Improving Effectiveness and Efficiency in Wagner's Modularity-Evolving Artificial Gene Regulatory Networks

Zhenyue Qin*, Rouyi Jin*, R. I. (Bob) McKay , and Tom Gedeon

Australian National University, Australia
*Equal contribution
zhenyue.qin@anu.edu.au

Abstract—

Although no consensus has been reached on the conditions under which modularity emerges, we find the idea of specialisation driving modularity highly plausible. Current abstractions have demonstrated emergence of modularity in scenarios evolving Gene Regulatory Networks to show simple targeted behaviours, but these methods are less successful in more complex, real-world-level scenarios. We aim to study methods that are more efficient and effective in these simple scenarios, aiming to extend them to higher complexity. We engineer two modifications to the system that advance this aim. Progressive selection allows more than an order of magnitude reduction in computational cost without damaging the emergence of modularity, while dynamic specialisation produces significant improvements in modularity while also modestly reducing computational costs. We also provide some insight into the first of the two evolutionary stages in this approach, providing some explanation of why dynamic specialisation works.

1. Introduction

The Road to Modularity

Modularity is the divisibility of structures or functions "into multiple sets of strongly interacting parts that are relatively autonomous with respect to each other" [17] It is ubiquitous in complex systems, from well-engineered machines to naturally-evolved creatures [2]. The advantages of modular systems seem endless. They include: that a modular system can be more robust against both internal and external corruptions, since modularity can contain perturbations within small regions, leaving other parts of the system unaffected [11]; that more modular structures may be more evolvable [12]; and that different modules can operate cohesively with each other to perform more complex functions [33].

Compared with engineered systems, natural creatures implement the spirit of modularity even more heavily [4]: modularity is omnipresent not only in all biological beings, from simple single-cell bacteria to highly-complex animals [16], but is also pervasive at all levels of structures within a creature [18]. For example, humans have different organs as modules undertaking different functions, and each organ can be further divided into various tissues as modules that are relatively independent from each other [29].

It is intriguing to investigate the conditions under which biological systems acquire such high levels of modularity. This is non-trivial: compared with engineered systems, under evolutionary theory, biological creatures are not designed [27]. The only feedback from nature is to fit into the environment by competition with others. Nevertheless, non-modular candidates can perform some (especially complex) biological functions better than modular ones so that they should be more competitive [22], [37]. Furthermore non-modular changes are more likely to arise from random mutations and recombinations than modular ones, making it even harder to explains the ubiquity of modularity [31]. Thus the origins of modularity are elusive, yet their discovery could bring huge inspiration to engineering: we may build systems that can spontaneously adjust themselves to be modular and autonomously fit into novel environments [13].

Despite long-term interest, there is no consensus on the conditions for modularity emergence [7]. Among various theories, four stand out since their proposed conditions seem biologically plausible. They are: mitigating conflicts between preserving and modifying genes [32]; modularity-varying evolutionary environments [10]; biological parsimony pressures [3] and specialisation in biological organisms [7]. We employ the latter because we view it as the most plausible theory for the emergence of modularity: It has no obvious conflicts with nature; moreover it is compatible with multi-causal theories for the origins of modularity.

Motivations and Contributions

In [7], homeostasis is modeled by randomly generating perturbations of a target state, and observing the ability of systems to recover that target. In recent work [22], [24], we observed that this sampling is binomial. This study arises from a longer term program: currently, Wagner's scenario works well for simulating simple homeostasis problems. However it does not extend well to more realistically-sized problems [1]. We believe this results from the combination of high dynamicity (from sampling noise) and a complex fitness function. In [7], the number of possible perturbations

978-1-7281-2547-3/20/$31.00 ©2020 IEEE

grows exponentially with target size, and sample size needs to follow this to keep noise within limits: this is clearly intractable. We propose two modifications that we evaluate on the small gene regulatory network (GRN) problems of [7], but designed to extend to larger problems.

We note that in [7] the majority of gene regulatory networks (GRNs) encounter, and evolve to restore, very highly perturbed patterns. This is biologically unrealistic: biological systems can typically recover from small disturbances, but are less likely to recover from large ones, unless anomalies such as cancer occur [8]. We propose a modified selection mechanism, progressive selection, that evaluates individuals in a progressive manner. It aims to produce similar behaviour to full distributional evaluation while evaluating far fewer perturbations (thus reducing execution cost and simplifying the fitness landscape), while also removing dynamicity.

We also note that in [7], a change of target occurs in a fixed generation (500), by which time the population has already achieved high performance and convergence on the single-target problem. This has a number of potential problems. The solutions have already specialised to restore even very highly perturbed targets; the population has reduced diversity which it will need to recover after the change of target; and the extended period wastes scarce computational resources. Exogenous changes such as meteor strikes (which fixed generation change simulates) do occur. But endogenous changes are probably more common. For example, the changed evolutionary pressures of emergent epidemics often arise from population over-convergence [5], [28], [36]. In addition, specialisation often emerges as a result of creatures becoming sufficiently complex [26]. Motivated by these observations, we propose dynamic specialisation, which avoids regulating overly perturbed homeostasis by introducing a second target as soon as the system achieves sufficient performance on the first.

We verify our hypotheses with a variety of experiments, supported by statistical tests.

2. Related Work

Before explaining the evolutionary framework, we review four theories explaining the evolutionary origins of modularity, explaining why we view [7] as most plausible.

