Are Artificial Mutation Biases Unnatural?

Seth Bullock

Center for Adaptive Behavior and Cognition
Max Planck Institute for Human Development
Lentzeallee 94, 14195 Berlin, Germany

Abstract. Whilst the rate at which mutations occur in artificial evolutionary systems has received considerable attention, there has been little analysis of the mutation operators themselves. Here attention is drawn to the possibility that inherent biases within such operators might artefactually affect the direction of evolutionary change. Biases associated with several mutation operators are detailed and attempts to alleviate them are discussed. Natural evolution is then shown to be subject to analogous mutation "biases". These tendencies are explicable in terms of (i) selection pressure for low mutation rates, and (ii) selection pressure to avoid parenting non-viable offspring. It is concluded that attempts to eradicate mutation biases from artificial evolutionary systems may lead to evolutionary dynamics that are more unnatural, rather than less. Only through increased awareness of the character of mutation biases, and analyses of our models' sensitivity to them, can we guard against artefactual results.

This paper explores the potential for artefactual evolutionary simulation results to derive from biases inherent within the artificial mutation operators they employ. As an example, consider a recent coevolutionary simulation model which has suggested that signals exhibiting complex symmetry could evolve merely as a side effect of selection for distinctiveness [1].

The model involved multicoloured, composite patterns coevolving with simple artificial neural networks under a mutualist selection regime. Networks which were able to discriminate signal patterns from distractor patterns were favoured, as were discriminable signal patterns. This discrimination had to be achieved despite patterns being presented in various orientations and positions on each network's artificial "retina". After a period of simulated coevolution, the authors report that networks were able to distinguish signals from distractors almost perfectly, and that the coevolved signal patterns displayed "marked symmetries" (p. 171). The authors note that, like many natural displays, the evolved signals consisted of "purer, brighter colours" (p. 171) than average signals. An evolutionary-functional account for the evolved symmetry was proposed – symmetrical signals persist because they are invariant under the various transformations involved in their presentation, and hence easier to discriminate from distractors. The evolved signals' bold coloration was explained as the result of selection pressure to diverge from random distractor patterns.

However, a replication of the study demonstrated that the complex symmetry of the evolved signals resulted from an unnaturally structured presentation regime [2]. Might the boldness of the evolved signals result from a similar simulation bias? One possible source of such a bias is the simulation's mutation operator.

During reproduction, each of the parent signal's colour components was subject to a small chance of mutation. A mutation event, when it occurred, perturbed the parental value by a small increment. Since each colour component was coded for by a real value lying in the interval $[0, 1]$, some of these perturbations resulted in mutant values which lay outside the legal range. The authors do not report the measures taken to deal with such illegal mutations. However, a mutation operator which replaced illegal mutant values with the nearest legal value might favour extreme colour component values merely through evolutionary drift. Perhaps this, rather than some pressure to deviate from the average, explains the simulation's results?

How might mutation biases occur in general? What are the possible effects of mutation bias? How can we test for the presence of mutation biases, and eliminate them from evolutionary simulations? These questions are of practical significance for any evolutionary simulation modeller.

1 Artificial Mutation

Within individual-based evolutionary computer simulations, artificial mutation operators are relied upon to provide the genetic diversity upon which selection may act. Simulations involving sexual populations augment mutation with crossover operators which allow offspring to inherit passages of genetic material from each parent. However, mutation continues to be the major source of genetic *novelty* in such simulations, since crossover operators reshuffle existing genes, rather than introduce new ones.

Many studies have explored the effect on artificial evolution of employing various genetic operators (e.g., one-point or two-point crossover, the transposition, repetition and excision of genetic material), varying rates at which these operators are employed, and even allowing the types of operators and the rates at which they are employed to themselves be subject to adaptation. For example, the proceedings of one early conference on genetic algorithms [3] contains five papers which between them address all of these concerns. More recent studies have developed these ideas [4–6].

However, since much of this work takes place within engineering contexts where there need be little concern for evolutionary plausability, little attention has been paid to the biases inherent within even the simplest mutation operators. Although the construction of these operators is not typically regarded as a complex matter, as will be argued below, they differ significantly from natural mutational processes, and their design therefore involves issues which are unique to the production of evolutionary simulations.

