the majority of the experiments: JC. Analyzed and interpreted the data: JC RS. Discussions to improve the experimental designs and interpretation of the results: JC DM CO REL SFE RS. Edited the manuscript: JC DM CO REL SFE RS. Designed the competition experiments: REL. Performed the competition experiments: DM. Prepared the graphics: DM. Conducted the mutation rate bias experiments: DM. Worked with JC on self-reflective mutation rates: DM.

## References


1. Chang DK, Metzgar D, Wills C, Boland CR (2001) Microsatellites in the eukaryotic DNA mismatch repair genes as modulators of evolutionary mutation rate. Genome Res 11: 1145–1146.
2. Kunkel TA, Bebenek K (2000) DNA replication fidelity. Annu Rev Biochem 69: 497–529.
3. Miller JH (1996) Spontaneous mutators in bacteria: insights into pathways of mutagenesis and repair. Annu Rev Microbiol 50: 625–643.
4. Schaaper RM (1998) Antimutator mutants in bacteriophage T4 and *Escherichia coli*. Genetics 148: 1579–1585.
5. Schofield MJ, Hsieh P (2003) DNA mismatch repair: molecular mechanisms and biological function. Annu Rev Microbiol 57: 579–608.
6. Fisher RA (1930) The genetical theory of natural selection. Oxford, UK: Oxford University Press.
7. Johnson T, Barton NH (2002) The effect of deleterious alleles on adaptation in asexual populations. Genetics 162: 395–411.
8. Orr HA (2000) The rate of adaptation in asexuals. Genetics 155: 961–968.
9. Sniegowski PD, Gerrish PJ, Johnson T, Shaver A (2000) The evolution of mutation rates: separating causes from consequences. Bioessays 22: 1057–1066.
10. André JB, Godelle B (2006) The evolution of mutation rate in finite asexual populations. Genetics 172: 611–626.
11. Johnson T (1999) The approach to mutation-selection balance in an infinite asexual population, and the evolution of mutation rates. Proc Biol Sci 266: 2389–2397.
12. Kimura M (1967) On the evolutionary adjustment of spontaneous mutation rates. Genet Res 9: 23–34.
13. de Visser JA (2002) The fate of microbial mutators. Microbiology 148: 1247–52.
14. Giraud A, Matic I, Tenaillon O, Clara A, Radman M, Fons M, Taddei F (2001) Costs and benefits of high mutation rates: adaptive evolution of bacteria in the mouse gut. Science 291: 2606–2608.
15. Sniegowski PD, Gerrish PJ, Lenski RE (1997) Evolution of high mutation rates in experimental populations of *Escherichia coli*. Nature 387: 703–705.
16. de Visser JAGM, Zeyl CW, Gerrish PJ, Blanchard JL, Lenski RE (1999) Diminishing returns from mutation supply rate in asexual populations. Science 283: 404–406.
17. Pal C, Maciá MD, Oliver A, Schachar I, Buckling A (2007) Coevolution with viruses drives the evolution of bacterial mutation rates. Nature 450: 1079–1081.
18. Chao L, Cox EC (1983) Competition between high and low mutating strains of *Escherichia coli*. Evolution 37: 125–134.
19. Denamur E, Matic I (2006) Evolution of mutation rates in bacteria. Mol Microbiol 60: 820–827.
20. Kivisaar M (2003) Stationary phase mutagenesis: mechanisms that accelerate adaptation of microbial populations under environmental stress. Environ Microbiol 5: 814–827.
21. Kashi Y, King DG (2006) Simple sequence repeats as advantageous mutators in evolution. Trends Genet 22: 253–259.
22. Pfeiffer JK, Kirkegaard K (2005) Increased fidelity reduces poliovirus fitness and virulence under selective pressure in mice. PLoS Pathog 1: e11. doi:10.1371/journal.ppat.0010011.
23. Vignuzzi M, Stone JK, Arnold JJ, Cameron CE, Andino R (2006) Quasispecies diversity determines pathogenesis through cooperative interactions in a viral population. Nature 439: 344–348.
24. Furió V, Moya A, Sanjuán R (2005) The cost of replication fidelity in an RNA virus. Proc Natl Acad Sci U S A 102: 10233–10237.
25. Ofria C, Wilke CO (2004) Avida: a software platform for research in computational evolutionary biology. Artif Life 10: 191–229.
26. Chow SS, Wilke CO, Ofria C, Lenski RE, Adami C (2004) Adaptive radiation from resource competition in digital organisms. Science 305: 84–86.
27. Lenski RE, Ofria C, Collier TC, Adami C (1999) Genome complexity, robustness and genetic interactions in digital organisms. Nature 400: 661–664.
28. Lenski RE, Ofria C, Pennock RT, Adami C (2003) The evolutionary origin of complex features. Nature 423: 139–144.
29. Misevic D, Ofria C, Lenski RE (2006) Sexual reproduction reshapes the genetic architecture of digital organisms. Proc Biol Sci 273: 457–464.
30. Adami C, Ofria C, Collier TC (2000) Evolution of biological complexity. Proc Natl Acad Sci U S A 97: 4463–4468.
31. Wilke CO, Wang JL, Ofria C, Lenski RE, Adami C (2001) Evolution of digital organisms at high mutation rates leads to survival of the flattest. Nature 412: 331–333.
32. Ofria C, Adami C, Collier TC (2003) Selective pressures on genomes in molecular evolution. J Theor Biol 222: 447–483.
33. Adami C (2006) Digital genetics: unravelling the genetic basis of evolution. Nature Rev Genet 7: 109–118.
34. Goings S, Clune J, Ofria C, Pennock RT (2004) Kin-selection: the rise and fall of kin-cheaters. In: Pollack J, Bedau M, Husbands P, Ikegami T, Watson R, eds. Proceedings of Artificial Life Nine. Cambridge (Massachusetts): MIT Press. pp 303–308.
35. Whitlock MC, Phillips PC, Moore FBG, Tonsor SJ (1995) Multiple fitness peaks and epistasis. Annu Rev Ecol Syst 26: 601–629.
36. Wolf JB, Brodie ED III, Wade MJ (2000) Epistasis and the evolutionary process. Oxford, UK: Oxford University Press.
37. Bäck T, Schütz M (1996) Intelligent mutation rate control in canonical genetic algorithm. In: Ras ZV, Michalewicz M, eds. Foundations of Intelligent Systems, 9th International Symposium. Berlin: Springer. pp 158–160.
38. Poelwijk FJ, Kiviet DJ, Weinreich DM, Tans SJ (2007) Empirical fitness landscapes reveal accessible evolutionary paths. Nature 445: 383–386.
39. Drake JW, Charlesworth B, Charlesworth D, Crow JF (1998) Rates of spontaneous mutation. Genetics 148: 1667–1686.
40. Taddei F, Radman M, Maynard-Smith J, Toupance B, Gouyon PH, Godelle B (1997) Role of mutator alleles in adaptive evolution. Nature 387: 700–702.
41. Domingo E, Holland JJ (1997) RNA virus mutations and fitness for survival. Annu Rev Microbiol 51: 151–178.
42. Furió V, Moya A, Sanjuán R (2007) The cost of replication fidelity in human immunodeficiency virus type 1. Proc Biol Sci 274: 225–230.
43. Gerrish PJ, Colato A, Perelson AS, Sniegowski PD (2007) Complete genetic linkage can subvert natural selection. Proc Natl Acad Sci U S A 104: 6266–6271.
44. Arnold FH (2001) Combinatorial and computational challenges for biocatalyst design. Nature 409: 253–257.
45. Lipson H, Pollack JB (2000) Automatic design and manufacture of artificial lifeforms. Nature 406: 974–978.
46. Eiben AE, Hinterding R, Michalewicz Z (1999) Parameter control in evolutionary algorithms. IEEE Trans Evol Comput 3: 124–141.
47. Rand W, Riolo R (2005) The problem with self-adaptive mutation rates in some environments. In: Byer H-G, O'Reilley U-M, Arnold DV, Banzhaf W, Blum C, et al., eds. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation. Washington, DC: Association for Computing Machinery. pp 1493–1500.
48. Clune J, Goings S, Goodman ED, Punch W (2005) Investigations in Meta-GAs: panaceas or pipe dreams? In: Byer H-G, O'Reilley U-M, Arnold DV, Banzhaf W, Blum C, et al., eds. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation. Washington, DC: Association for Computing Machinery. pp 235–241.
49. Travis JM, Travis ER (2002) Mutator dynamics in fluctuating environments. Proc Biol Sci 269: 591–597.

# On the Performance of Indirect Encoding Across the Continuum of Regularity

Jeff Clune, Kenneth O. Stanley, Robert T. Pennock, and Charles Ofria

***Abstract*—This paper investigates how an evolutionary algorithm with an indirect encoding exploits the property of phenotypic regularity, an important design principle found in natural organisms and engineered designs. We present the first comprehensive study showing that such phenotypic regularity enables an indirect encoding to outperform direct encoding controls as problem regularity increases. Such an ability to produce regular solutions that can exploit the regularity of problems is an important prerequisite if evolutionary algorithms are to scale to high-dimensional real-world problems, which typically contain many regularities, both known and unrecognized. The indirect encoding in this case study is HyperNEAT, which evolves artificial neural networks (ANNs) in a manner inspired by concepts from biological development. We demonstrate that, in contrast to two direct encoding controls, HyperNEAT produces both regular behaviors and regular ANNs, which enables HyperNEAT to significantly outperform the direct encodings as regularity increases in three problem domains. We also show that the *types* of regularities HyperNEAT produces can be biased, allowing domain knowledge and preferences to be injected into the search. Finally, we examine the downside of a bias toward regularity. Even when a solution is mainly regular, some irregularity may be needed to perfect its functionality. This insight is illustrated by a new algorithm called HybrID that hybridizes indirect and direct encodings, which matched HyperNEAT's performance on regular problems yet outperformed it on problems with some irregularity. HybrID's ability to improve upon the performance of HyperNEAT raises the question of whether indirect encodings may ultimately excel not as stand-alone algorithms, but by being hybridized with a further process of refinement, wherein the indirect encoding produces patterns that exploit problem regularity and the refining process modifies that pattern to capture irregularities. This paper thus paints a more complete picture of indirect encodings than prior studies because it analyzes the impact of the continuum between irregularity and regularity on the performance of such encodings, and ultimately suggests a path forward that combines indirect encodings with a separate process of refinement.**

***Index Terms*—Artificial neural networks, developmental encodings, generative encodings, HyperNEAT, indirect encodings, regularity.**

## I. Introduction and Motivation

A LONG-STANDING challenge for those who work with evolutionary algorithms (EAs) is to synthetically evolve entities that rival or surpass the complexity of both natural organisms and designs produced through human engineering. A key design principle critical to the success of both is regularity. *Regularity* refers to the compressibility of the information describing a structure, and typically involves symmetries and the repetition of modules or other design motifs [1]. Regularities allow solutions to sub-problems to be reused in a design, as in the cells of a body or the wheels of a car.

The level of regularity that an EA tends to produce is affected by the *encoding*, which is how information is stored in the genome and the process by which that information produces the phenotype. Regularity in evolved solutions is less likely to emerge with *direct encodings*, wherein each element in the genotype encodes an independent aspect of the phenotype. In contrast, regularities are common with *indirect encodings* (also known as generative or developmental encodings), wherein information in the genome can be *reused* to affect many parts of the phenotype [2]. Natural organisms are regular largely because they are encoded indirectly [3]–[5].

Reusing genetic information also facilitates scalability. With indirect encodings, evolution can search in a low-dimensional space yet produce phenotypes with many more dimensions. For example, only about 25 000 genes encode the information that produces the trillions of cells that make up a human [6].

Many prior studies have shown that indirect encodings often outperform direct encodings [2], [7]–[14]. However, in each case the problem domain is highly regular, or the regularity of the problem is unspecified and ambiguous. To date, no systematic study has been performed on how indirect encodings perform across a continuum from regular problems to irregular problems. This gap in our knowledge raises the question of whether indirect encodings achieve their increased performance on regular problems at the expense of performing

Manuscript received April 30, 2010; revised October 1, 2010; accepted December 5, 2010. This work was supported in part by NSF, under Grants CCF-0643952, CCF-0750787, CNS-0751155, CCF-0820220, CNS-0915855, and DBI-0939454, by the Cambridge Templeton Consortium, the U.S. Army Grant W911NF-08-1-0495, a Quality Fund Grant from Michigan State University, the DARPA FunBio Program, and a NSF Postdoctoral Research Fellowship in Biology to J. Clune (Award DBI-1003220).

J. Clune was with the Department of Computer Science, Michigan State University, East Lansing, MI 48824 USA. He is currently with the Department of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14850 USA (e-mail: jeffclune@cornell.edu).

K. O. Stanley is with the School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816 USA (e-mail: kstanley@eecs.ucf.edu).

R. T. Pennock is with the BEACON Center for the Study of Evolution in Action, Department of Computer Science and Engineering and the Department of Philosophy, Lyman Briggs College, Michigan State University, East Lansing, MI 48824 USA (e-mail: pennock5@msu.edu).

C. Ofria is with the BEACON Center for the Study of Evolution in Action, Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: ofria@msu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2010.2104157

poorly on problems with intermediate or low regularity. To fill that gap, this paper provides a systematic comparison of an indirect encoding and two direct encoding controls on multiple problems as problem regularity is varied from high to low.

The indirect encoding in this paper is motivated by a key concept from developmental biology. Nature builds complex organisms by producing increasingly complex geometric coordinate frames, and then determining the fate of phenotypic elements as a function of their location within such frames [3]. This process produces phenotypes with regularity, modularity, and hierarchy [3], [15], which are beneficial design principles [1]. Yet the exploitation of geometric information in natural organisms is challenging because each phenotypic element (e.g., a cell) does not have access to its global geometric position. Developing organisms therefore first have to generate such positional information via chemical gradients before exploiting it. In synthetic evolution, however, we can skip this first step by assigning geometric coordinates to phenotypic elements and allowing a genome to determine the fates of phenotypic elements as a function of these coordinates [16]. This technique allows evolution to start with geometric information and immediately exploit it, instead of first needing to discover how to produce geometric information before exploiting it.

One encoding that harnesses this idea is hypercube-based neuroevolution of augmenting topologies (HyperNEAT). HyperNEAT's explicit incorporation of geometry enables it to exploit geometric relationships in a problem domain [14], [17]–[19]. HyperNEAT has performed well on a wide range of problems, such as generating gaits for legged robots [20], pattern recognition [14], controlling multiagent teams [21], [22], and evaluating checkers boards [18], [19]. Additionally, on a soccer keepaway task that is a common benchmark for reinforcement learning algorithms, HyperNEAT produced the highest score to date for any type of algorithm [23]. Because HyperNEAT abstracts a key geometric ingredient that drives the success of natural development, and because empirically it has been shown to be a promising encoding, it is interesting to observe the regularities that HyperNEAT produces. Specific questions can be addressed, such as what types of regularities HyperNEAT generates (e.g., repeating, symmetric, and so on), whether it can create variations on repeated themes instead of being restricted to identical repetitions, and what kinds of irregular exceptions it can make to the patterns it generates. An additional question of interest is whether the experimenter can bias the regularities that HyperNEAT produces to inject domain knowledge into the algorithm. All of these questions are addressed in this paper.

Results in this paper show that HyperNEAT exploits even intermediate problem regularity, and thus increasingly outcompetes direct encoding controls as problem regularity increases. HyperNEAT achieves this success by producing regular artificial neural networks (ANNs) that in turn produce regular behaviors. HyperNEAT also proves more evolvable than direct encoding controls and its solutions generalize better.

However, an important accompanying result is that HyperNEAT's performance decreases on irregular problems, partly because of its bias toward producing regularities. To investigate this effect further, we introduce a new algorithm called HybrID that allows the HyperNEAT indirect encoding to produce regular patterns in concert with a direct encoding that can modify these patterns to produce irregularities. HybrID matches HyperNEAT's performance on regular problems, but outperforms HyperNEAT on problems with irregularity, which demonstrates that HyperNEAT struggles to generate certain kinds of irregularity on its own. The success of HybrID raises the interesting question of whether indirect encodings may truly excel not as stand-alone algorithms, but in combination with a further process that refines their regular patterns. Intriguingly, this further process in nature could be lifetime adaptation via learning.

In the following sections, we introduce HyperNEAT, two direct encoding controls, and the three problem domains. We then relate the experimental results and discuss them before offering concluding remarks.

## II. HyperNEAT and the Direct Encoding Controls

In this section, we describe the indirect encoding HyperNEAT and two direct encodings that serve as controls.

### A. HyperNEAT

In 2007 an encoding was introduced called compositional pattern producing networks (CPPNs), which abstracts the process of natural development without requiring the simulation of diffusing chemicals [16]. When CPPNs encode ANNs, the algorithm is called HyperNEAT [14], which is described in detail below. A key idea behind CPPNs is that complex patterns can be produced by determining attributes of their phenotypic components as a function of their geometric location. This idea is based on the belief that cells (or higher-level modules) in nature often differentiate into their possible types (e.g., kidney, liver, and so on) as a function of where they are situated in geometric space. For example, for some insects, a segment at the anterior pole should produce antennae and a segment at the posterior pole should produce a stinger.

Components of natural organisms cannot directly determine their location in space, so organisms have evolved developmental processes that create chemical gradients that organismal components use to figure out where they are and, thus, what to become [3]. For example, early in the development of embryos, different axes (e.g., anterior-posterior) are indicated by chemical gradients. Additional gradients signaled by different proteins can exist in the same area to represent a different pattern, such as a repeating motif. Downstream genes, such as Hox genes, can then combine repeated and asymmetric information to govern segmental differentiation [3]. Further coordinate frames can then be set up within segments to govern intra-module patterns.

1) *CPPNs:* One of the key insights behind CPPNs is that cells *in silico* can be directly given their geometric coordinates. The CPPN genome is a function that takes geometric coordinates as inputs and outputs the fate of an organismal component. When CPPNs encode 2-D pictures, the coordinates of each pixel on the canvas (e.g., $x = 2$, $y = 4$)

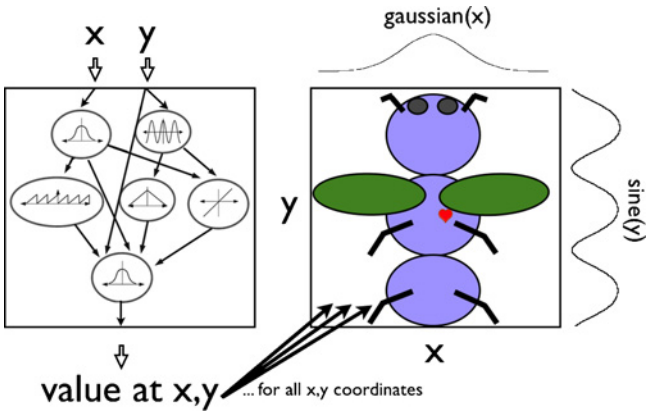


Fig. 1. Compositional pattern producing networks. CPPNs compose mathematical functions to generate regularities, such as symmetries and repeated modules, with and without variation. This figure is adapted from Stanley [16].

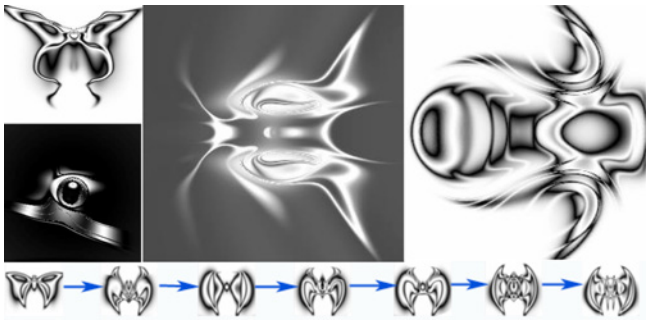

Fig. 2. Images evolved with CPPNs. Displayed are pictures from picbreeder.org [24], a website where visitors select images from a population evolved with the CPPN indirect encoding, which is also used in HyperNEAT. The bottom row shows images from a single lineage. Arrows represent intermediate forms that are not pictured.

are iteratively passed to the CPPN genome, and the output of the function is the color or shade of the pixel (Fig. 1).

Each CPPN is a directed graph in which every node is itself a single function, such as sine or Gaussian. The nature of the functions can create a wide variety of desirable properties, such as symmetry (e.g., a Gaussian function) and repetition (e.g., a sine function) that evolution can exploit. Because the genome allows functions to be made of other functions, coordinate frames can be combined and hierarchies can develop. For instance, a sine function early in the network can create a repeating theme that, when passed into the symmetrical Gaussian function, creates a repeating series of symmetrical motifs (Fig. 1). This procedure is similar to the natural developmental processes described above [3].

The links that connect and allow information to flow between nodes in a CPPN have a weight that can magnify or diminish the values that pass along them. Mutations that change these weights may, for example, give a stronger influence to a symmetry-generating part of a network while diminishing the contribution from another part.

When CPPNs are evolved artificially with humans performing the selection, the evolved shapes look complex and natural (Fig. 2) [24]. Moreover, these images display the features in natural organisms that indirect encodings were designed to produce, namely, symmetries and the repetition of themes, with and without variation.

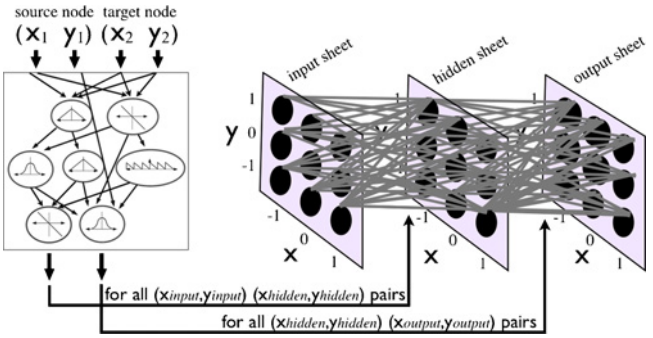


Fig. 3. HyperNEAT produces ANNs from CPPNs. Weights are specified as a function of the geometric coordinates of the source node and the target node for each connection. The coordinates of these nodes and a constant bias are iteratively passed to the CPPN to determine each connection weight. If there is no hidden layer, the CPPN has only one output, which specifies the weight between the source node in the input layer and the target node in the output layer. If there is a hidden layer in the ANN, the CPPN has two output values, which specify the weights for each connection layer as shown. This figure is adapted from Gauci and Stanley [18].

2) *Encoding ANNs with CPPNs:* In the HyperNEAT algorithm, CPPNs encode ANNs instead of pictures, and evolution modifies the population of CPPNs [14], [19]. HyperNEAT evolves the weighs for ANNs with a fixed topology. The ANNs in the experiments in this paper feature a 2-D, $m \times n$ Cartesian grid of inputs and a corresponding $m \times n$ grid of outputs. If an experiment uses an ANN with hidden nodes, the hidden nodes are placed in their own 2-D layer between the input and output grids. Recurrence is disabled, so each of the $m \times n$ nodes in a layer has a link of a given weight to each of the $m \times n$ nodes in the proximate layer, excepting output nodes, which have no outgoing connections. Link weights can be zero, functionally eliminating a link.