2.1. Conflicts Between Preserving and Modifying Behaviours Induce Modularity

The theory asserts that directional selections favouring changing some traits while maintaining others may explain the origin of modularity [32]. Pleiotropic interactions, *i.e.*, genes that influence two seemingly unrelated phenotypic traits, can obstruct both the constancy of some and the alternation of others [9]. Modularity of pleiotropic genes thus emerges to avoid compromising the related phenotypic traits. Modularity supports change of one trait while maintaining the behaviours of others. This explanation seems plausible because the proposed phenomenon is commonly observed in nature: only a few traits are likely to change, most remain stable during evolution [32], [35]. Nevertheless extended experiments failed to validate this theory. This may result from the genotype-phenotype map being overly simple [34].

2.2. Modular Environmental Changes Drive Modularity

Kashtan and Alon propose another theory: environmental modular changes may drive the emergence of modularity [10]. In more detail, repetitive and constant alternations of sub-components within an environment result in organisms evolving a higher-level of modularity. Two factors make this plausible. The proposed conditions are common in nature: environmental changes, such as fluctuations in temperatures and changes in salinity, are ubiquitous [7]. Perhaps more persuasively, the explanation is supported by observations on real-world creatures: metabolic networks of bacteria living in variable environments exhibit more modularity than those in steady environments [20]. However although natural environments often change continuously, it is unclear to what extent they vary modularly, leaving the quantitative relationship between modularly-varying environments and the emergence of modular organisms unclear [7].

2.3. Pasimony-Based Modularity-Inducing Models

Clune *et al*. argue that the evolutionary origin of modularity comes from the cost associated with every connection in the network [3]. They demonstrate this theory's plausibility by evolving artificial neural networks with or without imposing penalties on the number of edges. The former evolves significantly more modular networks. However the theory has a critical problem as an explanation for biological modularity. It is well known from early studies on parsimony pressures in genetic programming [21], [38] that too-large pressures lead to overly simple solutions: becoming small generally has a much smoother search gradient than becoming fit. Balancing these pressures is difficult, not least because later stages of evolution benefit from high parsimony pressures that would be destructive earlier.

Clune *et al*. in [3] avoid the issue by using a multi-objective algorithm, NSGA2 [6]. However the undoubted strengths of NSGA2 do not include biological plausibility: it incorporates non-dominated sorting, a population-wide operation that in nature would require a *deus in machina*. In reality, biological organisms evolve to optimise a single combined objective, being selected to fit the (constantly changing) environment. While [3] has a lot to tell us about how to ensure the emergence of modularity in engineered systems, its relevance to understanding the emergence of modularity in biological systems is more limited.

2.4. Modularity Emerges from Specialisation

Espinosa-Soto and Wagner claim that modularity emerges from specialisation: modular structures arise as a

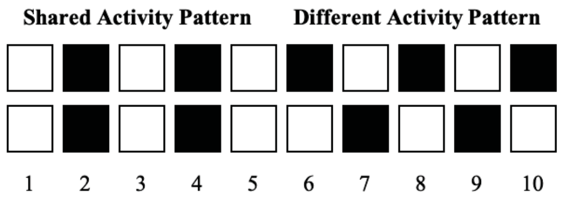


Figure 1. Visualisation of two gene activity patterns. White and black boxes stand respectively for activation and repression.

by-product of gene specialisation when networks try to regulate toward multiple different gene activity patterns [7]. They observe changes in the structure of Wagner's GRN model as evolution proceeds. Conflicts between incompatible regulation patterns favour the emergence of independent modules, with fewer inter-module connections. We view this as more plausible than the three preceding explanations for three reasons. There is an observed correlation between modular creatures and levels of specialisation [15]. In the simulated experiments, increasing the number of gene activity patterns can further improve the modularity, suggesting the potential for evolving extremely complex creatures as seen in the biological world [7]. Finally, the theory can be viewed as a refinement of the first two theories: relative to the first, sharing and specialised patterns can be seen as phenotypes to preserve and modify; relative to the second, organisms may evolve different gene patterns for different environments.

3. Evolutionary Framework

We briefly describe the procedures within the evolutionary framework: fitness evaluation, selection, recombination and mutation. We also explain how modularity is measured.

3.1. Multi-Stage Evolution

We employ the evolutionary framework of [7], simulating the acquisition of new gene activity patterns (GAPs) as evolution proceeds. The emergence of novel GAPs is common in natural evolution: for example unicellular organisms must evolve multiple GAPs to develop into multicellular creatures with diverse functions and morphologies among different cells. In [7], a GAP with N genes is represented as a discrete row vector $s_t = [s_t^1, \dots, s_t^N]$. The gene at each location i can be either active ($s_t^i = 1$) or inactive ($s_t^i = -1$).

GAPs are incrementally introduced as evolutionary targets. In [7], only two GAPs are used, depicted in Figure 1. It evolves with only the first GAP in the fitness function for 500 generations, then the second is added for 1500 more.

3.2. Discrete Gene Regulatory Network Model

Despite heterogeneity in functions and morphologies among cells within an organism, as discussed in Figure 3.1, to a first approximation all the cells share the same set of genes. The differences between GAPs in cells are largely controlled by gene-gene regulation. Networks of such gene-gene relationships (activation or repression) are termed gene regulatory networks (GRNs).