2 Legal Bounds in Natural and Artificial Genetic Systems

Consider a phenotypic trait which may vary over some range. We may distinguish between two classes of such a trait. Unbounded traits may vary limitlessly. Such traits might include the significance afforded to a male display by a female onlooker, which might vary from zero (non-significant), through positive infinity (infinitely attractive), or negative infinity (infinitely repellent). Bounded traits are those for which the range of legal values is in some way limited. Many traits suffer either a lower or an upper limit, and are thus partially bounded. For example, one cannot have fewer than zero legs. Some traits are bounded at both extremes, e.g., a trait governing the time of day at which one begins to forage might vary between dawn and dusk. Such a notion of boundedness raises the associated notion of legal and illegal genotypes, the latter being those that code for traits which transgress their legal limits.

The distinction being made here is an unnatural one which cannot easily be applied to natural genetic encodings. For example, although natural genes may code for what appears to be a bounded phenotypic trait (e.g., the redness of a signal), the manner in which they do so may logically preclude the occurrence of illegal values for this trait. Additive polygenic traits, for example, might code for the varying degree of a phenotypic trait with a varying number of genes. Since such a genotype cannot contain a negative number of these genes, it cannot code for an illegal phenotypic trait (e.g., a signal with negative redness).

In one sense however, an "illegal genotype" is one which does not result in a viable organism. Clearly, such mutants will not leave offspring. Their genotypes will be selected out of the population. This selective process is the same one that excludes viable mutants which are less well adapted to their niche than their competitors. From the perspective of natural selection, there is no difference (ignoring indirect fitness effects) between failing to be born living, failing to survive to reproductive age, failing to mate, or failing to reproduce before dying of old age – all such failures are awarded zero fitness.

Genotype legality within artificial evolutionary algorithms falls somewhere between the accounts given in the previous two paragraphs. It is true that, as in the first account, genotypes with illegal trait values are never realised as organisms, and are thus never subjected to the same selective pressures as their valid conspecifics. However, illegal genotypes *are* generated by many evolutionary algorithms and are thus not excluded out of logical necessity as in the first account, but are selected as invalid, as in the second account. But the grounds upon which illegal genotypes are selected are not the same as those which govern phenotypic selection. The legality of genotypes is assessed, prior to any morphogenetic, developmental or ontogenetic performance, on the basis of the genotype itself, rather than the performance of the associated (unrealisable) phenotype. If found to be legal, the genotype is translated into a phenotype and assessed as normal. If found to be illegal, some alternative course of action must be taken. It is this lack of correspondence between artificial evolutionary algorithms and natural evolution which raises the possibility that the character of artificial mutation operators may be unnatural due to the biases which they introduce.

3 Mutation Operators

In this section, the biases of various mutation operators will be characterised. These operators differ in the manner in which they treat illegal mutant values. In order to describe the repercussions of these differences, I will use the terms *departure rate* to denote the relative rate at which a particular parental value is altered by a mutation operator, and *arrival rate* to denote the relative rate at which a particular mutant value is generated by a mutation operator.

First, a distinction must be made between *context-free* and *context-sensitive* operators. Whilst context-free operators generate mutant values which are independent of parental values, context-sensitive operators are predicated on such variables. This latter class of operator (which will be the focus of this paper) typically generate mutant values which are close to the relevant parental values in an effort to mimic natural evolution. Many of the issues discussed here will be most pertinent to either continuous-valued genes or discrete (typically binary) encodings. However, most are to some degree applicable to both.

Genes coding for bounded phenotypic traits may be subjected to context-free mutations through drawing a random value from the range of legal values available for a trait. This operator will be referred to as the "Flat" mutation operator throughout this paper. For example, when mutating a trait which governs the hirsuteness of an organism, a Flat mutation operator might ignore the degree of body hair possessed by the parent and simply assign the mutant offspring a random number between 0% and 100%. Such an operator is flat in that both the departure rate and arrival rate are uniform across the valid range of the trait.

Similarly, genes coding for unbounded, or partially bounded, phenotypic traits may be mutated independently of their parental value through imposing some arbitrary *mutant range* upon the phenotypic trait and picking a value from this mutant range. Such a solution is unsatisfactory, however, as this method introduces a mutation *bias*. Despite there being an equal probability of any parental value being mutated (a flat departure rate), there is not an equal probability of any legal mutant value being generated by the mutation operator, the arrival rate outside the mutant range being zero, whilst that inside this range is positive and flat. Such a bias may lead to artefactual results. For example, if the optimal trait lies outside the mutant range, a population of organisms may evolve to lie clustered at the mutant value nearest to this optimum. Although the population may appear to have converged on some stable phenotype, this appearance is an illusion created by the mutation bias.

