NTNU
Innovation and Creativity

Learning and Evolution in Complex Fitness Landscapes

Ero Stig Karlsen

Master of Science in Informatics
Submission date: April 2007
Supervisor: Jørn Hokland, IDI
Co-supervisor: Sule Yildirim, IDI
Andrew Criswell, Bangkok University

Norwegian University of Science and Technology
Department of Computer and Information Science

Learning and Evolution in Complex Fitness Landscapes

Ero Karlsen
erostig@stud.ntnu.no

April 25, 2007

Problem statement

Assess the current status of the Baldwin effect by a literature review. Investigate the trade-off between learning and evolution in fitness landscapes of different complexity, and with different costs of learning.

Abstract

The Baldwin effect is the notion that life time adaptation can speed up evolution by 1) identifying good traits and 2) by genetic assimilation inscribing the traits in the population genetically. This thesis investigates the Baldwin effect by giving an introduction to its history, its current status in evolutionary biology and by reviewing some important experiments on the Baldwin effect in artificial life. It is shown that the Baldwin effect is perceived differently in the two fields; in evolutionary biology the phenomenon is surrounded by controversy, while the approach in artificial life seems to be more straight forward. Numerous computer simulations of the Baldwin effect have been conducted, and most report positive findings. I argue that the Baldwin effect has been interpreted differently in the literature, and that a more well-defined approach is needed.

An experiment is performed where the effect of learning on evolution is observed in fitness landscapes of different complexity and with different learning costs. It is shown that the choice of operators and parameter settings are important when assessing the Baldwin effect in computer simulations. In particular I find that mutation has an important impact on the Baldwin effect.

I argue that today's computer simulations are too abstract to serve as empirical evidence for the Baldwin effect, but that they nevertheless can be valuable indications of the phenomenon in nature. To assure the soundness of experiments on the Baldwin effect, the assumptions and choices made in the implementations need to be clarified and critically discussed. One important aspect is to compare the different experiments and their interpretations in an attempt to assess the coherence between the different simulations.

Acknowledgements

I would like to thank my supervisor Sule Yildirim for valuable support and encouragement. I would like to thank Ellen Ersfjord for proofreading and for being my girlfriend.

Cover photo by Jill Greenberg [http://www.manipulator.com/].

Contents

List of Figures

1 Introduction

The Baldwin effect suggests that lifetime plasticity can enhance evolution. Since Hinton and Nowlan's seminal paper (Hinton and Nowlan, 1987), a lot of experiments have been conducted confirming the effect in computer simulations. However, in evolutionary biology the Baldwin effect is still controversial, and there is scant evidence for the effect in biological populations. There is also confusion about what the Baldwin effect actually is. This project aims to clarify what the Baldwin effect is and is not, and experiments on how evolution is affected by learning will be performed in adaptive landscapes of different complexity. Lastly, the validity of artificial life (ALife) models for evolutionary biology will be discussed.

1.1 Motivation

The relationship between evolution and lifetime learning has been much explored in ALife research, and most experiments have supported the Baldwin effect. A variety of models have been utilized, ranging from abstract models with closely correlated genotype-phenotype mappings to situated simulations using robots controlled by neural networks.

Usually the experiments have focused on the the synergy between learning and evolution. The cost of learning, which is a prerequisite for the Baldwin effect to occur, has often been downplayed (Turney, 1996). Furthermore, it need not be only learning that contributes to the Baldwin effect, but *phenotypic plasticity* (ontogeny) in general, where development plays a significant part (Downing, 2004).

There is a trade-off between lifetime plasticity and instinct, and general-purpose learning is not the ultimate goal of natural evolution. Under some circumstances, given a high degree of learning in the population combined with a low cost, learning can inhibit selective pressure and reduce the innate fitness in a population, and generally it would be advantageous for a population that the learned behaviour eventually be performed by instinct. Learning might make it possible for a mechanism to arise in incremental steps and is often advantageous in the former steps of evolving a mechanism; in later stages learning might inhibit innate fitness.

Thus, there are several trade-offs between learning and evolution; the right balance depends on several factors, such as selective pressure and the rate of change in the environment. Evolution also guides learning, by determining what the animal will be able to learn (what stimuli will trigger learning) (Bryson and Hauser, 2002).

The general tendency seems to have been an overemphasis on learning in

ALife experiments. For instance Parisi et al. (1992) claimed to show that learning could speed up evolution also when the learning task was not correlated to the evolutionary task. While this claim is largely refuted (Harvey, 1997), it is still a belief held by some, and even reiterated in introductory texts (McLeaod et al., 1998).

The enthusiasm about the Baldwin effect in ALife is markedly different from the way it is perceived in evolutionary biology, where there is still debate about how to define it, what is needed for it to occur, and whether or not it occurs at all. Compared to the disputes in evolutionary biology, it is striking with which ease the Baldwin effect is discussed in ALife literature. While ALife researchers are not necessarily claiming to perform experiments that can inform biology, they are at least utilizing a theoretical construct borrowed from evolutionary biology. To that extent there are important factors in the interplay between evolution and learning that should be discussed, including the importance of the complexity of the adaptive landscape, the degree and cost of learning, the selective pressure, the rate of mutation, the mapping from genotypic to phenotypic space, the complexity of the genome, development, and a variety of environmental factors affecting selection.

1.2 Objective

I will argue that plasticity in general have been overemphasised as a means to boost evolution in artificial life experiments. A number of factors need to be included in a discussion of the Baldwin effect. I will put forward my argument in two steps:

- By assesing the status of the Baldwin effect in current ALife and evolutionary biology research. This involves an attempt at clarifying just what we are (and are not) talking about when we talk about the Baldwin effect. The underlying assumptions made in the computer simulations will be studied.
- Perform computer simulations that study the Baldwin effect where:
  - The fitness landscape is of different complexity
  - Evolutionary and learning parameters are varied.

1.3 Structure

This thesis's goal can be divided into three subtasks:

1. Review literature on the Baldwin effect to gain an understanding of the status of the phenomenon. The review will include both texts in evolutionary biology and computer science. Important questions which will be a starting point for this investigation is:

    - What is the scientific status of the Baldwin effect in evolutionary biology? Is it one agreed upon theory? Is it an empirically observable phenomenon? What are its historical roots?
    - How is the Baldwin effect perceived in computer science? Trace the different approaches to exploring the Baldwin effect in computer simulations. Are computer scientists and biologists talking about the same thing when they talk about the Baldwin effect?
    - Show that there has been a development from emphasizing the benefits of learning, to a stronger focus on the cost of learning and the trade-off between evolution and learning. Show that different researchers have perceived the Baldwin effect quite differently. Also, experimental results hav been interpreted differently.

2. Perform experiments using genetic algorithms in fitness landscapes of tunable complexity to assess the validity of some of the findings on the Baldwin effect in previous experiments. This includes observing how the Baldwin effect is affected by:

    - The complexity of the fitness landscape.
    - Different learning rules.
    - Different costs of learning. Different evolutionary parameters.
    - The genotype-phenotype mapping.

3. Discuss the results and the validity of the models. Put forward suggestions on how the soundness of computer simulations on the Baldwin effect can be improved.

1.4 Conclusions

The Baldwin effect is not an empirically observable phenomenon, and the term means different things to scientists from different backgrounds. In evolutionary biology there is considerable debate as to what the phenomenon is, whether it exists at all, and how it relates to other important evolutionary factors. This is in contrast to the more relaxed acceptance of the phenomenon in the ALife community, who has reported positive findings of the Baldwin

effect in a number of computer simulations over the years. However, some of the research seem to have overemphasized the positive role of learning in the Baldwin effect, and many experiments lack a clarification of what they actually mean by the Baldwin effect. To a varying degree the simulations have been ascribed biological relevance, but the underlying assumptions and the simplifications made in the experiments often lack a critical discussion.

In a series of experiments it is shown that the effect and relevance of learning on evolution depends on several factors, including the complexity of the fitness landscape, the initial rate of learning, the learning rule, the cost of learning, the selective pressure and the mutation rate. Learning proved advantageous in more complex fitness landscapes and a higher fitness was reached in fewer generations, but there was persistent plasticity in the population. This was interpreted as a consequence of a lack of genetic variability in the population, probably related to a small population size. I found that higher rates of mutation counteracted this tendency and contributed to a more pronounced Baldwin effect.

Computer simulations represent one source of information about the Baldwin effect, but the level of abstraction compared to natural occurring phenomena is very high, and so they should be interpreted with great care and not taken as empirical evidence. One way of improving the soundness of computer simulations of the Baldwin effect in ALife is to focus on the reliability of the simulations and of the coherence between the interpretations.

2 Background

In this section I will give a short introduction to the historical development of the theory behind the Baldwin effect. Current controversies in evolutionary biology regarding the Baldwin effect will be introduced, and lastly I will review important literature on computer simulations of the Baldwin effect. Assoiciated important terms will be introduced and explained as needed.

2.1 Two perspectives on the Baldwin effect

Because of its counterintuitive nature the Baldwin effect can be difficult to grasp. On first encounter it can also easily be confused with Lamarckism, the idea that an organism's acquired traits can be directly inscribed in its genes – but the similarities are only superficial. In an attempt to make things clearer, I will provide two definitions, one intuitive and one computational.

2.1.1 Intuitive definition

In short, the Baldwin effect states that individual lifetime changes (by learning or other means) can affect evolution. But how can this happen without the acquired characteristics being inscribed in the individual's genome?

If the change an individual undergoes during its lifetime is adaptive, this (generally speaking) gives the individual a better chance of reproducing. While the individual undergoes a change, to successfully aquire a trait it will also often need some innate tendency towards the trait, and will adapt the trait by a combination of nature and nurture. If the trait is beneficial the individual will have a higher chance of mating, which means that its *innate tendency* towards the trait is reproduced. Life time adaptation thus potentially plays a role in picking up on traits that are advantageous.

2.1.2 Computational definition

In computational terms the Baldwin effect is often explained as a 'smoothing' of the fitness landscape. The concept of a fitness landscape, or adaptive landscape, is often used when describing genetic algorithms, and I will give a more detailed introduction in the part describing the implementation (see section 3.2). Still, evolutionary computation is just one machine learning method among others, and 'fitness landscape' means nothing more nor less than the more traditional AI concept of a search space.

When a learning rule is applied in combination with an evolutionary algorithm, in computational terms one is using a local search inside a global

search. This means the search is more fine-grained and have the potential of picking up subtler solutions which the global search might overlook. But, the solutions are not retained by the local search; indeed, they literally die with it during the selection process. What the local search does though, is give a lookahead into the search space of a particular candidate solution. If the surrounding search space yields a good solution, and the individual moves in that direction during its lifetime, it is more likely to be selected. And because it is the individual's genome *in accordance* with the learning rule that has moved the individual in that direction, there is a fair chance the genome, by genetic operators such as recombination and mutation, is changed slightly to transform the individual's offspring into a slightly better candidate solution from birth.

This way the learning rule implicitly informs the evolutionary algorithm's global search about areas in the search space which the evolutionary search might otherwise have overlooked.

2.2 A short history of learning and evolution

2.2.1 The ambiguity of the Baldwin Effect

Since Hinton and Nowlan's seminal experiment in 1987 (Hinton and Nowlan, 1987), numerous artificial life experiments investigating the Baldwin effect have been performed, most of which report positive findings. It is often unclear whether the experiments are seen as relevant for evolutionary biology, or if they are simply investigating a way of boosting evolutionary search. Usually though, some references are made to biology.

Hinton and Nowlan's original experiment was backed up by the evolutionary biology authority John Maynard Smith in an article in Nature (Maynard Smith, 1987; Depew, 2003). The philosopher Daniel Dennet (2003) also sees this early experiment as a sort of empirical evidence that the Baldwin effect is a natural occurring phenomenon.

In the following, though, we shall see that the Baldwin effect is by and large not well established nor well understood in evolutionary biology. According to (Downes, 2003) the various defenses and versions of the Baldwin effect can generally be seen as attempts to expand the explanatory repertoire in evolutionary biology. There also seems to be confusion as to whether the Baldwin effect is a natural occurring phenomenon or an explanatory mechanism. This is in accordance with Depew's (2003) claim that the Baldwin effect is neither a theory-neutral empirical phenomenon, nor a unified or agreed upon theoretical concept.

According to Downes (2003) there is a lack of empirical evidence for the

Baldwin effect, and though he admits the concept might be entirely theoretically grounded, he criticizes some authors for assuming the Baldwin effect to be an explanatory mechanism without pointing out what phenomena calls for such an explanation. While some authors in evolutionary biology recognize computer simulations as potentially interesting indications of possibilities in biology, computer simulations are generally disregarded as empirical evidence for the Baldwin effect. Since the concept was introduced, it seems to have gone through a Promethean development, changing shape under the interpretation of different scientists.

2.2.2 The Baldwin effect, according to Baldwin

Baldwin introduced his idea about the relationship between learning and evolution, or rather intelligence and instinct, at a time where Lamarck's notion of the heredity of acquired traits had strong hold. Darwin himself was ambiguous towards Lamarckism, but Baldwin rejected the idea; and in this sense his theory was more Darwinian than Darwin's own in this particular respect. While Baldwin's main interest was human behaviour and evolution, he aimed at a theory that was general enough to describe evolution for all species (Downes, 2003).

Similar ideas was presented at the same time by Lloyd Morgan and Osborn, and there is some controversy about the relative contribution of each author (Godfrey-Smith, 2003). Still, for simplicity I will refer solely to the Baldwin effect in the following.

Baldwin's original argument has the following steps (Depew, 2003; Baldwin, 1896):

1. There exists 'ontogenetic adaptations,' that is adaptations through an organism's lifetime. These are ways of exploring the environment. The adaptations are produced by physical, neurological or more intelligent forces such as imitation, pain/pleasure reinforcement or even means-end-reasoning.

2. Ontogenetic adaptations adapt the instincts that permit them to changing and stochastic environments.

3. Through ontogenetic adaptations the individual increases its chances of survival. Through the ability to adapt through lifetime the individual can withstand environmental challenges (including, but not exclusively, environmental change).

4. Ontogenetic adaptiveness increases an organism's chances of reproducing.

5. In certain species ontogenetic adaptability is enhanced through 'social heritability'. Cultural knowledge is transferred to new generations.

6. Adaptations through social heritability can be maintained indefinitely, but might turn into instincts if germinal[1] elements coincide with their influence.

7. Newly evolved congenital[2] instincts forms the base for further ontological adaptation.

After Baldwin's original proposal the theory has been presented, interpreted and elaborated on in a number of ways. Godfrey-Smith (Godfrey-Smith, 2003) summarizes the Baldwin effect succinctly by dividing it into three main stages:

1. A new environmental condition arises.

2. Natural selection favours plasticity for the population to be able to adapt to the new environment.

3. Through mutation, recombination and selection the genotypes that can produce the best behaviour without plasticity will proliferate in the population.

In the following I will use Godfrey-Smith's three stages as a reference point.

2.2.3 Baldwin and the modern evolutionary synthesis

Baldwin's original theory of evolution relied on a view of natural selection as a relentless life-and-death struggle. The 'modern synthesis' integrates Darwin's classic theory with Mendel's theory of genetics and mathematical population genetics. It diverts from the classical view of evolution in a number of ways (Depew, 2003):

- Physiological and behavioural adaptiveness is not adaptation, but an instantiation of genetically based inherited adaptations.