The inputs to the CPPNs are a constant bias value and the coordinates of both a source node (e.g., $x_1 = 0, y_1 = 0$) and a target node (e.g., $x_2 = 1, y_2 = 1$) (Fig. 3). The CPPN takes these five values as inputs and produces one or two output values, depending on the ANN topology. If there is no hidden layer in the ANN, the single output is the weight of the link between a source node on the input layer and a target node on the output layer. If there is a hidden layer, the first output value determines the weight of the link between the associated input (source) node and hidden-layer (target) node, and the second output value determines the weight of the link between the associated hidden-layer (source) node and output-layer (target) node. All pairwise combinations of source and target node coordinates are iteratively passed as inputs to the CPPN to determine the weight of each ANN link. HyperNEAT can thus produce patterns in weight space similar to the patterns it produces in 2-D pictures (Fig. 2).

An additional novel aspect of HyperNEAT is that it is one of the first neuroevolutionary algorithms capable of exploiting the geometry of a problem [14], [17]–[19]. Because the connection weights between nodes are a function of the geometric positions of those nodes, if those geometric positions represent aspects of the problem that are relevant to its solution, HyperNEAT can exploit such information. For example, when playing checkers, the concept of adjacency

(on the diagonals) is important. Connection weights between neighboring squares may need to be different than weights between distant squares. HyperNEAT can create this kind of connectivity motif and repeat it across the board [18], [19]. Producing such a regularity would be more difficult with an encoding that does not include geometric information, because there would be no easy way for such an algorithm to identify which nodes are adjacent.

Variation in HyperNEAT occurs when mutations or crossover change the CPPNs. Mutations can add a node, which results in the addition of a function to the network, or change its link weights. The allowable functions for CPPNs in this paper are sine, sigmoid, Gaussian, and linear. The evolution of the population of CPPN networks in HyperNEAT occurs according to the principles of the widely used neuroevolution of augmenting topologies (NEAT) algorithm [25]. NEAT, which was originally designed to evolve ANNs, can be fruitfully applied to CPPNs because a population of CPPNs is similar in structure to a population of ANNs.

The NEAT algorithm contains three major components [25], [26]. First, it starts with small genomes that encode simple networks and slowly complexifies them via mutations that add nodes and links to the network. This complexification enables the algorithm to evolve the network topology in addition to its weights. Second, NEAT uses a fitness-sharing mechanism that preserves diversity in the population and allows new innovations time to be tuned by evolution before forcing them to compete against rivals that have had more time to mature. Finally, historical information stored in genes helps to perform crossover in a way that is effective, yet avoids the need for expensive topological analysis. A full explanation of NEAT can be found in Stanley and Miikkulainen [25].

### B. FT-NEAT, a Direct Encoding Control for HyperNEAT

A common direct encoding control for HyperNEAT is *fixed-topology NEAT* (FT-NEAT, also called perceptron NEAT or P-NEAT when it does not have hidden nodes) [14], [17], [20], [27]–[29]. FT-NEAT is similar to HyperNEAT in all ways, except it directly evolves each weight in the ANN independently instead of determining link weights via an indirect CPPN. In other words, for FT-NEAT there is no distinction between the genotype and phenotype, in contrast to HyperNEAT (where the CPPN genome network encodes a different ANN phenotype). All other elements from NEAT (e.g., its crossover and diversity preservation mechanisms) remain the same between HyperNEAT and FT-NEAT. Additionally, the number of nodes in the ANN phenotype is the same between HyperNEAT and FT-NEAT. Mutations in FT-NEAT cannot add nodes, making FT-NEAT a degenerate version of NEAT. Recall that the complexification in HyperNEAT is performed on the CPPN genome, but that the number of nodes in the resultant ANN is fixed. The end product of HyperNEAT and FT-NEAT are thus ANNs with the same number of nodes, whose weights are determined in different ways.

### C. NEAT, a Second Direct Encoding Control for HyperNEAT

While FT-NEAT is a good control for HyperNEAT because it holds the number of nodes in the ANN constant, HyperNEAT should also be compared against a cutting-edge direct encoding neuroevolution algorithm, such as regular NEAT [25]. The only difference between FT-NEAT and NEAT is that in NEAT hidden nodes and connections can be added during evolution. In those experiments in this paper where the optimal number of hidden nodes is not known *a priori*, we compare HyperNEAT to NEAT in addition to FT-NEAT.

### D. Parameter Settings

The parameters for all of the experiments below follow standard HyperNEAT and FT-NEAT conventions [14], [17], [20], [27]–[29] and can be found online at http://devolab.msu.edu/SupportDocs/Regularity. The results in this paper were found to be robust to moderate variations of these parameters.

## III. The Three Experimental Problem Domains

The first two problem domains are *diagnostic problems* that are designed to determine how the algorithms perform as regularity is varied from low to high. The first problem, called *target weights*, enables regularity to be scaled from zero to complete. However, this problem has no interactions between the different problem components. The second problem, *bit mirroring*, adds such interactions between problem components, making it a challenging problem with different types of regularity that can each be adjusted, yet the optimal solution in each case is known. The third problem, called the *quadruped controller* problem, is a challenging, real-world problem in which the regularity can be scaled and the optimal solution is not known. While we have previously reported some results in these domains [17], [20], [28], [29], this paper provides a more extensive investigation of the performance of HyperNEAT and direct encoding controls on these problems, including additional experiments, analyses, controls, and alternate versions of the problems.

### A. Target Weights Problem

One way to create a completely irregular problem is to challenge evolution to evolve an ANN phenotype ($P$) that is identical to a target neural network ($T$), where $T$ is completely irregular. Regularity can then be scaled by increasing the regularity of $T$ (Fig. 4). We call this the target weights problem because evolution is attempting to match a target vector of weights (recall that the number of nodes is constant, so the vector of weights in $T$ fully describes $T$). Fitness is a function of the difference between each weight in $P$ and the corresponding weight in $T$, summed across all weights. The lower this summed error is, the higher the fitness value. Specifically, the proximity to the target is calculated as

$$\textit{proximity to target} = \sum_{i=1}^{N} M - |P_i - T_i| \qquad (1)$$

where $N$ is the number of weights, $M$ is the maximum error possible per weight (which is 6 because weights could range from $-3$ to 3), $P_i$ is the value of the $i$th weight in the phenotype, and $T_i$ is the value of the $i$th weight in the target

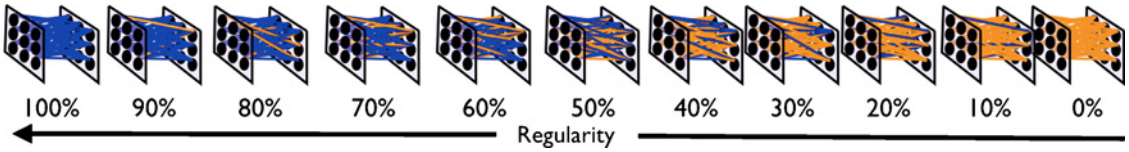


Fig. 4. Scaling regularity in the target weights problem. Regularity is scaled by changing the percentage of weights $S$ (which increases from right to left) that are set to $Q$, a single randomly-chosen value (shown as dark/blue lines). The remaining weights are each set to a random number (shown as light/orange lines).

ANN. To amplify the importance of small improvements, fitness is then calculated as

$$fitness = 2^{proximity\ to\ target}. \quad (2)$$

To scale the regularity of this problem, some randomly chosen subset of the target weight values, $S$, are assigned $Q$, a single randomly-chosen value. All remaining weight values are independently assigned a random value. Changing the number of weights in $S$ scales the regularity of the problem. When $S$ is set to 0%, all of the target weight values are chosen independently at random. When $S$ is set to 50%, half the weights have the value $Q$ and the rest have values independently chosen at random. In the most regular version of the problem, $S$ is set to 100% and all weights have the value $Q$. There are 11 treatments, with $S$ values of 0, 10, 20...100, and ten runs per treatment. Target vectors are constant for each evolutionary run, but are different between runs (due to differences in randomly-generated weight values, including $Q$). Trials last 1000 generations with a population size of 1000. The ANNs have $3 \times 3$ grids of input and output neurons. Because the number of nodes in this problem does not change, only FT-NEAT is tested as a direct encoding control.

The target weights problem is useful because it allows regularity to be scaled from zero to complete, and because the regularity of the solution is known *a priori*. It is also a simplistic problem because it has no interactions (epistasis) between weight elements (i.e., changing a given weight will not affect the optimal value of other weights). While this property makes it a good starting point for investigating regularity, other more epistatic problems are also required to understand performance in more realistic scenarios.

### B. The Bit Mirroring Problem

The bit mirroring problem is intuitively easy to understand, yet provides multiple types of regularities, each of which can be scaled independently. For each input, a target output is assigned [e.g., the input $x_1 = -1$, $y_1 = -1$ could be paired with output $x_2 = 0$, $y_2 = 1$, Fig. 5(a)]. A value of one or negative one is randomly provided to each input, and the fitness of an organism is incremented if that one or negative one is reflected in the target output. Outputs greater than zero are considered 1, and values less than or equal to zero are considered $-1$. The correct wiring is to create a positive weight between each input node and its target output and, importantly, to set to zero all weights between each input node and its non-target output nodes [Fig. 5(a)]. To reduce the effect of randomness in the inputs, in every generation each organism is evaluated on ten different sets of random inputs and these scores are summed to produce the fitness for that organism. The max fitness is thus $10n^2$, where $n$ is the number of nodes in the input sheet. Each

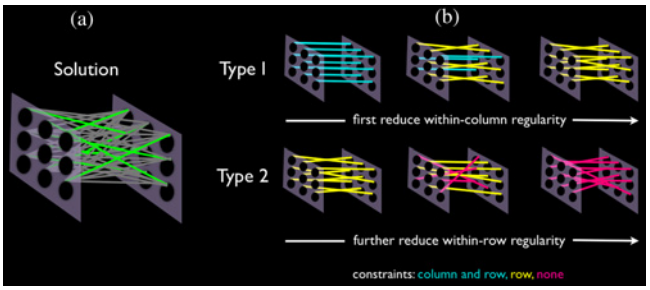


Fig. 5. Bit mirroring problem. (a) The correct wiring motif for the links projecting from each input node is to create an excitatory connection (light/green) to the correct target output node, and to turn off all other links (dark/gray). (b) Within-column regularity (Type 1) is highest when all targets are in the same column, and can be lowered by decreasing the number of targets in the same column (by assigning unconstrained targets to columns at random). For the experiments in this paper, within-column regularity is scaled while keeping within-row regularity at its highest possible level, with all targets in the same row. Within-row regularity (Type 2) is reduced by constraining fewer targets to be in the same row. By first reducing within-column regularity, then further reducing within-row regularity, the overall regularity of the bit mirroring problem can be smoothly scaled from high to low. Note that the treatment with the lowest Type 1 regularity and the highest Type 2 regularity have identical constraints.

run lasted 2000 generations and had a population size of 500. As in target weights, the number of nodes does not change on this problem (additional nodes would only hurt performance), so only FT-NEAT is tested as a control.

The first type of regularity in the problem is *within-column regularity*. This regularity is high when targets are in the same column, and low when there is no expectation about which column a target will be in [Type 1 in Fig. 5(b)]. The second type of regularity is *within-row regularity*, which is the likelihood that a target is in the same row [Type 2 in Fig. 5(b)]. While these two regularities are intuitively related, evolutionary algorithms must compute them independently, which means they are distinct. Each of these types of regularity can be scaled by constraining a certain percent of targets to be in the same column or row, and assigning the remaining targets randomly.

The third type of regularity, called *inherent regularity*, arises from the fact that, for each node, the same pattern needs to be repeated: turning one link on and all other links off. This type of regularity can be reduced by decreasing the number of nodes in the network, and hence the number of times that pattern needs to be repeated.

While the bit mirroring problem is easy to conceptualize, it is challenging for evolutionary algorithms. It requires most links to be turned off, and only a few specific links to be turned on. Moreover, links between input nodes and non-target nodes, which are likely to exist in initial random configurations and to be created by mutations, can complicate fitness landscapes. Imagine, for example, that a mutation switches the weight

on a link between an input node and its target output from zero to a positive number. The organism is now closer to the ideal wiring, but it may not receive a fitness boost if other incorrect links to that output node result in the wrong net output. The bit mirroring problem is useful, therefore, because it is challenging, yet its three main regularities are known, and can be independently adjusted.

### C. Quadruped Controller Problem

The quadruped controller problem is to evolve fast gaits for simulated four-legged robots. This problem is challenging because creating dynamic gaits for legged robots is a complicated non-linear problem that is infeasible to compute mathematically; for example, effective gaits are difficult and time-consuming for human engineers to program [30], [31]. Given how sensitive gait controllers are to slight changes in the configuration of a robot, a new gait must be created each time a robot is changed, which can lead to substantial delays in the prototyping stage of robotic development [32]. It would therefore be beneficial to automate the process of gait creation.

The problem of legged locomotion also contains many regularities; each leg is a repeated module and various gaits have different regularities, such as left-right or front-back symmetry. The regularity of this problem can also be scaled by changing the number of legs that are slightly different, as can happen due to inconsistencies in manufacturing processes. The quadruped controller problem is thus a challenging real-world problem that has regularities that can be varied. It will help validate whether conclusions drawn from target weights and bit mirroring generalize to more challenging real-world problems.

Before describing the specific implementation of the quadruped controller problem in this paper, we will review previous work in this area. Many researchers have successfully evolved controllers for legged robots, typically by evolving neural network controllers [32]–[39]. Evolved gaits are often better than those produced by human designers; one was even included on the commercial release of Sony's AIBO robotic dog [32], [39]. However, many researchers have found that evolutionary algorithms cannot handle the entire problem because the number of parameters that need to be simultaneously tuned to achieve success is large [32], [38]–[43]. Many of these scientists report that while it is possible to evolve a controller to manage the inputs and outputs for a single leg, once evolution is challenged with the inputs and outputs of many legs, it fails to make progress.

One solution that has worked repeatedly is to help the evolutionary algorithm "see" that there are regularities and symmetries to the problem. This approach involves manually decomposing the problem by, for example, evolving a controller for one leg and then copying that controller to every other leg, with some variation in phase. Unfortunately, this tactic imposes a specific type of regularity on the network instead of allowing evolution to produce different regularities. It would be better to completely automate the process and thereby remove the need for human engineers to spend time decomposing the problem. Furthermore, such manual decomposition potentially introduces constraints and biases that could preclude the attainment of better solutions [44]. Finally, if we can employ algorithms that can automatically discover and exploit the regularities in a problem, such algorithms may be able to do the same for complex problems with regularities humans are not aware of. Indirect encodings should be well-suited to this task of automatically discovering regularities, so it is worthwhile to study their capabilities on this front.

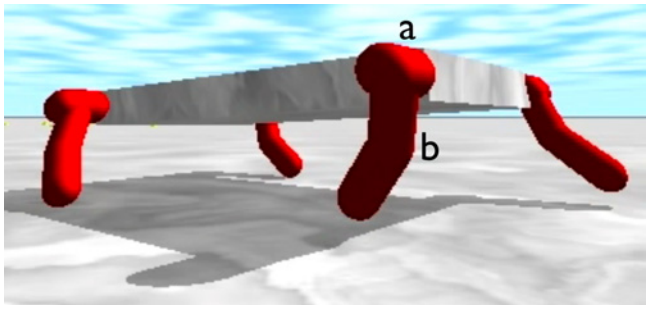


Fig. 6. The simulated robot in the quadruped controller problem. (a) Two joints (HipFB and HipIO, see the text) approximate a universal hip joint that can move in any direction. (b) The knee joint can swing forward and backward.

While it has not been the norm, indirect encodings have occasionally been used to evolve the gaits of legged creatures. In at least two cases, an indirect encoding evolved both the gaits and the morphologies of creatures [11], [37]. In both cases, the morphologies and behaviors were regular. While the regularity of these ANNs was not systematically studied, one of these papers contained an anecdotal report of a regular ANN [11]. However, the regularity of the problem was not scaled in either of these works. In another study, an indirect encoding and a direct encoding were compared for their ability to evolve a gait for a legged creature in an attempt to determine whether the indirect encoding can exploit the regularity of the domain without the problem being simplified or manually decomposed [34]. However, this project used a simple model of a six-legged insect that had only two degrees of freedom per leg. Nevertheless, the work showed that the indirect encoding could automatically discover the regularity of the problem and decompose it by encoding a neural submodule once and expressing it repeatedly. The indirect encoding also outperformed a direct encoding by solving the problem faster. Unfortunately, computational limits at the time meant that such results were anecdotal and not statistically significant because so few trials could be performed.

The quadruped controller problem is a challenging engineering problem with scalable regularity. Investigating the performance of HyperNEAT and direct encoding controls will therefore illuminate how an indirect encoding handles problem regularities differently than direct encodings. We next describe the implementation details of the specific quadruped controller problem in this paper.

The robots (Fig. 6) are evaluated in the open dynamics engine (ODE) physics simulator (www.ode.org). The rectangular torso of the organism is (in arbitrary ODE units) 0.15 wide, 0.3 long, and 0.05 tall. For a point of reference, the right side of the robot from the viewer's perspective in Fig. 6 is designated as the robot's front. Each of four legs is composed of three cylinders (length 0.075, radius 0.02) and three hinge joints. The first cylinder functions as a hip bone. It is parallel to

the proximal-distal axis of the torso and barely sticks out from it. The second cylinder is the upper leg and the last cylinder is the lower leg. There are two hip joints and one knee joint. The first hip joint (HipFB) allows the legs to swing forward and backward (anterior-posterior) and is constrained to 180° such that at maximum extension it is parallel with the torso. The second hip joint (HipIO) allows the leg to swing in and out (proximal-distal). Together, the two hip joints approximate a universal joint. The knee joint swings forward and backward. The HipIO and knee joints are unconstrained.

Each quadruped is simulated for 6 seconds (6000 time steps). Trials are cut short if any part of the robot save its lower leg touches the ground or if the number of direction changes in joints exceeds 960. The latter condition is an attempt to roughly reflect that servo motors cannot be vibrated incessantly without breaking. The fitness of a controller is

$$fitness = 2^{d^2} \tag{3}$$

where $d$ is the maximum distance traveled during the allotted time. An exponential fitness function is used so that even small increases in the distance traveled result in a sizable selective advantage.

For HyperNEAT and FT-NEAT, the ANN configuration on this problem features three 2-D, $5 \times 4$ Cartesian grids forming input, hidden, and output layers (Fig. 7). The NEAT control has the same number of inputs and outputs, but the number of hidden nodes can evolve. There are no recurrent connections. All possible connections between adjacent layers exist (although weights can be zero, functionally eliminating the link). There are thus 800 links in the ANN of each individual for HyperNEAT and FT-NEAT. The number of links in NEAT can evolve. Link weights are in the range of $[-3, 3]$.

The inputs to the ANN are the current angles (from $-\pi$ to $\pi$) of each of the 12 joints of the robot, a touch sensor that provides a 1 if the lower leg is touching the ground and a 0 if it is not, the pitch, roll and yaw of the torso, and a modified sine wave (which facilitates periodic behavior). The sine wave function is

$$sin(t/120)\pi \tag{4}$$

where $t$ is the number of milliseconds that have passed since the start of the experiment. Multiplying by $\pi$ produces numbers between $-\pi$ and $\pi$, which is the range of the unconstrained joints. The constant 120 was chosen because it was experimentally found to produce fast yet natural gaits. While changing this constant can affect the types of gaits produced, doing so never altered any of the qualitative conclusions of this paper. Preliminary tests determined that the touch, pitch, roll, yaw, and sine inputs all improved the ability to evolve fit gaits.

The outputs of the ANNs are the desired joint angles for each joint, which are fed into a PID controller that simulates a servo. The controller subtracts the current joint angle from the desired joint angle. This difference is then multiplied by a constant force (2.0), and a force of that magnitude is applied to the joint such that the joint moves toward the desired angle. Such PID-based control systems have been shown to be effective in robot control [32], [36], [45].

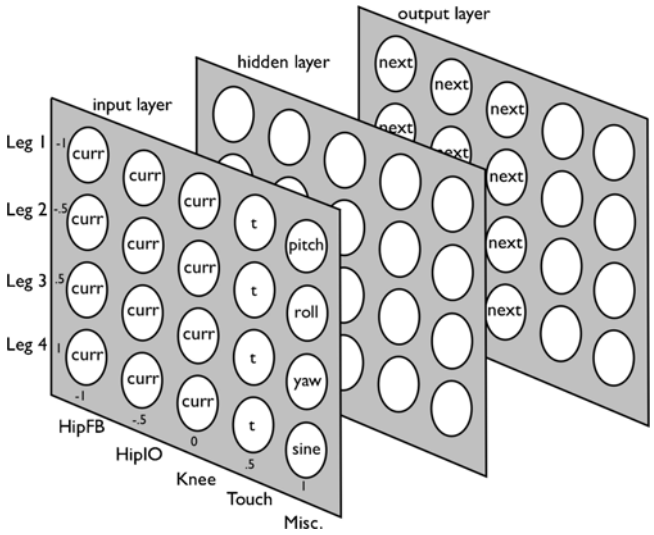


Fig. 7. ANN configuration for HyperNEAT and FT-NEAT treatments. The first four columns of each row of the input layer receive information about a single leg (the current angle of each of its three joints, and a 1 or 0 depending on whether the lower leg is touching the ground). The final column provides the pitch, roll, and yaw of the torso as well as a sine wave. Evolution determines how to use the hidden-layer nodes. The nodes in the first three columns of each of the rows in the output layer specify the desired new joint angle. The joints move toward that desired angle in the next time step as described in the text. The outputs of the nodes in the rightmost two columns of the output layer are ignored.

Regularity in the quadruped controller problem can be scaled by changing the number of faulty joints. A faulty joint is one in which, if an angle $A$ is requested, the actual desired angle sent to the PID controller is $A + E$, where $E$ is an error value in degrees within the range $[-2.5, 2.5]$. The value of $E$ is chosen from a uniform random distribution in this range for each faulty joint at the beginning of a run, and is constant throughout the run. Such errors are analogous to the inconsistencies of robotic joints produced by manufacturing processes. The more faulty joints, the less regularity there is in the problem because fewer legs behave identically. The default version of this problem [20] is the most regular version with zero faulty joints, which is the version referred to throughout the rest of the paper unless the regularity is specified. Each algorithm was run 50 times for experiments with 0, 1, 4, 8, and 12 faulty joints. Runs lasted 1000 generations and had a population size of 150.

## IV. Results

The following sections describe experiments and analyses that investigate how the HyperNEAT indirect encoding compares to direct encodings with respect to exploiting problem regularities and producing phenotypic regularities.