In contrast, context-sensitive mutation operators which *perturb* the parental genotype by some small amount have the potential to generate illegal values for bounded traits. For such mutation operators, decisions concerning the treatment of these illegal mutations must be made. There are a number of straightforward options that an operator designer might take:

Absorb Illegal mutant values are truncated to the nearest boundary.

Repeat Mutant values are repeatedly generated, until a legal value is obtained.

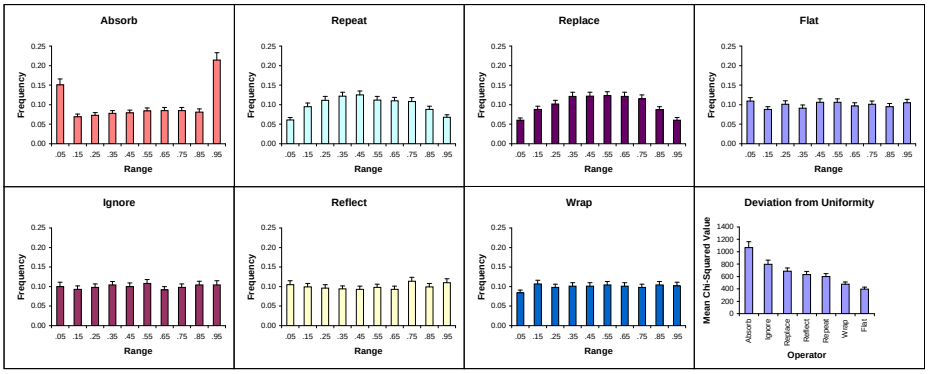

Fig. 1. Average distribution of trait values across 20 asexually reproducing populations of 1000 single-trait organisms after 5000 generations of evolution on a flat fitness surface, for each of seven mutation operators (see text). Traits were real values in the range $[0, 1]$. Mutation events occurred with probability 0.01, and consisted of real-valued perturbations drawn from a normal distribution with zero mean and standard deviation 0.2. Absorb, Repeat and Replace show clear deviation from uniformity in their aggregate performance. A Flat, context-free operator is shown for comparison. Three of the lower panels suggest that over many runs the behaviour of Ignore, Reflect and Wrap is roughly equivalent to Flat. However, the fourth lower panel shows that their mean deviations from uniformity (see text) differ significantly from each other, and from that of traits evolved under a Flat mutation operator. All graphs show mean frequencies with attendant standard errors.

Replace Any offspring for which illegal trait values are generated is replaced by a new offspring, re-choosing parents.

Each of these three operators result in mutation biases which either favour or resist extreme-valued traits (Fig 1). The "Absorb" operator will effect a constant departure rate across the range of trait values, save that it falls to half the nominal mutation rate at the extremes of the trait's legal range. In addition, the arrival rates at these extremes are increased by their absorbent nature. This ensures that trait values near the legal boundaries of the trait will tend to reach them and be kept there. Conversely, the "Repeat" operator will maintain a constant departure rate across the legal range of trait values, but will tend to mutate extreme parental values away from the boundaries of the trait. This ensures that arrival rates decrease as trait values approach their legal limits. The "Replace" operator, although seemingly the most accurate reflection of natural mutation processes, in that illegal or non-viable offspring are simply rejected, results in a selection pressure that resists the evolution of extreme trait values due to the increased likelihood that extreme genotypes will generate illegal offspring. Several mutation operators might be constructed specifically to alleviate these edge-effect biases.

Ignore Mutation events which transgress legal bounds are ignored. Rather than inherit an illegal mutant value, offspring inherit the parental value.

Reflect Mutant values lying a distance of x above (or below) the legal range are replaced by vaues a distance of x below (or above) the nearest boundary.

Wrap The trait is treated as if it were periodic. The edges of its legal range "wrap" around. Mutant values are calculated modulo the trait's range.