- Natural selection is based on mean reproduction rates, and only indirectly based on the life and death of individuals.

[1]Germinal: relating to the origin of the animal, i.e. fetus.
[2]Congenital: present at birth.

- The alleles code for phenotypes on the organismic level[3], not the germinal.
- Evolution by natural selection is not inherently directional.

From the perspective of the modern synthesis, the Baldwin effect cannot be understood the way it was initially proposed. According to the modern synthesis there is unlikely to be any selective pressure for traits that are effectively transferred over generations by social inheritance. If learned behaviours do become genetically underwritten, in the view of the modern synthesis this would mean that a population would be swapping short term plastic behaviour for rigid long term adaptations – thus subverting the very point of the Baldwin effect (Depew, 2003).

The modern synthesis views culture as much more autonomous than did Baldwin in his original proposal (and many of his proponents today). Culture is seen as natural selection's greatest achievement, but this also implies there is no patch back from culture to instinct (Depew, 2003). To elaborate on this we now turn to the second part of the Baldwin effect, i.e. how development of new traits by ontogenetic plasticity relates to genetic assimilation.

2.2.4 The Baldwin effect and genetic assimilation

Waddington (Waddington, 1953; Stearns and Hoekstra, 2000) demonstrated that for some traits organic plasticity (due to in uterus environmental stimuli) would produce phenotypes that eventually, over generations, would be genetically encoded in the population. Waddington saw genetic assimilation as nothing more than a combination of traditional Darwinism and embryology that can give results that look like inheritance of acquired characteristics; he did not consider genetic assimilation 'a new factor in evolution'. Waddington did not experiment with learning, but manipulated the developmental process of individuals, thus influencing the development of certain traits.

In his original experiment Waddington (1953) showed that fruit flies (*Drosophila*) embryos exposed to ether at a particular stage would develop wing-like halters. These were then (artificially) selected for reproduction, and after twenty generations the wing-like halter phenotype had been genetically encoded in the population, so there was no longer a need for exposure to ether for the trait to develop. Phenotypes that through lifetime adaptation mimic a genetically produced phenotype are called *phenocopies*. Waddington's observations have been confirmed in experiments demonstrating that

[3]The organismic level relates to the grown up animal.

environmental stimuli such as heat or cold can develop phenotypes of butterflies with particular wing patterns (Waddington, 1953; Stearns and Hoekstra, 2000).

Genetic assimilation can be divided into three steps:

1. Some factor in the environment (usually embryonic factors) change the phenotype in a certain way.

2. The changed phenotypes are selected for (either by artificial or natural selection).

3. Over generations the trait is encoded in the genome (it becomes instinctive), and the environmental stimuli is no longer needed for the trait to develop.

The last step is known as genetic canalisation. An essential prerequisite for genetic assimilation to take place is a correlation between the phenotypic and genotypic space. Waddington's genetic assimilation describes how phenotypic adaptation can canalise a trait genetically – but it is not the plasticity itself that facilitate the canalisation. While it is true that learning can be seen as a local search in the phenotypic space, this does not imply that the phenotypic variety discovered by learning has corresponding genotypes nearby the genotype of the individual. Mayley (1996b) coins the needed correlation between genotypic and phenotypic space 'neighbourhood correlation', and defines it as the correlation between the distance travelled in phenotypic space to a new learned phenotype and the distance moved in genotypic space for the same phenotype to be generated without learning (Mayley, 1996a). So, while Waddington showed that genetic canalisation can happen, this does not imply it will happen in all situations.

We shall see that in many of the computer simulations of the Baldwin effect, this correlation is not accounted for, and even that the lack of it is interpreted as a special case of the Baldwin effect (see section 2.3.4). In many of the simulations there is no established correlation between the *plasticity* itself and the trait(s) the plasticity enables the phenotype to attain. Also, several authors have referred to genetic assimilation and the Baldwin effect as the same phenomenon, which they are not (Downes, 2003). In short, there is no obvious reason why plasticity – in itself – would indicate that an individual's genotype is closer to the 'ideal' genotype that codes more directly for the trait. If phenotypic fitness is to inform the evolutionary search, there must be a certian degree of correlation between the genotype and the phenotype.

2.3 Computer simulations of the Baldwin effect

2.3.1 Artificial life

The term artificial life was coined in 1987 by Chris Langton, and originated primarily as a means to simulate biological systems. ALife has its root in cybernetics (the theory of control and communication in the animal and the machine), and according to Risan (1997) also traditional AI (GOFAI) was born from the cybernetic movement, of which it 'rejected the holism of the systemic perspective and emphasized the formal and logical aspects of human cognition' (Risan, 1997, p. 9). In that respect, today's ALife can be seen as a realignment with the early cybernetic movement, since it emphasizes bottom-up modelling and decentralized and interconnected systems. ALife is an interdisciplinary field, including sciences such as biology, psychology, philosophy, ethology and computer science (Risan, 1997).

In addition one might talk about three main directions within ALife: 1) engineering focused methods that borrows biological principles, but where the focus is on improving system efficiency; 2) a scientific modelling tradition, which is concerned with simulating naturally occurring phenomena; and 3) a direction more concerned with what constitutes life in general, often promoting the idea that life does not necessarily need to be biological.

The works to be discussed in the following are mainly in the second category. The focus is, whether explicitly stated or not, mainly on modelling evolution in biology, and the efficiency of the implementations are seldom discussed.

2.3.2 Hinton and Nowlan – the needle in the haystack

Hinton and Nowlan's (1987) paper is regarded as the first demonstration of the Baldwin effect in computer simulation. Probably, much of the paper's impact outside artificial life circles can be attributed to the evolutionary biologist John Maynard Smith's presentation of the paper in *Nature* (Maynard Smith, 1987; Depew, 2003).

The set-up of the experiment is as follows. A genome represents 20 connections in a feedforward neural net. The alleles consists of 0, 1, ?, where 0 means there is no connection; 1 means there is a connection; and ? means the connection value is undecided and will be set through life time learning. See figure 1 (Hinton and Nowlan, 1987).

There exists only *one* net configuration with optimal fitness (a net consisting of all 1s) – all other configurations have equally low fitness. Since each net has 20 connections, if only the final alleles, 0 and 1, are considered, there are 2^{20} possible genotypes, which yields a relatively large search space

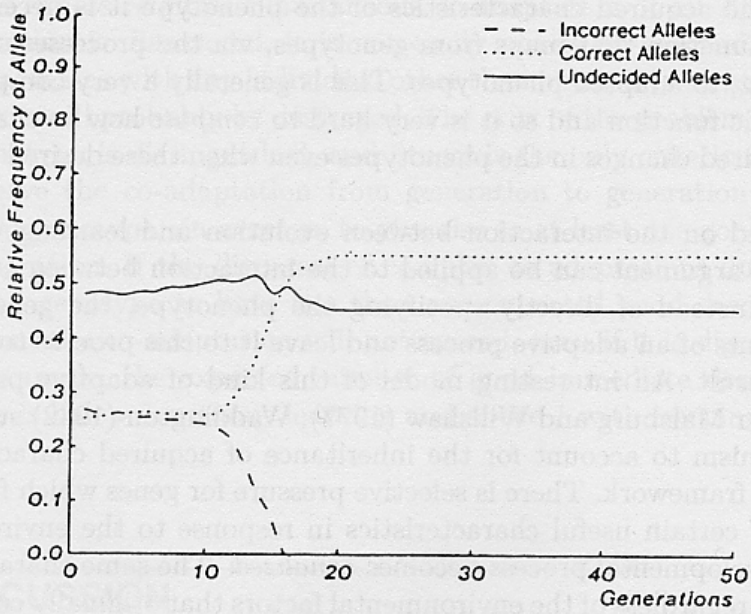

FIGURE 2 The evolution of the relative frequencies of the three possible types of allele. There are 1000 organisms in each generation, and each organism performs 1000 learning trials during its lifetime. The initial 1000 genotypes are generated by selecting each allele at random with a probability of 0.5 for the ? allele and 0.25 for each of the remaining two alleles. A typical genotype, therefore, has about ten decisions genetically specified and about ten left to learning. Since we run about 2^{10} learning trials for each organism, there is a reasonable chance that an organism which has the correct genetic specification of ten potential connections will learn the correct specification of the remaining ten. To generate the next generation from the current one, we perform 1000 matings. The two parents of a mating are different individuals which are chosen at random from the current generation. Any organism in the current generation that learned the good net has a much higher probability of being selected as a parent. The probability is *proportional* to $1 + 19n/1000$, where n is the number of learning trials that remain after the organism has learned the correct net. So organisms which learn immediately are 20 times as likely to be chosen as parents than organisms which never learn. The single offspring of each mating is generated by randomly choosing a cross-over point and copying all alleles from the first parent up to the cross-over point, and from the second parent beyond the cross-over point.

Figure 1: Hinton and Nowlan's experiment (Hinton and Nowlan, 1987).

with over one million possible combinations. We are indeed dealing with a needle in a haystack search, for there is only one single spike in the fitness

landscape.

Two experiments were performed, one with and one without learning. The population size was 1000, and for each generation 1000 matings (crossover) were performed (i.e. the whole of the current generation was replaced by the new one). No mutation was used. See figure 1 for further details.

In the experiment without learning the population never reached a higher fitness than its starting point, but when learning was introduced, the solution was reached in about 15 generations. The explanation lies in the set-up, with its particular landscape and fitness function. Fitness is determined by the following function, F = 1 + 19 * n / 1000, where n is the number of learning trials remaining after the goal string is encountered. When the function is applied to an ordinary evolutionary search without learning, the search becomes purely hit-and-miss, no better than a random search; since only the single right solution is ascribed a higher than usual fitness there is *no selection pressure*.

The situation changes when learning is introduced. Initially 50% of the alleles are plastic, and during each generation an individual is allowed 1000 learning trials. Learning is conducted by randomly setting the plastic alleles in the phenotype to either 0 or 1 and then testing it for fitness. Learning is stopped once the correct goal string is encountered.

Firstly, the high amount of learning introduced gives learning a fair chance to find the target string by random search; secondly, because the fitness function rewards phenotypes with fewer learning trials higher, this gives an indirect approximation of the genotype's fitness, since phenotypes which are initially closer to the goal string will tend to use fewer trials.

In this perspective, it is not surprising that the algorithm is able to find the optimum using learning, because at the instant an individual gets a higher fitness score than another, the evolutionary algorithm is guided, and selective pressure is introduced. Thus, learning's main function in this experiment is to introduce diversity in the fitness scores in the population, thereby enabling selective pressure which will guide the evolutionary search.

From a purely computational viewpoint, a simple random search would be much more effective for this fitness landscape. What Hinton and Nowlan's experiment illustrates though, is that learning can help identify good evolutionary solutions and inform the evolutionary search. In that resepct the experiment is no doubt elegantly set up to illustrate a point – but in a senese the extreme set-up makes the already blind watchmaker even blinder. Therefore its relevance for evolutionary biology is probably limited. In particular what makes the model unrealistic is that a phenotype cannot be found by incremental improvement. As (Sterelny, 2004) points out few developments in nature tend to be genuine spikes. In his words, Hinton and Nowlan's

model is 'not just oversimplified, it is positively misleading' (Sterelny, 2004, p. 297). The unimodal [4] quality of the landscape is also an implausible model of evolution in nature.

To summarize, because of the structure of the landscape in Hinton and Nowlan's (1987) model, learning plays two different roles:

- In the first phase, the genetic algorithm has no information to guide its search. In effect, the genetic algorithm is performing a pure random search. Since for each generation 1000 learning trials are performed for an average of 50% of the phenotype, learning is much more likely than evolution to stumble upon 'the good trick'.

- Once learning has found the good trick (usually after about 15 generations in this set-up), the genetic algorithm performs an informed search, and the average fitness in the population rises. With genotypes increasingly closer to the goal string, they will be selected for since a cost is associated with learning.

While Hinton and Nowlan's (1987) intention was probably to illustrate a point, which they elegantly do with their set-up, it is probably misleading to see their experiment as a realisitc representation of evolution in nature.

2.3.3 When the learning task and the evolutionary task are not correlated

Parisi et al. (1992) suggested that an instance of the Baldwin effect also occurs when the learning task and the evolutionary task are not correlated. In Hinton and Nowlan's (1987) model there was a close resemblance between genotypic and phenotypic space; in fact the genotype and the phenotype had identical representations. Several models have been suggested where there is a larger gap between genotypic and phenotypic space, and often neural networks models have been used, where the genotype have coded for the set of weights in the networks.

This is also the approach taken by Parisi et al. (1992). They simulated a foraging environment with evolved agents situated in a 2-D grid world. The agents receive sensory information from food items randomly placed in the grid. For each time step an agent can either turn or move forward from its current position. Any time an agent reaches a square with a food item, its fitness is increased and the item removed. The agents are controlled by a feed forward neural network with initial weights set randomly.

[4] See section 3.2.

Only one agent operates in the grid world at a time, so for each generation the foraging has to be repeated for each agent. In other words there is no co-evolution. There are 100 individuals in each generation, and the 20 best individuals are selected for reproduction. This is done by copying an individual's genome and applying a mutation operator. There is no crossover of genomes between individuals. The genome is the set of weight values for the connections in the neural network. The results showed that the agents' fitness increased steadily over generations (Parisi et al., 1992).

In a second experiment a learning task was introduced: to predict how an agent's sensory information about a food element will change after the agent has executed its action. This is implemented by adding two prediction units to the network's output layer. Given the current input and the network's current output (i.e. its planned action), the prediction units are trained by backpropagation to predict the sensory input in the next time step (Parisi et al., 1992)

Parisi et al. (1992) found that the agents in the second simulation got better at finding food, even though the learning task and the evolutionary task were not the same. While there was an increase in both lifetime fitness and inherent fitness, the increase in inherent fitness cannot be wholly attributed to the effect of learning.

To explain the effect Parisi et al. (1992) ask us to imagine a fitness landscape with two different individuals at different points in the landscape, but with the same fitness. When an individual is selected for reproduction its genome is mutated, thereby moving it to a slightly different point on the fitness landscape. Until the next generation is selected evolution has no way of knowing whether this point is better or worse than the current point. But as learning changes the connection weights of the individual this is a way of exploring the surrounding fitness landscape of that individual. This means that if the individual is situated nearby an even higher fitness peak, this 'good location' will be identified, and mutation will have a higher probability of placing the offspring of that individual in a location with higher fitness, thus leading to progressively higher innate fitness. In other words they claimed that a smoothing of the fitness landscape occurred also when the correlation between the learning task and the evolutionary task was weak.

The authors go on to show that even learning which is a random search (i.e. where the learning might render the individual's fitness worse than before learning) will boost evolution, though not as much as with informed learning. Again, Parisi et al. (1992) claim this would inform evolution about which individuals are at the better locations in the fitness landscape when the surrounding landscape is also taken into consideration. They hypothesise that informed learning is better because it operates in a weight space which

share some characteristics with the the evolutionary space, thereby giving a slight guidance also to the evolutionary search.