### *A. HyperNEAT Outcompetes Direct Encodings as Problem Regularity Increases*

1) *Target Weights:* The results from the target weights experiments (Fig. 8) reveal that HyperNEAT performs better as the regularity of the problem increases, especially in early generations, where mean performance perfectly correlates with

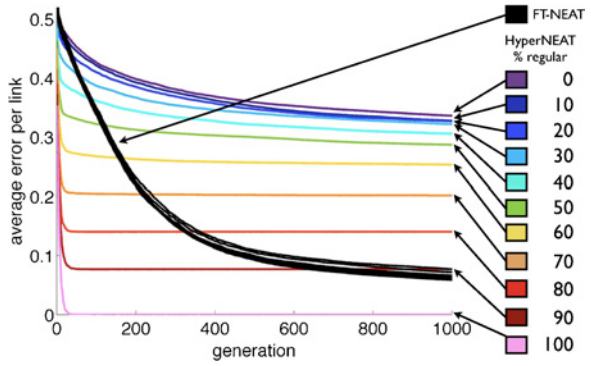


Fig. 8. Mean performance of HyperNEAT and FT-NEAT on a range of problem regularities for the target weights problem. HyperNEAT lines are colored for each regularity level, and in early generations are perfectly ordered according to the regularity of the problem (i.e., regular treatments have less error). The performance of FT-NEAT (black lines) is unaffected by the regularity of the problem, which is why the lines are overlaid and mostly indistinguishable.

problem regularity. Interestingly, after 1000 generations of evolution, the performance of HyperNEAT is statistically indistinguishable below a certain regularity threshold ($p > 0.05$[1] comparing the final performance of the $S = 0\%$ treatment to treatments with $S \leq 30\%$). Above that regularity threshold, however, HyperNEAT performed significantly better at each increased level of regularity ($p < 0.01$ comparing treatment with values of $S > 30\%$ to the treatment with a value of $S$ 10% higher). FT-NEAT, on the other hand, is blind to the regularity of the problem: the results from different treatments are visually and statistically indistinguishable ($p > 0.05$). Early on HyperNEAT outperformed FT-NEAT on regular versions of the problem ($p < 0.01$ comparing treatments of $S \geq 60\%$ at generation 100), but at 1000 generations FT-NEAT outperforms HyperNEAT on all but the two most regular versions of the problem ($p < 0.001$). HyperNEAT outperforms FT-NEAT on the most regular version of the problem at generation 1000 ($p < 0.001$), and the algorithms are statistically indistinguishable on the $S = 90\%$ treatment ($p > 0.05$).

Overall, this experiment provides evidence for several interesting observations. While this evidence comes from only the target weights problem, which is a simple diagnostic problem, we highlight them here because they will also be supported by data from the bit mirroring and quadruped controller problems. They are given as follows.

1) FT-NEAT outcompetes HyperNEAT when problem regularity is low.
2) As problem regularity increases, HyperNEAT's performance rises to, and then surpasses, that of FT-NEAT, demonstrating that HyperNEAT can exploit problem regularity.
3) FT-NEAT is blind to problem regularity.
4) HyperNEAT exploits problem regularity only above a certain regularity threshold.

A final result of interest from this experiment is the lack of progress HyperNEAT makes after the early generations on the problems that are mostly regular, but have some irregularity (e.g., $S = 90\%$). HyperNEAT is easily able to produce weights similar to the repeated weight $Q$, and thus exploits the regularity of the problem, but over hundreds of generations HyperNEAT did not discover how to make exceptions to the pattern to produce the irregular link values. This evidence suggests HyperNEAT is biased toward producing regular solutions and has difficulty producing certain irregular patterns, a subject we will revisit later in the paper.

2) *Bit Mirroring:* The first two bit mirroring experiments have $7 \times 7$ grids of input and output nodes. In the first experiment, both within-column and within-row regularity start at 100%, and within-column regularity decreases per treatment by constraining fewer targets to be in the same column as their respective inputs [Type I in Fig. 5(b)]. The most irregular treatment in this experiment features zero within-column regularity, but has 100% within-row regularity. The second experiment picks up where the first left off by lowering within-row regularity per treatment [Type II in Fig. 5(b)]. The least regular treatment in experiment two has no within-column or within-row regularity. For each treatment, ten runs of evolution are performed.

The results from the bit mirroring experiment support many of the conclusions from the target weights experiment. Initially, FT-NEAT is blind to both within-column and within-row regularity (Fig. 9, right two columns, $p > 0.05$ comparing all within-column and within-row treatments to the treatments with no column or row constraints, respectively). The performance of HyperNEAT, however, increases with the regularity of the problem (Fig. 9, left two columns). HyperNEAT perfectly solves the problem in all but two runs on the most regular version of the problem, where targets are in the same column and row. Once again, the performance of HyperNEAT within a type of regularity does not increase until that type of regularity is above a threshold. For both within-column and within-row regularity, HyperNEAT's performance advantage is statistically significant only once that type of regularity is above 50% (only treatments with more than 50% of targets column-constrained or row-constrained statistically outperform treatments with 0% of targets column-constrained or row-constrained, respectively: $p < 0.05$).

It is also interesting that the magnitude of the *range* of fitness values is correlated with the regularity of the problem for HyperNEAT. This phenomenon might occur because, when regularity is present, the indirect representation either discovers and exploits it, which would result in high fitness values, or it fails to fully discover the regularity, at which point its fitness more closely resembles lower-performing, less-regular treatments.

HyperNEAT outperforms FT-NEAT in all versions of this problem ($p < 0.05$). This result is likely due to the inherent regularity of the problem, which arises because the same pattern (turning one link on and the rest off) must be repeated for each of the 49 input nodes. Inherent regularity should decrease with the number of nodes in the network, a hypothesis we test in experiment three, where experiments are performed on grid sizes from $8 \times 8$ down to $3 \times 3$ (Fig. 10). For all grid sizes, both within-column and within-row regularity levels are

[1]This $p$ value and all others in this paper were generated with MATLAB's non-parametric Mann-Whitney $U$-test, unless otherwise specified.

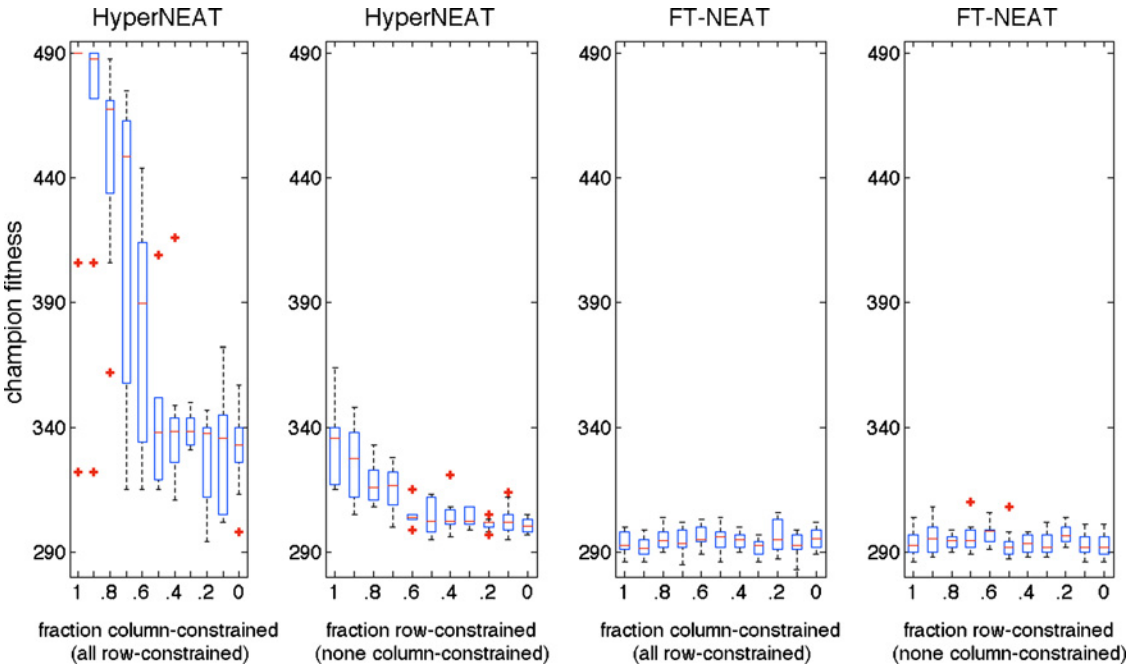


Fig. 9. HyperNEAT and FT-NEAT on versions of the bit mirroring problem with different levels of regularity. For each treatment, from left to right, within-column regularity is first decreased (left panel) and then within-row regularity is further decreased (right panel). The data plotted are collected from populations at the end of evolutionary runs, but plots over time (not shown) reveal the same qualitative story: HyperNEAT's performance is consistently higher on more regular problems and the performance of FT-NEAT is unaffected by the regularity of the problem. For HyperNEAT, the performance gaps between more and less regular treatments are evident early and increase across evolutionary time.

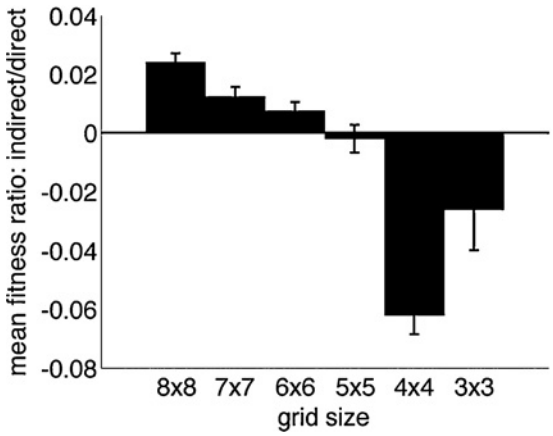


Fig. 10. HyperNEAT versus FT-NEAT as the inherent regularity of the bit mirroring problem is decreased. Reducing the grid size reduces the amount of inherent regularity in the problem. Error bars show one standard error of the mean. Ratios are used instead of absolute differences because the allowable fitness ranges change with grid size.

0%, leaving only the inherent regularity of the problem. Due to the high variance between runs, 40 runs are conducted per treatment.

As the grid size is lowered, the relative performance of HyperNEAT degrades to and then falls below that of FT-NEAT. The general trend is significant ($p < 0.05$ comparing the ratios on the $3 \times 3$ problem versus those with $6 \times 6$ and larger grids). It is not clear why HyperNEAT performed relatively better on the $3 \times 3$ grid than the $4 \times 4$ grid. This experiment reinforces the conclusion that once problems become irregular enough, FT-NEAT can outperform HyperNEAT. It also provides a further demonstration that HyperNEAT can exploit increasing problem regularity to gain a relative edge over FT-NEAT.

3) *Quadruped Controller:* The data from the quadruped controller problem also generally support the conclusions from target weights and bit mirroring. HyperNEAT outperforms both FT-NEAT and NEAT on the two most regular versions of this problem, where there are 0 or 1 faulty joints (Fig. 11, $p < 0.001$). That HyperNEAT outperforms NEAT is noteworthy, given that NEAT is one of the most successful direct encoding neuroevolution algorithms.

As with target weights and bit mirroring, HyperNEAT's performance increases with the regularity of the problem, but only above a certain threshold: the 0 faulty joint treatment significantly outperforms the 1 faulty joint treatment ($p < 0.001$) which, in turn, outperforms the 4 faulty joint treatment ($p < 0.001$) which, in turn, outperforms the 8 faulty joint treatment ($p < 0.001$). However, the 8 and the 12 faulty joint treatments are statistically indistinguishable ($p > 0.05$). In contrast to target weights and bit mirroring, FT-NEAT is not blind to the regularity of this problem, although it is less sensitive to the regularity than HyperNEAT. The treatment with 0 faulty joints is statistically indistinguishable from the 1 faulty joint treatment ($p > 0.05$), but performance on both of these treatments is higher than on the 4 faulty joint treatment ($p < 0.001$) which is, in turn, higher than the 8 faulty joint treatment. As is the case for HyperNEAT, performances for FT-NEAT on the treatments with 8 and 12 faulty joints are statistically indistinguishable ($p > 0.05$). The statistical comparisons for NEAT are the same as those for FT-NEAT.

One reason that regularity may affect the direct encodings on this problem is because weights tend to be near

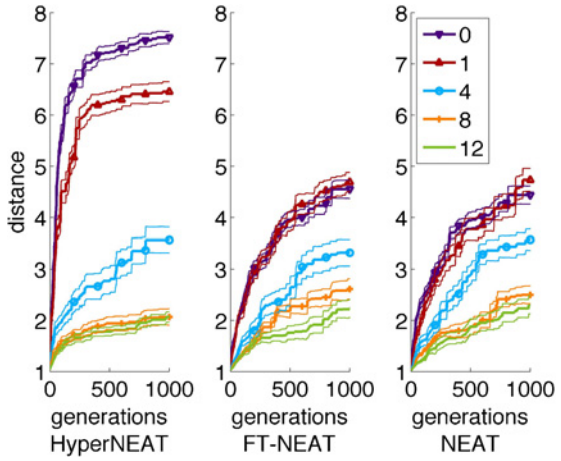


Fig. 11. Performance of HyperNEAT, FT-NEAT, and NEAT on the quadruped controller problem with 0, 1, 4, 8, and 12 faulty joints.

the maximum or minimum allowable value, which is partly because mutations that create links outside of this range are set to the maximum or minimum value. This method makes extreme values more likely, which can facilitate coordination in the joints because different joints are controlled by links with similar weights. If normal joints are controlled by links with the maximum or minimum link value, to have faulty joints behave the same as normal joints, link values would either have to be outside the allowable range, or inside the range at non-extreme (and thus harder to set) values. Faulty joints thus increase the difficulty of the problem for both indirect and direct encodings, and can help explain why the problem regularity appears to benefit the direct encodings. Exploring ways to reduce this bias is an interesting avenue for future work.

As with bit mirroring and target weights, FT-NEAT is able to outperform HyperNEAT on the quadruped controller problem once the regularity of the problem is sufficiently low. FT-NEAT and NEAT outperformed HyperNEAT on both the 8 and 12 faulty joint treatments. On the treatment with 8 faulty joints, the difference is significant for FT-NEAT ($p < 0.05$), but not for NEAT. On the treatment with 12 faulty joints, the difference for FT-NEAT is almost significant ($p = 0.066$), but the difference for NEAT is highly significant ($p < 0.01$).

The difference in fitness in the first generation of randomly-generated organisms is interesting. HyperNEAT begins with an advantage over FT-NEAT because even randomly-generated CPPNs are sometimes able to produce the coordination of legs that facilitates movement. Some of these randomly-generated organisms in HyperNEAT display impressive coordination and appear to be on the road toward rudimentary locomotion. Randomly-generated FT-NEAT and NEAT organisms do not provide this impression.

Overall, the results from the target weights, bit mirroring, and quadruped controller problems show that the direct encodings outperform HyperNEAT when problem regularity is low. They also show that as problem regularity increases, HyperNEAT can exploit that regularity whereas the direct encodings mostly do not. This ability to exploit problem regularity means that HyperNEAT increasingly outperforms direct encoding controls as problem regularity increases. We now investigate further how HyperNEAT is able to exploit regularity.

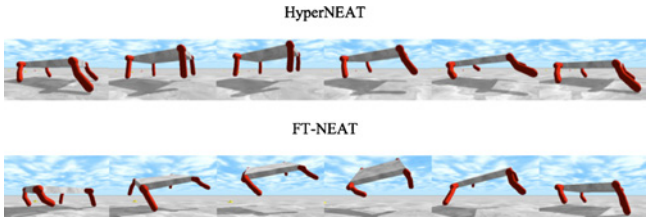


Fig. 12. A time series of images from typical gaits produced by HyperNEAT and FT-NEAT. HyperNEAT robots typically coordinate all of their legs, whether all legs are in phase (as with this robot) or with one leg in anti-phase. A short sequence involving a bound or gallop is repeated over and over in a stable, natural gait. FT-NEAT robots display far less coordination among legs, are less stable, and do not typically repeat the same basic motion. NEAT gaits are qualitatively similar to FT-NEAT gaits.

### B. HyperNEAT Produces More Regular Behaviors

We focus our analysis of regularity in ANNs and behaviors on the quadruped controller problem because target weights and bit mirroring are diagnostic problems wherein regularity is explicitly built into the problem (i.e., any phenotypic regularity in fit solutions is unsurprising). Moreover, there is no meaningful behavior associated with the two diagnostic problems. The quadruped controller problem, on the other hand, does have interesting behaviors with different levels of regularity. Moreover, the problem does not explicitly require or reward regularity, which means that any regularities that develop do so because of the encoding and because such regularities happen to produce fast gaits.

The first method we employ to analyze the behaviors produced by the different algorithms is based on videos of the highest performing gaits from all 50 runs in the treatment with 0 faulty joints (available at http://devolab.msu.edu/SupportDocs/Regularity). The HyperNEAT gaits are all regular. They feature two separate types of regularity: coordination between legs, and repetition of the same movement pattern across time. Generally, the gaits are one of two types. The first type has four-way symmetry, wherein each leg moves in unison and the creature bounds forward repeatedly (Fig. 12, top row). This gait implies that HyperNEAT is reusing neural information in a regular way to control all of the robot's legs. The second gait resembles a horse gallop and features the back three legs moving in unison, with the fourth leg moving in opposite phase. This *3-1 gait* demonstrates that HyperNEAT can reuse neural information with some variation, because the same behavioral pattern exists in each leg, but is inverted in one leg. The ability to produce repetition with variation is a desirable feature in genetic encodings [2].

Overall, the HyperNEAT gaits resemble those of running natural organisms because they are coordinated and graceful. These observations are noteworthy because they indicate that HyperNEAT is automatically exploiting the regularities of a challenging, real-world problem. This accomplishment is significant given that researchers have previously needed to

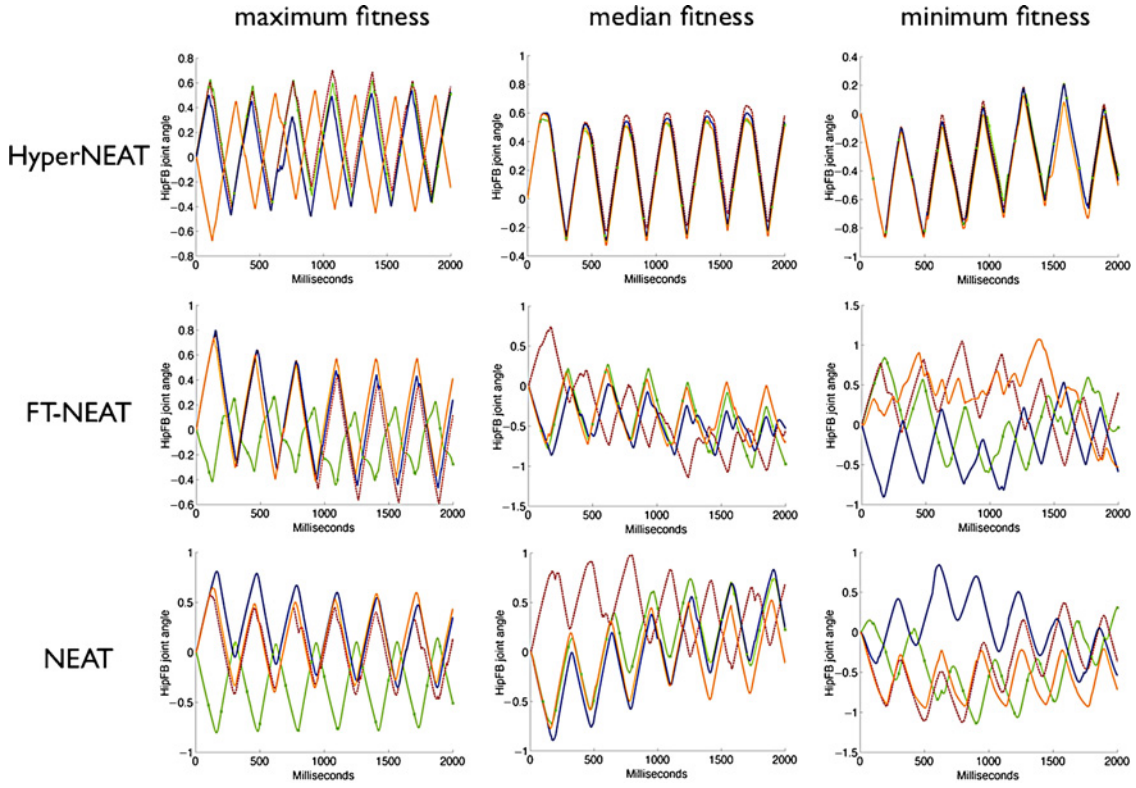


Fig. 13. HipFB joint angles observed in robots evolved with HyperNEAT, FT-NEAT, and NEAT. The possible range for this joint is $-0.5\pi$ to $0.5\pi$. The $y$-axis shows radians from the initial down (0) position. For clarity, only the first 2 s are depicted. For HyperNEAT, the best gait is an example of the 3-1 gait, where three legs are in phase and one leg is in opposite phase, which resembles the four-beat gallop gait. The other two HyperNEAT gaits are four-way symmetric, with all legs coordinated in a bounding motion (Fig. 12). The best direct encoding gaits are mostly regular. However, the median and worst gaits, which are representative of most direct encoding gaits, are irregular: while some legs are synchronized, other legs prevent the coordinated repetition of a pattern.

manually decompose legged locomotion tasks for evolutionary algorithms to perform well [32], [38]–[43].

The gaits of FT-NEAT and NEAT, on the other hand, are mostly uncoordinated and erratic, with legs often appearing to operate independently of one another (Fig. 12, bottom row). A few of the best-performing gaits do exhibit coordination between the legs, and the repetition of a basic movement pattern, but most of the gaits are irregular. Even the regular gaits are not as natural and impressive as the HyperNEAT gaits, which is reflected in their lower objective fitness values. For most gaits, some legs flail about, others trip the organism, and some work against each other by pushing in opposite directions. The robots frequently cartwheel and trip in unstable positions until they finally fall over. There is much less repetition of a basic movement pattern across time. Coordination between legs is often rare and temporary.

The few examples of regular gaits produced by FT-NEAT and NEAT show that it is sometimes possible for direct encodings to produce regularities. Overall, however, HyperNEAT is much more consistent at producing regular gaits. All of the HyperNEAT gaits are regular, whereas only a few FT-NEAT and NEAT gaits are. It is important for algorithms to be consistent, especially when computational costs are high, so that high-quality results can be obtained without performing many runs. A test of the reliability of each encoding is to watch the median- and least-fit gaits of the 50 champions for each encoding: for HyperNEAT these gaits are coordinated and effective, whereas for FT-NEAT and NEAT they are discombobulated.

In general, the gaits reveal a greater gap in performance between HyperNEAT and the direct encodings than is suggested by the fitness scores, especially for all but the best runs for each algorithm. Most of the direct encoding gaits do not resemble stable solutions to quadruped locomotion, whereas HyperNEAT produces a natural gait in all trials with a small variety of different solutions.

A second method for investigating how HyperNEAT is able to outperform the direct encodings is to look at the angles of the leg joints during locomotion. This technique is a different way of estimating the coordination, or lack thereof, of the different legs for each encoding. Plots of each leg's HipFB joint from the best, median, and worst runs for each algorithm corroborate the descriptive evidence (Fig. 13). The legs in all HyperNEAT organisms exhibit a high degree of both of the two main regularities: at any point in time most legs are in similar positions (except the out-of-phase leg in the 3-1 gait, which is opposite), and a basic movement pattern is repeated across time. The direct encoding gaits are less regular in both ways, except for the highest-performing gaits. The median and worst gaits are representative of most of the direct encoding gaits: there is little coordination between legs or across time (Fig. 13). While only the HipFB joint is shown, plots of the other two joints are consistent with these results.