The aggregate behaviour of these operators is uniform (Fig 1). However, they differ in how they achieve this aggregate uniformity. For example, under the "Ignore" regime, more illegal values will be generated (and ignored) for parental values near the extremes of the legal range. This ensures that the departure rate will decrease as parental values approach legal extremes. However, this reduction in departure rate is prevented from systematically biasing the population by a correlated reduction in arrival rate. To the extent that a trait value x is extreme and hence suffers a low departure rate, it will be in an extreme neighbourhood also suffering a low departure rate. Since it is through mutations affecting parents within this neighbourhood that x is likely to arise, this low departure rate will lower x's arrival rate. As a result, extreme values are less likely to be generated by the operator, but when generated, are less likely to be mutated. Whilst this balance ensures that populations are not systematically biased toward or away from some areas of the trait space, it also results in individual distributions with a certain character. In order to determine whether, and to what extent, this character differs from that of populations under different mutation regimes, a measure of *deviation from uniformity* was calculated.

A χ^2 value estimating the degree of deviation from a uniform expected distribution was calculated for each individual evolved population contributing to the aggregate distributions graphed in Fig 1. The mean χ^2 value for each operator is shown in the lower far-right panel. This measure of deviation from uniformity differentiates between the three pseudo-flat operators, demonstrating that despite their similar aggregate performance, they each exert a unique influence on the character of individual populations. In addition, this metric demonstrates that aggregate flat performance can disguise individual distributions which tend to be very far from flat. The Ignore operator, for example, generates distributions which are on average less flat than either Repeat or Replace, despite there being no systematic bias to this non-uniformity.

4 Discrete Encodings

Two kinds of discrete encoding must be distinguished. Both allow a trait to take one of a (possibly infinite) number of discrete values. The first treats these values as unitary wholes for the purposes of mutation (and crossover), whereas the second represents these values with a number of discrete digits, each independently exposed to the possibility of mutation. Whilst the issues discussed above apply fairly straightforwardly to the first kind of discrete encoding, the second kind raises new issues.

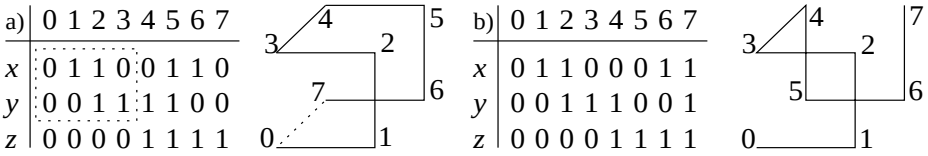

Fig. 2. Two three-bit Gray codes expressed over *(a)* the canonical Hamiltonian circuit, and *(b)* a mere Hamiltonian path. An n-bit canonical Gray code may be constructed by reversing the (n-1)-bit canonical Gray code which it contains (dashed box), and appending a 0 to the first 2^{n-1} bit strings and a 1 to the remaining 2^{n-1} bit strings.

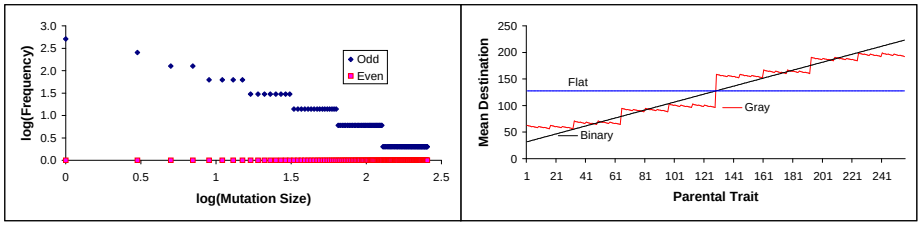

Fig. 3. *Left*: frequency distribution of mutation sizes for an 8-bit Gray-coded trait. *Right*: distributions of mean mutation destination by parental trait value for Gray, Binary and Flat mutation operators. Discontinuities could impede a population's evolution.

Encoding traits as a vector of discrete digits, and moving between trait values through manipulations at the level of these individual digits, imposes a structure on the mutation space of a model. There are a^n possible values represented by an n-dimensional code utilising an alphabet of a symbols, but each individual vector has only $n(a-1)$ immediate neighbours. This ensures that the code's neighbourhood relationships might systematically bias evolutionary change in particular ways.

Traits encoded as groups of binary digits which are interpreted as phenotypic values are typically mutated via random bit flips. When such a mutation operator acts upon conventional binary numbers, mutated phenotypic traits are poorly correlated with parental phenotypic traits, since a single bit-flip may result in a large change in the value for which the bit-string codes. Gray coding [7], through ensuring that consecutive integers are coded for by adjacent binary strings, increases the correlation between mutants and their parents. In addition to sharing the general appeal of context-sensitive mutation operators, this coding scheme is advocated by genetic algorithm designers, who argue that it makes evolutionary search more effective [8].