To test this hypothesis the XOR problem was introduced as the life time task to learn. The expectation was that this would not affect the learning ability of successive generations, but also this learning task turned out to improve innate fitness. This was explained by the notion that two arbitrary tasks, one evolutionary and one learning task, could be accidentally correlated on some sub-regions of the weight space, thereby causing this effect.

Thus, Parisi et al. (1992) claim to simulate the Baldwin effect both on tasks that are correlated to the evolutionary task and tasks that are not so. However, Harvey (1996) claims that this is not an example of the Baldwin effect, but something else – in his terms 'Another new factor (ANF)'. Using a geometric analysis, Harvey shows that the effect has to do with how weights in a neural network are perturbed by mutation and then restored through relearning. The conditions needed for this to occur are very restricted, and so the experiment of Parisi et al. (1992) should not be generalized to imply that the Baldwin effect will occur for learning tasks that are totally unrelated to the evolutionary task. Rather, the finding is a side effect to the peculiarities of artificial neural networks (Harvey, 1997). Other experiments have also shown that learning does not speed up evolution if the learning task is not related to the evolutionary task (Menczer and Belew, 1994).

The model also lacks an important aspect of the Hinton and Nowlan (1987) model, namely the evolution of plasticity itself. In Parisi et al.'s (1992) model, learning is intrinsic to the model; it is not evolved. The agents will always learn, whether learning is "called for" by evolution or not.

To summarize:

- The fact that tasks uncorrelated to the evolutionary task seem to fascilitate evolution, is probably not an instance of the Baldwin effect, but can be explained by the peculiarities of ANNs.
- While uncorrelated learning *might* boos evolution, this is highly dependent on the particular landscape to be searched.
- The model does not account for the evolution of plasticity.

2.3.4 Learning and evolution in a simulated ecological system

Ackley and Littman (1992) implemented a more advanced ecological grid-world with carnivores, plants, trees and walls. Agents living in the world are governed by two ANNs, one *action network* and one *evaluation network*. The action network controls the agent's behaviour, while the evaluation network

represents the agent's goals, and so also applies the feedback for the training of the action network. An important property of the model is that the action network is only modifiable during the agent's lifetime (by reinforcement learning), while the evaluation network can only be modified genetically through evolution. Thus, unlike in the model of Parisi et al. (1992) learning and evolution operates on two different representations.

Each agent has a fitness component which is increased by eating plants or dead carnivores in the environment, and a health component that is decreased for instance by hitting a wall, and which increase spontaneously with time. When an agent has a high enough fitness it is selected for mating with another agent nearby. Thus, the fitness is endogenous to each agent. Reproduction is done by recombination and a low degree of mutation (Ackley and Littman, 1992).

Ackley and Littman (1992) found that the agents achieved the highest fitness when both learning and evolution was used. Though evolution and learning combined only proved better after 100 000 time steps. When taking the most successful agent and letting it run for more than the one million time steps, which was the ordinary limit, they found that genes that were important for survival tended not to mutate as much. They also observed the Baldwin effect in that evolved characteristics tended to mimic learned ones, but it was not a very pronounced effect (Ackley and Littman, 1992). Probably, this is related to the different representations used for the genotypic and the phenotypic space, and therefore the lack of correlation between the two spaces.

2.4 Trade-offs between learning and evolution

There has probably been a tendency to overestimate the importance of learning in the early experiments on learning and evolution. Later experiments and theories indicate that there are many trade-offs between learning and evolution that need to be considered. Also, in some situations evolution can guide learning just as much as learning can guide evolution. In the following I will review some important papers on the trade-off between learning and evolution.

2.4.1 The cost of learning

Turney (1996) points out that the Baldwin effect is surrounded by many myths in the ALife community. In most experiments the benefits of the synergy between learning and evolution has been emphasized, but learning also

has costs. There also exists several misunderstandings about the relationship – or lack thereof – between the Baldwin effect and Lamarckian inheritance.

An important notion is also that learning has come to be identified with the Baldwin effect, while the effect really is related to *phenotypic plasticity in general*. Learning is an instance of phenotypic plasticity, but there are many other ways an organism's phenotype can change during its lifetime that do not involve behavioural changes; one example would be the human ability to tan in sunny environments. It is important to remember that the Baldwin effect is a theory of the way *any* phenotypic change can influence evolution, not just learning (Turney, 1996).

When the Baldwin effect has been explored in evolutionary computation the focus has usually been how learning can benefit evolution. This is done by the smoothing of the fitness landscape that constitute the Baldwin effect as interpreted in computational terms.

However, there are situations where phenotypic plasticity could also inhibit the evolutionary search. Phenotypic plasticity can be timely and costly (because of metabolism). For instance, in a sunny environment it would be better to be born with dark skin than to have to develop it through tanning. Furthermore the experimentation involved in learning could potentionally be very costly; being instinctively afraid of snakes is advantageous for most species. This is in accordance with Bryson et al.'s (2002) observation that even highly adaptable species, such as primates, show persistent failures to learn. Such failures of learning might actually be an adaptive strategy; the animal's behaviour might reflect a local optimum, an "adaptive island" surrounded by ineffective or even dangerous behaviour strategies (Bryson and Hauser, 2002). Thus, the animal's failure to learn functions as a safety mechanism, securing that the animal stay put on it's island of behaviours – a mechanism Bryson et al. (2002) coin 'safe learning'.

According to Bryson et al. (2002), many AI researchers have a naive, pre-Darwinian conception of learning: it is seen as general, ideal, without limits, and evolution's ultimate goal. Instead, Bryson et al. (2002) suggest that learning in nature is generally restricted and specialized to particular tasks. An animal cannot learn to associate *any* stimuli with *any* behaviour; on the contrary, which behaviour can be paired with which other behaviour is species specific. For instance, pigeons cannot be trained to peck to avoid shock or to flap their wings to get food; but they *can* be trained to flap their wings to avoid shock and peck to get food. The animal's learning is biased towards information likely to be relevant. Neurophysiologic findings also support this; for instance, poison avoidance in rats is tied to a specific learning mechanism in the olfactory section of their amygdala (Bryson and Hauser, 2002).

In this perspective, learning is evolution's last resort, there is selective pressure for genetic coding to replace individual learning. According to Bryson et al. (2002) the only reason for learning to persist is when the animal is required to adapt to changes on a less than evolutionary time scale, i.e. during the animal's lifetime.

Still, the fact remains that quite a few animals, such as dolphins or humans, rely heavily on learning. In such cases the animals have come to inhibit a niche in which a greater capacity for learning *was* advantageous. Often, animals with complex social structures, such as primates, require learning (though in the insect world there are many examples of social structures that do not require learning). Also, some genetic characteristics are selected for primarily to enhance the animal's sexual attractiveness, and might be counter-adaptive in all other areas, even contributing to the extinction of species(!).

I will now try and identify factors that contribute to the trade-off between learning and evolution, starting with the benefits of learning:

- Temporal adaptability. If there are significant environmental changes during the time-scale of a generation, individuals that are able to adapt during their lifetime will have an selective advantage and increase the fitness of the population (Mayley, 1996b; Turney, 1996; Bryson and Hauser, 2002).

- Organic selection. This is the Baldwin effect in the classic interpretation. Lifetime adaptability helps evolution identify the fitter individuals.

- Less complex genotype. If the environment is relatively predictable, information can be 'stored' in the environment and accessed through learning instead of being stored in the genotype (Todd, P. and Miller; Mayley, 1996a).

- Preserves genetic variation. Because individuals with different genotypes through lifetime adaptation can attain the same fitness, the selective pressure is relaxed, resulting in a higher genetic diversity in the population (Mayley, 1996b).

- In an extension of the above, if a trait cannot be represented genetically, it might still be possible for the population to achieve the trait through lifetime adaptation (Mayley, 1996b).

Mayley (1996b) also identifies several different costs of learning:

- An individual born with a trait will reach higher fitness faster than an individual who has to spend time learning the trait.

- Delay of reproductive ability. Typically, in a species individuals will not be able to reproduce until after development/important learning periods.

- Energy spent. Learning requires more energy because of the associated behavioural and metabolic costs. Also, there might be increased ontogenetic costs, as a more complex organisms are needed for learning to take place.

- Like Bryson (2002), Mayley (1996b) comments on the dangers of learning. Firstly, learning is more stochastic than evolution. For instance, learning can be prevented if an important stimuli in the environment is lacking, thereby preventing the individual from acquiering an important trait; or the individual might simply learn to do the wrong thing. In highly plastic species, such as humans or chimpanzees, psychopathology might be explained by 'learning gone awry'.

- Dangerous behaviour. When an individual encounters a situation for which it has not yet learnt the right response, it might behave dangerously. Examples would include eating poisonous food or closing in on a snake 'to see what happens'.

- Population costs. Costs that affect not only the individual but also the population as a whole include time and energy spent by parents or other members of the population to teach younger or more inexperienced members of the population, and the increased genome length that might be needed to represent the learning ability. Note that this is in contrast to the suggestion by Todd and Miller (1991) that learning can decrease the genome length, so there is a trade-off between these two effects.

Turney (1996) identifies several factors that contribute to the trade-off between learning and evolution:

- Time scale of environmental change. Some environments change too fast for evolution to adapt; if so, learning will be beneficial.

- Variance and reliability. Learning requires the right kind of experience, which might not be available; therefore learning is more stochastic than instinct.

- Energy consumption. Because learning requires acquisition of data, an organism or agent must expend energy in order to learn.
- Length of learning period. An organism is more vulnerable before it has fully learned a behaviour. Therefore, all other factors being equal, evolution will select for shorter learning periods.
- Global vs. local search. Evolution can be seen as a form of global search, while learning can be seen as a local search. The right combination depends on the properties of the fitness landscape.
- Fitness landscape. Learning can smooth the fitness landscape (the Baldwin effect), but this smoothing might not be necessary.
- Reinforcement vs. supervised learning. Evolution can be seen as a form of reinforcement learning; while lifetime learning can be seen as supervised (because of relatively immediate feedback from the environment).
- Bias direction. A certain bias of learning, a direction, is a prerequisite for learning. An instinct-based agent will have a stronger bias than a more learning-based agent (the learner-agent will consider more hypotheses than the instinct-agent). If the bias is correct, the agent will benefit from an instinctive approach, otherwise learning is better.
- Global vs. local goals. The immediate goals of learning might not be in accordance with the goal of evolution (i.e. maximize fitness). The goals at the learning level must be simplified to more immediate sub-goals.

The role of learning will vary with different sequences in the evolutionary process. In 'times of evolutionary fitness' individuals who has the ability for lifetime adaptation might be able to reach fitness levels others cannot and be selected for (Mayley, 1996b). But, this is not straightforward, as a high degree of learning might decrease genetic variation, which would make it harder for a population to adapt to a changing environment, as noted above. In a relatively stable environment selective pressure will inhibit learning and select for genotypes with higher fitness.

To summarize: Many ALife experiments claiming to investigate the Baldwin effect have focused on the benefits of learning combined with evolution. But the Baldwin effect is more than the synergy between learning and evolution. The costs of learning and genetic assimilation also need to be taken into consideration. While there is evidence that learned behaviours can facilitate

the evolution of physical structures, learning can be expensive, and evolution will select for the best balance between learning and instinct. Under certain circumstances learning will not boost evolution, but slow it down. If a species so to speak can get away with instinct, plasticity will not be selected for. There are many examples in nature species that relies little on learning. Snakes and crocodiles are examples of highly successful species that do not need high degrees of learning to succeed in their ecological niches.

2.4.2 Cost of learning and genetic assimilation

Mayley (1996b) puts forward much the same argument as Turner (1996) when he points out that most of the research on the Baldwin effect has focused too narrowly on the benefits of learning. In a two-step conception of the Baldwin effect[5], learning corresponds to the first step, while the costs of learning is a prerequisite for the second step, genetic assimilation, to take place.

As we have seen in the previous section, under certain conditions learning can slow evolution down, and there will always be a trade-off between the costs and the benefits of learning. For instance, given a situation where individuals can adapt well through learning and the costs of learning are minimal, there will be less selective pressure for evolution to favour the individuals who would have a innately high fitness. Mayley (1997) coins this antipode of the Baldwin effect "the Hiding effect", and shows in computer simulations that when both cost of learning and epistasis is low[6], the Hiding effect will dominate; while if the cost of learning and epistasis is high, the Baldwin effect will be more pronounced (Mayley, 1997).

Mayley (1996b) lists two prerequisites for the Baldwin effect to take place:

- Learning (or plasticity in general) must have a cost. If there is no such cost, there will not be any selective pressure for the learned trait to become innate (genetic assimilation).
- There must be a neighbourhood correlation between genotypic and phenotypic space.

The conception of learning as a local search in phenotypic space which can guide evolution is the most common explanation of the Baldwin effect in computational terms. This is not always the case though; learning might take place in a search space quite different from the genotypic space. For

[5]As we have seen in section 2.2.2 there might be more than two steps. Suzuki (2004) performs experiments where they observe a three step Baldwin effect; this will be described later.

[6]I.e. the fitness landscape has low complexity.

instance, in Ackley and Littman's experiment (1992), described in section 2.3.4, the evolutionary and the lifetime search takes place in quite different search spaces.

Mayley (1996b) introduces the concept of neighbourhood correlation as a way of formally describing the overlap between the search distance travelled in pehnotypic space versus the distance travelled in genotypic space. Intuitively, neighbourhood correlation is a measure of how closely the genotype to phenotype mapping during lifetime is related to the genotype to genotype mapping over generations. Basically this implies that individuals that are close in genotype space are also close in phenotype space. More formally the concept is defined thus: Given an original phenotype p_i, a trained phenotype p_j and a learning rule L, the distance in phenotypic space is defined as the probability that $L(p_i) = p_j$, multiplied by the number of times the learning rule has to be applied to change p_i into p_j. In the same way, in genotypic space the distance between two genotypes is defined as the number of genetic operations it takes to transform g_i into g_j. The more the distance in phenotypic space is correlated with the distance in genotypic space, the higher the neighbourhood correlation (Mayley, 1996b).

By using a version of the NK fitness landscape (see section 3.3) Mayley (1996b) vary the complexity of the adaptive landscape in different experiments. This is done by setting the degree of epistasis [7] in the landscape. The degree of neighbourhood correlation is defined by a parameter L, which designates the number of loci in the phenotype which are plastic. These loci are distinct and chosen at random. The size of L is negatively correlated with the degree of neighbourhood correlation. At the start of each program run, a lookup table the size of the genome is generated, which indexes each hold L distinct random phenotypic loci.

The learning mechanism is a steepest ascent hill-climb in the landscape. Initially an individual's phenotype is identical to its genotype. A learning operation consists of flipping the L bits and testing the phenotypes fitness; if it is higher the transformed phenotype becomes the current. Learning continues with the next L bits in the lookup table until no higher fitness is gained. The distance travelled in phenotypic space thus is a Hamming distance of size L. The larger L is, the farther away from the original genotype the learning will move the phenotype (Mayley, 1996b).