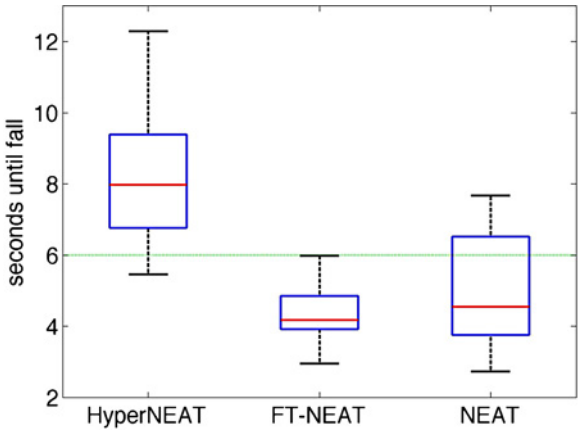


Fig. 14. Gait generalization. HyperNEAT gaits generalize better than FT-NEAT and NEAT gaits, which means that they run for longer before falling over. The allotted time during evolution experiments is 6 s (dashed horizontal line). Only the HyperNEAT gaits exceed that amount of time in the generalization tests. For clarity, outliers are not shown.

### C. HyperNEAT Behaviors are More General

One of the benefits of regularity is generalization. On the quadruped controller problem, for example, repeating the same basic pattern of motion is a type of regularity that is likely to generalize, because its success in one cycle makes it probable that it will be successful in the next cycle. A non-repeating sequence of moves, however, may be less likely to generalize because its past is less likely to predict its future. Generality, then, can be a test of regularity. It is also a desirable property in its own right.

During the evolution experiment, the robots are evaluated for six simulated seconds. One test of generality is to remove the time limit of 6 s and measure how long the evolved robots are able to move before they fall. Fig. 14 reports that HyperNEAT champion gaits are significantly more general than FT-NEAT and NEAT champion gaits ($p < 0.001$). HyperNEAT gaits are the only ones on average that keep moving beyond the number of seconds simulated during evolution.

### D. HyperNEAT Regularities can be Influenced, Allowing Domain Knowledge to be Injected

HyperNEAT genomes create regularities in geometric space that affect phenotypic elements based on the geometric coordinates of those elements. Changing the geometric arrangement of these elements may make it easier for HyperNEAT to produce one type of behavior versus another [17]. For example, it might be easier to group two elements together if those elements are close to each other, whereas this grouping may be more difficult if the elements are far away from each other, especially if elements that should not be included in the group lie in-between. If this hypothesis is correct, arranging phenotypic elements such as sensors or outputs in different geometric configurations may be a way for the experimenter to influence the types of regularities HyperNEAT produces.

The ordering of the legs in HyperNEAT for the quadruped controller problem offers a way to test this hypothesis. In the default configuration (Fig. 7) the legs are ordered, from lowest to highest $Y$ coordinate value, FL-BL-BR-FR, where F = front, B = back, L = left, and R = right. This ordering may make it easier to group the left legs into one group and the right legs into another, since they are closer to each other. To test this hypothesis, we performed experiments with the following alternate orderings: FL-FR-BL-BR and FL-FR-BR-BL, which may encourage front-back symmetry, and FL-BR-FR-BL, which may encourage diagonal symmetry. For each of these four orderings, we performed 50 runs, each with a population size of 150 that lasted 1000 generations. Table I reports the classifications of the highest performing gait at the end of each run.

TABLE I
RESULTANT GAIT TYPES FOR DIFFERENT LEG ORDERINGS

| | 4way Sym | L-R Sym | F-B Sym | One Leg Out of Phase | | | |
|---|---|---|---|---|---|---|---|
| | | | | FL | BL | BR | FR |
| FL-BL-BR-FR (default) | 36 | 4 | | | | | 9 |
| FL-BR-FR-BL | 47 | | | | 2 | | 1 |
| FL-FR-BL-BR | 44 | | 3 | | | 1 | |
| FL-FR-BR-BL | 36 | | 4 | | 9 | | 1 |

Gaits are placed into the following categories: 4way Sym(metry) (all legs in synchrony), L-R Sym (the left legs are in phase and the right legs out of phase), F-B Sym (the front legs are in phase and the back legs are out of phase), and one leg out of phase (three legs moved in synchrony and one is out of phase, which resembles a gallop). If two legs are motionless, they are considered in synchrony. Two gaits do not fit into these categories and are not tabulated. FL: front left, BL: back left, BR: back right, and FR: front right.

The most common gait in all runs exhibits four-way symmetry, which is not expected to be biased by leg ordering. The other gaits, however, do tend to reflect the geometric ordering of the legs in each treatment. For example, all four examples of left-right symmetry evolved in the *L*L*R*R treatment (where * stands for any symbol), and all seven cases of front-back symmetry evolved in the treatments that ordered the legs F*F*B*B*. It appears it is easier for HyperNEAT to bisect the $Y$ dimension once to group neighboring legs, instead of creating the more complex pattern required to group legs with non-adjacent $Y$ coordinate values.

It is interesting to observe which is the exception leg in the gaits that had three legs in synchrony and one leg in opposite phase. In 23 out of 25 cases, the exception leg is the one with the highest $Y$ coordinate value, although which leg that is changes based on the geometric ordering (Table I). Different geometric representations, therefore, can probabilistically bias evolution to make different legs be the exception leg, which is an example of how a HyperNEAT user can inject a preference into the algorithm. It is not clear why exceptions are typically made for the leg with the highest $Y$ coordinate value. This result may be due to the nature of the mathematical functions in the CPPNs.

### E. HyperNEAT ANNs are More Regular, Which is Visually Apparent

Beyond behavioral regularities, it is also interesting to examine the ANNs produced by HyperNEAT and FT-NEAT. NEAT ANNs are not visualized because their variable number of hidden-node layers make such visualization difficult.

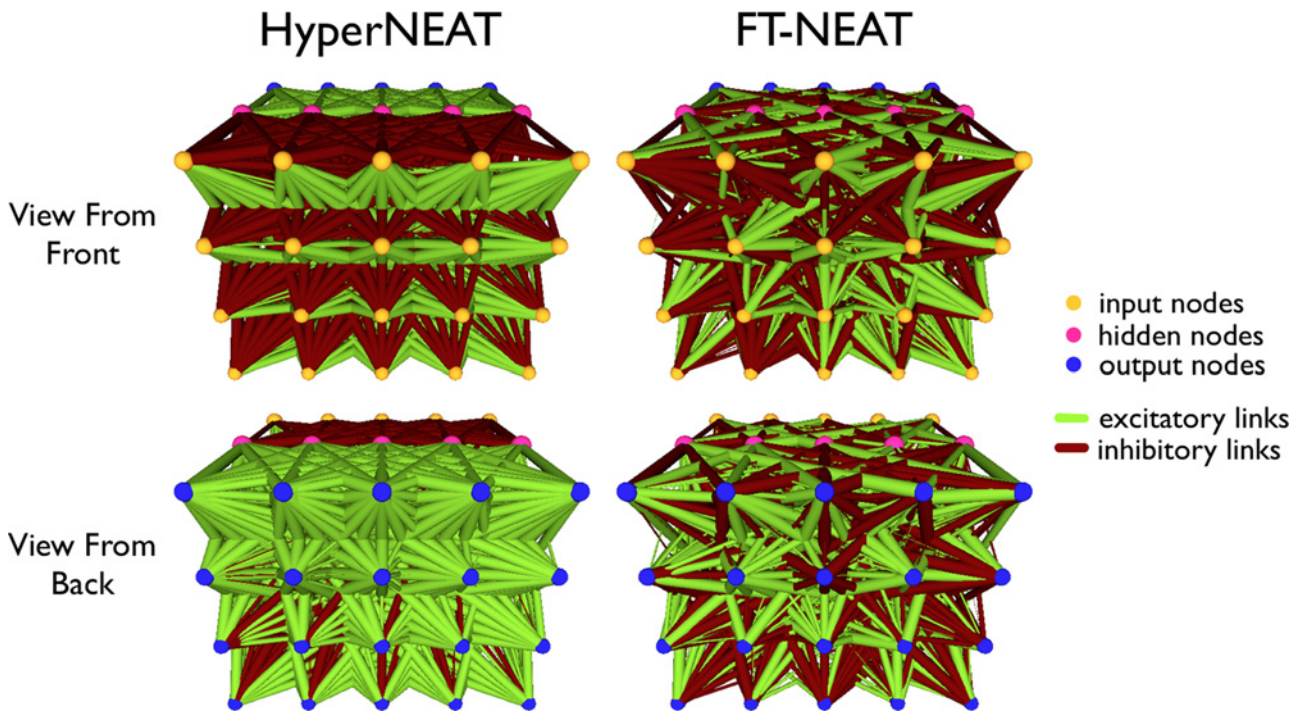


Fig. 15. Example ANNs produced by HyperNEAT and FT-NEAT. Views from the front (looking at the inputs) are shown in the top row, and from the back (looking at the outputs) are shown in the bottom row. The thickness of the link represents the magnitude of its weight.

Visualizations reveal that all HyperNEAT ANNs are regular while all FT-NEAT ANNs are irregular. Fig. 15 shows example ANNs produced by HyperNEAT and FT-NEAT (all 50 visualizations for HyperNEAT and FT-NEAT ANNs can be viewed at http://devolab.msu.edu/SupportDocs/Regularity). The FT-NEAT ANN has no obvious regularity, and is hard to visually distinguish from an ANN with randomly-generated weights. Such ANN irregularity occurs even when the behavior produced by the ANN is somewhat regular (e.g., despite producing the relatively regular gait seen in the left-most column of Fig. 13, the FT-NEAT ANN in Fig. 15 is irregular).

Because nodes do not have defined geometric coordinates in FT-NEAT, it would be equally appropriate to visualize its ANNs with any rearrangement of nodes within a layer. In this paper, all FT-NEAT nodes are plotted in the same arbitrary order. While different orderings may affect the visual appearance of regularity, the experimental evidence suggests that it is unlikely that reordering the nodes will reveal substantial regularity in the FT-NEAT ANNs.

The HyperNEAT ANN in Fig. 15, on the other hand, displays multiple different regularities. Initially, looking from the front, the links emanating from each input node repeat a pattern with inhibitory (dark red) links on top and excitatory (light green) links below. Additionally, a clear distinction is made between the different layers of connections, at least on the top of the ANN, where the input-to-hidden layer is inhibitory and the hidden-to-output layer is excitatory. An additional regularity is observable (viewing from the back) in links projecting from the outputs, where the nodes toward the bottom-left corner have a pattern of a few inhibitory links surrounded by excitatory links. Interestingly, this pattern is repeated in nearby nodes, but the strength of the pattern decreases roughly with the distance from the bottom-left corner, such that eventually there are few or no remaining inhibitory links. The decrease is faster in the $Y$ (height) dimension, and slower in the $X$ (width) dimension. This pattern shows that HyperNEAT can produce complex geometric regularities.

A diverse set of regularities are observable in the 50 HyperNEAT champion ANNs. Some of this diversity is shown in Figs. 16 and 17. Left-right, top-bottom, and diagonal symmetries are produced. Additionally, exceptions (different patterns) are made for single columns, rows, and individual nodes. Some networks appear completely regular, with every viewable link having a similar value, and others feature complicated, yet still regular, patterns. Some networks have predominantly large weights (e.g., the top-left ANN in Fig. 16), others have mainly small weights (e.g., the top-center ANN in Fig. 16), and some have a broad range of weights. Importantly, many of these regularities include some variation. For example, the top-left ANN in Fig. 16 has a motif that is repeated for each node, but that motif varies in regular ways (e.g., the number and strength of excitatory links emanating from nodes in the second column increases from bottom to top). Other variation is less regular, such as the exception made for a single node (Fig. 16, bottom left), but even in this case the node itself contains a regular pattern.

This diversity of regularities demonstrates that HyperNEAT can explore many different types of regularities to solve a problem. It is also interesting that, while HyperNEAT is able to find many different regular solutions that perform and behave similarly, FT-NEAT is unable to consistently discover *any* of these regular patterns. In other words, there are many solutions in the search space, but the regularity of those solutions means

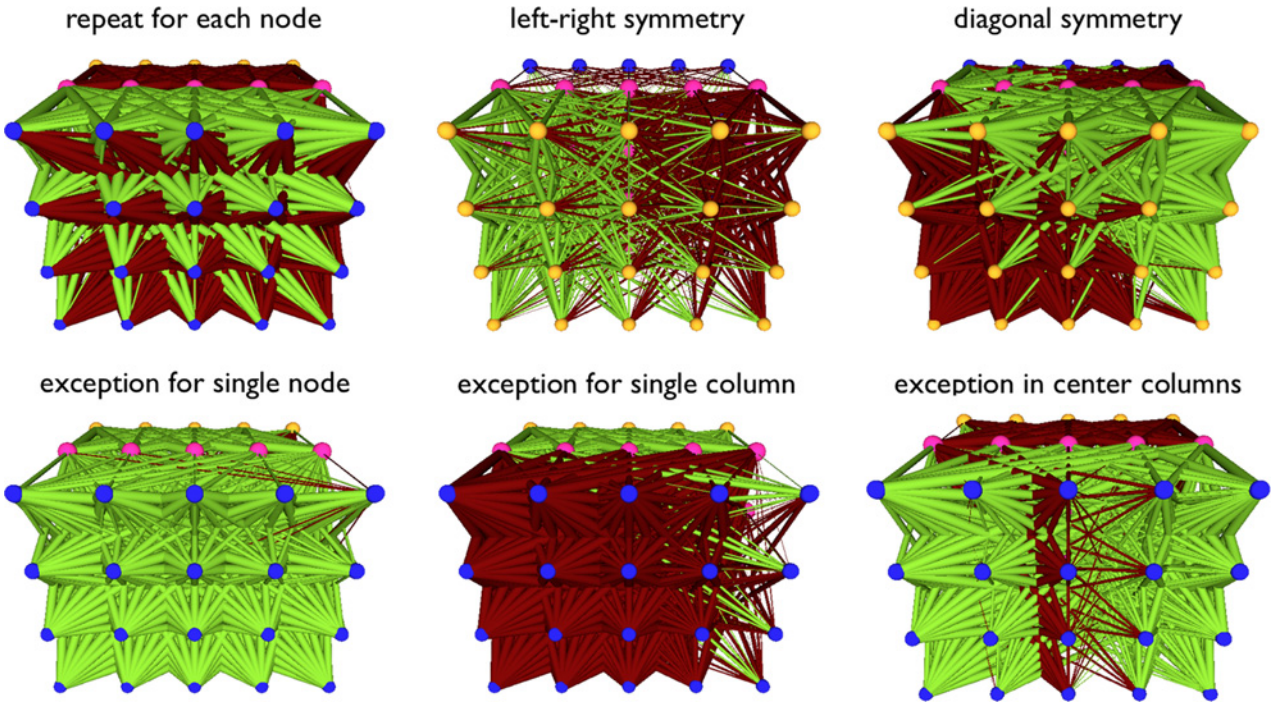


Fig. 16. Diverse set of ANN regularities produced by HyperNEAT, with and without variation. Fig. 15 explains what different colors/shades and line thicknesses represent.

that an indirect encoding frequently encounters them while the direct encoding rarely does.

As mentioned previously, HyperNEAT predominantly produces two different gaits on the quadruped controller problem: one with all legs in synchrony and the other with three legs in synchrony and the exception leg in opposite phase. It is sometimes possible to infer the behavior of a robot just by viewing a visualization of the regularities in its neural wiring. For example, ANNs that produce four-way symmetric gaits display similar wiring patterns for each row of outputs (Fig. 17, top). Recall that each row of outputs controls a separate leg (Fig. 7). For the 3-1 gaits in this experiment, the exception leg is always the front right leg, which is controlled by outputs in the top row. ANNs that produce 3-1 gaits frequently have a different pattern of weights for the top row of outputs than for the other outputs (Fig. 17, bottom). While it is not always possible to classify gaits by viewing ANNs alone, an informal experiment by one of the authors made a correct prediction for most runs.

It is interesting to note that in most output rows, patterns generalize to output nodes that do not control any aspect of the robot, and are therefore irrelevant to selection. Recall that in the output layer, only nodes in the first three columns (counting from the right when viewing from the back) control joints in the robot. The outputs in the last two columns are ignored (Fig. 7). This design decision enables us to investigate whether the patterns that HyperNEAT evolves generalize to geometric areas that are not under selective pressure. In a direct encoding, one would expect that weights connected to nodes that are not under selective pressure would be random. Accordingly, these unused columns do appear random in the FT-NEAT ANN visualizations, but so do all of the columns (Fig. 15). The patterns for unused HyperNEAT output nodes, on the other hand, generally exhibit extensions of the patterns displayed across the entire network. This phenomenon occurs in nearly every case, although a few networks had a different pattern in unused nodes than in other nodes within the same row (e.g., the bottom-right ANN in Fig. 17).

This result suggests that if joints or legs were added to an evolved HyperNEAT ANN, HyperNEAT would likely produce similar behaviors in those new joints and legs as elsewhere in the robot. If that is right, HyperNEAT would also likely succeed more consistently than a direct encoding when evolving a pre-evolved ANN further to control additional legs or joints, provided that it is beneficial to have behavioral similarities between the original and newly added components. This prediction assumes that the new components are placed at geometric coordinates such that an appropriate regularity applies. This general idea has already been demonstrated to work in a domain where HyperNEAT evolved ANNs to control multiple agents of a team [22]. Such an ability to transfer skills from one task to a new task is an important goal of machine learning [46] that an indirect encoding like HyperNEAT can potentially perform well at [23], but where a direct encoding is likely to perform poorly.

### *F. HyperNEAT ANNs are Quantifiably More Regular*

Because regular structures require less information to describe them, regularity can be measured by compression [1]. One test of the regularity of a structure, then, is how much it can be compressed. We applied the standard unix gzip compression program (version 1.3.5) to text files that contain only the weights of each ANN phenotype for HyperNEAT and FT-NEAT. NEAT ANNs are not compared because the

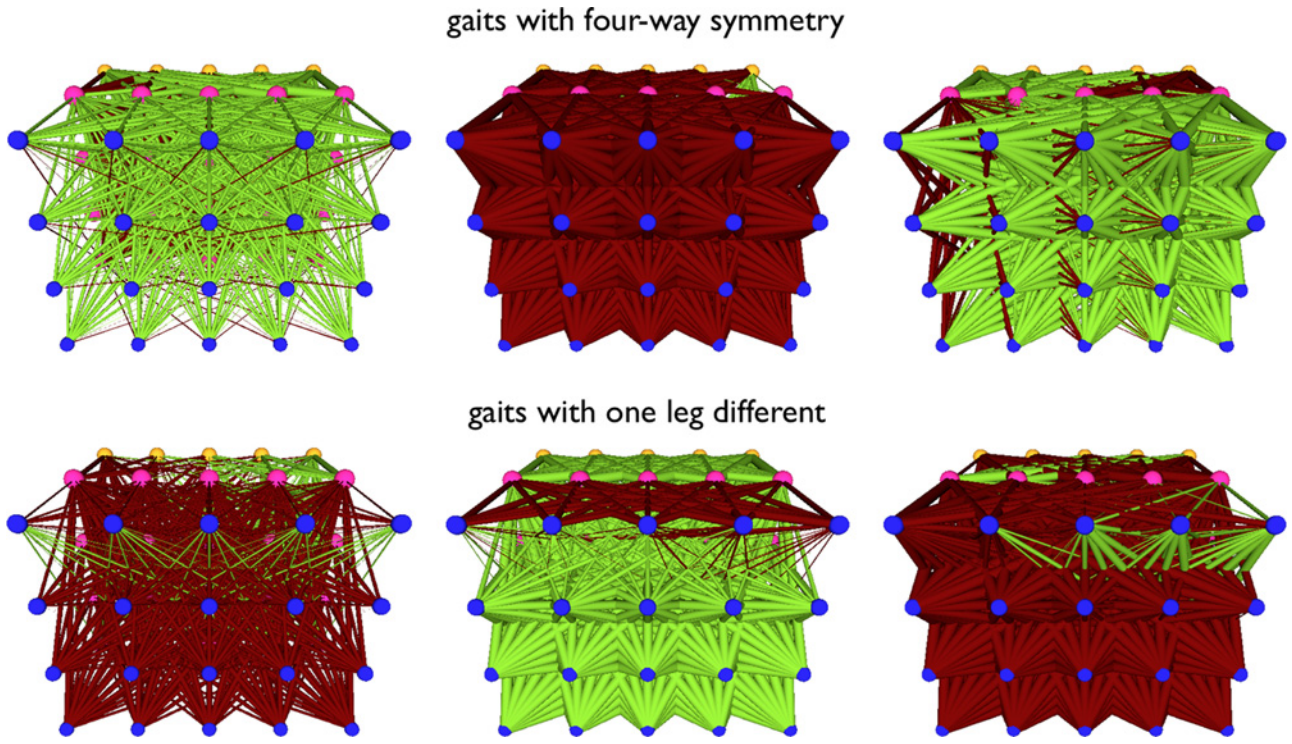


Fig. 17. Correlating ANN regularities to different behaviors. It is possible to recognize ANN patterns that produce different robotic gaits. The ANNs in the top row all generate a four-way symmetric gait. The weight patterns in these ANNs appear similar for all rows (legs are controlled by separate rows of nodes). The ANNs in the bottom row have three legs moving together and one leg in anti-phase. That exception leg is controlled by the nodes in the top row, which have a different pattern of weights than the other three rows. These views are from the back (looking at the outputs), as indicated by the color/shade scheme described in the caption of Fig. 15.

number of links in NEAT ANNs can vary, which means that compressibility alone is not isolated. This analysis is performed on the quadruped controller problem because it does not have regularity explicitly built in, as opposed to target weights and bit mirroring, where fitness scores already indicate ANN regularity.

The gzip algorithm is a conservative test of regularity because it looks for repeated symbols, but does not compress all mathematical regularities (e.g., each link weight increasing by a constant amount). Nevertheless, gzip is able to compress HyperNEAT ANNs on the regular quadruped controller problem significantly more than FT-NEAT ANNs: $p < 0.001$, comparing the difference between each uncompressed and compressed HyperNEAT file (mean 4488 bytes $\pm$710 SD) and each FT-NEAT file (mean 3349 bytes $\pm$37 SD). This quantifiable result confirms the clear yet subjective observation from visually inspecting HyperNEAT and FT-NEAT ANNs (Fig. 15), namely, that HyperNEAT ANNs are more regular.