In [7], a GRN over patterns with N genes is abstracted as an $N \times N$ matrix [1] . In more detail, The state transition is regulated by GRN A as:

$$s_{t+1}^i = \sigma[\sum_{j=1}^{N} A_{ji} s_t^j], \tag{1}$$

where $\sigma(x) = +1$ if $x > 0$ and $\sigma(x) = -1$ otherwise. Figure 3 shows the regulating roles of each GRN region, *i.e.*, the pairwise regulating relationships between GAP locations.

3.3. Fitness Evaluation

Espinosa-Soto and Wagner estimate the fitness of a GRN based on their robustness against a randomly sampled set of perturbations, leading to a noisy evaluation [7]. To accurately evaluate a GRN's fitness, we exploit the essence of the perturbation sampling process: sampling from a binomial distribution, to design a new fitness evaluation that considers the entire perturbation set and weights the contribution of each perturbation to the fitness based on its probability in the distribution, eliminating the noise from sampling [23]. We briefly explain these two evaluation methods, referring to the former and the latter respectively as stochastic and distributional fitness evaluation.

Stochastic Evaluation. [7] evaluates fitness as the capability of a GRN to regulate randomly sampled perturbed gene activity patterns back to their original forms. To simulate the disturbance of gene activity patterns, [7] generates a set of P random perturbations of a pattern, with a probability of 0.15 of perturbing each gene to its opposite state. The GRNs are recursively applied to each perturbed state until the resultant pattern stops changing. This stabilised result is called an attractor. [30] shows that, except for very rare cases, it takes less than 30 transitions to reach an attractor. If an attractor can be attained, the system returns the Hamming distance D between the attractor and the original unperturbed pattern, otherwise it returns the maximum Hamming distance D_{max}. It then calculates $\gamma_i = (1 - D/D_{max})^5$ for each perturbed patterns within the set P. Finally, it computes the fitness of GRN g over gene activation pattern t as

$$f_t(g) = 1 - e^{-3\bar{\gamma}}, \tag{2}$$

where $\bar{\gamma}$ is the mean value over all γ_i.

1. We caution readers over a terminological problem. We are discussing abstracted GRNs applied to abstracted GAPs which consist of abstracted (biological) genes. However in evolutionary computation terms, the genotype is the GRN, so it would be natural to refer to the N^2 GRN locations as genes, rather than the N GAP locations. We will eschew the latter terminology: when we refer to a gene, it has the abstracted biological meaning, not the typical evolutionary computation meaning

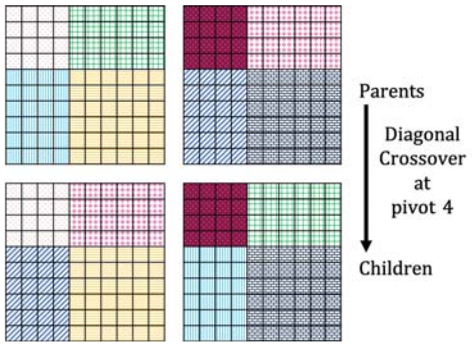


Figure 2. Illustration of a diagonal crossover with the pivot at 4. The same patterns represent the GRN entries that the children share with the parents.

Distributional Evaluation. Instead of randomly sampling perturbations from a binomial distribution, we replace $\bar{\gamma}$ in Equation 2 with the directly evaluated expectation of γ, thus removing the stochasticity of sampling. The process is described in greater detail in [23]).

3.4. Mutation

Mutation in [7] applies to a gene [1] in a GAP (*i.e.*, to a row of a GRN), and either adds an interaction to, or deletes an interaction from, that row. In GAP terms, it either adds an incoming interaction to, or deletes one from, the corresponding gene. Gene u has a probability of $\mu = 0.2$ to mutate in each generation, corresponding to a per-GRN-location mutation rate of 0.02 and a per-genome mutation probability of 0.893. However [7] uses a biased mutation to preserve sparsity of edges in the GRN. If gene u is to mutate, it has a probability $p(u)$ of losing an interaction and $q(u) = 1 - p(u)$ of gaining one. $p(u)$ is defined as

$$p(u) = \frac{4r_u}{4r_u + (N - r_u)} \tag{3}$$

where N is the number of genes in the target pattern, and r_u is the number of regulators (promoters or repressors) of gene u *i.e.*, the number of nonzero entries in the row of the GRN. When a regulator is added, it is equally likely to be a promoter (+1) or a repressor (-1). If there are no regulators to delete, or no openings to add one, mutation is a no-op. Note that Equation 3 predicts a fixpoint of $r_u = 0.2$, but because the preceding point leads to an additional bias (rows are more likely to be empty than full), the equilibrium edge density for selection-free evolution is around 0.22.

3.5. Recombination

In [7], no recombination (crossover) was implemented. However homologous crossover is well recognised in both biology and evolutionary computation as an effective means to combine genetic material. Horizontal crossover, which exchanges whole rows, was used in [14]. However it ignores the diagonal symmetry of the GRN effect (through matrix multiplication). We implemented a crossover (diagonal crossover) to exchange blocks diagonally, as illustrated in Figure 2. A "pivot" i is randomly selected from $\{1, \ldots, 10\}$, giving a "pivot point" $[i, i]$. When matrices A_1, A_2 are chosen to cross over at pivot i, the sub-matrices: $A[1:i, 1:i]$ and $A[i+1:10, i+1:10$ are left untouched while the remainders are exchanged.