Constructing an n-bit Gray code can be thought of as assigning each of 2^n values to each of the vertices of an n-dimensional binary hypercube such that adjacent values are assigned to adjacent vertices lying on one of the hypercube's

Hamiltonian paths (paths which visit each vertex once and visit all vertices). Typical Gray code algorithms [8, 9] use a *Hamiltonian circuit*, in which the terminal vertices are also adjacent (Fig 2). It is unclear to what extent the claims of improved performance under Gray code schemes can be generalised from this *canonical Gray code* to other instances [8].

Though Gray codes ensure that there always exists a mutation event capable of perturbing a trait's value by a unit, the character of the remaining distribution of single-bit mutation events is unspecified. In fact, the canonical Gray code achieves a distribution of mutation events which decays roughly exponentially with mutation size (calculated as the absolute difference between pre-mutation and post-mutation trait values). This distribution exhibits strong discontinuities (Fig 3); there are no mutation events of even-valued magnitude. This implies that any mutation event changes both the parity and magnitude of the inherited value. For instance, if the former determined an organism's handedness whilst the latter coded for degree of handedness, this mutation bias would ensure that any mutant would exhibit both a (typically small) change in the degree of handedness *and* a change in which hand was preferred. Care must thus be taken to ensure that the constraints on mutation imposed by a discrete encoding do not systematically interfere with evolutionary change.

5 Implications

The threat to simulation modelling posed by the biases described above may seem overworked. However, the analyses clearly reveal the potential for the design of mutation operators to systematically influence trajectories of evolutionary change. One kind of solution to this problem has already been described, namely, to attempt the construction of bias-free operators. However, we have seen that while such attempts may *alter* the biases exhibited by an operator, these biases are typically not extinguished. It is also the case that whereas the most naturalistic mutation operators (e.g., Replace, or Absorb) might generate the most striking effects, the biases exhibited by these operators are explicable in terms of selection pressures which are present in naturally evolving populations.

For example, natural selection, *ceteris paribus*, favours genotypes which (i) have relatively low mutation rates, and (ii) are reasonably far from non-viable mutants [10, 11] (but see [12, 13]). Between them, these two selective forces can account for the biases exhibited by the most straightforward mutation operators considered here (Absorb, Repeat, and Replace). However, the biases of the operators designed specifically to negate these selection pressures (e.g., Ignore, Reflect and Wrap) are more difficult to account for in terms of natural selection pressures, as a result of their unnatural treatment of trait boundaries.

How are simulation modellers to decide between competing mutation operators – should they choose mutation operators that are clearly biased, but in a manner underwritten by natural selection, or choose alternative operators which although in some sense are "less biased", exhibit idiosyncrasies that have no biological analogue? I propose that this problem can be resolved by addressing

a more pressing question: how are simulation modellers to guard against the artefactual results which may be caused by such biases?

Ultimately, in order to answer this question, the kinds of biases exhibited by various classes of mutation operator must be better understood, and attention must be paid to the conditions in which these biases will affect simulation results most severely. For instance, it appears that the influence of mutation bias will be strongest when populations are under only weak selection pressure. This situation may occur when populations drift across fitness plateaux, or during initial evolutionary transients which may be especially sensitive to population make-up. However, mutation biases may still influence the evolution of populations under strong selection pressures if these pressures conflict, through favouring one rather than the other. These are issues demanding further analysis.

In the absence of a principled understanding of mutation bias, however, two steps can be taken to minimise their influence. The presence of potential artefacts can be detected through exploring (i) the sensitivity of the simulation's behaviour to variation of its initial conditions (a procedure which has a number of other advantages and should thus be undertaken in any case), and (ii) the sensitivity of the simulation's behaviour to variation of the mutation operator employed (a less generally useful technique, but one which may be necessary if mutation bias is suspected).

Here the application of these two techniques to the example with which this paper opened will be described. In order to explore whether the bold coloration of the coevolved signal patterns was an artefact brought about by mutation bias, a previous study [2] explored the sensitivity of the result to manipulation of the initial ancestral population: populations of signals were able to evolve away from maximally bold ancestors, but they tended to re-converge on bold colours, although not necessarily those with which the populations were seeded.