### *G. Regularity of HyperNEAT ANNs Correlates with the Regularity of the Problem*

Another measure of the regularity of HyperNEAT ANNs is the number of nodes or links in the CPPN genome. Some of the HyperNEAT end-of-run champions on the quadruped controller problem, for example, have as few as 9 nodes and 26 links in their genome. Given that this genome encodes an ANN with 60 nodes and 800 links, the compression in this case is 6.6-fold and 30.8-fold, respectively. Unfortunately, this measure of regularity cannot be used for comparisons between HyperNEAT and direct encodings. It is unfair to compare genome sizes between HyperNEAT and FT-NEAT, given that the FT-NEAT genome is the same size as the final ANN. A comparison to NEAT is similarly unfair, because its genomes at least need to contain one node for every input and output node in the final ANN.

It is interesting, however, to investigate whether HyperNEAT genomes are smaller on more regular problems. As expected, the number of CPPN nodes in end-of-run champions tends to increase with the irregularity of the problem, meaning there is more compression (i.e., smaller genomes) on more regular problems. We focused this analysis on genomic *nodes* only, because the number of genomic links is correlated with the number of nodes. In target weights, the correlation between problem irregularity and number of CPPN nodes is positive ($r = 0.54$), although the trend is slightly insignificant ($p = 0.08$, using MATLAB's corrcoef correlation coefficients test). In bit mirroring, the correlation is more dramatic (Fig. 18). For both the experiment that reduces column regularity, and the experiment that further reduces row regularity, the trend is positive ($r = 0.91$) and highly significant ($p < 0.001$, using MATLAB's corrcoef correlation coefficients test). On the quadruped controller problem the trend is also positive ($r = 0.58$), although insignificant ($p > 0.05$, using MATLAB's corrcoef correlation coefficients test).

We also performed this same analysis on data from a previous publication that created irregularity in the quadruped

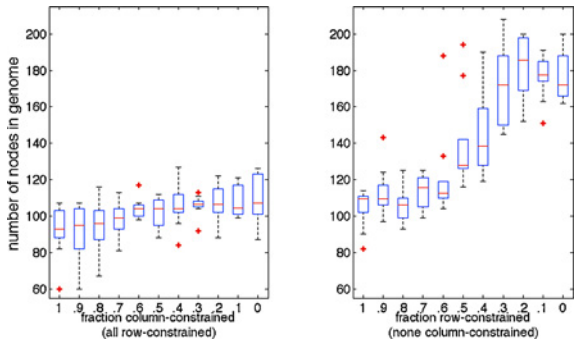


Fig. 18. Genome size increases with problem irregularity. The number of nodes in the CPPN genomes for HyperNEAT on the bit mirroring problem as irregularity is scaled from low (left) to high (right). The number of genomic nodes increases as the problem becomes more irregular.

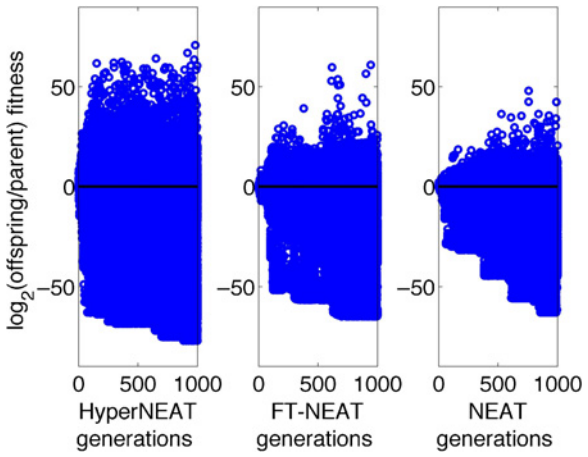


Fig. 19. Fitness changes caused by mutations with different encodings. Each circle represents the ratio of parent fitness over offspring fitness. Positive values indicate an offspring that is more fit than its parents, and higher numbers indicate larger fitness improvements. The inverse is true for negative numbers.

controller problem in a different way, by randomizing the geometric locations of each of the input and output nodes [17]. In the regular version of the problem, the nodes are laid out in a configuration that appears regular to a human engineer (e.g., the representation in Fig. 7). In the irregular version, the geometric locations of the nodes are randomized in each trial. We analyze data from three different experiments, where the nodes are represented with geometric coordinates in one, two, and three dimensions, respectively [17]. We previously reported that the performance of HyperNEAT is significantly higher in all three experiments on the regular version of the problem [17]. An analysis of the number of nodes in the CPPN genomes reveals that, for all three experiments, genome sizes are significantly smaller in the regular treatment than the irregular treatment ($p < 0.05$).

### H. HyperNEAT is More Evolvable

One of the touted benefits of indirect encodings is that the reuse of genetic information that produces regularity also enables coordinated mutational effects, which can be beneficial [20], [35]. It has previously been shown that a different indirect encoding based on L-systems produces more beneficial mutations than a direct encoding control [35]. This section investigates whether HyperNEAT similarly tends to produce a higher distribution of fitness values in mutated offspring than its direct encoding controls.

We analyzed the difference in fitness between organisms and their offspring in the quadruped controller problem in all cases where offspring were produced solely by mutation. While the majority of organisms were produced by crossover and mutation, this analysis isolates the impact of mutational effects. Over 1.3 million, 1.7 million, and 1.5 million organisms were produced solely via mutation for HyperNEAT, FT-NEAT, and NEAT treatments, respectively, providing a substantial sample size.

Overall, the indirect encoding HyperNEAT produces a wider *range* of fitness changes than the direct encodings (Fig. 19). HyperNEAT also has a distribution of fitness values with a higher median than both FT-NEAT and NEAT ($p < 0.001$). While HyperNEAT also produces more extreme negative fitness changes, they are balanced by more extreme positive fitness changes. For example, with respect to the ratio of parent fitness to offspring fitness, 5.8% of HyperNEAT offspring have a positive value greater than 20, whereas for FT-NEAT and NEAT only 0.35% and 0.21% do, respectively (Fig. 19). Despite the many extreme negative fitness changes, it appears that the continuous production of organisms that are much more fit than their parents fuels the success of HyperNEAT over FT-NEAT and NEAT on this problem.

### I. HyperNEAT's Bias Toward Regularity Hurts its Performance on Problems with Irregularity, as Demonstrated by a New Algorithm Called HybrID

The previous sections have documented that HyperNEAT's performance decreases as the irregularity of a problem increases. One explanation is that HyperNEAT's bias toward producing regular solutions makes it less likely to create the phenotypic irregularities necessary to account for problem-irregularities. The clearest example of this phenomenon is on the target weights problem in the treatment in which 90% of the weights had the same value, but 10% of the weights had different randomly-assigned values (Fig. 8). In a few generations, HyperNEAT discovers and exploits the regularity of the problem by setting 100% of its weights to the value that is correct for 90% of them. For the remaining hundreds of generations in the experiment, however, HyperNEAT fails to make exceptions to that regular pattern to account for the 10% of irregular link values. While the patterns observed in visualizations of the ANNs produced by HyperNEAT demonstrate that HyperNEAT can in fact produce some variation on overall patterns, many of those exceptions are themselves regular and affect whole geometric regions (Section IV-E). The target weights problem shows that changing specific weights to match an irregular target is challenging for HyperNEAT. However, such fine-tuning of neural connections may be required for real-world problems that have some degree of irregularity.

That HyperNEAT's performance generally decreases with problem irregularity suggests the hypothesis that HyperNEAT

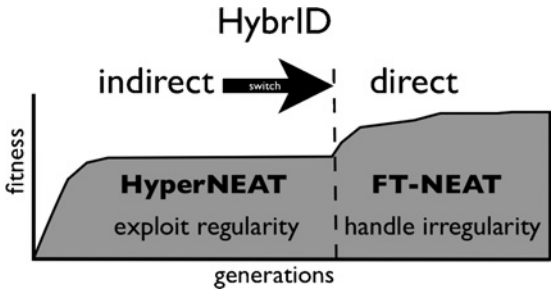


Fig. 20. Hybridizing indirect and direct encodings in the HybrID algorithm. The HybrID implementation in this paper evolves with HyperNEAT in the first phase until a switch is made to FT-NEAT. The idea is that the indirect encoding phase can produce regular weight patterns that can exploit problem regularity, and the direct encoding phase can fine tune that pattern to account for problem irregularities. In this hypothetical example, large fitness gains are initially made by the indirect encoding because it exploits problem regularity, but improvement slows because the indirect encoding cannot adjust its regular patterns to handle irregularities in the problem. Fitness increases again, however, once the direct encoding begins to fine-tune the regular structure produced by the indirect encoding.

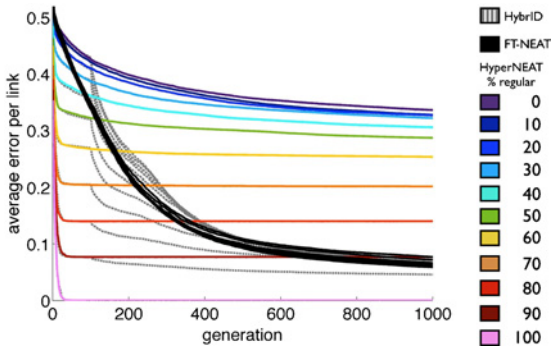


Fig. 21. Comparison of HyperNEAT, FT-NEAT, and HybrID on a range of problem regularities for the target weights problem. For each regularity level, a HybrID line (dashed gray) departs from the corresponding HyperNEAT line (colored) at the switch point (generation 100). The performance of FT-NEAT (black lines) is unaffected by the regularity of the problem, which is why the lines are overlaid and indistinguishable. HybrID outperforms HyperNEAT and FT-NEAT in early generations on versions of the problem that are mostly regular but have some irregularities.

would perform better on such problems if it were able to both generate regularities and irregularities. An alternate explanation for the performance drop is that the less-regular problems are simply harder. One way to test whether the first hypothesis is correct is to create an algorithm that can generate regularities and then adjust those regularities to create irregularities.

Because indirect encodings excel at producing regular patterns, and direct encodings excel at producing irregular patterns, the combination of the two may produce both. The hybridization of indirect and direct encodings (HybrID) algorithm [28] is based on this idea (Fig. 20).

While we are not aware of any prior work that specifically combines direct and indirect encodings, researchers have previously altered representations during evolutionary search, primarily to change the precision of values being evolved by genetic algorithms [47]. Other researchers have employed non-evolutionary optimization techniques to fine-tune the details of evolved solutions [48]. However, such techniques do not leverage the benefits of indirect encodings.

While the name HybrID applies to any combination of indirect and direct encodings, this paper reports results for one specific implementation called a *switch-HybrID* [28], wherein an indirect encoding is run first and then a switch is made to a direct encoding. Specifically, HyperNEAT is the encoding for each generation until the switch point, when each HyperNEAT ANN phenotype is converted into an FT-NEAT genome. Evolution then continues as normal with FT-NEAT until the end of the experiment. HyperNEAT and FT-NEAT serve as the indirect and direct encodings in the HybrID implementation in this paper to provide fair comparisons to the results from previous sections. For each of the HybrID experiments in this paper, a different switch point is chosen for illustrative purposes. In future applications of the HybrID algorithm, it may be more effective to choose a separate switch point for each run automatically based on the rate of fitness improvement (or lack thereof).

1) *Target Weights:* HybrID's performance on the target weights problem reveals that it does combine the regularity-generating attribute of indirect encodings with the irregularity-generating attribute of direct encodings (Fig. 21). At the switch point of 100 generations, HybrID immediately makes noticeable gains over HyperNEAT, and 150 generations later these gains are significant on all treatments except the perfectly regular one ($p < 0.001$). This result confirms the hypothesis that HyperNEAT can do better on some irregular problems if a further process of refinement creates some irregularity within its regular patterns. It is additionally interesting that HybrID significantly outperforms the direct encoding FT-NEAT on regular versions of the problem early in the experiment ($p < 0.01$ at generation 250 on the 70%, 80% and 90% regular problems); The HyperNEAT phase of HybrID first discovers the regularity of the problem, giving the FT-NEAT phase of HybrID a head start over FT-NEAT on these problems. While the performance of HybrID and FT-NEAT is similar by the end of the experiment, that result is likely because this problem is simple and has no interactions (epistasis) between individual link weight values. Direct encodings are expected to perform well on problems without epistasis, but most real world problems are highly epistatic.

2) *Bit Mirroring:* On the bit mirroring problem, which does have epistasis, HybrID's performance ties HyperNEAT's performance on regular versions of the problem, and significantly outperforms HyperNEAT on problems with a certain level of irregularity (Fig. 22, see figure for statistical significance per treatment). This result further highlights that HyperNEAT produces regular patterns that can benefit from a refining process that generates irregularity. The performance gap between HybrID and HyperNEAT is largest on problems with intermediate levels of regularity. The gap in performance narrows on the most irregular treatments because such configurations are difficult and both algorithms perform poorly.

These data are pooled across ten runs per treatment on a $7 \times 7$ grid. Each experiment lasted 5000 generations with a switch point at 2500 generations. A comparison to FT-NEAT is not shown because HyperNEAT outperforms FT-NEAT on all versions of this problem (Section IV-A2).

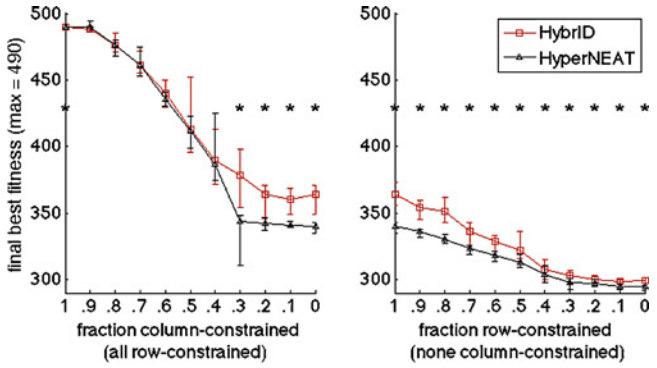


Fig. 22. Performance of HybrID versus HyperNEAT on the bit mirroring problem. Regularity decreases from left to right. Plotted are median values ± the 25th and 75th quartiles. Asterisks indicate $p < 0.05$.

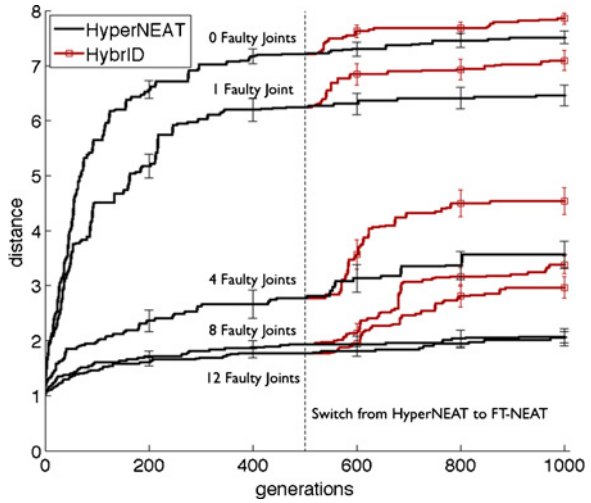


Fig. 23. Performance of HybrID versus HyperNEAT on the quadruped controller problem. Error bars show one standard error of the median. HybrID outperforms HyperNEAT on all versions of the quadruped controller problem. The increase generally correlates with the number of faulty joints.

3) *Quadruped Controller:* HybrID outperforms HyperNEAT on every version of the quadruped controller problem (Fig. 23), although the difference is significant only on problems with a certain amount of irregularity ($p < 0.01$ on treatments with four or more faulty joints). HybrID increases performance over HyperNEAT by 5%, 10%, 27%, 64%, and 44%, respectively, for the treatments with 0, 1, 4, 8, and 12 faulty joints. These substantial performance improvements on the quadruped controller problem, which is a challenging engineering problem, highlight the degree to which HyperNEAT's inability to produce irregularity on its own can harm its performance.

HybrID also outperforms both direct encodings on all treatments of the problem ($p < 0.05$). HyperNEAT significantly outperforms both direct encodings only on the two most regular versions of the problem ($p < 0.01$). That HybrID outperforms the direct encodings on irregular problems underscores that it does not just act like a direct encoding on irregular problems, but instead first leverages the indirect encoding's ability to exploit available regularities and then improves upon those by accounting for problem irregularities via the direct encoding.

It is instructive to examine how the FT-NEAT phase of HybrID changes the patterns provided to it by the HyperNEAT phase. Visualizations of ANNs at the end of each HyperNEAT phase and the ANN for that same run after the FT-NEAT phase can provide clues to how HybrID generates its performance improvements. Examples from runs in the treatment with one faulty joint are shown in Fig. 24. In all cases, the FT-NEAT phase of HybrID makes no major changes to the overall regular pattern produced by the HyperNEAT phase (visualizations of ANNs after the HyperNEAT and FT-NEAT phases for each HybrID run are available at http://devolab.msu.edu/SupportDocs/Regularity). Evolution thus maintains the regular pattern HyperNEAT generates even while that pattern is being fine-tuned by the direct encoding.

The types of exceptions HybrID produces are different from those seen by HyperNEAT alone. In many cases, only a few weights are noticeably changed by the FT-NEAT phase of HybrID, and these changes occur in an irregular distribution. For example, in the run depicted in the left column of Fig. 24, HyperNEAT produces the regular pattern of inhibitory connections to all of the output nodes. FT-NEAT switches some of those to excitatory connections, which may have been difficult for HyperNEAT to do without changing many other weights. In another example run, depicted in the middle column of Fig. 24, the only noticeable change FT-NEAT made is the creation of a single, strong, excitatory connection. Of course, in both cases there are subtle changes in many of the link weights that do not stand out to the human eye.

In the third example run, changes were made during the FT-NEAT phase to many different weights, yet the overall patterns remained (Fig. 24, right column). Many of these changes are irregular, such as the weights switched from excitatory to inhibitory and vice versa in the top-left node, and the few links that switch to excitatory in the bottom row. What is unusual and fascinating about this example run, however, is that the direct encoding makes many *regular* changes. For example, most of the links in the top row proportionally increase in strength, which preserves the regular patterns. These visualizations demonstrate a rare case of a direct encoding producing coordinated phenotypic changes. It might be the case that the indirect encoding discovered regularities that put this organism on the side of a hill in a fitness landscape, but climbing that hill is difficult for the indirect encoding because mutations to the genome that increase the strength of connections in these nodes may change other weights and thus decrease fitness overall. The direct encoding does not have such constraints, and thus can increase the magnitude of all of these links. This hypothetical explanation illustrates how evolution can produce coordinated change through direct encodings if it starts in a place in the fitness landscape where there is a positive fitness gradient in the same direction for many link values. Interestingly, it is unlikely that the direct encoding would have discovered this starting point without the indirect encoding.

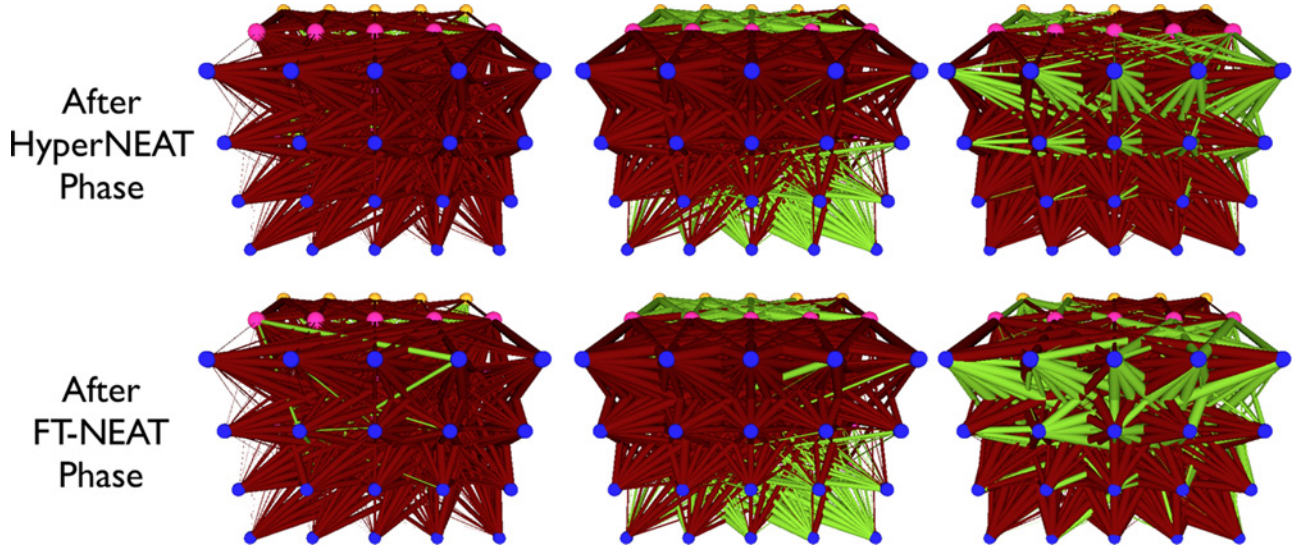


Fig. 24. Visualizations of the ANNs produced at the end of the HyperNEAT phase and FT-NEAT phase of HybrID for three example runs.

## V. Discussion

While it is well-established that indirect encodings can outperform direct encodings on regular problems [2], [7]–[14], no prior studies have investigated how indirect encodings compare to direct encodings on problems as regularity scales from high to low. Given that most realistic problems will not be at the extreme regular end of this continuum, it is important to understand how indirect encodings perform on problems with different levels of regularity.

On three different problems we have shown that indirect encodings are able to automatically exploit intermediate amounts of problem regularity, and that their performance improves as the regularity of the problem increases. Moreover, the indirect encodings outperform direct encoding controls on regular problems. We have also shed light on why indirect encodings perform better, which is because they produce both regular ANNs and regular behaviors that exploit problem regularity. These results suggests that indirect encodings may be an attractive alternative to direct encodings as evolutionary algorithms are applied to increasingly complicated engineering problems, because such problems are likely to contain regularities. The results from the bit mirroring problem also reveal that HyperNEAT is able to independently exploit different types of regularity within the same problem; scaling three different regularities (within-row, within-column, and inherent) independently contributes to overall performance [29]. One interesting caveat we discovered on all three problems, however, is that HyperNEAT does not exploit a particular type of regularity until the level of regularity within that type is above a threshold. Tests on additional problems and with different indirect encodings are required to discover how general this finding is. Additionally, further quantitative analyses of regularity in both evolved solutions and in the problems themselves would shed additional light on how general the findings in this paper are.

It is particularly noteworthy that HyperNEAT is able to automatically exploit the regularity of the challenging quadruped controller problem. This result is important because evolutionary algorithms have previously performed poorly when evolving gaits for legged robots because they could not automatically exploit the problem's regularities: to perform well, researchers needed to manually identify such regularities and force the encoding to exploit them [32], [38]–[43]. This manual approach is time consuming, restrictive, and may fail to identify helpful regularities.