3.6. Modularity Evaluation

We used the Q score introduced by [19], which compares the abundance of intra-module connections of a known network with that a randomly connected network. Formally, Q is defined as:

$$Q = \sum_{i}^{K} [\frac{l_i}{L} - (\frac{d_i}{2L})^2] \tag{4}$$

where i represents one of the K prospective modules in a network, L means the total number of edges in the network, l_i is the number of connections in module i, and d_i the sum of the number of interactions that each node in module i has [7]. Thus a network with high modularity has more intra-module connections and fewer inter-module connections. For two modules, this Q value lies in the range $[-\frac{1}{2}, 1)$, but the upper bound depends on the number of modules, making it difficult to compare experiments with different numbers of modules. Instead we used the normalised Q_n value of [10], which normalises the modularity of a network by comparison with a number of random networks with the same attributes. It is calculated as:

$$Q_\text{n} = \frac{Q - Q_\text{ran}}{Q_\text{max} - Q_\text{ran}} \tag{5}$$

where Q_ran represents the average of Q score over 10000 random networks, with the same number of nodes and edges as the known network and Q_max is the maximum Q score among those random networks.

4. Framework Extensions

In this section, we detail two extensions to the framework: progressive selection and dynamic specialisation.

4.1. Progressive Selection

In the first stage of evolution, when there is only one GAP target, the system readily achieves optimum fitness, as seen in Figure 4. In doing so, it regulates all perturbations back to the target. However once a second target is introduced, such performance is impossible. The second target is actually a perturbation (of weight 5) of the first, so that if the system regulates the second target back to the first (which it will do if it was optimal for the first stage), it by definition also regulates a weight zero perturbation of the second target to the first. This is suboptimal, since a weight zero perturbation carries far more influence in the fitness function than a weight 5 perturbation. This leads to a conflict in the fitness function: whenever more than half

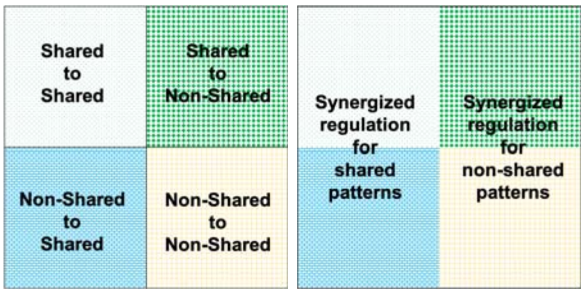

Shared to Shared
Shared to Non-Shared
Non-Shared to Shared
Non-Shared to Non-Shared
Synergized regulation for shared patterns
Synergized regulation for non-shared patterns

Figure 3. Illustrations of the expected GRN functional regions and a particular GRN example. Left: Four functional regions of a GRN to regulate two GAPs as illustrated in Figure 1, indicating what GAP locations regulate what. Right: Illustrations of which entries of a GRN regulate shared GAP locations and which regulate non-shared ones.

of the second half of a target is perturbed, it will be more beneficial to regulate back to the "other" target than itself.

This leads to a plausible improvement. Since longer perturbations will be rarely sampled by binomial sampling (and thus carry little weight in the fitness function), why bother to sample these confusing perturbations at all? Instead, we introduce a perturbation size limit l, and only evaluate GRNs on perturbations of weight up to l. In this work, since perturbations of length greater than 2 can lead to confusion if the whole of the perturbed part falls within the non-shared part of the two targets, we set $l = 2$. This saves enormously on computational cost. To gain even further speedup, we introduce a further wrinkle: since the perturbations of weights 0 and 1 have the most influence on the fitness function, when we perform tournament selection, we only evaluate perturbations of weights 0 and 1. If one of the GRNs wins the tournament based on this evaluation, we go no further. Only if two GRNs are tied on this evaluation do we evaluate them on weight 2 perturbations. We call this overall process progressive selection. Combining these two facets aims to reduce the number of executions of the GRN, the inner loop of this algorithm.

4.2. Dynamic Specialisation

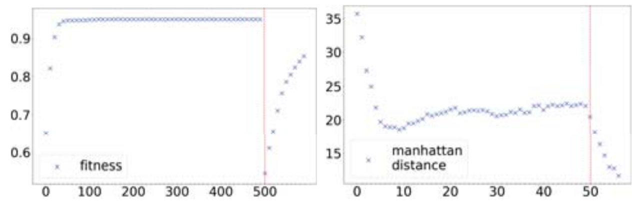


Figure 4. Mean best-of-population fitnesses and mean population-wide pairwise Manhattan distances between GRNs over 100 runs. The vertical lines show the generation at which the second GAP is added to the target using the mechanism of [7].

Dynamic specialisation replaces the fixed introduction of new target patterns of [7] with a more flexible scheme. Theoretically, this might best be based on fitness population convergence, but since we already know the optimum fitness, it is simpler and equally effective to base it on convergence to the optimum. We introduce a new hyper-parameter k that determines when to add a new GAP to the target: when the proportion of the population that has attained maximum fitness reaches k. For example, $k = 0.5$ sets the threshold as when 50% of individuals have achieved the maximum fitness. Preliminary testing showed that this value, $k = 0.5$, is an effective choice. As a fail-safe, if this fitness convergence is not achieved by generation 500, the second GAP is added to the target anyway.