Learning is also ascribed a cost by subtracting a constant representing the cost of learning multiplied by the number of learning trials from the phenotype. Experiments were performed with genotypes/phenotypes of length 20 and with a population size of 50. The initial generation was constructed

[7] The interaction between alleles in a genome.

by copying a randomly set genotype to all the individuals in the population. All the individuals were then mutated with a probability of 0.05. This was done to simulate a converged population, with the same starting point in genotypic space. The successive generations were generated using standard genetic algorithm techniques. Selection was done by simple linear ranking, crossover probability was 0.7 and mutation was done with an average rate of 0.1 bits per genotype (1.0/genome size per bit). The experiments were run for 150 generations (Mayley, 1996b).

Experiments were run with varying degrees of learning costs, neighbourhood correlation and landscape complexity (epistasis). Mayley (1996b) found that when there was no cost of learning no genetic assimilation occurred, as there was no selective pressure for it to happen. With high degrees of epistasis, learning and evolution together was much more effective than evolution alone in populations with no or low learning costs. When learning costs are set high, learning is penalized, and the population converges faster because of genetic assimilation and selective pressure against learning.

Mayley's (1996b) notion of neighbourhood correlation is in direct conflict with Parisi et al.'s (1992) claim that the Baldwin effect can occur also when the learning task is uncorrelated with the evolutionary task (see section 2.3.4) [8].

In an experiment also utilizing a version of the NK fitness landscape, Suzuki and Arita (2004) observe a three step Baldwin effect. They claim that learning has three distinct roles in different stages of the Baldwin effect.

As in Mayley (1996b) Suzuki and Arita (2004) use the NK fitness landscape, but they modify the genome so that each allele represent a quantitative trait in the range [0.0, 1.0], not the usual binary representation. They also measure the degree of phenotypic variation, which is the absolute difference between an individual's phenotype at birth and its phenotype after the learning trials are completed. A simple gradient descent learning rule is used, which gradually adjust the plastic genes towards fitter genotypes. This is possible since the traits are quantitative, and not binary. It is also worth noticing that learning is not penalized by any explicit cost. Suzuki and Arita (2004) point out that with epistasis there will still be implicit costs associated with learning. This is so because the learning rule is designed so that each plastic trait is assessed independently, and then all the plasticity genes are updated at the same time. As the learning rule adjusts each trait at a time it loses the overall picture and the effect of epistasis. This is an important difference from Mayley's (1996b) experiment where the cost of learning

[8]The misconception seems to have taken hold and has also been repeated in textbooks (McLeaod et al., 1998).

was set explicitly. Also, unlike in Mayley's (1996b) experiment, where the degree of learning was set explicitly before every program run, the degree of plasticity is evolved using a plasticity genome that code for which alleles in the phenotype will be plastic.

In the first experiment, with no epistasis, Suzuki and Arita (2004) observed that after the genetic assimilation had taken place, phenotypic plasticity increased and stayed high, even though this did not increase the individual's phenotypic fitness. In other words some other role of learning must occur after the Baldwin effect (Suzuki and Arita, 2004).

In their second experiment, with epistasis (N=15, K=4), they found that phenotypic plasticity decreased faster. In this case the adjustment of each plastic trait during learning is done independently from its influence on other plastic traits (i.e. one could say learning in this case is blind to the epistasis); we therefore have an implicit cost of learning because of the epistasis between loci. This implicit cost of learning was confirmed by maximising the degree of plasticity – the individuals then achieved a lower lifetime fitness than if they behaved on instinct alone. (Suzuki and Arita, 2004).

Suzuki and Arita (2004) suggest that learning has the following roles:

- First, learning acts as a local search that can aid the more global evolutionary search.
- The search performed by learning eventually become more guided; the phenotypic plasticity decreases while the phenotypic variation increases. The implicit cost of learning due to epistasis has narrowed the area where the search is performed.
- When the phenotypic fitness approaches its optima, genetic assimilation occurs, as evolution will be guided towards this optima because of the costs of learning.
- Suzuki and Arita (2004) find that even when the genotypes has reached optimum learning still persists. They suggest that learning now acts as a way of preventing random mutation of disrupting the population from its optimum.

In later experiments, Suzuki and Arita elaborates on the three steps of the Baldwin effect (Suzuki and Arita, 2007a) and on repeated occurrences of the Baldwin effect through an evolutionary process (Suzuki and Arita, 2007b).

There are some interesting differences to be observed between Suzuki and Arita's (2004) and Mayley's (1996b) experiments. While there are differences

in the set-ups, the experiments share many of the same characteristics. The main difference is that Suzuki and Arita (2004) use quantitative traits (i.e. real numbers) in the genotype/phenotype encoding, while Mayley (1996b) uses the standard binary encoding of the NK fitness landscape. Also, Suzuki and Arita (2004) use a plasticity genome, so that also the degree of learning is evolved, while Mayley (1996b) sets the degree of learning explicitly before each program run. As opposed to Mayley (1996b), Suzuki and Arita (2004) do not include any explicit costs of learning.

One interesting difference between the experiments is the observations and interpretations of cost-free learning. Mayley (1996b) found that when learning was cost-free no genetic assimilation took place. This was attributed to the fact that there would be no selective pressure against learning. Mayley also observed that with high degree of plasticity, individuals tended to rely on learning, and the innate fitness decreased. In contrast, Suzuki and Arita (2004) found that even when there was no epistasis (K=0), and thus no cost of learning, genetic assimilation took place, which is explained as an effect of genetic drift[9]. After the genetic assimilation had taken place, as mentioned, Suzuki and Arita (2004) observed persistent plasticity in the population, which they explain as a mechanism against genetic vounerability caused by mutation; but it is also possible to explain this as another effect of genetic drift, especially given the small population size (20 individuals). In my experiment I observed the same persistance of plasticity, and in section 5.1.5 I will present an explanation that differs from Suzuki and Aritas (2004); I will suggest that countrary to Suzuki and Aritas (2004) explanation, higher rates of mutation will actually make the deselection of plasticty more pronounced.

2.5 What is the status of the Baldwin effect?

In this part I have given a short introduction to the controversies surrounding the Baldwin effect in evolutionary biology, as well as a review of important experiments on the Baldwin effect in the ALife literature. I fear though, that we are not so much closer to a set definition of the Baldwin effect, and neither can we be, as the phenomenon is still so controversial. Indeed, one of the fundamental research questions regarding the Baldwin effect is probably how to define it.

I have shown that the Baldwin effect is perceived quite differently in evolutionary biology and in ALife. The Baldwin effect is often perceived as more unproblematic in ALife, but that might be because the different areas

[9]Random fluctuations in the genome not due to selection pressure.

of research are not talking about quite the same thing. In the next section I will describe the basis for computer simulations of the Baldwin effect, as well as present my own model. The work of (Mayley, 1996b; Suzuki and Arita, 2004; Puentedura, 2003) are all important inspirations for my model.

With my implementation I hope to shed some light on the interaction between learning and evolution in fitness landscapes of different complexity, as well as the importance of the genetic operators used and parameter setting. My model is in no way intended as a solution to unsolved problems regarding modelling of the Baldwin effect; on the contrary I think it will serve to illustrate the complexity of the task, the arbitray and abstract level of modeling, and the long way ahead before a computer model can, if ever, shed light on the Baldwin effect as natural phenomenon.

3 Model

In this section I will describe the basis for the implementation of my model, namely genetic algorithms and the NK fitness landscape, as well as the model itself.

3.1 Genetic algorithms

The idea that principles borrowed from biological evolution could be used as an optimization tool for engineering problems was investigated already in the 1950s. By the end of the 1960s, Rechenberg introduced his evolution strategies; but it was Holland (1992) who first described genetic algorithms as we know them today. Holland saw genetic algorithms as an abstraction of biological evolution, and he introduced the theoretical framework for adaptation using genetic algorithms (Mitchell and Forrest, 1994; Holland, 1992).

Today, genetic algorithms have been applied in a number of fields, including optimization, automatic programming, machine and robot learning, economic models, immune system models, ecological models, population genetic models, models of social systems, and interactions between evolution and learning. What unites these diverse areas of research is the inspiration from biological evolution and Hollands original GA. (Mitchell and Forrest, 1994).

Problems for which genetic algorithms are typically applied can be divided into three main areas (Eiben and Smith, 2003):

- Optimization. Here the goal is to minimize the costs associated with some task. The traveling salesman problem would be a typical example.
- Modeling. Here input and output sets are known and the goal is to find a function that maps from the input to the correct output. Such systems can be used for prediction tasks.
- Simulation. Here the system model and at least some of the input is known, and we want the system to produce output.

The model in this thesis fall into the last category, as it will be an abstract representation of evolutionary processes in nature. It is important to keep in mind that GA simulations of natural phenomena are highly idealized.

3.1.1 Representation

The choice of representation of the genome in a genetic algorithm is highly dependent on the problem to be solved. Also, one should take into account

the mapping from genotype to phenotype. A representation consists of a string (usually of fixed length, although it does not have to be), and an alphabet which constitutes the string. Common alphabets are binary values and sets of integers and real values, but also letters or other characters might be used. Obviously, the size of the search space of the genetic algorithm increases with the size of the genome and the size of the alphabet set.

Scheduling problems, like the traveling salesman problem, represent special cases where the most natural representation is a permutation of a set of integers. In such problems a member of the alphabet can only occur once in the genome, and special considerations have to be taken in the choice of crossover and mutation operators (Eiben and Smith, 2003).

In our implementation, though, we use a simple binary encoding, with an alphabet of 0, 1, as this is the standard encoding in the NK fitness landscape. (Though also quantitative encodings has been used with the NK fitness landscape, see (Suzuki and Arita, 2004)).

3.1.2 Recombination

Recombination is the method used to combine the genomes of two (or more) individuals to form a new individual. This corresponds to mating in nature, and recombination is considered the most important genetic operator (Eiben and Smith, 2003).

There are numerous ways of recombining parent individuals to form offspring. One method is to probabilistically vary whether the parents will reproduce by crossover, i.e. a combination of each parent's genes, or asexually by copying the parent's genes unmodified to the offspring (Eiben and Smith, 2003). The choice of recombination method should be considered with the population's selection method in mind. With asexual reproduction there will be less genetic variation and good solutions are better perserved in the population; but this could also be achieved by certain selection methods.

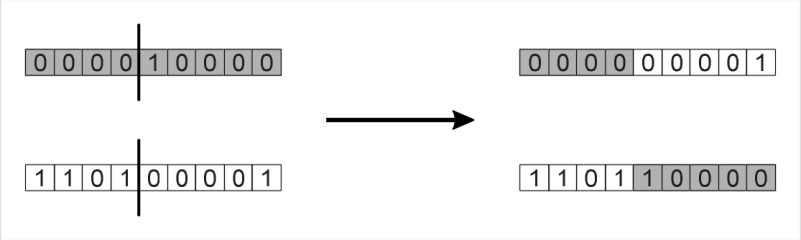

Figure 2: One point crossover. (Eiben and Smith, 2003).

As is the case with genome representation, the choice of recombination

method is highly dependent on the problem at hand. In particular, special methods need to be used if the genome represents a permutation problem. A very simple crossover method for binary representations is one-point crossover, where an integer between 0 and the length of the genome minus 1 is used to divide the parent genomes into two parts which are then spliced together to form the offspring (Eiben and Smith, 2003). This for instance was the crossover method used in Hinton and Nowlan's (1987) experiment.

For my implementation though, I have used uniform crossover for the binary representation. Uniform crossover works by generating an offspring by randomly choosing the next gene in the genome. A distribution over [0, 1] is used to choose the parent from which to get the next gene. If the value is below 0.5 the gene is taken form the first parent, else it is taken from the second parent (Eiben and Smith, 2003).

Because uniform crossover does not split each parent genome into parts that it transfers to the offspring, the method is not prone to so called *positional bias*, which is the tendency to keep together clusters of genes from one parent. Whether or not this is better depends on the problem. (Eiben and Smith, 2003).

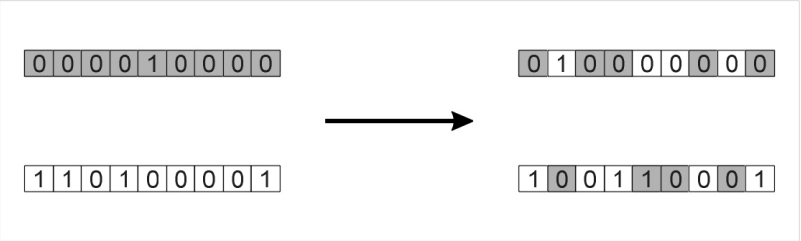

Figure 3: Uniform crossover (Eiben and Smith, 2003).

3.1.3 Mutation

Because mutation is a chance process it contributes to genetic divergence between populations and genetic variations in a population. In nature most mutations arise from errors during DNA replication, and they come in several forms. Point mutations is a DNA base pair change that usually has little effect on fitness, and deletion or insertion mutations removes or adds chromosomal segments. A third type of mutations, that changes the amount of DNA or the number of genes, usually has large effects on fitness (Stearns and Hoekstra, 2000).

In organisms mutations occur at much higher rates in some parts of the genome than others, and they can also be triggered by signals; but there

is no correlation between their effect on the phenotype and a higher fitness (Stearns and Hoekstra, 2000). (Had that been the case mutations would have been a Lamarckian mechanism.) Mutations are random, and that is the reason the promote genetic variation.

Mutation in computer systems is usually very idealized representations of mutations in living organisms. For instance, the length of the genome can seldom be affected by mutation. In my system I use a simple form of bitwise mutation. Given a genome consisting of binary values, each bit is flipped with a small probability. Usually the rate is somewhere between one mutation per generation and one mutation per offspring (Eiben and Smith, 2003).

Figure 4: Bitwise mutation. (Eiben and Smith, 2003).

3.1.4 Selection and replacement

The selection process in genetic algorithms is an idealized representation of natural sexual selection. Though there are examples of agent based genetic algorithms that model sexual selection (Todd and Miller, 1993; J. Sanchez-Velazco and J. A. Bullinaria, 2003), usually the process of two individual's seeking each other out for mating is abstracted away.

The most common method of selection in artificial systems is to rank the individuals based on fitness and selecting individuals for reproduction probabilistically based on their ranking. There are numerous selection methods, each with different strengths and weaknesses.

Replacement is the process of exchanging old individuals in the population with new individuals. Again there are numerous methods. The choice of both the selection method and the replacement method, and the parameters used for each method (for instance how many new individuals to generate and how many old individuals to replace in each generation), will obviously affect the selection pressure and thereby the genetic variation in the population (Eiben and Smith, 2003).

3.2 Adpative landscapes

The notion of an *adaptive landscape* originated in evolutionary biology, and is much used also to describe the search space of the genetic algorithm. (Jones, 1995) points out that even though genetic algorithms are based on principles borrowed from biology, they can still be analysed as heuristic state space search algorithms, and they are fully understandable as computational phenomena.

Formally, an adaptive landscape is constituted by 1) a representation space $\Re$, 2) a genetic operator, 3) a function $(f) : M(\Re) \rightarrow F$ for some set F, and 5) a partial order $>_F$ over F. The adaptive landscape is the graph that arises from this five tuple[10] (Jones, 1995).

The height dimension stands for fitness while the other dimensions represent biological traits. See figure 5 for a generic adaptive landscape.