Because indirect encodings are biased toward producing regular patterns in solutions, it is important to understand the degree to which they can vary and make exceptions to these patterns. Two separate lines of evidence confirm that HyperNEAT has difficulty making certain types of exceptions. Initially, its performance decreases as a problem becomes more irregular. Second, the HybrID algorithm boosts performance over HyperNEAT on intermediately regular problems, confirming that HyperNEAT is not creating some irregularities that would aid performance. On the other hand, visualizations of HyperNEAT ANNs reveal that HyperNEAT can create variations on patterns, and even exceptions for single nodes. However, these variations and exceptions themselves are regular, suggesting that HyperNEAT creates its exceptions by adding one regularity to another, with the result being an overall regularity with a regular variation within it. The HyperNEAT visualizations rarely demonstrate cases where single weights violate the prevailing pattern. HybrID, on the other hand, does provide examples of such single-link exceptions. The significant boost in performance by HybrID implies that the ability to make such radical exceptions at the single-link level is sometimes important.

HybrID's performance increase over HyperNEAT, combined with investigations into the different types of exceptions HybrID and HyperNEAT make, suggest that HyperNEAT can benefit from a process of refinement that adjusts individual link patterns in an irregular way. While a direct encoding provides such refinement in this paper, there are other candidate refinement processes. One intriguing possibility is that lifetime

adaptation via learning can play a similar role [44], [49]–[51]. Lifetime learning algorithms could adjust the overall regular patterns produced by HyperNEAT to account for necessary irregularities. Having a learning algorithm serve as the refining process may be superior to a direct encoding, especially as ANNs scale closer to the size of brains in nature. While HybrID works well on the problems in this paper, its direct encoding component may ultimately encounter the same scaling challenges that all direct encodings face on high-dimensional problems.

The ANN weight patterns produced by HyperNEAT alone are also interesting because they demonstrate the types of regularities that can be produced by an indirect encoding that incorporates geometric concepts from developmental biology. The pictures evolved with CPPNs reveal that CPPNs can encode many features observed in natural animal bodies, such as symmetry and serial repetition, with and without variation (Fig. 2) [24]. The visualizations presented here of ANNs evolved with CPPNs demonstrate that HyperNEAT can generate these same properties in ANNs. It is clear by looking at the different visualizations that different geometric patterns are being created and combined to produce complex neural patterns, which is reminiscent of how nature produces complex brains and bodies [3].

Outside the field of neuroevolution, other techniques have improved performance by biasing neural networks toward regular weight patterns, such as weight sharing [52], [53] and convolutional networks [54]. However, such methods typically involve the researcher imposing a certain regularity on the ANN (such as a subset of weights all being identical). The fact that these techniques were successful demonstrates that regular patterns in neural connections can be beneficial. However, these methods typically do not automatically discover which regularities to create. HyperNEAT is novel because it explores a large space of possible geometric regularities to find those that enhance performance. This ability to discover and encode different regularities suggests that HyperNEAT can potentially adapt the regularities it produces to different domains, instead of needing to be manually tuned for each domain.

## VI. Conclusion

This paper contains the first extensive study in the field of evolutionary computation comparing an indirect encoding to direct encoding controls across a continuum of problems with different levels of regularity. On three different problems the performance of the indirect encoding improved with the regularity of the problem, and the indirect encoding outperformed the direct encodings on more regular versions of problems. The indirect encoding was able to exploit problem regularity by generating regular neural networks that produced regular behaviors. The specific indirect encoding studied is based on concepts from developmental biology involving geometric patterning, which enabled domain knowledge and preferences to be injected into the algorithm. Moreover, this generation and combination of geometric coordinate frames created regular weight patterns in neural networks that are visually complex and resemble regularities seen in natural brains.

The indirect encoding's bias toward regularity hurt its performance on problems that contained some irregularity. A new algorithm that first evolves with an indirect encoding and then switches to a direct encoding was able to outperform the indirect encoding alone. This HybrID algorithm outperformed the indirect encoding because it made subtle adjustments to regular patterns to account for problem irregularities. The success of this approach suggests that indirect encodings may be most effective not as stand-alone algorithms, but in combination with a refining process that adjusts regular patterns in irregular ways to account for problem irregularities.

Overall, this paper provides a more comprehensive picture of how indirect encodings compare to direct encodings by evaluating them across a continuum from high to low problem regularity. This paper also suggests a path forward that combines the pattern-producing power of indirect encodings with a process of refinement to account for the irregularities that are likely to exist in challenging problems.

## Acknowledgment

The authors would like to thank the members of the Digital Evolution Laboratory, Michigan State University, East Lansing, in particular P. K. McKinley, B. E. Beckmann, and D. Knoester.

## References

[1] H. Lipson, "Principles of modularity, regularity, and hierarchy for scalable systems," *J. Biol. Phys. Chem.*, vol. 7, no. 4, p. 125, 2007.
[2] K. Stanley and R. Miikkulainen, "A taxonomy for artificial embryogeny," *Artif. Life*, vol. 9, no. 2, pp. 93–130, 2003.
[3] S. Carroll, *Endless Forms Most Beautiful: The New Science of Evo Devo and the Making of the Animal Kingdom*. New York: Norton, 2005.
[4] S. Carroll, J. Grenier, and S. Weatherbee, *From DNA to Diversity*. Malden, MA: Blackwell Science, 2001.
[5] R. Raff and J. Slack, *The Shape of Life: Genes, Development, and the Evolution of Animal Form*. Chicago, IL: University of Chicago Press, 1996.
[6] C. Southan, "Has the yo-yo stopped? An assessment of human protein-coding gene number," *Proteomics*, vol. 4, no. 6, pp. 1712–1726, 2004.
[7] D. D'Ambrosio and K. Stanley, "A novel generative encoding for exploiting neural network sensor and output geometry," in *Proc. Genet. Evol. Comput. Conf.*, 2007, pp. 974–981.
[8] J. Gauci and K. Stanley, "Generating large-scale neural networks through discovering geometric regularities," in *Proc. Genet. Evol. Comput. Conf.*, 2007, pp. 997–1004.
[9] F. Gruau and E. Lip, "Genetic synthesis of Boolean neural networks with a cell rewriting developmental process," in *Proc. Int. Workshop Combin. Genet. Algorithms Neural Netw.*, 1992, pp. 55–74.
[10] F. Gruau, D. Whitley, and L. Pyeatt, "A comparison between cellular encoding and direct encoding for genetic neural networks," in *Proc. 1st Annu. Conf. Genet. Programming*, 1996, pp. 81–89.
[11] G. Hornby and J. Pollack, "Creating high-level components with a generative representation for body-brain evolution," *Artif. Life*, vol. 8, no. 3, pp. 223–246, 2002.
[12] J. Miller, "Evolving a self-repairing, self-regulating, French flag organism," in *Proc. Genet. Evol. Comput. Conf.*, 2004, pp. 129–139.
[13] J. Reisinger and R. Miikkulainen, "Acquiring evolvability through adaptive representations," in *Proc. Genet. Evol. Comput. Conf.*, 2007, p. 1052.
[14] K. Stanley, D. D'Ambrosio, and J. Gauci, "A hypercube-based encoding for evolving large-scale neural networks," *Artif. Life*, vol. 15, no. 2, pp. 185–212, 2009.
[15] G. Striedter, *Principles of Brain Evolution*. Sunderland, MA: Sinauer Associates, 2005.
[16] K. Stanley, "Compositional pattern producing networks: A novel abstraction of development," *Genet. Programming Evol. Mach.*, vol. 8, no. 2, pp. 131–162, 2007.

[17] J. Clune, C. Ofria, and R. Pennock, "The sensitivity of HyperNEAT to different geometric representations of a problem," in *Proc. Genet. Evol. Comput. Conf.*, 2009, pp. 675–682.
[18] J. Gauci and K. Stanley, "A case study on the critical role of geometric regularity in machine learning," in *Proc. AAAI Conf. Artif. Intell.*, 2008, pp. 628–633.
[19] J. Gauci and K. Stanley, "Autonomous evolution of topographic regularities in artificial neural networks," *Neural Comput.*, vol. 22, no. 7, pp. 1860–1898, 2010.
[20] J. Clune, B. Beckmann, C. Ofria, and R. Pennock, "Evolving coordinated quadruped gaits with the HyperNEAT generative encoding," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 2764–2771.
[21] D. D'Ambrosio and K. Stanley, "Generative encoding for multiagent learning," in *Proc. Genet. Evol. Comput. Conf.*, 2008, pp. 819–826.
[22] D. D'Ambrosio, J. Lehman, S. Risi, and K. Stanley, "Evolving policy geometry for scalable multiagent learning," in *Proc. Conf. Autonomous Agents Multiagent Syst.*, 2010, pp. 731–738.
[23] P. Verbancsics and K. O. Stanley, "Transfer learning through indirect encoding," in *Proc. Genet. Evol. Comput. Conf.*, 2010, pp. 547–554.
[24] J. Secretan, N. Beato, D. D Ambrosio, A. Rodriguez, A. Campbell, and K. Stanley, "Picbreeder: Evolving pictures collaboratively online," in *Proc. Comput. Hum. Interaction Conf.*, 2008, pp. 1759–1768.
[25] K. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evol. Comput.*, vol. 10, no. 2, pp. 99–127, 2002.
[26] K. Stanley and R. Miikkulainen, "Competitive coevolution through evolutionary complexification," *J. Artif. Intell. Res.*, vol. 21, no. 1, pp. 63–100, 2004.
[27] J. Clune, B. Beckmann, P. McKinley, and C. Ofria, "Investigating whether HyperNEAT produces modular neural networks," in *Proc. Genet. Evol. Comput. Conf.*, 2010, pp. 635–642.
[28] J. Clune, B. Beckmann, R. Pennock, and C. Ofria, "HybrID: A hybridization of indirect and direct encodings for evolutionary computation," in *Proc. Eur. Conf. Artif. Life*, 2009, pp. 1–8.
[29] J. Clune, C. Ofria, and R. Pennock, "How a generative encoding fares as problem-regularity decreases," in *Proc. Parallel Problem Solving From Nature*, 2008, pp. 358–367.
[30] C. Ridderstrom, "Legged locomotion control: A literature survey," Royal Instit. Technol., Stockholm, Sweden, Tech. Rep. TRITA-MMK, ISSN 1400-1179, 1999.
[31] D. Wettergreen and C. Thorpe, "Gait generation for legged robots," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Jul. 1992, pp. 1413–1420.
[32] G. Hornby, S. Takamura, T. Yamamoto, and M. Fujita, "Autonomous evolution of dynamic gaits with two quadruped robots," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 402–410, Jun. 2005.
[33] J. Gallagher, R. Beer, K. Espenschied, and R. Quinn, "Application of evolved locomotion controllers to a hexapod robot," *Robot. Autonomous Syst.*, vol. 19, no. 1, pp. 95–103, 1996.
[34] F. Gruau, "Automatic definition of modular neural networks," *Adaptive Behavior*, vol. 3, no. 2, pp. 151–183, 1994.
[35] G. Hornby, H. Lipson, and J. Pollack, "Generative representations for the automated design of modular physical robots," *IEEE Trans. Robot. Autom.*, vol. 19, no. 4, pp. 703–719, Aug. 2003.
[36] H. Liu and H. Iba, "A hierarchical approach for adaptive humanoid robot control," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2004, pp. 1546–1553.
[37] K. Sims, "Evolving 3-D morphology and behavior by competition," *Artif. Life*, vol. 1, no. 4, pp. 353–372, 1994.
[38] R. Téllez, C. Angulo, and D. Pardo, "Evolving the walking behavior of a 12 DOF quadruped using a distributed neural architecture," in *Biologically Inspired Approaches to Advanced Information Technology*. Berlin, Germany: Springer, 2006, pp. 5–19.
[39] V. Valsalam and R. Miikkulainen, "Modular neuroevolution for multilegged locomotion," in *Proc. Genet. Evol. Comput. Conf.*, 2008, pp. 265–272.
[40] R. Beer and J. Gallagher, "Evolving dynamical neural networks for adaptive behavior," *Adaptive Behav.*, vol. 1, no. 1, p. 91, 1992.
[41] D. Filliat, J. Kodjabachian, and J. Meyer, "Evolution of neural controllers for locomotion and obstacle avoidance in a six-legged robot," *Connection Sci.*, vol. 11, no. 3, pp. 225–242, 1999.
[42] J. Kodjabachian and J. Meyer, "Evolution and development of neural controllers for locomotion, gradient-following, and obstacle-avoidance in artificial insects," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, pp. 796–812, Sep. 1998.
[43] M. Raibert, M. Chepponis, and H. Brown, Jr., "Running on four legs as though they were one," *IEEE J. Robot. Autom.*, vol. 2, no. 2, pp. 70–82, Jun. 1986.
[44] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. Cambridge, MA: MIT Press, 2000.
[45] K. Wolff and P. Nordin, "An evolutionary based approach for control programming of humanoids," in *Proc. 3rd Int. Conf. Humanoid Robots*, 2003, pp. 1–19.
[46] M. Taylor, P. Stone, and Y. Liu, "Transfer learning via inter-task mappings for temporal difference learning," *J. Mach. Learning Res.*, vol. 8, no. 1, pp. 2125–2167, 2007.
[47] A. Eiben, Z. Michalewicz, M. Schoenauer, and J. Smith, "Parameter control in evolutionary algorithms," in *Parameter Setting in Evolutionary Algorithms*. Berlin, Germany: Springer, 2007, pp. 19–46.
[48] J. Grimbleby, "Automatic analogue circuit synthesis using genetic algorithms," *IEE Proc. Circuits Devices Syst.*, vol. 147, no. 6, pp. 319–324, Dec. 2000.
[49] A. Soltoggio, J. Bullinaria, C. Mattiussi, P. Durr, and D. Floreano, "Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios," in *Proc. Int. Conf. Artif. Life*, vol. 11. 2008, pp. 569–576.
[50] A. Soltoggio, P. Durr, C. Mattiussi, and D. Floreano, "Evolving neuromodulatory topologies for reinforcement learning-like problems," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2007, pp. 2471–2478.
[51] S. Risi and K. Stanley, "Indirectly encoding neural plasticity as a pattern of local rules," in *Proc. 11th Int. Conf. Simul. Adaptive Behav.*, to be submitted.
[52] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
[53] D. Rumelhart, G. Hinton, and R. Williams, *Learning Internal Representations by Error Propagation, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. Cambridge, MA: MIT Press, 1986.
[54] S. Lawrence, C. Giles, A. Tsoi, and A. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98–113, Jan. 1997.

**Jeff Clune** received the Bachelor's degree in philosophy from the University of Michigan, Ann Arbor, and the Master's degree in philosophy and the Ph.D. degree in computer science, both from Michigan State University, East Lansing.

He is currently a Post-Doctoral Fellow with Hod Lipson's laboratory in the Department of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY, funded by a Postdoctoral Research Fellowship in Biology from the U.S. NSF. He studies generative encodings, which enhance evolutionary algorithms by augmenting them with concepts from developmental biology. Such concepts enable the assembly of complex forms from compact genomes. He combines these generative encodings with neuroevolution, a technology that harnesses the power of evolution to construct artificial neural networks (ANNs). Evolving ANNs with generative encodings creates large-scale, structurally organized ANNs that produce sophisticated, coordinated behaviors. He demonstrates the capabilities of such ANNs in robotic control problems. He also develops evolutionary algorithms to investigate open questions in evolutionary biology, and has published work on the evolution of altruism, phenotypic plasticity, and evolvability. His research contributes to biology, because it improves our knowledge of evolving systems, and enhances computer science, because it helps design better evolutionary algorithms. He is also a co-author of Avida-ED, a software tool for teaching evolution. Articles about his research have appeared in many news publications, including *New Scientist*, *the Daily Telegraph*, *Slashdot*, and *U.S. News and World Report*.

**Kenneth O. Stanley** received the B.S.E. degree from the University of Pennsylvania, Philadelphia, in 1997, and the Ph.D. degree from the University of Texas, Austin, in 2004.

He is currently an Assistant Professor with the School of Electrical Engineering and Computer Science, University of Central Florida, Orlando. He is an inventor of the neuroevolution of augmenting topologies (NEAT) and HyperNEAT algorithms for evolving complex artificial neural networks. His main research contributions are in neuroevolution

(i.e., evolving neural networks), generative and developmental systems, co-evolution, machine learning for video games, and interactive evolution.

Dr. Stanley has won separate best paper awards for his work on NEAT, NERO, NEAT Drummer, HyperNEAT, novelty search, and Galactic Arms Race. He is the Chair of the IEEE Task Force on Computational Intelligence and Video Games and is an Associate Editor of the IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES. He recently joined the Editorial Board of the *Evolutionary Computation* journal.

**Robert T. Pennock** is currently a Professor with Michigan State University, East Lansing, where he is on the faculty of Lyman Briggs College, the Philosophy Department, the Department of Computer Science and Engineering, and the Ecology, Evolutionary Biology and Behavior Graduate Program. He is one of the leads of the BEACON Center for the Study of Evolution in Action. His scientific research in experimental evolution and evolutionary computation focuses especially on the emergence of complexity and intelligent behavior, and has been featured in *Discover*, *New Scientist*, *Slashdot*, and many other periodicals. His philosophical research interests include philosophy of biology, artificial life, and the relationship of epistemic and ethical values in science. He speaks regularly around the country on these topics, and has been named a National Distinguished Lecturer by Sigma Xi, the Scientific Research Society.

**Charles Ofria** received the Bachelor's degree, with a triple major in pure mathematics, applied mathematics, and computer science, from the State University of New York at Stony Brook, Stony Brook, in 1994, and the Ph.D. degree in computation and neural systems from the California Institute of Technology, Pasadena, in 1999, followed by a post-doctoral position in the Center for Microbial Ecology, Michigan State University (MSU), East Lansing.

He is the Director of the MSU Digital Evolution Laboratory, Deputy Director of the BEACON Center for the Study of Evolution in Action, and an Associate Professor with the Department of Computer Science and Engineering, MSU. He is the Architect of the Avida Digital Evolution Research Platform, which is downloaded over a thousand times per month for use in research and education at dozens of universities around the world. His current research interests include the intersection of computer science and evolutionary biology, developing a two-way flow of ideas between the fields. In particular, he is focused on understanding the evolution of biological complexity and complex behaviors, including evolvability, navigation, cooperation, division of labor, and intelligence.

PROCEEDINGS OF THE ROYAL SOCIETY B BIOLOGICAL SCIENCES

# Selective pressures for accurate altruism targeting: evidence from digital evolution for difficult-to-test aspects of inclusive fitness theory

Jeff Clune, Heather J. Goldsby, Charles Ofria and Robert T. Pennock

*Proc. R. Soc. B* 2011 **278**, 666-674 first published online 15 September 2010
doi: 10.1098/rspb.2010.1557

PROCEEDINGS OF THE ROYAL SOCIETY B

*Proc. R. Soc. B* (2011) **278**, 666–674
doi:10.1098/rspb.2010.1557
*Published online* 15 September 2010

# Selective pressures for accurate altruism targeting: evidence from digital evolution for difficult-to-test aspects of inclusive fitness theory

**Jeff Clune[1,2,5,*], Heather J. Goldsby[2,3,5], Charles Ofria[2,3,5] and Robert T. Pennock[1,2,3,4,5]**

[1]*Department of Philosophy,* [2]*Department of Computer Science and Engineering,* [3]*Ecology, Evolutionary Biology and Behavior Program,* [4]*Lyman Briggs College, and* [5]*The BEACON Center for the Study of Evolution in Action, Michigan State University, East Lansing, MI 48824, USA*

Inclusive fitness theory predicts that natural selection will favour altruist genes that are more accurate in targeting altruism only to copies of themselves. In this paper, we provide evidence from digital evolution in support of this prediction by competing multiple altruist-targeting mechanisms that vary in their accuracy in determining whether a potential target for altruism carries a copy of the altruist gene. We compete altruism-targeting mechanisms based on (i) kinship (*kin targeting*), (ii) genetic similarity at a level greater than that expected of kin (*similarity targeting*), and (iii) perfect knowledge of the presence of an altruist gene (*green beard targeting*). Natural selection always favoured the most accurate targeting mechanism available. Our investigations also revealed that evolution did not increase the altruism level when all green beard altruists used the same phenotypic marker. The green beard altruism levels stably increased only when mutations that changed the altruism level also changed the marker (e.g. beard colour), such that beard colour reliably indicated the altruism level. For kin- and similarity-targeting mechanisms, we found that evolution was able to stably adjust altruism levels. Our results confirm that natural selection favours altruist genes that are increasingly accurate in targeting altruism to only their copies. Our work also emphasizes that the concept of targeting accuracy must include both the presence of an altruist gene and the level of altruism it produces.

**Keywords:** kin selection; inclusive fitness; altruism; green beard; digital evolution; Avida

## 1. BACKGROUND

Inclusive fitness theory, also known as kin selection theory, describes when a trait will be favoured by natural selection [1]. Applied to altruistic traits, inclusive fitness theory explains that an altruist gene is selected for if it is altruistic (assists another at a cost to itself) towards relatives when the cost of altruism is less than its benefit diluted by the chance that the beneficiary does not have the altruist gene [1]. In its more general form, inclusive fitness theory holds that any gene that directs a net benefit towards other copies of itself will be favoured by selection, even if the altruistic and beneficiary genes do not share common descent [1–7]. Altruist genes can, with varying degrees of reliability, identify carriers of the altruism gene in nature in three ways: (i) by recognizing kin, who are likely to share the altruist gene, (ii) in viscous populations, where surrounding organisms are often related, and (iii) by directly sensing the presence of the altruist gene [8].

Putting the point anthropomorphically, in order to determine whether to target altruism towards an organism (i.e. select it as the beneficiary of altruism), an altruist gene would like to have *perfect* genetic information as to whether a copy of itself exists in that organism. Given imperfect information, however, altruist genes are forced to settle on less accurate targeting mechanisms. *Accuracy* in this context is the probability that the recipient of altruism has a copy of the altruistic gene. Targeting altruism based on identifying kinship, which we call *kin targeting*, is the most commonly used indicator of the presence or absence of an altruist gene [9,10]. Altruism via kin recognition can be evolutionarily stable, although only in certain situations ([11–13]; reviewed in [14]). However, if more accurate indicators of the presence of altruistic genes were available, inclusive fitness theory predicts that natural selection should use them in addition to, or in lieu of, kinship indicators [1,4,5,7,15,16]. This pressure to be as accurate as possible is strongest when donations are costly or otherwise limited, which is the case we focus on in this paper.

Aside from kinship, one possible mechanism for determining whether an organism has a copy of an altruistic gene is to sense the *genetic similarity* between the potential donor and the recipient, which we will call *similarity targeting*. One may envision mechanisms based on sensing biochemical signals that would allow the inference of genetic similarity irrespective of kinship. For instance, mice, fishes and humans use scent to preferentially choose mates with genetically dissimilar major histocompatibility complexes, suggesting that genetic information can be

* Author for correspondence (jclune@msu.edu).

Electronic supplementary material is available at http://dx.doi.org/10.1098/rspb.2010.1557 or via http://rspb.royalsocietypublishing.org.

*Received* 21 July 2010
*Accepted* 24 August 2010

acquired and exploited by evolution [17]. Additionally, social amoebas are more likely to form cooperative relationships with genetically similar organisms [18]. If it were possible for organisms to target altruism based on high genetic similarity (greater than expected for kin), inclusive fitness theory predicts that this mechanism would be selected for over kin targeting because it is more accurate.