The value of k affects two different aspects of the evolution. If the value of k is close to zero, it will switch targets with a population that has few optimal members, which can easily disappear after the change of fitness function, disrupting the feature of the overall scenario that is expected to generate modularity. On the other hand, it is well-known that diversity is critical in dynamic evolution [25], so that a value of k close to 1 affects the ability of the population to recover from the change of target. Between these two limits, values of k in the mid-range primarily affect the running time of the algorithm rather than its behaviour, but in that mid-range convergence is rapid, so that small changes in k do not much affect run time, nor do they greatly change behaviour.

Enhancing Evolutionary Efficiency and Effectiveness. As Figure 4 illustrates, in a typical run, optimum fitness is first attained very early, long before 500 generations, and the population diversity is substantially reduced. This has two consequences. The first is reduced efficacy: an over-converged population makes it more difficult for the population to re-adapt when a second target pattern is introduced. The second is reduced efficiency: why spend time running further generations when a sufficient level of convergence to the first target has been attained.

5. Experiment Settings

The target gene activity patterns and relational parameters in our evolutionary simulations are listed in Tables 1 and 2. The Wilcoxon Signed-Rank Test was used to validate the results of experiment. Each experiment had 100 runs and our evaluation measured the eventual fitness as well as the normalised score Q_{n} in the final generation.

In comparison with [7], we added an additional hyper-parameter k, and used distributional fitness evaluation,

TABLE 1. GENE ACTIVITY PATTERNS

Target Pattern	Generation to add Pattern
+1 -1 +1 -1 +1 -1 +1 -1 +1 -1	0
+1 -1 +1 -1 +1 +1 -1 +1 -1 +1	500 or k

6. Experimental Results

In this section, we report the experimental performance of our modifications to the evolutionary framework.

TABLE 2. Parameters in Evolutionary Simulation

Max Steps	Edge Size	Perturbation Rate
30	20	0.15
Mutation Rate	Population Size	Selection Type
0.2	100	Tournament
Tournament Size	Reproduction Rate	Maximum Generation
3	0.2	1000
k	Limitation Size of Perturbation	Significant Test
0.5	2	Wilcoxon Signed Rank

TABLE 3. Parameters Explanations in Evolutionary Simulation

Gene Activity Patterns	the patterns to which GRN evolves
Pattern Addition	the generation where a target is added
Edge Size	the initial number of edges
Perturbation Rate	the expected proportion of genes perturbed
Mutation Rate	the probability of a GRN node to lose or gain an interaction
Population Size	the number of individuals in a generation
Selection Type	the type of selection
Tournament Size	the number of tournaments in selecting a parent
Reproduction Rate	the proportion of offspring generated by crossover
Maximum Generation	the generation where the evolution terminates
Max Steps	the maximum time for an individual to reach an attractor
Limitation Size of Perturbation	The maximum perturbation size in progressive

6.1. Progressive Selection Improves Efficiency without Sacrificing Effectiveness

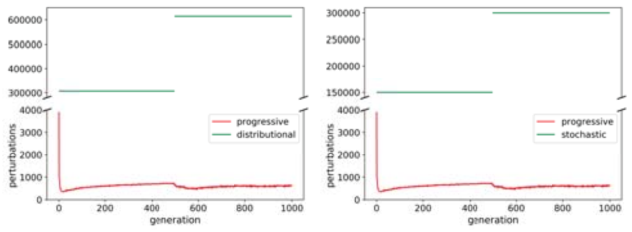


Figure 5. Number of perturbations evaluated over the course of evolution. Left panel: progressive selection vs distributional evaluation; right panel: progressive selection vs stochastic evaluation.

We compared three treatments: tournament selection with distributional evaluation; tournament selection with stochastic evaluation; and progressive selection using 100 trials for each. All introduced the second GAP target at generation 500 as per [7]. We recorded the total number of perturbations evaluated (which is the inner loop of the whole algorithm, and hence dominates computational cost) in each generation. The results in Figure 5 shows that while distributional evaluation has roughly twice the computational cost of stochastic evaluation in both phases, progressive selection is roughly 20 times faster in the first phase, and closer to 40 times in the second.

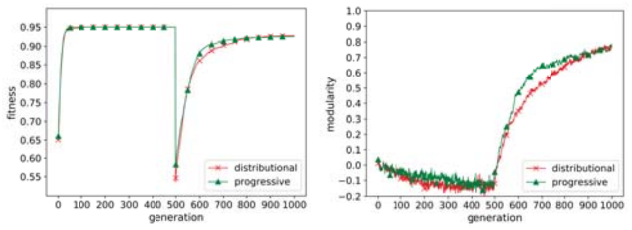


Figure 6. Mean fitness (left) and modularity (right) for the second stage of evolution over 100 trials: distributional fitness vs progressive selection

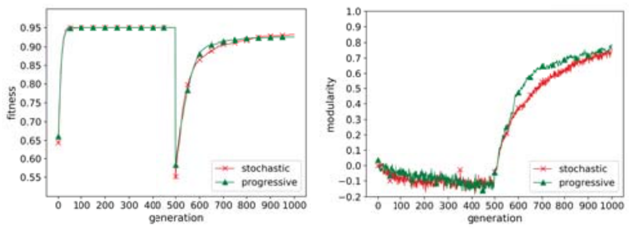


Figure 7. Mean fitness (left) and modularity (right) for the second stage of evolution over 100 trials: stochastic fitness vs progressive selection

To test the effectiveness of progressive selection, for each generation we extracted the fitness and Q_n values of the fittest individual, and averaged over 100 trials. Comparing this with the corresponding values for distributional and stochastic evaluation in Figure 6 and Figure 7, we see that at the end of the run (1000 generations), despite the very large gain in efficiency, progressive selection performs at least comparably with the other two treatments, and even generates high modularity somewhat earlier.