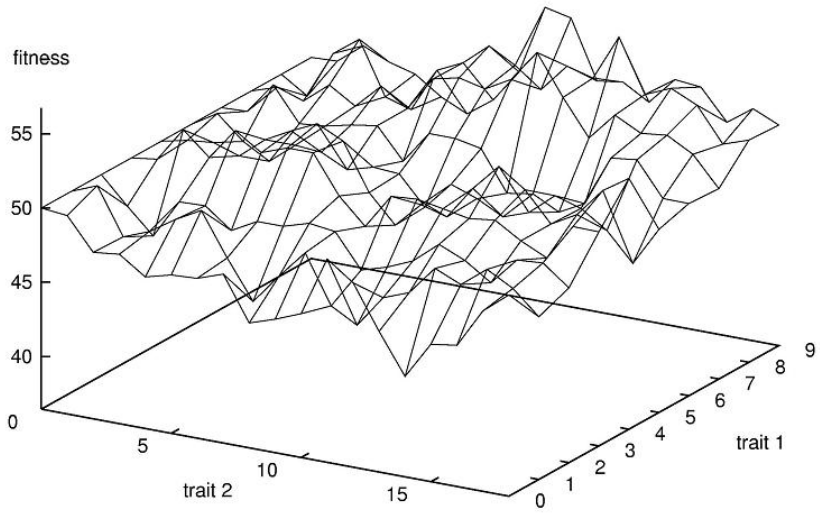

Figure 5: A generic adaptive landscape. (Eiben and Smith, 2003).

The attributes of an adaptive landscape requires the notion of a neigh-

[10]For further details and a formal definition of adaptive landscapes, see (Jones, 1995).

bourhood. For instance, a peak is defined as a point which neighbours are of less fitness. Other important attributes of landscapes include (Jones, 1995):

- Global optimum. A vertex which is at least as fit as any other possible vertex.
- Local optimum. A vertex which is not a global maximum, but which is fitter than all its neighbours.
- Plateau. A set of minimum two neighbour vertexes with the same fitness.
- Mesa. A pleateu where no point on the plateau has a neighbour with higher fitness.

When a landscape has several local optima it represents a *multimodal problem*, whereas if it has only one point which is fitter than all its neighbours it is known as a *unimodal problem* (Eiben and Smith, 2003). As we can see, the "ruggedness" of a landscape (the number of local optima) gives us an intuitive notion of the complexity of the search space the genetic algorithm operates in. Next, we turn to ways of specifically tune the ruggedness of an adaptive landscape.

3.3 Kauffman's NK fitness landscape

Kauffman's NK landscape provides a way of defining a test landscape for genetic algorithms where the complexity, or "ruggedness", of the search space can be explicitly set. In the NK landscape each genome is a binary string of length N. In addition a parameter K defines how many genes interact with how many other genes (epistasis). Every possible combination of gene interaction is assigned a random fitness value uniformly in the range [0.0, 1.0]. The values are stored in a lookup table with 2^{K+1} indexes. This is done for every locus in the genome, and thus the number of fitness values equals $2^{K+1}N$ (Back et al., 1997).

Fitness is computed by averaging the N fitness components F_i from each locus. The value of each F_i is determined by its own allele x_i *and* K other alleles. Thus the overall fitness of a genome is:

$$\frac{\sum_{k=1}^{N} F_i(x_i; x_{i_1}, ..., x_{i_K})}{N}$$

Kauffman investigated two strategies for choosing the genes a particular gene interacts with: *adjacant neighbourhoods* and *random neighbourhoods*.

Using the adjacant neighbourhoods the K bits next to the one in question will contribute to a genes fitness; using random neighbourhoods the K bits are chosen from the genome at random (Eiben and Smith, 2003).

Figure 6 shows the genome and the lookup table for the first locus in the genome. Adjacent neighbourhoods is used. N = 5 (size of genome) and K = 1 (number of other genes with which a gene interact). For this particular genome, the fitness for the gene in the first locus is 0.45, since its neighbour gene is 1, which leads us to the index {1,1} in the lookup table

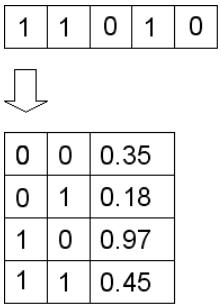

Figure 6: The genome and the lookup table for an NK fitness landscape with N = 5, K = 1

Since the NK fitness landscape is much used and fairly well understood (Eiben and Smith, 2003), and since it offers a convenient way to tune the complexity of a GA's search space, I chose this model for my experiments.

3.4 Model

In this section I will describe the system I implemented to perform the experiments.

I chose Python as the implementation language, due to its high level characteristics. All classes are written from scratch, with two exceptions – a support class that made it possible to implement singletons[11] in Python, and a library PyX which I use to generate graphs displaying the results.

In addition to a traditional genetic algorithm the system has the following characteristics:

- An extra plasticity genome that codes for what alleles can be modified by learning and not.
- A learning rule.

[11] A singleton is a class that can only be instantiated once.

- A measure of the cost of learning.
- An expanded version of the NK fitness landscape where the global optimum can be explicitly set.

3.4.1 Classes

The implementation consists of four main classes, depicted in figure 7.

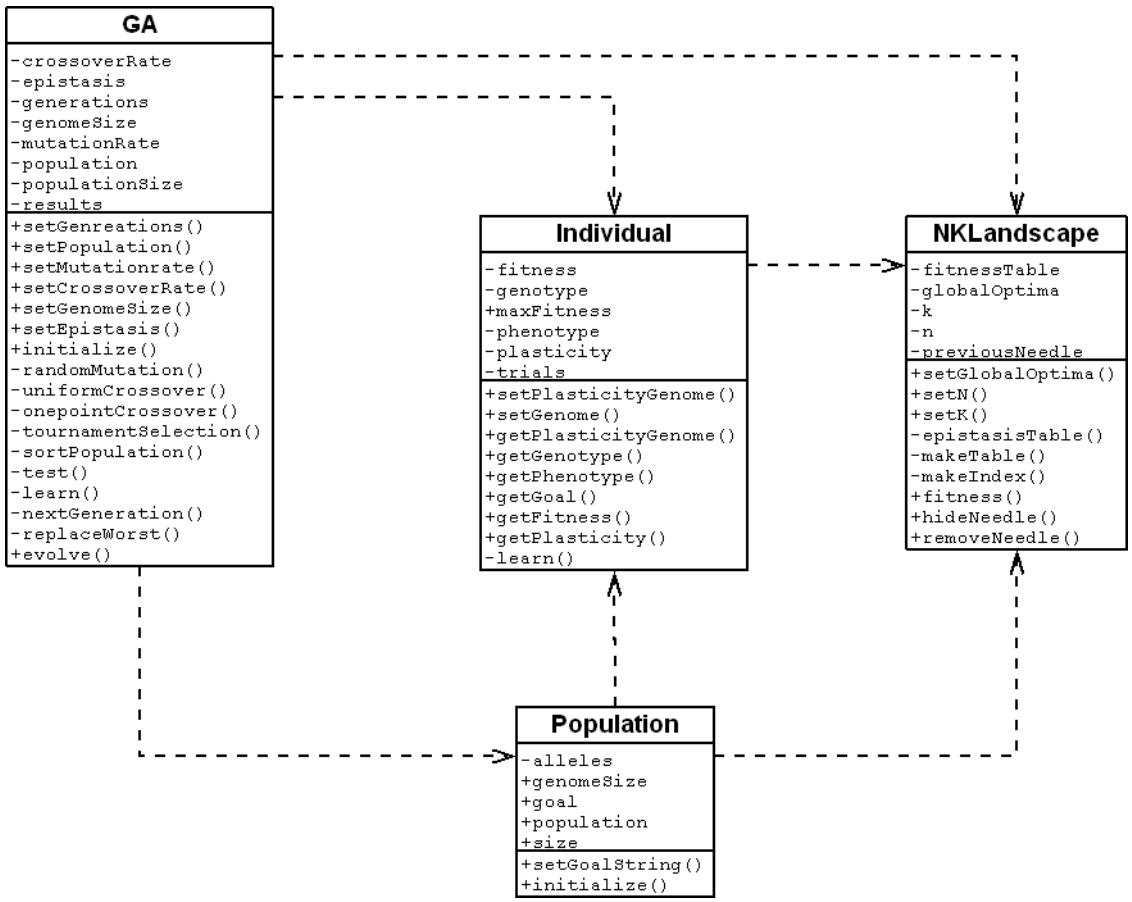

Figure 7: Class diagram.

- *GA* is the main class, and it utilizes most of the other classes. By instantiation this class sets parameters for genome size, number of individuals in the population, number of generations, and whether or not learning will take place.

- *Population* initializes all of the individuals in the population by instantiating objects of the Individual class and assigning each Individual-instance genomes of random values (both for the plasticity genome and the main genome).
- *Individual* keeps track of its current fitness and its genotype and phenotype values. It also implements the learning rule.
- *NK fitness landscape*. In addition to implementing the original fitness landscape with tunable epistasis, this class also have a method for altering the landscape by defining a global optimum.

3.4.2 Representation

The representation for the model used in this thesis is binary; each allele will have a value of 1 or 0. An individual's phenotype is represented by a genome which length can be set arbitrary, but for the experiments here I will use the length 20. An individual's genotype is represented by *two* genomes, one 'main genotype' which will also be the individual's initial phenotype, and one 'plasticity genome' which determines which alleles in the phenotype are plastic. See figure 8.

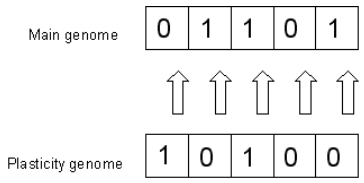

Figure 8: Ordinary genome and plasticity genome.

Here, the first and the third alleles of the phenotype can be flipped during learning, while the remaining alleles will be unchanged through the individual's lifetime. This solution is inspired by works by (Puentedura, 2003; Suzuki and Arita, 2004).

3.4.3 Crossover

Both one-point crossover and uniform crossover was implemented and tested. There seemed to be little difference between them, so I ended up using uniform crossover. See 3.1.2 for details.

3.4.4 Mutation

Simple random mutation was used. In an individual chosen for mutation, 20% of its bits were flipped (i.e. an allele with the value 0 would change to a value of 1). This was applied to both the main genome and the plasticity genome. The alleles that were flipped were chosen at random.

I experimented with two rates of mutation in the population, 0.05 and 0.25 (meaning that 5% or 25% of the population was chosen for mutation, respectively). The results of this will be described later.

3.4.5 The learning rules and costs of learning

The learning rule is implemented as a simple random search in the NK fitness landscape. Each learning trial consists of resetting all the plastic alleles in the phenotype at once, and then evaluate the phenotype's fitness. The default maximum number of trials is 100.

The pseudo code is as follows:

```
currentPhenotype := genotype
bestPhenotype := currentPhenotype
trials := 0
WHILE i < maxNumberOfTrials DO
        Reset all alleles in currentPhenotype at random
        trials := trials + 1
        IF fitness(currentPhenotype) > fitness(bestPhenotype) THEN
            bestPhenotype = currentPhenotype
            lastChange := trials
        ENDIF
ENDWHILE
```

Two different schemes were used to compute the cost of learning: 1) explicit cost; and 2) continual assessment, where the cost of learning is implicit (Mayley, 1996b).

The explicit cost of learning was calculated as follows:

originalFitness + ((currentFitness originalFitness) * (remainingTrials / maxNumberOfTrials)) - (learningCost * numberOfPlasticityGenomes / totalNumberOfGenomes)

In other words, an individual's fitness is penalized proportionally to the number of learning trials used and its degree of plasticity. For instance, if an individual uses half of the available trials, half of the gain in fitness is added. From this calculation is subtracted the proportion of plasticity alleles in the phenotype, multiplied by a constant cost of learning. In sum, both

the number of learning trials used and the degree of plasticity affects the individual's final fitness negatively.

The *continual assessment* scheme is taken from (Mayley, 1996b), and is the average of the fitness scores achieved searching for best fitness, added to the average of the fitness scores achieved after maximum fitness is found, for 100 (maximum) number of trials. If for instance an individual achieves the highest fitness on the 10^{th} trial, its fitness becomes the average fitness up to the 10^{th} trial added to the maximum acheivable fitness for the remaining 100 - 10 trials. In other words, the sooner an individual achieves its maximum fitness, the better its final fitness becomes (Mayley, 1996b).

3.4.6 Selection and replacement

Tournament selection was used to select members of the population for mating. In each selection round, 30% of the population was sampeled randomly, and the best member was chosen as a parent; this was repeated for each parent selection. Replacement was done by simply replacing the worst 80% individuals in the population with the new individual's created with crossover.

3.5 The modified NK landscape

A standard NK landscape where the N and K values could be set by parameters was created, with an adjacant neighbourhoods implementation of epistasis (see section 3.3). In addition a method to insert a goal string in the landscape was added. The goal string was assigned a fitness value of 1.0 if reached.

There were two motivations for doing this: 1) With such a goal string I could create the same 'needle in a haystack'-scenario as in Hinton and Nowlan's (1987) original experiment, while being able to explicitly set the complexity of the landscape; and 2) defining such a goal string of maximal fitness made it easy to assess how near a population was to reaching the global optimum. Also, for the continual assessment learning rule, the global optimum needed to be known; the only other method of getting this information in a randomly created NK landscape would be to perform a exhaustive search.

Also, a method to save and load the NK landscape to disk was implemented. This way, the same landscape could be used for all the experiments performed, assuring that random differences between the generated landscapes would not obscure the results.

4 Experiments

4.1 Overview

I will here outline the experiments and shortly reiterate some of the main points of the preceding sections as motivation for the set-ups.

Evolutionary simulations with and without learning will be run and the results compared. A number of parameters will be measured. For experiments without learning the following will be measured: fitness of best individual in the population, average fitness in the population and fitness of the worst individual in the population. Experiments with learning will measure: phenotypic fitness of best individual (i.e. the fitness achieved after learning), phenotypic fitness of worst individual, average phenotypic fitness, innate fitness of best individual (i.e. the genotypic fitness, before learning), and best innate fitness. In addition the degree of plasticity is measured; this is simply the number of plastic alleles divided by the number of alleles in the genotype, averaged for the whole population. To minimize random fluctuations, each simulation is averaged over 10 runs. The Baldwin effect will be investigated with relation to degree of learning, cost of learning, mutation rate, and the complexity of the fitness landscape.

4.2 Simulations

4.2.1 Motivation

Hinton and Nowlan's (1987) experiment does demonstrate that given a certain set-up, computational evolution can be expedited by adding a learning rule. As I have shown, the experiment is designed in a way that initially makes the learning rule much more efficient than the evolutionary search (which in their set-up is initially uninformed and therefore in effect a random search). It is still an open question whether the Baldwin effect occurs in biology, and if so to what extent; and while Hinton and Nowlan do not claim their model to be a valid representation of evolution in nature, others, like Dennet (2003) and Maynard Smith (1987), have used the experiment to support such a claim.