It is also possible to have altruism-targeting mechanisms that would be even more accurate than those based on high genetic similarity. The most accurate targeting mechanism would be a gene that can identify with certainty whether other organisms possess (and express) its copy, irrespective of kinship or overall genetic similarity. This is the idea behind green beard genes, which were proposed by Hamilton [1], named by Dawkins [4] and subsequently found in nature [19–27]. Green beard genes do two things: (i) display a marker (e.g. a green beard) and (ii) target altruism towards entities bearing that marker and no others. Green beard genes are the ideal implementation of inclusive fitness theory—they direct altruism only towards organisms that contain and express their copies. We refer to this strategy as *green beard targeting*.

A prediction, then, of inclusive fitness theory is that if more accurate altruism-targeting methods become available, selection will favour their use over less accurate targeting methods. Specifically, inclusive fitness theory predicts that organisms will use kin targeting if it is the only type of altruism targeting available. If genes for both kin targeting and similarity targeting exist in a population, and if the similarity targeting is more accurate, then similarity targeting should have a selective advantage over kin targeting. Finally, if the genes for kin targeting, similarity targeting and green beard targeting all exist in a population, and are mutually exclusive, selection will favour green beard targeting because it is the most accurate. We confirm all of these predictions in this paper. Interestingly, however, we had to implement a novel instantiation of green beard targeting before natural selection favoured it over similarity targeting. Natural selection did not favour green beard targeting over similarity targeting in most situations because green beard targeting by itself cannot evolve the *amount* of altruism conferred from the altruist to the recipient. Except in rare, contrived situations, natural selection switched away from similarity targeting only when different beard colours existed that each reliably indicated different levels of altruism (a mechanism we call *identical beard colour targeting*).

To our knowledge, a test of these predictions has not been conducted either in computer models or in natural systems. While it would be ideal to confirm these predictions in natural systems, such tests would be difficult, if not impossible, to perform. As such, the closest we may come to empirically testing these predictions of inclusive fitness theory is in digital evolution systems.

We conduct such tests in Avida, a digital evolution software platform that instantiates evolutionary processes in a computer [28,29]. Avida has repeatedly served as a tractable system to investigate the general properties of evolving systems [30–37].

## 2. METHODS

In Avida, self-replicating computer programs (i.e. *digital organisms*) evolve through random mutations and selective pressures. To self-replicate, an organism must copy its own *genome*, which is a sequence of computer instructions. The copy process is imperfect, however, such that a genomic instruction has a probability of mutating to another instruction when copied. These mutations can alter the execution of the genome and change its behaviour. When an organism self-replicates, a copy of it is placed at random either in one of its parents' cells or in a neighbouring cell of a parent (replacing the resident organism if extant). There are a limited number of cells, creating a competition for space. In the experiments described here, digital organisms obtain extra *metabolic units* through altruistic donations (explained subsequently), allowing them to execute their genomes more rapidly, which increases their ability to compete for space and thus their fitness.

As Dennett [38] has noted, evolution will occur in any system that has heritable variation and differential fitness. Avida exhibits these traits and can therefore be used to study the general principles of evolving systems [33]. The remainder of §2 describes how Avida was configured for the experiments described in this paper. A general, more thorough description of Avida can be found in [28]. The Avida software is free and can be obtained from http://devolab.cse.msu.edu/software/avida.

Each experiment featured 50 trials that differed only in the seed for the random number generator, which affected stochastic aspects of the trial, such as mutations. Trials began by filling 3600 cells of a virtual toroidal grid with identical copies of an organism that could self-replicate, but exhibited no other behaviours. Each cell within the toroidal grid was adjacent to eight *neighbouring* cells. When an organism successfully executed a `divide` command, it reproduced with another organism that had successfully divided. Specifically, offspring resulted from the sexual recombination of two copied genomes using two-point crossover, where both points were randomly chosen from each organism's circular genome and the sections of the genome between each point were swapped [36]. During replication, each instruction in the genome had a 0.75 per cent chance of mutating to any other instruction in the instruction set. Lower mutation rates of $10^{-3}$, $10^{-4}$ and $10^{-5}$ (following Rousset & Roze [13]) did not qualitatively change our results, except evolution was slower and similarity targeting was less competitive versus kin targeting because organisms were more similar, making similarity less informative. The standard Avida instruction set [28] includes instructions that allow organisms to manipulate numbers, copy their instructions, and modify execution flow (e.g. jump to, or skip over, instructions in their genomes). All genomes were fixed at a length of 100 instructions. Organisms died of 'old age' and were removed from the population if they executed 2000 instructions prior to completing replication. Organisms started their lives with 100 metabolic units.

For these experiments, we extended Avida to include altruistic instructions that allowed organisms to donate their metabolic units to neighbouring organisms. Organisms lost five metabolic units per donation made and gained 50 per donation received. This asymmetry created the possibility of non-zero sum gains, which are necessary for the evolution of altruism. Altering the ratio or magnitude of cost and benefit did not qualitatively change the results of the experiments, provided the benefit was at least approximately three times the cost. For all experiments, the number of donations of any type that an organism could

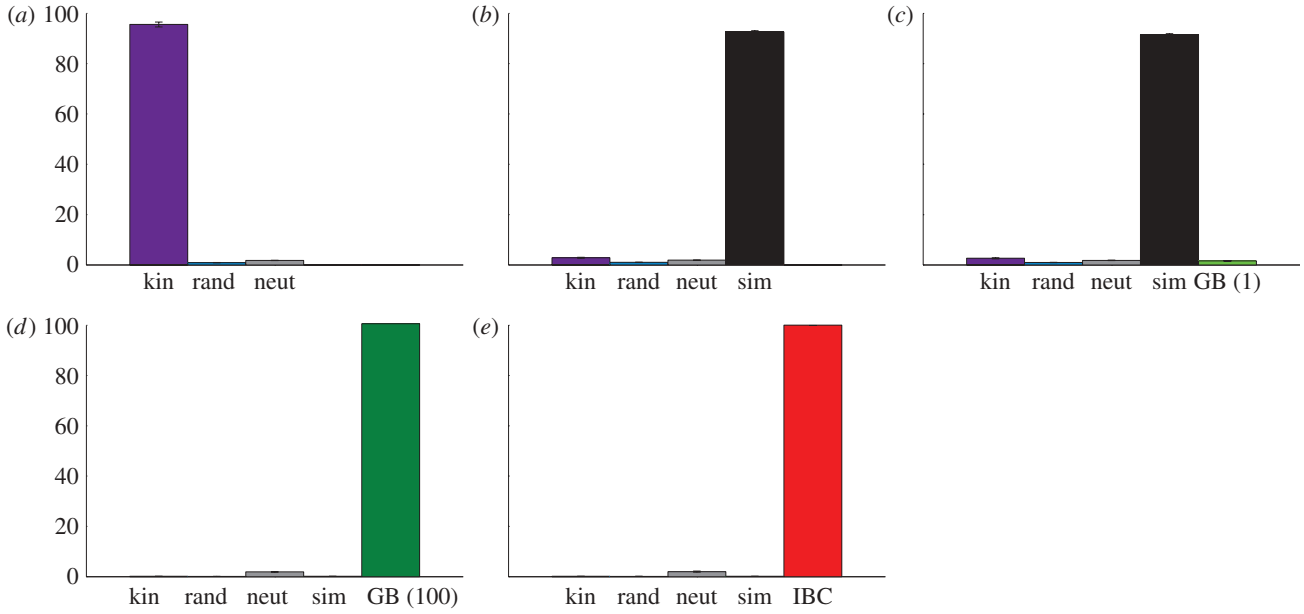


Figure 1. Evolved altruism levels for different targeting mechanisms. Plotted is the average number of donation instructions per type executed by organisms in the final populations of 50 trials ($\pm 1$ s.e., often too small to distinguish). The maximum number of donations was capped at 100. (*a*) Targeting altruism based on kinship was selected for over two controls. (*b*) Targeting altruism based on high genetic similarity was favoured over targeting based on kinship. (*c*) Selection did not favour targeting altruism via a green beard mechanism (with an implicit threshold of 1, see text) over kin and similarity targeting. (*d*) Selection favoured a green beard mechanism with a threshold of 100 (the maximum number of donations allowed) over kin and similarity targeting. (*e*) Selection favoured identical beard colour targeting over kin and similarity targeting. Purple, kin; blue, random (rand); grey, neutral (neut); black, similarity (85%); light green, green beard (GB) (1); dark green, green beard (100); red, identical beard colour (IBC).

perform was capped at 100 (by ignoring subsequent attempts), which makes it easier to determine which donation strategies are the most effective in any given setup. If an organism executed a donation instruction but there was not a suitable recipient in the surrounding eight cells, the organism was credited with performing the donation, but no energy transfer or deduction took place.

## 3. RESULTS

To test the prediction that natural selection would favour the most accurate altruism-targeting mechanism available, we ran a series of experiments with different instructions that enable organisms to altruistically donate metabolic units using different targeting mechanisms.

### (a) *Kin targeting*

We initially wanted to confirm that kin targeting would evolve in Avida. For this first experiment, we added three instructions to the standard Avida instruction set [33]. When an organism executed `donate-kin`, it made a donation to a neighbouring parent or offspring. We excluded full siblings for simplicity (they are extremely rare because mates are chosen randomly from the population, meaning that it is unlikely that two organisms will have the same two parents). The other two additional instructions were controls: executing `donate-random` donated to a random neighbour, and executing `neutral` had no altruistic effect. The latter provided a baseline of how often instructions with no selective advantage or disadvantage were executed. These three instructions were not present in the genome of the starting organism. Because individual organisms can execute multiple donation instructions, we estimate the evolutionary success of a donation instruction by looking at how many times it is executed by the organisms in the final population of each experiment, which is analogous to the expression level for a gene. Expression levels significantly above those for `neutral` are evidence that an instruction has been selected for. We also report on the frequency of the various donation instructions (alleles) in the populations at the end of evolutionary trials. Those alleles that are significantly more frequent in final populations than the `neutral` control have probably been selected for.

The results show that organisms that donated to their kin were selected for (figure 1*a*, $p < 0.001$ comparing the average number of `donate-kin` executions versus `donate-random` and `neutral` in final populations; all *p*-values in this paper are generated using Matlab's Mann–Whitney test). The average number of donations per organism is nearly the maximum number of donations allowed (100). The average number of `donate-kin` instructions per genome in the final populations was 7.95 ($\pm 0.58$ s.d.), which was significantly higher than both the `donate-random` ($2.13 \pm 0.22$ s.d.) and `neutral` ($2.72 \pm 0.25$ s.d.) controls ($p < 0.001$). These results confirm that kin targeting is selected for in Avida, as predicted by inclusive fitness theory. Changing which relatives were considered kin (e.g. including cousins) did not qualitatively change the results (data not shown). By repeating this experiment without the `donate-kin` instruction, we also found that indiscriminate altruism is selected against in our experimental setup (`donate-random` expression is significantly lower than `neutral`, $p < 0.001$), although this might

not have been expected because the population is viscous [39]. The increased accuracy of kin targeting over indiscriminate altruism thus enables the elevated levels of altruism observed with `donate-kin`.

### (b) *Similarity targeting*

We next investigated whether selection favours altruism-targeting mechanisms that are more accurate than those based on kinship. To test this prediction, we added a `donate-similar` instruction to the instruction set. When executed, the `donate-similar` instruction donated to a neighbour that had an edit distance of fewer than 15 (i.e. 85% genetically similar), where *edit distance* is the number of point, insert and/or delete mutations needed to transform one genome into another [40]. We selected this edit distance of 15 so that `donate-similar` more accurately targets altruism than `donate-kin` (otherwise there would be no expected selection for the former over the latter). Specifically, at equilibrium in the previous experiment, 34.69 per cent of donations using `donate-kin` went to organisms that had an edit distance greater than 15 ($\pm$0.0033% s.d.). The average edit distance between the donor and the recipient for `donate-kin` was 14.44 ($\pm$0.12 s.d.), whereas it was 7.48 ($\pm$0.05 s.d.) for `donate-similar`, which is significantly different ($p < 0.001$).

In accordance with the prediction of inclusive fitness theory, selection did favour this greater accuracy in altruism targeting (figure 1*b*, $p < 0.001$ comparing the average number of `donate-similar` executions versus other donation types in final populations). On average, `donate-similar` was executed 90 times per organism, whereas `donate-kin` was executed fewer than 10. The genomic instruction frequencies are consistent with the expression levels: The average number of `donate-similar` instructions per genome in the final populations was 5.1 ($\pm$0.45 s.d.), which was significantly higher ($p < 0.001$) than `donate-kin` (3.16 $\pm$ 0.32 s.d.), `donate-random` (2.15 $\pm$ 0.26 s.d.) and `neutral` (2.80 $\pm$ 0.24 s.d.). Varying the similarity (edit distance) threshold did not qualitatively change the result as long as it remained below approximately 50, i.e. above about 50 per cent genetic similarity (data not shown).

### (c) *Green beard targeting*

In the next experiment, we provided direct knowledge of the presence of expressed altruist genes by adding a `donate-greenbeard` instruction. When executed, it caused a donation to a neighbouring organism that (i) had `donate-greenbeard` in its genome and (ii) executed the `donate-greenbeard` instruction at least once. The latter condition was added to prevent an organism from having a green beard phenotypic marker (i.e. having `donate-greenbeard` in its genome), but not being altruistic because the instruction was included in a 'junk' section of the genome that was never executed. We determined whether an organism executed the `donate-greenbeard` instruction by testing each new organism in a separate test environment prior to placing it in the population.

At first pass, it seems that inclusive fitness theory predicts that natural selection should favour the use of `donate-greenbeard` over both `donate-similar` and `donate-kin` because `donate-greenbeard` is perfectly accurate: it donates only to other green beard donors. In contrast, `donate-similar` and `donate-kin` will sometimes donate to non-donors (a false-positive error), as well as fail to donate to others that share their altruism genes (a false-negative error). For example, an organism may donate to kin that did not receive the altruism gene, or an organism may fail to recognize and donate to a cousin that shares the altruism gene. To test the prediction that altruism using the green beard-targeting mechanism will be selected for over kin and similarity altruism targeting, we repeated the previous experiment with the addition of `donate-greenbeard` to the instruction set.

Contrary to our initial expectations, `donate-greenbeard` was not competitive with `donate-similar` (figure 1*c*). The `donate-similar` instruction was selected for over all other donation types ($p < 0.001$ comparing the average number of `donate-similar` executions versus other donation types in final populations). The genomic instruction frequency data also show that selection favoured similarity targeting over all of the alternatives, including green beard targeting: The average number of `donate-similar` instructions per genome in the final populations was 4.90 ($\pm$0.46 s.d.), which was significantly higher ($p < 0.001$) than `donate-greenbeard` (2.60 $\pm$ 0.25 s.d.), `donate-kin` (3.11 $\pm$ 0.33 s.d.), `donate-random` (2.03 $\pm$ 0.22 s.d.) and `neutral` (2.63 $\pm$ 0.30 s.d.).

Later in the paper, we demonstrate a variant of the green beard concept that does outcompete `donate-similar`, but it is first instructive to learn why `donate-greenbeard` was not selected for. A possible explanation for this result is the disincentive an organism has for performing more than the minimum number of green beard donations necessary to receive green beard donations, which was only 1 in this case. For instance, in a population where all organisms perform two green beard donations, an organism that donates only once would receive more than it donates and thus have a competitive advantage. Such an organism is a variant of a 'falsebeard' cheater because it has the altruism-signifying marker, but is not altruistic to the same degree [19]. We hypothesized that green beard organisms are under selective pressure to be as selfish as possible while being just altruistic enough to qualify to receive donations from other green beard donors.

We tested this hypothesis by creating a *threshold*, which is the number of green beard donations an organism needs to make in order to qualify to receive green beard donations. The original `donate-greenbeard` instruction has an implicit threshold of 1, but this requirement can be set to any value. If our hypothesis is correct, the amount of green beard altruism should rise as a function of the threshold ($T$), but should not rise far above $T$; values slightly above $T$ are expected owing to a pressure for robustness under mutation–selection balance [32].

We tested this hypothesis by repeating the previous experiment, but using only the default instruction set and a new `donate-threshold-gb` instruction, which is identical to `donate-greenbeard`, but with a threshold that can be set to any integer. We tested four threshold values ($T = 1$, 25, 50, 100) and the hypothesis was confirmed: the level of altruism rose to the threshold, but did not substantially exceed it (figure 2). These results indicate that green beard targeting is unable to evolve persistent levels of altruism above whatever fixed and arbitrary threshold of altruism is required to qualify

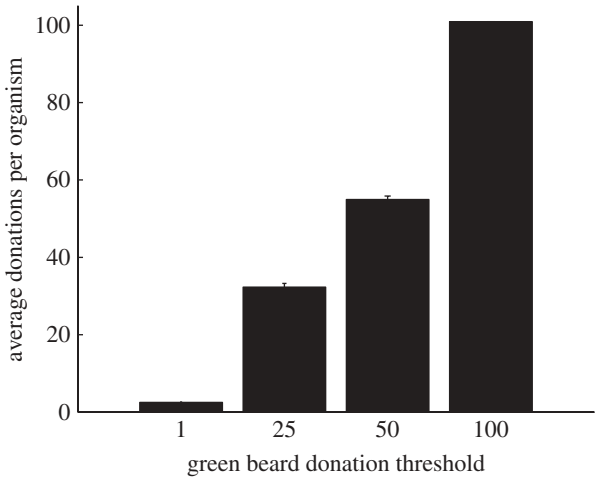


Figure 2. Evolved altruism levels for different green beard thresholds. Plotted is the average number of donations executed per organism for different threshold values ($T$) of the `donate-threshold-gb` instruction (averaged from the final populations of 50 trials per treatment $\pm 1$ s.e., often too small to distinguish). Organisms evolved to perform enough donations to surpass the threshold and thus qualify to receive altruism, but did not perform substantially more than $T$ donations.

for green beard donations. The inability to evolve sustained altruism levels above level $T$ provides a plausible explanation for why green beard targeting did not outcompete similarity targeting in the experiment plotted in figure 1*c*. In that experiment, the threshold of 1 resulted in a low level of green beard altruism, which prevented selection from taking advantage of the greater accuracy of the green beard-targeting mechanism in all but a few donation opportunities. Natural selection took advantage of the remaining non-zero sum opportunities with similarity targeting.

These results suggest that in order for an altruist gene to accurately identify another gene as its copy, the *level* of altruism must also be taken into account. This is because two altruist genes that are altruistic to different degrees are not copies of one another. Previously in this paper, we have discussed altruism-targeting mechanisms solely based on a targeting *method*, such as targeting based on kinship, shared genetic similarity or the mutual possession of a green beard. In addition to targeting methods, however, targeting mechanisms can also have a *discrimination level*, which discriminates based on the level of altruism (electronic supplementary material, figure S1). Targeting methods and discrimination levels can work in conjunction to filter out the subset of organisms that will receive altruism. In our experiments, `donate-kin` and `donate-similar` had no discrimination level. Green beard targeting has a discrimination level equal to its threshold: the instruction `donate-greenbeard` had an implicit discrimination level of 1 and `donate-threshold-gb` had a discrimination level equal to its threshold $T$. The reason inclusive fitness theory, at first pass, seemed to predict that `donate-greenbeard` would be favoured over `donate-kin` and `donate-similar` is because the altruism level was not being considered when green beard genes were labelled as perfectly accurate. Recognizing the importance of altruism levels reveals that `donate-greenbeard` is not perfectly accurate because it frequently donates to organisms with more selfish versions of its altruist gene.

In our original green beard experiment (figure 1*c*), even though kin- and similarity-targeting mechanisms were less accurate, they were employed to take advantage of the remaining donation opportunities. We hypothesized that if the green beard threshold were set to the maximum number of donations allowed, then selection would indeed use the green beard-targeting mechanism instead of similarity targeting owing to its greater accuracy. To test this idea, we used a green beard threshold of 100, which is approximately the level of altruism generated by kin and similarity targeting when they are dominant (figure 1*a*,*b*). The experiment reported in figure 1*c* was repeated, but using `donate-threshold-gb` ($T = 100$) instead of `donate-greenbeard`.

Our prediction was confirmed: selection employed the green beard-targeting mechanism significantly more than kin and similarity targeting (figure 1*d*, $p < 0.001$ comparing the average number of `donate-greenbeard` executions versus other donation types in final populations, and comparing the average number of `donate-threshold-gb` instructions per genome in the final populations ($5.61 \pm 0.54$ s.d.) versus `donate-similar` ($1.46 \pm 0.20$ s.d.), `donate-kin` 1.47 ($\pm 0.19$ s.d.), `donate-random` ($1.40 \pm 0.19$ s.d.) and `neutral` ($2.94 \pm 0.30$ s.d.)). Our results demonstrate that inclusive fitness theory is right that selection will favour the most accurate altruism-targeting mechanism available, but only if the altruism level is controlled for.

The green beard-targeting mechanism can evolve an altruism level near the maximum only if its discrimination level (threshold) is arbitrarily set to be near the maximum. It cannot evolve an altruism level near the maximum if its discrimination level happens to be much lower (figure 1*c*). By contrast, the altruism level for kin and similarity targeting automatically increased from zero to close to the maximum, exploiting nearly all of the non-zero-sum donation opportunities (figure 1*a*,*b*). These results reveal that kin and similarity targeting can evolve ever-higher altruism levels without explicit discrimination levels. One reason a discrimination level is unnecessary is that the organisms they cooperate with (i.e. kin or genetically similar organisms) have similar genomes and thus are likely to have similar altruism levels. Furthermore, organisms using kin and similarity targeting cooperate only with a small group of other organisms, limiting the success of 'kin-cheaters', which are organisms within a kin group that mutate to be less altruistic (figure 3*a*–*c*) [35,41]. Additionally, if a new kin group is created with a higher altruism level, it needs to survive via drift for only a few generations before some of its members are no longer close enough kin to donate to their less altruistic ancestors, and they can thus successfully evade exploitation (figure 3*c*,*d*). These attributes of kin and similarity targeting make possible the evolution of persistent increases in altruism. In other words, kin and similarity targeting do not have an explicit discrimination level, but instead possess an implicit discrimination level that occurs as a side effect of the genetic similarity inherent in these targeting methods.