6.2. Dynamic Specialisation Promotes Robustness and Modularity

TABLE 4. Final generation best fitness (over 100 trials) with distributional and stochastic fitness evaluation: fixed generation vs dynamic specialisation for second GAP inclusion. Bold numbers indicate p-values below 0.01.

	Fixed	Dynamic	p-value
Distributional Fitness	0.927	0.934	0.0200
Distributional Q_n	0.767	0.894	**0.0020**
Stochastic Fitness	0.931	0.938	**0.0005**
Stochastic Q_n	0.763	0.854	0.0300

For both distributional and stochastic fitness evaluation, we compared static with dynamic specialisation, running 100 trials of each treatment for 1000 generations. For each treatment, we collected the best-fitness individuals from the final generation of each trial, and compared their mean fitness and modularity, computing p-scores for any differences. For each metric, dynamic specialisation outperformed fixed-generation specialisation, though only the Q_n value for distributional evaluation and the fitness value for stochastic evaluation were significantly different at the $p = 0.01$ level (Table 4).

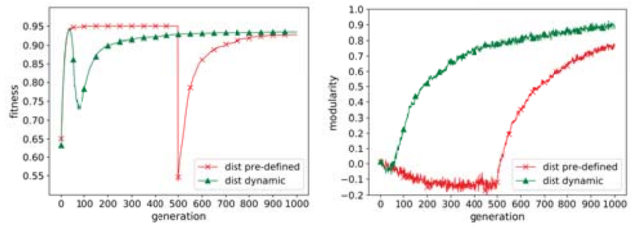


Figure 8. Mean fitness (left) and modularity (right) of the fittest individual in the final generation averaged over 100 runs using distributional evaluation: dynamic specialisation vs fixed generation

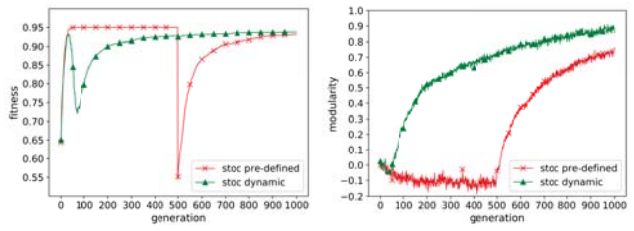


Figure 9. Mean fitness (left) and modularity (right) of the fittest individual in the final generation averaged over 100 runs using stochastic evaluation: dynamic specialisation vs fixed generation

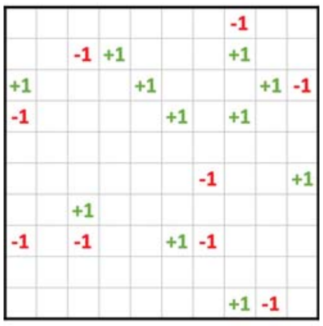
We show the progress of fitness and modularity for both distributional and stochastic fitness evaluation as generations proceed in Figure 8 and Figure 9. There is an improvement in fitness and modularity due to the change of the target happening earlier, so that evolution has longer to converge on the more difficult second stage problem. But it also appears that recovery after the change of target is more rapid, especially for fitness, suggesting that over-convergence, and hence the need to recover diversity, may play a part.

7. Discussion

In this section, we re-visit experimental phenomena that we could not comprehend at the time [24]. Further insight into the multi-stage evolutionary framework enable us to better understand these experimental results.

7.1. Regulating Processes of Stage 1

As Figure 4 illustrates, with few exceptions, almost all runs generate GRNs that can regulate any perturbations of the first GAP target back to itself – even for the extreme perturbation that inverts all GAP locations. In the cases we examined, these optimal GRNs contained a column made up



							-1		
		-1	+1				+1		
+1				+1				+1	-1
-1					+1		+1		
						-1			+1
		+1							
-1		-1			+1	-1			
							+1	-1	

Figure 10. An example of a GRN whose second column is an all-zero one. The entries of +1 and -1 stand for activation and repression respectively. Empty entries mean no edge.

entirely of zeros. This always corresponds to a location with target value -1; the zero column immediately regulates the corresponding location to -1. Other columns of the GRN then leverage this guaranteed -1 value to regulate other locations. Recursively, these newly corrected locations are further utilised to control more locations, until all locations are stable. Figure 10 shows an example of such an evolved GRN. From any input, this GRN recovers the GAP [1, -1, +1, -1, +1, -1, +1, -1, +1, -1]. We refer to such optimal evolved GRNs as zero-column based.

7.2. Limitations of Zero-Column Regulation

As we showed in [23], an optimal GRN for the two-target (second stage) optimisation regulates any perturbation of the first half of the target back to itself, and the second half to whichever of the two targets is closer (in Manhattan distance). If an optimal solution to the first target is zero-column based, there are two cases.