In any case, the experiment is doubtless very important as a first computer simulation of the Baldwin effect. While it has been much discussed and referred to, Puentedura (2003) points out that several possible critiques of Hinton and Nowlan's model have been little mentioned in the literature. Puentedura (2003) expands the original model in several respects and demonstrates that the model withstands the modifications. However, Puentedura

(2003) points out that his model still operates on a one-dimensional fitness landscape, and that it would be interesting to see how it would fare in a more complex landscape.

In the next section I will describe an expansion of the original model that incorporates this element. I have also incorporated Puentendura's (2003) expansion of an extra plasticity genome to deal with the so called *coupled genome problem*. In Hinton and Nowlan's (1987) model the allele that codes for plasticity replaces the allele that codes for a fixed location, so that one necessarily replaces the other. This can be remedied by using an extra plasticity genome in addition to the 'main genome'.

Also of interest is the fact that Hinton and Nowlan do not use mutation in their original experiment, only recombination. Furthermore it would be interesting to see how the model would fare with different parameter settings in a fitness landscape of tunable complexity.

4.2.2 Set-up

The fitness landscape – or if one wishes, simply the search space – of Hinton and Nowlan's (1987) model was indeed designed as a 'needle in the haystack' space, a hit-or-miss search. My model is designed to obtain that characteristic, but in a landscape with tunable complexity. This is done by constructing a standard NK fitness landscape, which is then modified by inserting a random goal string. The string is inserted in the landscape by modifying each of the values in the lookup tables representing the fitness landscape, assigning the index which corresponds to the goal string a fitness value of 1.0.

Firstly, this lets us see how the original model fares in a much more complex landscape, with many local optima; secondly, this lets us tune to what degree the goal string is hidden, how sharp of a spike the global optima will be in the landscape. The latter is a function of the degree of epistasis (the K value in the NK landscape). With lower values of K the global optima will be more easily identified, which will lead the search towards it faster; with higher values of K the search attains more of a hit-or-miss characteristic, and it is possible for a phenotype to be quite close to the goal string, as measured by Hamming distance, without this being reflected in its fitness.

A number of experiments were run on fitness landscape with the following K values: 0, 4, 9, 14 and 19. For each landscape configuration the cost and degree of learning were varied.

The following parameters were used for the genetic algorithm:

- Generations: 100
- Population size: 50

- Crossover rate: 80%
- Mutation rate: Two set-ups were tested, 5% and 25% of the population mutated, with 20% of mutated indivuals' genomes changed.

Tournament selection was used, and for each generation the population was sorted according to fitness and the worst 80% replaced with the new individuals. The fixed cost of learning was set to either no cost (0.0), low (0.05), medium (0.15), or high (0.3). Two different cost computations were also used: explicitly set cost or implicit cost by continual assessment (see section 3.4.5 for details).

4.3 Results

I here present selected graphs that illustrate the most important observations from the runs. All the graphs from the runs and the data files are contained in the attached file Results.rar.

4.3.1 No epistasis

With epistasis set to 0 the fitness landscape is unimodal, and while the search space is not small (there are more than a million possible combinations of a bit string of length 20), the search is informed. As can been seen in figure 9, the genetic algorithm reaches the global optima in less than 10 generations.

None of the runs using learning faired better than the run without learning, and except for the run with cost-free learning, plasticity soon decreased in the population. Neither did any of the runs with learning with associated cost reach maximum fitness. To do this they would have had to produce a phenotype which was identical to the genotype without utilizing learning. For a representative graph of a learning run, see figure 10.

It is not surprising that learning was not advantageous in this fitness landscape, since there is a strong correlation between fitness scores and the overall closeness of phenotypes in this landscape. Since learning is performing a local search, the benefits do not outweigh the costs. Still, plasticity persisted in the population to some degree and was never extinguished. Later this will be discussed in relation to genetic drift and mutation rates.

4.3.2 Medium epistasis

With epistasis set to 4 (K = 4 in the NK landscape), the run without learning reaches peak fitness at about the 60^{th} generation, with a score of 0.986, i.e.

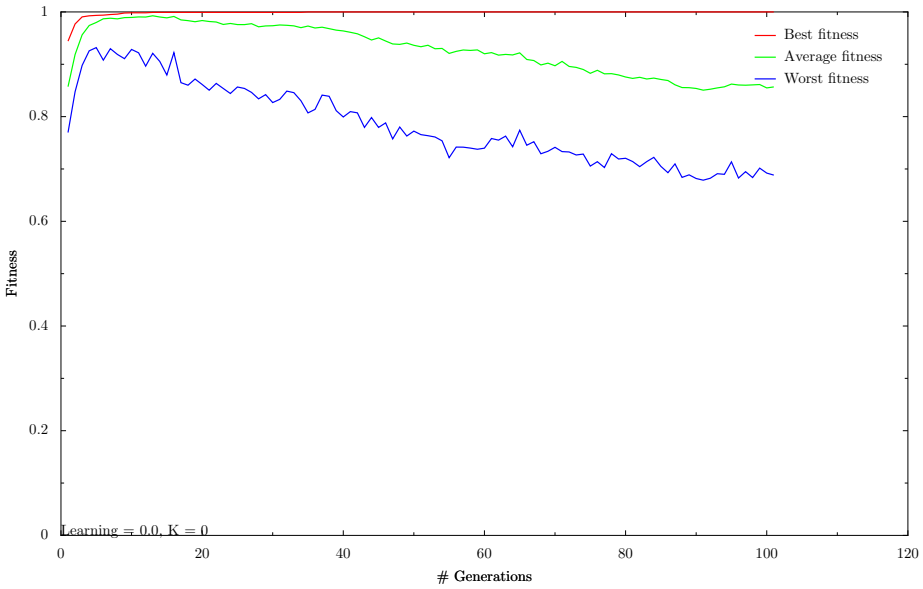

Figure 9: No learning, mutation 0.05, no epistasis.

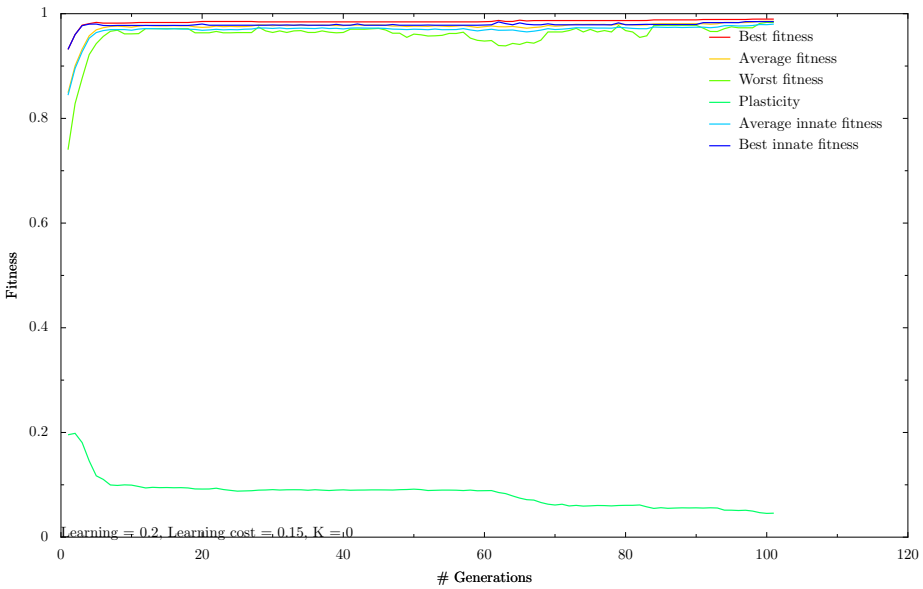

Figure 10: Learning with medium cost, mutation 0.05, no epistasis.

almost the global optimum (see figure 11). Learning without cost reaches optimum by the 40^{th} generation, and also reaches a high fitness quicker than the genetic algorithm without learning, so learning clearly enhances the search when there are epistatic interactions among genes. Not surprisingly, when

learning has no cost the innate fitness steadily decreases, while the degree of plasticity increases; so the population rely more and more on learning (figure 12).

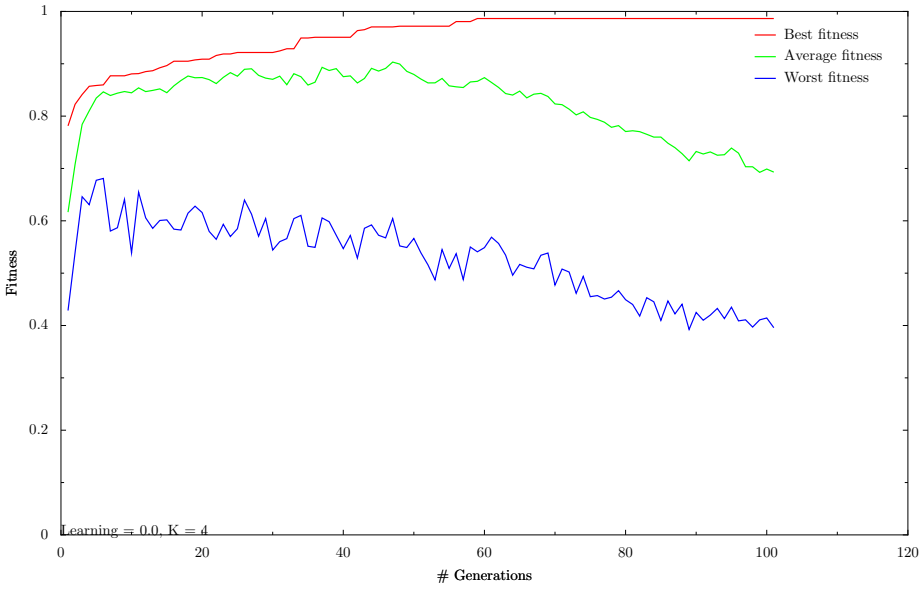

Figure 11: Evolution without learning, mutation 0.05, K = 4.

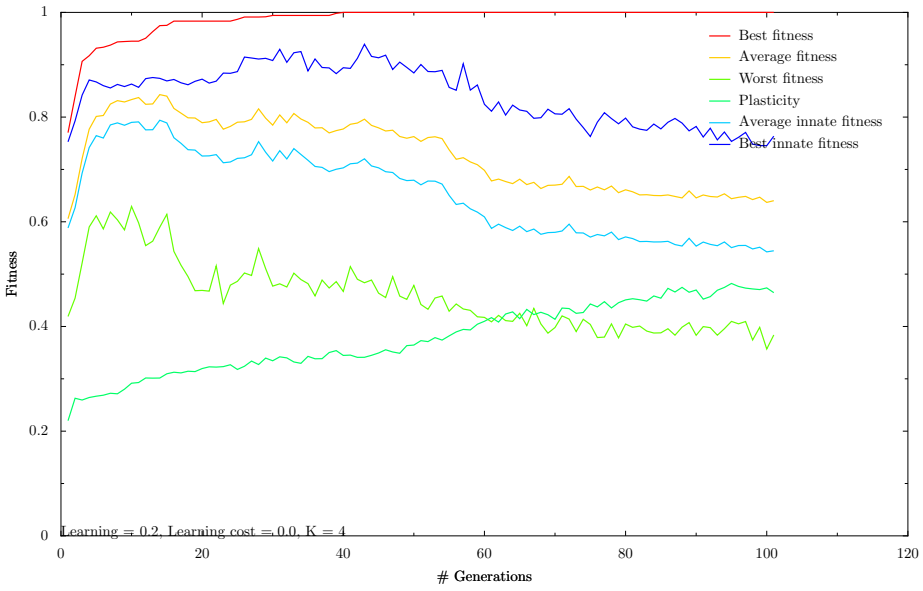

Figure 12: Evolution with cost-free learning, mutation 0.05, K = 4.

When learning with low cost (0.05) was introduced, again a high fitness is

reached sooner than with no learning. Here I also varied the mutation rate, and found that it had a notable impact on both how fast a high fitness was reached and on the degree of plasticity. With high mutation rates (0.25), the phenotypes achieved higher fitness sooner. In addition, there was a more noticeble selection in favour of plasticity early on in the evolutionary process, which then again decreased before it again increased slowly. In this case the plasticity curve was in accordance with what is expected in regards to the Baldwin effect. With lower mutation rates (0.05) this tendency was not as marked. See figures 13 and 14.

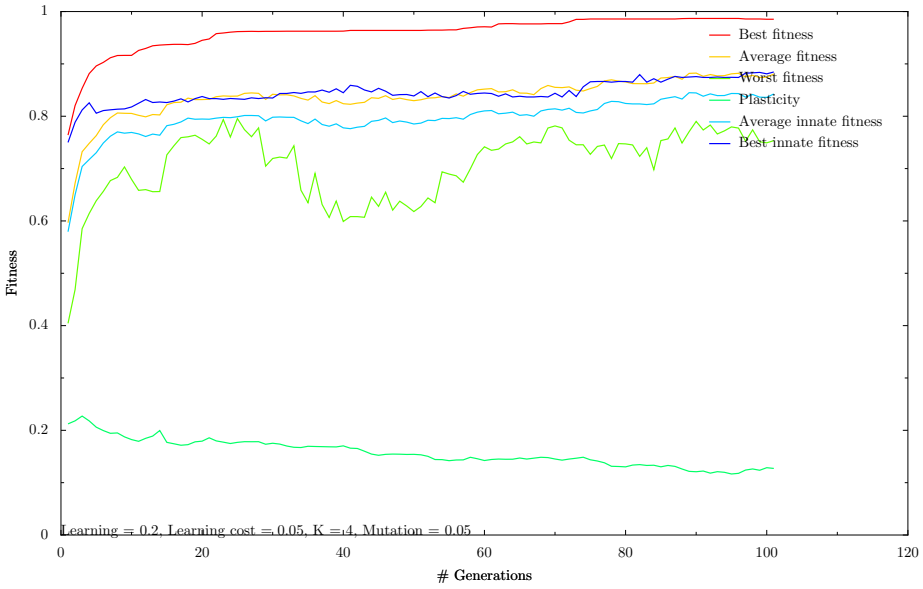

Figure 13: Evolution with low cost learning, mutation 0.05, K = 4.

With medium learning cost (0.15) there was also a noticable difference in the rate of plasticity given different mutation rates. With a low mutation rate of 0.05 learning was selected against early, and with this higher learning cost this yielded a higher fitness initially than with a mutation rate of 0.25, where learning was initially selected for, and then decreasing. See figure 15 and 16.

When the learning cost was set to 0.3, learning was deselected from the outset, but did not diminish completely, and the overall fitness was, not surprisingly, less than in the other runs. Learning with continual assessment showed that learning was utilized early in the evolutionary prosess, but decreased after a while; overall fitness was not very high compared to the other runs (see figure 17).