Here the second line of sensitivity analysis will be reported. Simulations identical to those carried out in [2] were implemented, save that the mutation operator was varied across three conditions – Flat, Repeat, and Absorb. Boldness, b, was calculated as $1 - \sqrt{4/3}\sum_{i=1}^{25}\sqrt{r_i^2 + g_i^2 + b_i^2}$, where r_i, g_i, and b_i are the distances of the i^{th} red, green, and blue colour components from their nearest boundary (i.e., 0 or 1). The mean boldness (across 100 simulation runs) of coevolved signals was sensitive to changes in mutation operator in the direction predicted by the analyses presented in this paper. Whilst the Absorb operator ($\bar{b} = 0.87$, $\sigma_{\bar{b}} = 0.006$) resulted in coevolved signals which were significantly bolder than those coevolved under either the Flat or Repeat regimes ($\bar{b} = 0.81$, $\sigma_{\bar{b}} = 0.004$, and $\bar{b} = 0.82$, $\sigma_{\bar{b}} = 0.004$, respectively), all three mutation operators generated signals which were bolder than random signals ($\bar{b} = 0.5$).

These results demonstrate that whilst the simulation is sensitive to the different characteristics of different mutation operators, mutation bias is *not* the underlying cause of the boldness of the evolved signals. While Enquist and Arak's original hypotheses may in fact be correct, an alternative explanation is that bold signals exploit an inherent preference for extreme-valued inputs on the part of the simple artificial neural networks with which they coevolve [14, 2].

6 Conclusion

The potential for a mutation operator's inherent biases to influence the dynamics of evolutionary simulation models was demonstrated for a range of prevalent mutation operators. Although these mutation biases may be the source of artefactual results, they are also part and parcel of natural selection. Rather than attempt to eliminate them from evolutionary simulations, their presence should be tolerated and controlled for. Greater understanding of the role of mutation bias in shaping the evolutionary dynamics of both natural and artificial evolutionary systems can only increase our understanding of the parallels and disimilarities between artificial and natural evolutionary processes.

Acknowledgments

Thanks to Valerie Chase, Martin Lages, Laura Martignon, Jason Noble, Peter Todd, Henrietta Wilson and two anonymous referees for helpful comments.

References

1. Enquist, M., Arak, A.: Symmetry, beauty and evolution. Nature **372** (1994) 169–172
2. Bullock, S., Cliff, D.: The role of 'hidden preferences' in the artificial co-evolution of symmetrical signals. Proc. Roy. Soc. Lond., *B* **264** (1997) 505–511
3. Schaffer, J.D., ed.: Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kauffman, San Mateo, CA (1989)
4. Bäck, T.: Self-adaptation in genetic algorithms. In Varela, F., Bourgine, P., eds.: Toward a Practice of Autonomous Systems, Cambridge, MA, MIT Press (1992) 263–271
5. Bäck, T.: Optimal mutation rates in genetic search. In Forest, S., ed.: Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, CA, Morgan Kauffmann (1993) 2–8
6. Maley, C.: The coevolution of mutation rates. In Morán, F., Moreno, A., Merelo, J.J., Chacón, P., eds.: Advanced in Artificial Life, Berlin, Springer Verlag (1995) 219–233
7. Gray, F.: Pulse code communication. U.S. Patent **2 632 058** (1953)
8. Caruna, R.A., Schaffer, J.D.: Representation and hidden bias: Gray vs. binary coding for genetic algorithms. [3] 153–161
9. Press, W.H., Teukolsky, S.A., Vetterling, W.Y., Flannery, B.P.: Numerical recipes in C: The art of scientific computing. CUP, Cambridge (1992)
10. Leigh, E.G.: Natural selection and mutability. Am. Nat. **104** (1970) 301–35
11. Leigh, E.G.: The evolution of mutation rates. Genetics **73** (1973) 1–18
12. Taddel, F., Radman, M., Maynard Smith, J., Toupance, B., Gouyan, P.H., Godelle, B.: Role of mutator alleles in adaptive evolution. Nature **387** (1997) 700–702
13. Sniegowski, P.D., Gerrish, P.J., Lenski, R.E.: Evolution of high mutation rates in experimental populations of *E. coli*. Nature **387** (1997) 703–705
14. Dawkins, M.S., Guilford, T.: An exaggerated preference for simple neural network models of signal evolution? Proc. Roy. Soc. of Lond, *B* **261** (1995) 357–360