Where the zero column occurs in the right half of the GRN, it immediately regulates the corresponding location in an input GAP to -1. If the majority of the right-hand of the target differs from the first target, a second-stage-optimal GRN should regulate it instead to +1. But it is difficult to make this change, because changing that column will then disrupt all regulation that depends on that zero column – including (directly or indirectly) all locations in the left-hand half of the GAP, which optimally should stay stable.

Conversely, if the zero column lies in the first half, this assignment should stay stable. But then, any links from that location to any location on the right hand side fall into the same problem as the zero column in the previous case: they must stop depending on the left hand columns they previously depended upon.

In effect, this arises because the second-stage-optimal GRN, for the first half of the target, needs only to solve a problem of recovering a specific target. But for the second, it needs to solve a majority-match problem.

These considerations suggest that over-convergence to zero-column solutions in the first stage is likely to lead to inefficient evolution in the second stage, providing further justification for dynamic specialisation.

7.3. How General are the Methods Described Here

Many computationally expensive applications of evolutionary algorithms involve fitness functions which are too expensive to evaluate precisely, but which can be approximated with increasing precision at the cost of more computation. The general idea of the progressive selection operator– to use a form of tournament selection, in which the fitness function is lazily evaluated only as far as is necessary to reasonably reliably decide the tournament – is very widely applicable. In most cases, it is not important to precisely estimate the tournament winner, since small random errors in this process merely add a small amount of stochasticity to a process (an evolutionary algorithm) that is deliberately stochastic in any case. This wider class of selection operators is correspondingly widely applicable. The variant here uses deep specific knowledge about the structure of the fitness function (based on binomial sampling) and of the relative influence of different perturbations; this specific structure is peculiar to Wagner's scenario, but fitness functions based on binomial sampling are quite common in artificial life simulations, so that very similar methods may be much more widely aplicable.

Dynamic specialisation applies to a much narrower range of simulations, where the behaviour of the system under consideration is episodically dynamic. In such scenarios, efficient evolution able to track the changing fitness function will occur when the population is partially converged at the time the target changes. This is reasonably common in artificial life simulations, but requires a computation of diversity at regular intervals whose cost is generally quadratic in the population size. Linear cost (as here) is generally only achievable when the exact optimum is known, and is likely to be found in all runs.

8. Conclusion

Emergence of modularity is a key issue for both understanding the evolution of natural systems, and engineering complex artificial systems. For example, recent studies suggest that human intelligence arises from modular brain structures. We study and improve Wagner's modularity inducing framework. Our long-term aim is to extend the complexity of the systems it can evolve, to approach that of simpler biological networks. Short-term, we focus on improving the computational efficiency and the effectiveness of the system for more complex problems. Here, section 6 reveals that progressive selection can provide very large speedups with no cost to effectiveness, while dynamic specialisation significantly improves modularity as well as providing modest computational speedups and section 7 reveals further insights into the computational task of [7] on which we hope to leverage future engineering improvements.

References

[1] Chris Adami. Comment on ALIFE 2019 presentation https://drive.google.com/file/d/1T2t6ETXLV_0uzIW2rwKT0-KZXLzKVHu_/view?usp=sharing, 2019.

[2] Uri Alon. Biological networks: the tinkerer as an engineer. *Science*, 301(5641):1866–1867, 2003.

[3] Jeff Clune, Jean-Baptiste Mouret, and Hod Lipson. The evolutionary origins of modularity. *Proceedings of the Royal Society b: Biological sciences*, 280(1755):20122863, 2013.

[4] Pedro H Constantino and Prodromos Daoutidis. A control perspective on the evolution of biological modularity. *IFAC-PapersOnLine*, 52(11):172–177, 2019.

[5] H. David Cooper, Charles Spillane, and Toby Hodgkin. Broadening the genetic base of crops: An overview. In H. David Cooper, Charles Spillane, and Toby Hodgkin, editors, *Broadening the Genetic Base of Crop Production*, chapter 1, pages 1–23. CABI Publishing, New York, January 2001.

[6] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In Marc Schoenauer, Kalyanmoy Deb, Günther Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature PPSN VI*, pages 849–858, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[7] Carlos Espinosa-Soto and Andreas Wagner. Specialization can drive the evolution of modularity. *PLoS Comput Biol*, 6(3):e1000719, 2010.

[8] Yejing Ge and Elaine Fuchs. Stretching the limits: from homeostasis to stem cell plasticity in wound healing and cancer. *Nature Reviews Genetics*, 19(5):311, 2018.

[9] Haiyang Hu, Masahiro Uesaka, Song Guo, Kotaro Shimai, Tsai-Ming Lu, Fang Li, Satoko Fujimoto, Masato Ishikawa, Shiping Liu, Yohei Sasagawa, et al. Constrained vertebrate evolution by pleiotropic genes. *Nature Ecology & Evolution*, 1(11):1722–1730, 2017.

[10] Nadav Kashtan and Uri Alon. Spontaneous evolution of modularity and network motifs. *Proceedings of the National Academy of Sciences*, 102(39):13773–13778, 2005.

[11] H Kirsten and Paulien Hogeweg. Evolution of networks for body plan patterning; interplay of modularity, robustness and evolvability. *PLoS Comput Biol*, 7(10):e1002208, 2011.