In a next series of experiments the epistasis was set to 9. This had a

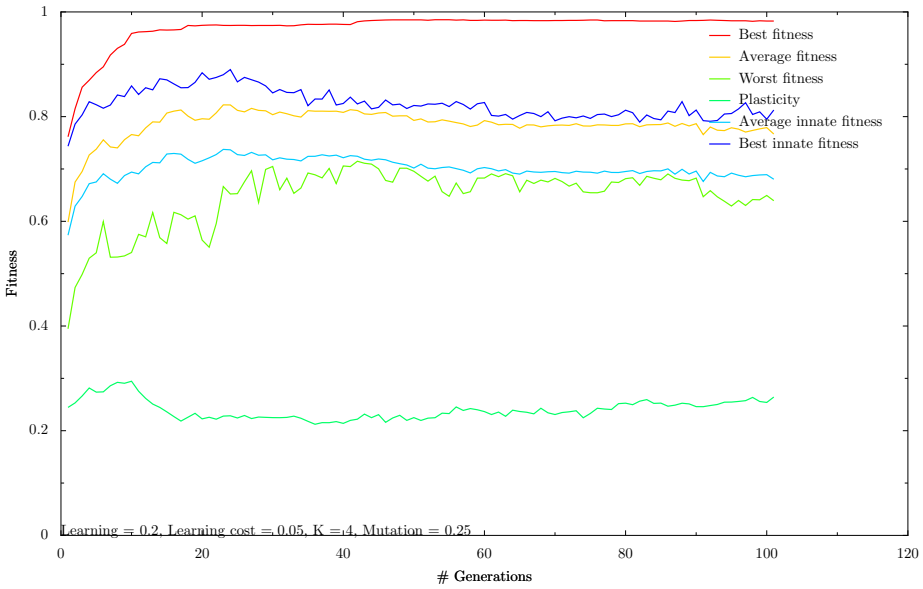

Figure 14: Evolution with low cost learning, mutation 0.25, K = 4.

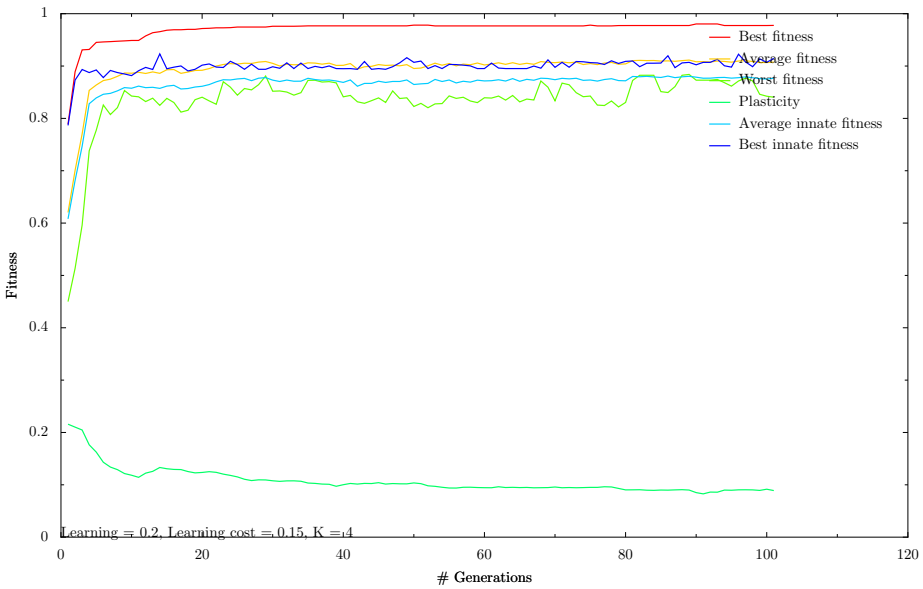

Figure 15: Evolution with medium learning, mutation 0.05, K = 4.

noticeable effect on the run with evolution without learning, which reached a maximum fitness of 0.724 compared to the score of 0.986 in the run with epistasis 4. The best result was again not surprisingly achieved by the combination of evolution and cost free learning, which peaked at a fitness score of

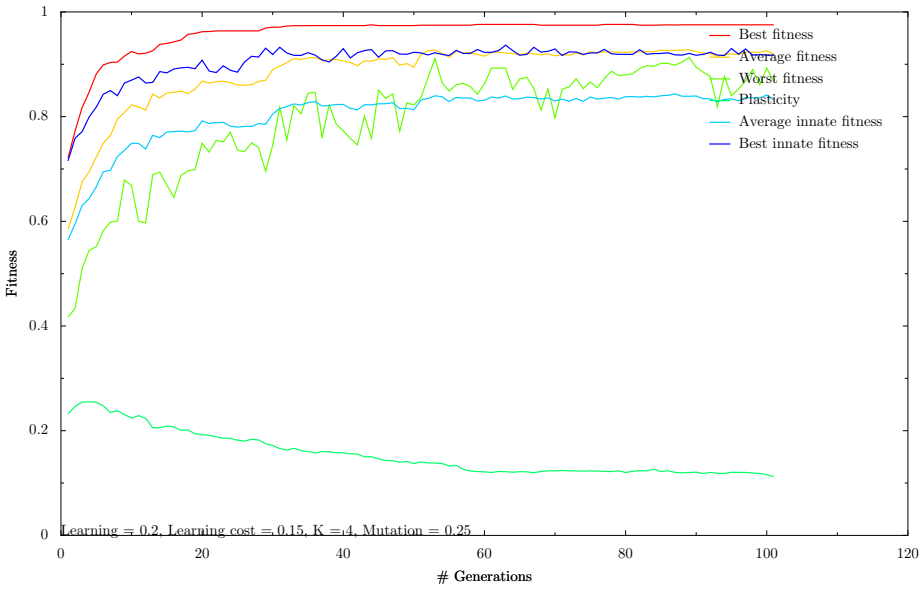

Figure 16: Evolution with medium cost learning, mutation 0.25, K = 4.

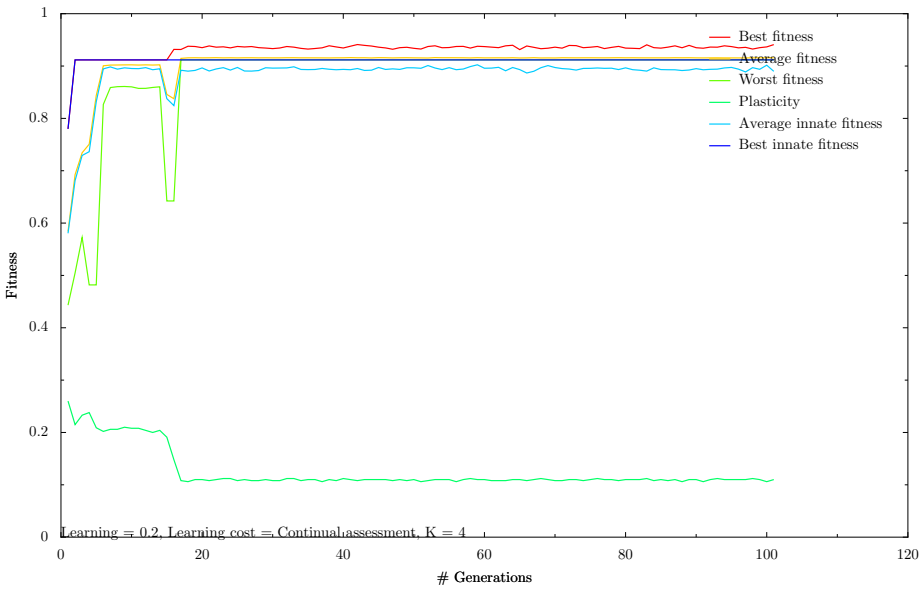

Figure 17: Evolution with learning and continual assessment, mutation 0.05, K = 4.

0.929. Also, the fitness increased stepwise continously over 100 generations, so had the experiment been run for further generations global optimum probably would have been achieved. See figure 18 and 19.

Learning with a cost of 0.05 also outperformed evolution without learn-

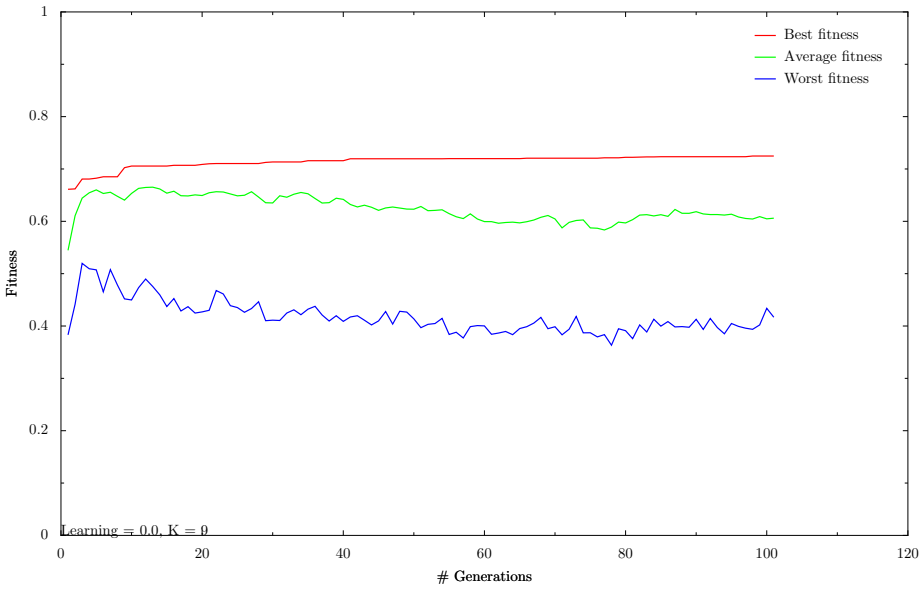

Figure 18: Evolution without learning, mutation 0.05, K = 9.

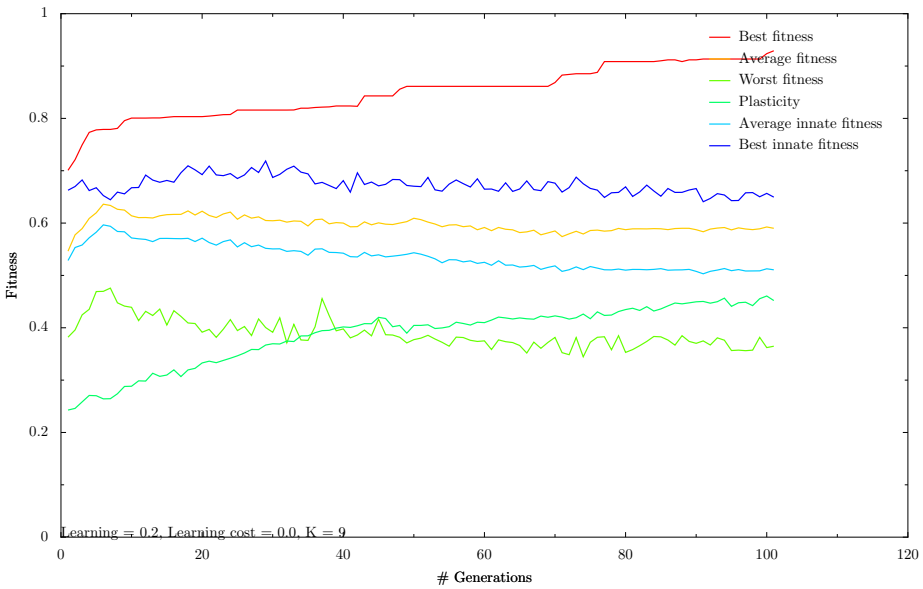

Figure 19: Evolution with no-cost learning, mutation 0.05, K = 9.

ing, and at least for the 100 generations the experiment was run, learning continued to stay high. When the learning cost was increased to 0.15, with a low mutation rate of 0.05, the combination of learning and evolution barely outperformed evolution alone (with a final fitness of 0.731 and 0.725 respec-

tively), though when learning was used a high fitness was achieved slightly faster (see figure 20). However, when the the mutation rate was changed to high (0.25), learning with cost 0.15 clearly outperformed evolution without learning (see figure 21). Also, with the high mutation rate, an increase in plasticity in the earlier generations, and an corresponding increase in phenotypic fitness, was marked; this was not the case with the lower mutation rate.

As can be seen in figure 22, evolution with continual assessment learning performed worst of all.

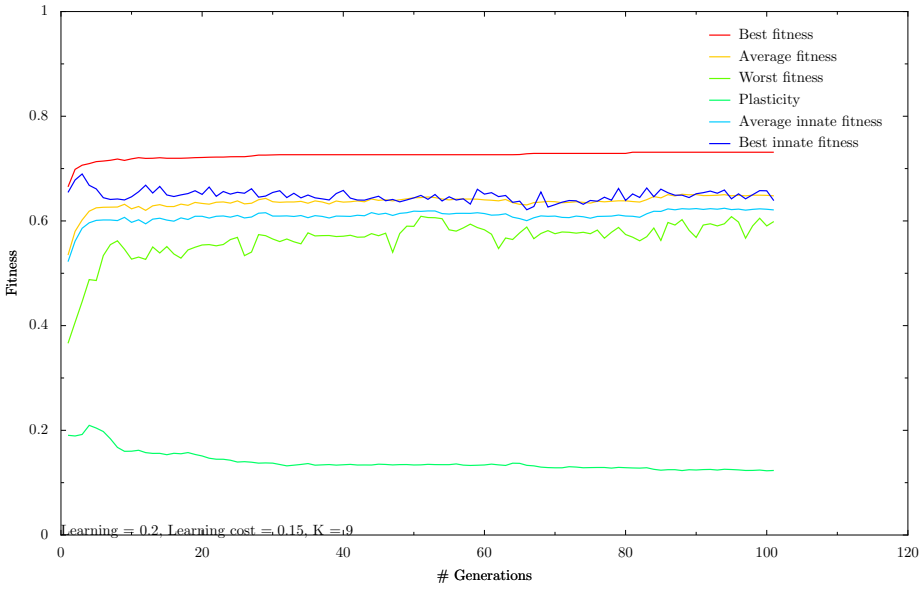

Figure 20: Evolution with medium cost learning, mutation 0.05, K = 9.

4.3.3 High epistasis

Runs with high degrees of epistasis includes K settings of 14 and 19. With a K setting of 14 and low mutation rate (0.05) the no-learning run achieved a max fitness score of 0.729. Not surprisingly, the cost-free learning run achieved the highest result (0.788). Of all the learning runs with cost, the low-cost learning was the only one to beat the no-learning search, with a score of 0.762 (though learning algorithms with higher costs did increase their fitness earlier). Learning with continual assessment as usual performed worst.

When the mutation rate was increased to 0.25, though, the medium-cost learning search beat the no-learning search, with a score of 0.739 (see figure 23

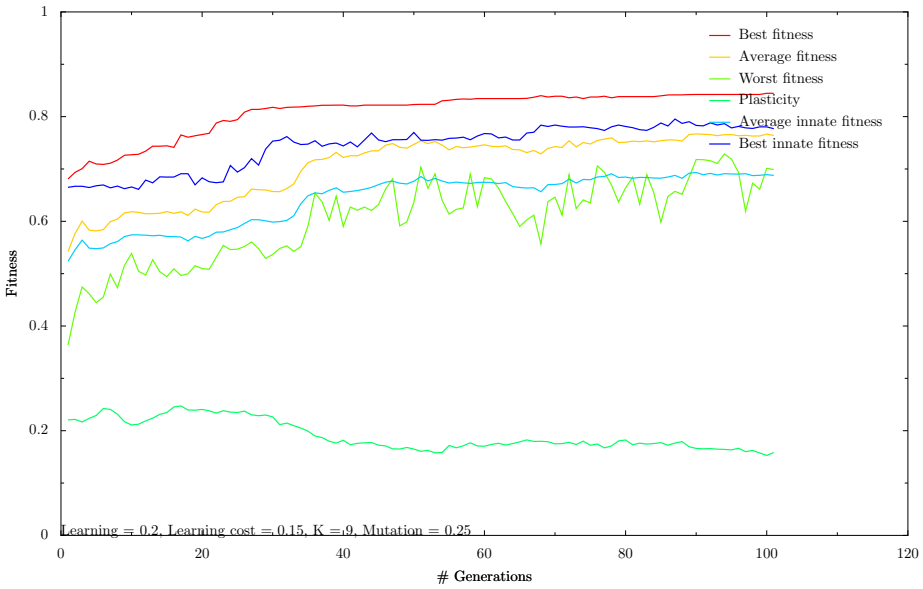

Figure 21: Evolution with medium cost learning, mutation 0.25, K = 9.