The previous experiment demonstrates that if a green beard-targeting mechanism *happens* to have a

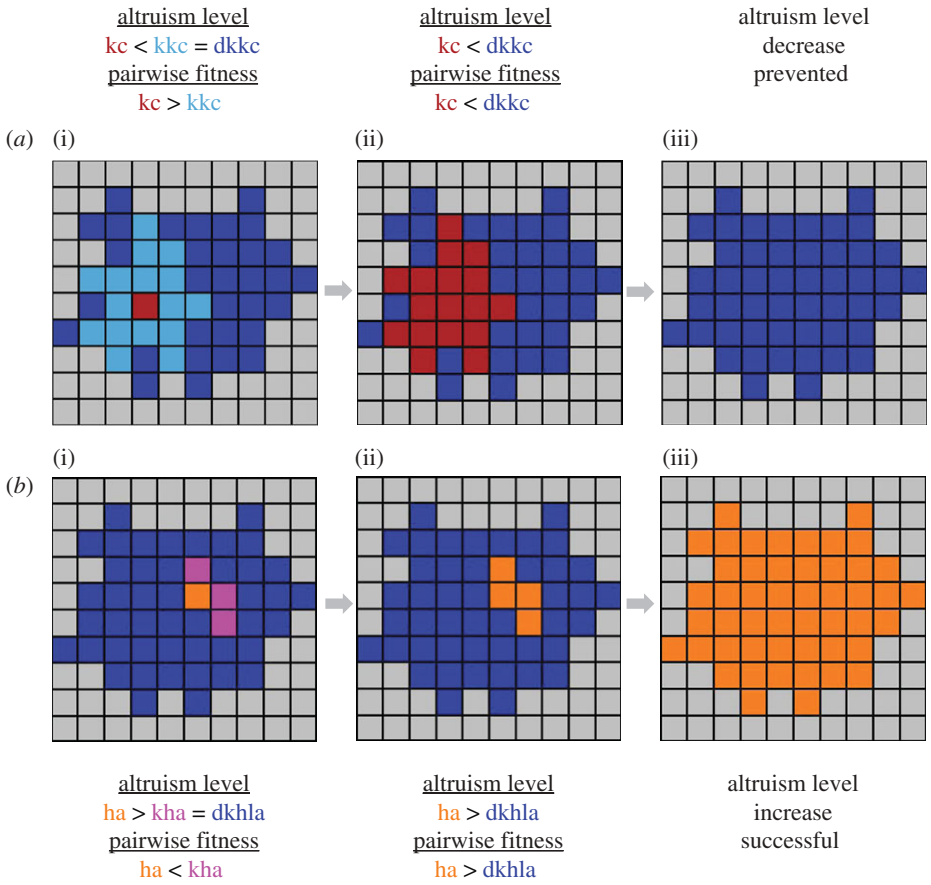


Figure 3. How kin and similarity targeting can evolve persistently high altruism levels. A thought experiment illustration showing how (*a*) kin-based altruism naturally thwarts kin-cheaters (kc) and (*b*) enables enduring increases in altruism levels. (*a*(i)) Consider a group of related organisms that are altruistic to each other (blue and light blue). One organism may mutate to be less altruistic, becoming a kin-cheater (red), but since only its closest relatives (light blue) will consider it kin, only they will be altruistic towards it. (*a*(ii)) The kin-cheater will tend to supplant its kin because it receives more donations from them than it gives. (*a*(iii)) Once the kin-cheater has replaced those that considered it kin, the kin-cheater is left receiving donations only from other kin-cheaters. This group (red) will have a lower altruism level than their distant kin (blue) and will come to be replaced by them. (*b*(i)) Now consider an organism (orange) that mutates to have a higher level of altruism (ha) than its ancestors (blue). Initially, it will be selected against because it gives more donations to those that it considers kin (pink) than it receives from them. (*b*(ii)) If the less-altruistic kin of the higher level altruist are killed off by drift, then the higher level altruist and its offspring (orange) will have a competitive advantage over their distant ancestors (blue). (*b*(iii)) While chance is required to start the process, once it has occurred, there will be selection for the higher level of altruism. There are additional factors that complicate all of these fitness comparisons, but for clarity, we have sketched these scenarios only in broad strokes. kc, kin cheater; kkc, kin of kin cheater; dkkc, distant kin of kin cheater; ha, higher-level altruist; kha, kin of higher-level altruist; dkhla, distant kin of higher-level altruist.

discrimination level near the optimum (here, the maximum number of opportunities to exploit non-zero sum gains), then the green beard mechanism will be favoured by natural selection over kin and similarity targeting (figure 1*d*). However, it is unlikely that random mutations would produce a green beard gene with a near-optimal discrimination level de novo. This small probability is compounded by the already unlikely prospect of random mutations producing a gene that both creates a phenotypic marker and targets altruism towards bearers of the marker [4]. These improbable requirements diminish the expectations of finding such a gene in natural settings. Furthermore, if the optimal altruism level changed over time, the descendents of a bearer of a green beard gene that happened to be near the optimum would no longer be optimal, and would probably be replaced by a kin- or similarity-targeting mechanism. An unchangeable discrimination level, therefore, appears to be an evolutionary disadvantage.

### (d) *Identical beard colour targeting*

If a green beard-targeting mechanism were able to automatically optimize its altruism level across generations, then such adaptability could make the green beard-targeting mechanism more competitive with kin- and similarity-targeting mechanisms. This type of adaptability is possible with a slight variation on the green beard idea wherein mutations to the altruism gene simultaneously change the level of altruism, the phenotypic marker (e.g. beard colour) and the level of discrimination. We call this targeting mechanism *identical beard colour targeting*, because organisms carrying the gene would be altruistic only towards others bearing the same beard colour. There would thus be many beard colours in a population, and the beard colour would perfectly indicate the altruism level of its bearer. This proposal is different from previous work with multiple beard colours in a population where the beard colour did not reliably indicate the altruism level [42]. We tested the efficacy

of such an identical beard colour targeting gene by creating a `donate-identical-beard-colour` instruction. This instruction, when executed, donated to another organism that both: (i) had the `donate-identical-beard-colour` instruction and (ii) donated the same number of times as the donor. Thus, the phenotypic marker accurately signifies both the altruism and discrimination levels.

We repeated the previous experiment, but substituted the `donate-identical-beard-colour` instruction for the `donate-greenbeard-threshold` instruction, and found that identical beard colour targeting was selected for over kin and similarity targeting (figure 1*e*, $p < 0.001$ comparing the average number of `donate-identical-beard-colour` executions versus other donation types in final populations), which confirms our prediction. The genomic instruction frequency data also confirmed that selection favoured identical beard colour targeting over all of the alternative targeting mechanisms. The average number of `donate-identical-beard-colour` instructions per genome in the final populations was 3.61 ($\pm$1.60 s.d.), which was significantly higher ($p < 0.001$) than `donate-similar` (1.50 $\pm$ 0.22 s.d.), `donate-kin` (1.46 $\pm$ 0.21 s.d.) and `donate-random` (1.44 $\pm$ 0.21 s.d.). In this experiment, the difference in genomic instruction frequency between `donate-identical-beard-colour` and `neutral` (3.01 $\pm$ 0.21 s.d.) was not significant ($p > 0.05$), but the significant difference in the level of execution of those instructions (figure 1*e*, $p < 0.001$) clearly shows that selection favoured a much higher expression of `donate-identical-beard-colour` via regulatory instructions. We also performed this experiment with a population structure where offspring are placed at random in the population and found that the level of `donate-identical-beard-colour` expression was significantly higher than all other donation types ($p < 0.001$).

Interestingly, the average edit distance between the donor and the recipient for `donate-identical-beard-colour` was 52.78 ($\pm$0.15 s.d.), which was significantly greater than the edit distances for either `donate-kin` (14.44 $\pm$ 0.12 s.d.) or `donate-similar` (7.48 $\pm$ 0.05 s.d.), indicating that the identical beard colour targeting mechanism did indeed find altruism recipients that kin- and similarity-targeting mechanisms would not have identified. The average number of identical beard colour donations per organism in the final population was near the maximum amount allowed, as was the case in previous experiments for the kin-, similarity- and (threshold) green beard-targeting mechanisms when they were dominant. This high average means that most of the organisms in the population donated nearly 100 times, which was the maximum allowed. A look at altruism levels across evolutionary time reveals that this high level of altruism was evolutionarily stable in the sense that it was maintained for thousands of generations (figure 4). Plots of altruism levels across evolutionary time look qualitatively similar from the previous experiments when other targeting mechanisms were dominant (data not shown). These time plots reveal that the altruism via the targeting mechanisms discussed in this paper was maintained at high levels across thousands of generations.

Another alternate way altruism levels could be adjusted via a green beard mechanism is with multiple, fixed-threshold green beard genes with different markers (e.g. a blue hand, a red foot etc.). While each has a fixed threshold, the overall organismal level of altruism could be adjusted by having as many of these genes as necessary. We implemented this concept, and it worked qualitatively the same as identical beard colour targeting in the previous experiments, and evolution probabilistically chose one or the other when both options were available (electronic supplementary material). This alternate implementation further shows that being able to adjust altruism levels is a key fitness component of the green beard mechanism.

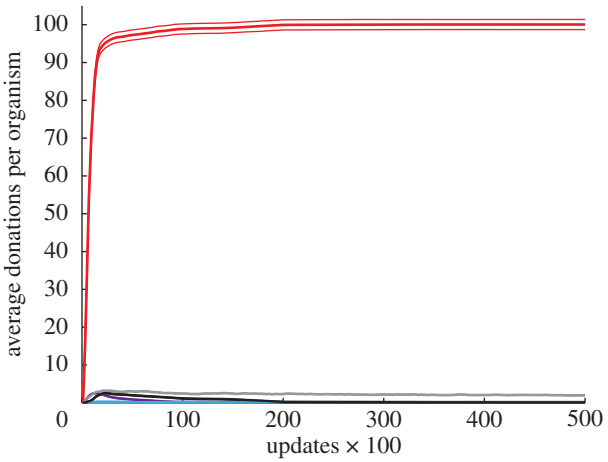


Figure 4. Identical beard colour altruism levels over evolutionary time. The data from the experiment plotted in figure 1*e* are shown here across evolutionary time. Plotted is the average number of donation instructions per type executed per organism for each update. The plot shows averages over 50 evolutionary trials ($\pm$1 s.e.). The altruism level of identical beard colour targeting rises early and remains at a high level for most of the experiment, which lasted thousands of generations. Purple, kin; light blue, random; grey, neutral; black, similarity (85%); red, identical beard colour.

## 4. DISCUSSION

It is commonly assumed that kin-based altruism involves poor, but adequate estimations of the presence of an altruist gene and that such altruism would be more effective if it were based on true knowledge of the presence of that gene [1,4,5,15]. In this paper, we provided organisms with such perfect knowledge via green beard targeting with a fixed and low discrimination level, but it was not selected for over altruism targeting based on 'imperfect' kin information. We discovered that this unexpected outcome is because kin targeting can naturally adjust its level of altruism, whereas green beard targeting using a single phenotypic marker (e.g. a single beard colour) cannot. This important feature of kin-based altruism may help explain its widespread occurrence in nature [9,10] versus the rare examples of green beard altruism [15,20–26]. It is interesting to note that the documented cases of green beard targeting in nature do involve binary decisions, for example, to kill or not [21] or to bind to or not [23,26].

Our results support that green beard targeting will be the method of choice only if the decision is binary, consisting of whether to be altruistic or not, or if the discrimination level of a green beard gene happens to be

near the optimal altruism level. In such cases, green beard targeting is indeed more accurate than kin or similarity targeting. However, when adjusting the level of altruism is advantageous, selection will favour kin targeting (and would favour similarity targeting were it possible) over green beard targeting. Green beard targeting is not favoured by selection in such situations because a gene with a single beard colour cannot evolve its discrimination level threshold. However, as we demonstrated with the identical beard colour targeting mechanism, if mutations can simultaneously change the phenotypic marker, the altruism level and the discrimination level, then the amount of altruism can be continuously optimized by evolution. While kin and similarity targeting have an advantage over green beard targeting in adaptability, they do not have this advantage over identical beard colour targeting. It is not surprising, therefore, that identical beard colour targeting, which is more accurate and as adaptive as kin and similarity targeting, was selected for over these alternatives in our experiments.

Jansen & van Baalen [42] found that non-zero altruism levels could be maintained with multiple beard colours, but only when altruism levels were averaged across a population. In their model, beard colours and altruism traits were unlinked, allowing cheaters to easily appear and replace any subpopulation of altruists that temporarily emerged. Frequent altruist evolution and extinction meant that at any given time it was likely there were some altruist organisms in the population. Jansen & van Baalen conclude that altruism levels become increasingly unstable as the linkage increases between the phenotypic marker (in their case, beard colour) and altruism level. By contrast, such linkage is perfect in our identical beard colour mechanism, but we find that altruism levels quickly evolve to be maximally high and persist at that high level across evolutionary time (figure 4). We believe the phenomena Jansen & van Baalen describe are not due to green beard dynamics, but instead resemble the dynamics of kin selection, wherein altruism is maintained by the constant formation of groups that have not yet been invaded by cheaters (figure 3). In this situation, altruism can be maintained at a population level only if new cheater-free groups are constantly being created. Such kin-based altruism could not be maintained in a well-mixed population structure, yet such a population structure should not preclude high levels of green beard altruism (and did not in our experiments with the identical beard colour mechanism). The stability of altruism levels reported by Jansen & van Baalen is thus a property of a constantly changing subset of the population, but no altruist genotype can persist across evolutionary time because it can be invaded once a cheater emerges. In contrast, the identical beard colour mechanism can produce genotypes that exhibit high levels of altruism and resist invasion indefinitely (figure 4).

Although we have shown that identical beard colour targeting can maintain high levels of altruism, it is unlikely that natural, biological organisms use it as an altruism-targeting mechanism. We consider it improbable because identical beard colour targeting requires an even more unlikely phenomenon to be caused by a single gene than is the case for green beard genes. In addition to the requirements of green beard targeting, the gene must respond to mutations in such a way that new beard colours are created simultaneously with changes in the bearer's altruism level and discrimination level. That said, green beard genes were not thought to exist when they were invented as thought experiments [4]. Another possible green beard mechanism involves a collection of different green beard genes, each with its own marker, that independently contribute a fixed level of altruism. The costs, benefits and existence in nature of this strategy remain interesting open areas of research.

## 5. CONCLUSION

The main contribution of this paper is to provide empirical verification of the prediction of inclusive fitness theory that natural selection will favour altruism-targeting strategies that are increasingly accurate in targeting copies of the altruist gene. Additionally, our experimental results underscore that the altruism level of a gene is an important aspect of an altruist gene. It has not been common in the literature to discuss *how* altruistic a green beard gene is, but this level of altruism is a key attribute when green beard-targeting mechanisms compete with kin- and similarity-targeting mechanisms. A further issue highlighted in the paper is the importance of the adaptability, or evolvability, of the altruism levels of different targeting mechanisms. The evolvability of altruism-targeting mechanisms plays a significant role in determining which ones will succeed in evolving populations. Finally, in this paper, we demonstrate that a variant on the green beard-targeting concept, identical beard colour targeting, provides both perfect accuracy and adaptability. When identical beard colour targeting is available, natural selection favours it over its less accurate rivals. However, the unlikelihood of an identical beard colour targeting mechanism arising by chance in nature is sufficiently high that it is likely to remain confined to the realms of thought experiments and *in silico* evolution.

Kevin Foster and Andy Gardner provided helpful reviews, including Andy Gardner's idea of multiple, fixed-threshold green beard genes with different markers. We also thank Elizabeth Ostrowski, Phil McKinley and the members of the digital evolution laboratory at Michigan State University. This work was supported by NSF grants CCF-0643952, CNS-0751155 and CNS-0915885, by the Cambridge Templeton Consortium Emerging Intelligence grant, by a Michigan State University Dean's Recruitment Fellowship to J.C. and by a fellowship to J.C. from the Quantitative Biology and Modelling Initiative at Michigan State University.

## REFERENCES

1 Hamilton, W. D. 1964 The genetical evolution of social behaviour. *I-II. J. Theor. Biol.* **7**, 1–52. (doi:10.1016/0022-5193(64)90038-4)
2 Hamilton, W. D. 1963 The evolution of altruistic behavior. *Am. Nat.* **97**, 354–356. (doi:10.1086/497114)
3 Maynard Smith, J. 1964 Group selection and kin selection. *Nature* **201**, 1145–1147. (doi:10.1038/201145a0)
4 Dawkins, R. 1976 *The selfish gene*. Oxford, UK: Oxford University Press.
5 Dawkins, R. 1982 *The extended phenotype*. Oxford, UK: Oxford University Press.

6 Foster, K. R., Wenseleers, T. & Ratnieks, F. L. W. 2006 Kin selection is the key to altruism. *Trends Ecol. Evol.* **21**, 57–60. (doi:10.1016/j.tree.2005.11.020)
7 Lehmann, L. & Keller, L. 2006 The evolution of cooperation and altruism—a general framework and a classification of models. *J. Evol. Biol.* **19**, 1365–1376. (doi:10.1111/j.1420-9101.2006.01119.x)
8 West, S. A., Griffin, A. S. & Gardner, A. 2007 Evolutionary explanations for cooperation. *Curr. Biol.* **17**, R661–R672. (doi:10.1016/j.cub.2007.06.004)
9 Hepper, P. G. 1991 *Kin recognition*. New York, NY: Cambridge University Press.
10 Holmes, W. 2004 The early history of Hamiltonian-based kin recognition research theory. Past and future. *Ann. Zool. Fennici* **41**, 691–711.
11 Crozier, R. H. 1986 Genetic clonal recognition abilities in marine invertebrates must be maintained by selection for something else. *Evolution* **40**, 1100–1101. (doi:10.2307/2408769)
12 Grafen, A. 1990 Do animals really recognize kin? *Anim. Behav.* **39**, 42–54. (doi:10.1016/S0003-3472(05)80724-9)
13 Rousset, F. & Roze, D. 2007 Constraints on the origin and maintenance of genetic kin recognition. *Evolution* **61**, 2320–2330. (doi:10.1111/j.1558-5646.2007.00191.x)
14 Gardner, A. & West, S. A. 2007 Social evolution: the decline and fall of genetic kin recognition. *Curr. Biol.* **17**, R810–R812. (doi:10.1016/j.cub.2007.07.030)
15 Crespi, B. & Springer, S. 2003 Ecology: social slime molds meet their match. *Science* **299**, 56–57. (doi:10.1126/science.1080776)
16 West, S. A., Griffin, A. S., Gardner, A. & Diggle, S. P. 2006 Social evolution theory for microorganisms. *Nat. Rev. Microbiol.* **4**, 597–607. (doi:10.1038/nrmicro1461)
17 Penn, D. J. 2002 The scent of genetic compatibility: sexual selection and the major histocompatibility complex. *Ethology* **108**, 1–21. (doi:10.1046/j.1439-0310.2002.00768.x)
18 Ostrowski, E. A., Katoh, M., Shaulsky, G., Queller, D. C. & Strassmann, J. E. 2008 Kin discrimination increases with genetic distance in a social amoeba. *PLoS Biol.* **6**, 2376–2382. (doi:10.1371/journal.pbio.0060287)
19 Gardner, A. & West, S. A. 2010 Greenbeards. *Evolution.* **64**, 25–38. (doi:10.1111/j.1558-5646.2009.00842.x)
20 Haig, D. 1997 *Behavioural ecology: an evolutionary approach* (eds J. R. Krebs & N. B. Davies), pp. 284–306. Cambridge, UK: Cambridge University Press.
21 Keller, L. & Ross, K. G. 1998 Selfish genes: a green beard in the red fire ant. *Nature* **394**, 573–575. (doi:10.1038/29064)
22 Lizé, A., Carval, D., Cortesero, A. M., Fournet, S. & Poinsot, D. 2006 Kin discrimination and altruism in the larvae of a solitary insect. *Proc. R. Soc. B* **273**, 2381–2386. (doi:10.1098/rspb.2006.3598)
23 Queller, D. C., Ponte, E., Bozzaro, S. & Strassmann, J. E. 2003 Single-gene greenbeard effects in the social amoeba *Dictyostelium discoideum*. *Science* **299**, 105–106. (doi:10.1126/science.1077742)
24 Sinervo, B. & Clobert, J. 2003 Morphs, dispersal behavior, genetic similarity and the evolution of cooperation. *Science* **300**, 1949–1951. (doi:10.1126/science.1083109)
25 Smukalla, S. *et al.* 2008 FLO1 is a variable green beard gene that drives biofilm-like cooperation in budding yeast. *Cell* **135**, 727–737.
26 Summers, K. & Crespi, B. 2005 Cadherins in maternal–foetal interactions: red queen with a green beard? *Proc. R. Soc. B* **272**, 643–649. (doi:10.1098/rspb.2004.2890)
27 West, S. A. & Gardner, A. 2010 Altruism, spite, and greenbeards. *Science* **327**, 1341–1344. (doi:10.1126/science.1178332)
28 Ofria, C. & Wilke, C. O. 2004 Avida: a software platform for research in computational evolutionary biology. *Artif. Life* **10**, 191–229. (doi:10.1162/106454604773563612)
29 Pennock, R. T. 2007 Models, simulations, instantiations and evidence: the case of digital evolution. *J. Exp. Theor. Artif. Intell.* **19**, 29–42. (doi:10.1080/09528130601116113)
30 Lenski, R. E., Ofria, C., Collier, T. C. & Adami, C. 1999 Genome complexity, robustness and genetic interactions in digital organisms. *Nature* **400**, 661–664. (doi:10.1038/23245)
31 Adami, C., Ofria, C. & Collier, T. C. 2000 Evolution of biological complexity. *Proc. Natl Acad. Sci. USA* **97**, 4463–4468. (doi:10.1073/pnas.97.9.4463)
32 Wilke, C. O., Wang, J., Ofria, C., Adami, C. & Lenski, R. E. 2001 Evolution of digital organisms at high mutation rates leads to survival of the flattest. *Nature* **412**, 331–333. (doi:10.1038/35085569)
33 Lenski, R. E., Ofria, C., Pennock, R. T. & Adami, C. 2003 The evolutionary origin of complex features. *Nature* **423**, 139–144. (doi:10.1038/nature01568)
34 Chow, S. S., Wilke, C. O., Ofria, C., Lenski, R. E. & Adami, C. 2004 Adaptive radiation from resource competition in digital organisms. *Science* **305**, 84–86. (doi:10.1126/science.1096307)
35 Goings, S., Clune, J., Ofria, C. & Pennock, R. T. 2004 Kin-selection: the rise and fall of kin-cheaters. *Proc. Artif. Life* **9**, 303–308.
36 Misevic, D., Ofria, C. & Lenski, R. E. 2006 Sexual reproduction reshapes the genetic architecture of digital organisms. *Proc. R. Soc. B* **273**, 457–464. (doi:10.1098/rspb.2005.3338)
37 Clune, J., Misevic, D., Ofria, C., Lenski, R. E., Elena, S. & Sanjuán, R. 2008 Natural selection fails to optimize mutation rates for long-term adaptation on rugged fitness landscapes. *PLoS Comput. Biol.* **4**, e1000187. (doi:10.1371/journal.pcbi.1000187)
38 Dennett, D. 2002 The new replicators. In *Encyclopedia of evolution* (ed. M. Pagel), pp. E83–E92. New York, NY: Oxford University Press.
39 Rousset, F. 2004 Genetic structure and selection in subdivided populations. Princeton, NJ: Princeton University Press.
40 Levenshtein, V. I. 1965 Binary codes capable of correcting deletions, insertions and reversals. *Dokl. Akad. Nauk SSSR* **163**, 845–848.
41 Sober, E. & Wilson, D. S. 1998 *Unto others*. Cambridge, MA: Harvard University Press.
42 Jansen, V. A. A. & van Baalen, A. 2006 Altruism through beard chromodynamics. *Nature* **440**, 663–666. (doi:10.1038/nature04387)