[12] Ben Kovitz, David Bender, and Marcela Poffald. Acclivation of virtual fitness landscapes. In *Artificial Life Conference Proceedings*, number 31, pages 380–387, 2019.

[13] Sam Kriegman, Nick Cheney, and Josh Bongard. How morphological development can guide evolution. *Scientific reports*, 8(1):1–10, 2018.

[14] Ari Larson, Anton Bernatskiy, Collin Cappelle, Ken Livingston, Nicholas Livingston, John Long, Jodi Schwarz, Marc Smith, and Josh Bongard. Recombination hotspots promote the evolvability of modular systems. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, pages 115–116, 2016.

[15] Dirk M Lorenz, Alice Jeng, and Michael W Deem. The emergence of modularity in biological systems. *Physics of life reviews*, 8(2):129–160, 2011.

[16] Diogo Melo. How does modularity in the genotype–phenotype map shape development and evolution? In *Old Questions and Young Approaches to Animal Evolution*, pages 237–249. Springer, 2019.

[17] Diogo Melo, Arthur Porto, James M Cheverud, and Gabriel Marroig. Modularity: genes, development, and evolution. *Annual review of ecology, evolution, and systematics*, 47:463–486, 2016.

[18] Henok Mengistu, Joost Huizinga, Jean-Baptiste Mouret, and Jeff Clune. The evolutionary origins of hierarchy. *PLoS computational biology*, 12(6):e1004829, 2016.

[19] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

[20] Merav Parter, Nadav Kashtan, and Uri Alon. Environmental variability and modularity of bacterial metabolic networks. *BMC evolutionary biology*, 7(1):169, 2007.

[21] Riccardo Poli and Nicholas Freitag McPhee. Parsimony pressure made easy. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, page 1267–1274, New York, NY, USA, 2008. Association for Computing Machinery.

[22] Zhenyue Qin, Tom Gedeon, and RI McKay. Anomalies in the behaviour of a modularity inducing problem domain. In *Artificial Life Conference Proceedings*, pages 228–235. MIT Press, 2019.

[23] Zhenyue Qin, Tom Gedeon, and R.I. (Bob) McKay. Just Use Distributions: Eliminating The Noise of Sampling-Based Fitness Estimation. *TechRxiv*, 10 2020. doi: 10.36227/techrxiv.13042289.v1.

[24] Zhenyue Qin, Tom Gedeon, and Robert I McKay. Why don't the modules dominate? In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 121–122, 2018.

[25] Hendrik Richter. Evolutionary optimization and dynamic fitness landscapes. In Ivan Zelinka, Sergej Celikovsky, Hendrik Richter, and Guanrong Chen, editors, *Evolutionary Algorithms and Chaotic Systems*, pages 409–446. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[26] Shunsuke Sogabe, William L Hatleberg, Kevin M Kocot, Tahsha E Say, Daniel Stoupin, Kathrein E Roper, Selene L Fernandez-Valverde, Sandie M Degnan, and Bernard M Degnan. Pluripotency and the origin of animal multicellularity. *Nature*, 570(7762):519–522, 2019.

[27] Ricard V Solé and Sergi Valverde. Spontaneous emergence of modularity in cellular networks. *Journal of The Royal Society Interface*, 5(18):129–133, 2008.

[28] Javier Tello, Roswitha Mammerler, Marko Čajić, and Astrid Forneck. Major outbreaks in the nineteenth century shaped grape phylloxera contemporary genetic structure in europe. *Scientific reports*, 9(1):17540, November 2019.

[29] Lisa A Urry, Michael Lee Cain, Steven Alexander Wasserman, Peter V Minorsky, and Jane B Reece. *Campbell biology*. Pearson Education, Incorporated, 2017.

[30] Andreas Wagner. Does evolutionary plasticity evolve? *Evolution*, 50(3):1008–1023, 1996.

[31] Andreas Wagner. *Arrival of the fittest: solving evolution's greatest puzzle*. Penguin, 2014.

[32] Günter P Wagner. Homologues, natural kinds and the evolution of modularity. *American Zoologist*, 36(1):36–43, 1996.

[33] Günter P Wagner and Lee Altenberg. Perspective: complex adaptations and the evolution of evolvability. *Evolution*, 50(3):967–976, 1996.

[34] Günter P Wagner and JG Mezey. The role of genetic architecture constraints in the origin of variational modularity. *Modularity in development and evolution*, pages 338–358, 2004.

[35] AltenbergL WagnerGP. Complexadaptationsand theevolutionofevolvability. *Evolution*, 50:967–976, 1996.

[36] Yiting Wang, Feng Chen, Jiali Wang, Yingwang Zhao, and Fang Liu. Two novel homozygous mutations in nphp1 lead to late onset end-stage renal disease: a case report of an adult nephronophthisis in a chinese intermarriage family. *BMC nephrology*, 20(1):1–5, 2019.

[37] Qiuhai Yue, Randi C Martin, Simon Fischer-Baum, Aurora I Ramos-Nuñez, Fengdan Ye, and Michael W Deem. Brain modularity mediates the relation between task complexity and performance. *Journal of cognitive neuroscience*, 29(9):1532–1546, 2017.

[38] Byoung-Tak Zhang and Heinz Mühlenbein. Balancing accuracy and parsimony in genetic programming. *Evolutionary Computation*, 3(1):17–38, 1995.