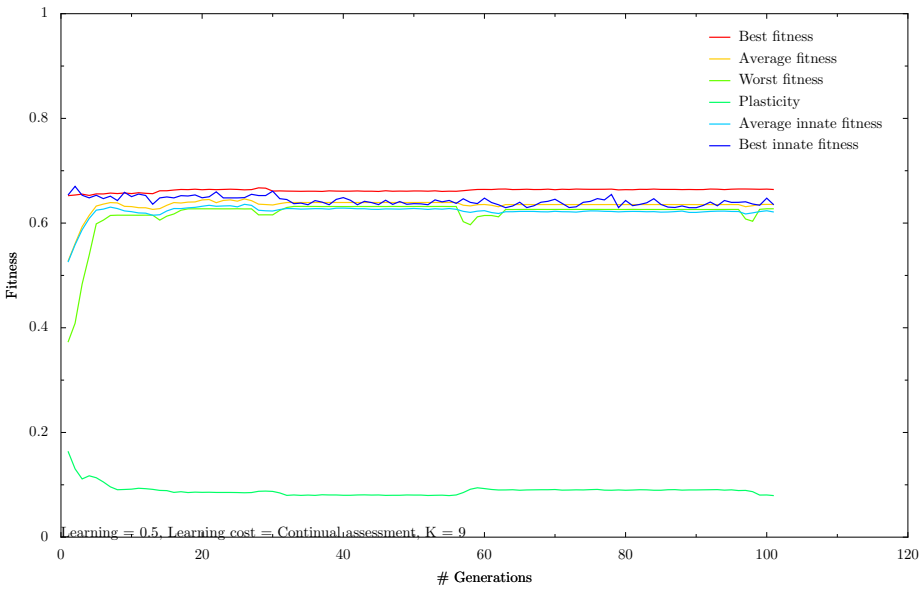

Figure 22: Evolution with continual assessment learning, mutation 0.05, K = 9.

and 24); while when high mutation was utilized on low-cost learning search, it performed slightly *worse* than with the lower rate of mutation. Here the plasticity increased in the population and stayed relatively high for all the 100 generations. Higher mutation rates was also tested for the non-learning

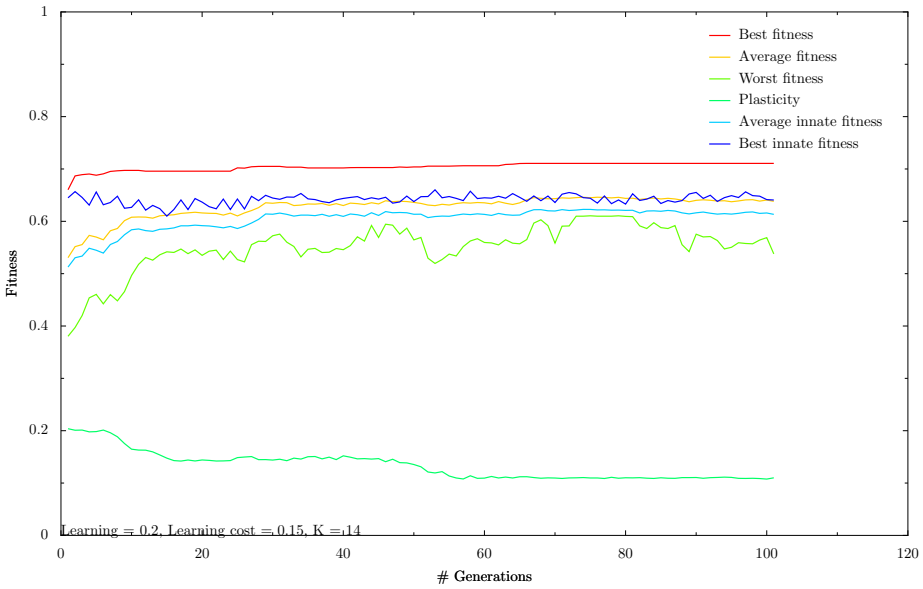

Figure 23: Evolution with medium cost learning, mutation 0.05, K = 14.

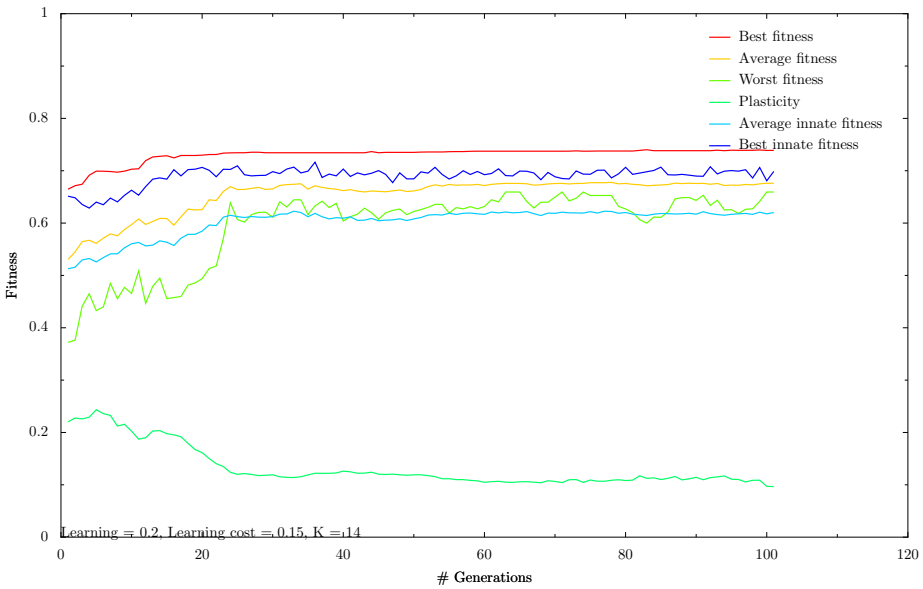

Figure 24: Evolution with medium cost learning, mutation 0.25, K = 14.

search, and this yielded a slightly worse result (0.713 vs. 0.729).

With epistasis set to 19 each allele interacts with all other alleles in the genome. Here evolution with no learning reached a maximum fitness of 0.698. In comparison, cost-free learning did significantly better with a final score of

0.781. Also low-cost learning (final fitness 0.721) and medium-cost learning did better than evolution alone. With a mutation rate of 0.05, medium cost learning scored 0.705; while it acheived 0.715 with a mutation rate of 0.25. As usual high-cost learning and continual assessment perfomed worst.

4.3.4 The effect of mutation

As we have seen above, the effect of mutation turned out to be of importance both for the overall fitness and for the 'rise and fall' of plasticity in the population. Also, learning tended to persist in the population to some degree. (Suzuki and Arita, 2004) make the same observations, which they interpret as an instance of learning protecting against 'genetic vulnerability' caused by mutation.

To further observe the effect of mutation, two lasts last experiments with no mutation was run in landscapes with epistasis of K = 4 and K = 14. The results are shown in figure 25 and 26.

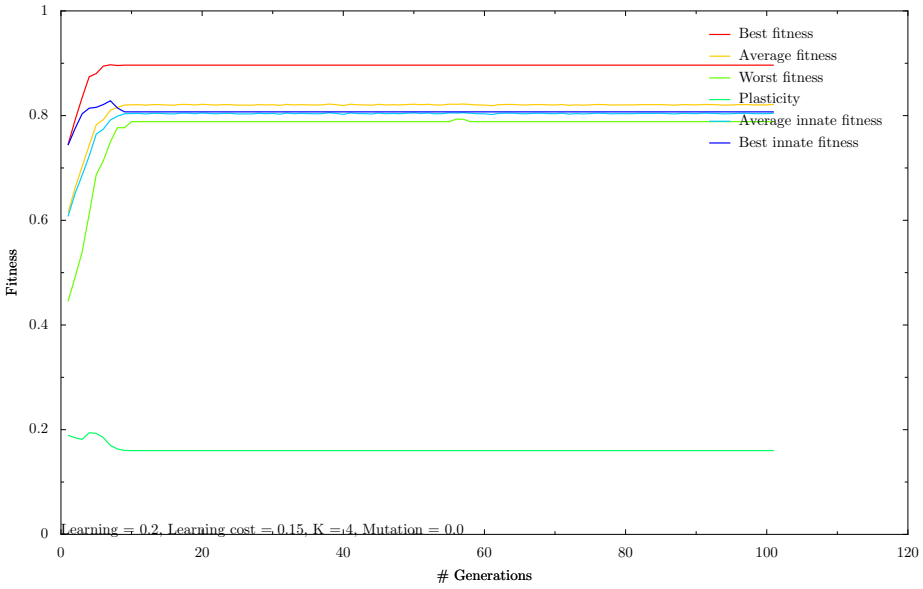

Figure 25: Evolution with medium cost learning, no mutation, K = 4.

Comparing with the runs with mutation, we see that the lack of mutation leads both to a decrease in overall fitness and more remaining plasticity. The decrease in plasticity is markedly less for the runs without mutation.

With K = 4 the final plasticity degree in the population is 0.16 for the run with no mutation. With a mutation rate of 0.05 the plasticity degree is 0.09; while with a higher mutation rate of 0.25 the plasticity degree is 0.11. See figure 15 and 16 for a comparison.

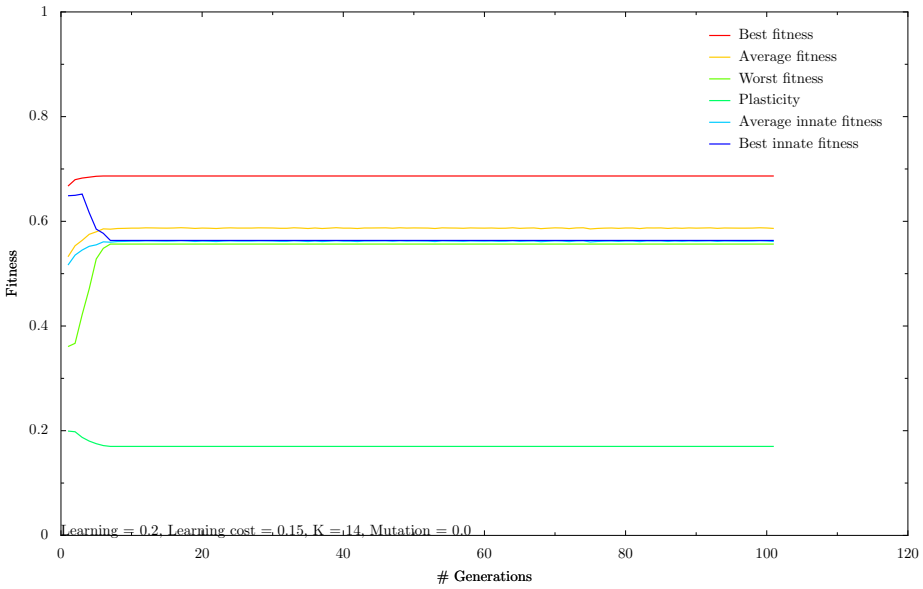

Figure 26: Evolution with medium cost learning, no mutation, K = 14.

The same holds true for the runs with K = 14. Here, the final plasticity in the population is 0.17 without mutation. With a mutation rate of 0.05 the remaining degree of plasticity is 0.14; with a mutation rate of 0.25 the final plasticity degree is 0.11. See figure 23 and 16.

While both high and low rates of mutation lead to less remaining plasticity than without mutation, for smaller values of K low mutation rates leads to less remining plasticity than higher mutation rates – while the opposite is true for high values of K. (This is confirmed be also looking at the results for the other runs. In the case of K = 9 a low mutation rate leads to a lower plasticity in the population (0.12 for low mutation versus 0.16 for high mutation); while in the case of K = 19 the situation is the opposite (0.12 for low mutation versus 0.11 for high mutation).

Suzuki and Arita (2004) suggest that learning persists to protect against genetic vulnerability caused by mutation, but as we can see, in complex fitness landscapes mutation leads to a lower degree of remeining plasticity, so this cannot be the full explanation. This will be discussed in more detail in section 5.1.5.

4.4 Observations

The following is a short summary of the observations, and will be elaborated on in the next section:

- Learning is more important with higher epistasis.
- Higher mutation rates significantly affects overall fitness and plasticity rates. It also yields the typical 'rise and fall' of the plasticity graph.
- Regardless of high mutation rates, some plasticity persists.
- Continual assessment did not work out well, as it seems to yield too high a cost of learning.
- The specific paramter tuning and the choice of genetic operators seem to play an important part regarding how learning affects evolution.

5 Discussion

5.1 Analysis

The experiments performed does show that learning can have a positive effect on evolution, i.e. that the Baldwin effect occurred. This was as expected, and in accordance with several earlier works (Hinton and Nowlan, 1987; Nolfi et al., 1994; Ackley and Littman, 1992; Mayley, 1996a; Puentedura, 2003; Suzuki and Arita, 2004). However, several factors contributed to the effect of learning.

Firstly, learning did not speed up evolution in unimodal fitness landscapes (K = 0). The reason is that in such 'smooth' landscapes the genetic algorithm has enough information to perform an efficient search without the help of learning, which would achieve worse fitness because of the associated costs. In fact, when cost-free learning was introduced it reached optimum only one generation before the no-learning run (59 versus 60 generations), so the benefit was marginal.

When epistasis was increased though, learning did have a positive impact on evolution, but only for relatively small costs of learning. For example, in the case of epistasis = 9 learning with low cost (0.05) outperformed evolution alone, while learning with a medium cost (0.15) did not outperform evolution with the standard mutation rate. However, when higher mutation rates were used, it did outperform evolution alone, and the tendency to an initial increase and a later decrease in learning, associated with the Baldwin effect, was more marked. Learning became more important as the complexity of the landscape increased (higher K-values).

5.1.1 The degree of plasticity

As shown in the section describing the results, the classical 'rise-and-fall' associated with the Baldwin effect was observed to a certain extent, but in many cases the change in plasticity was rather subtle. All in all, the rise in plasticity initially and the decrease caused by genetic assimilation was lower than expected.

Probably, this is related to genetic drift and 'hitch-hiking' because of the small population size (50 individuals) (Harvey, 1993). However the change in plasticity over generations was more marked in runs with higher mutation rates, which is elaborated in section 5.1.5.

5.1.2 Innate fitness

As a consequence of the relativley low genetic assimilation, in runs with learning the population tended to rely on learning, and the innate fitness was overall lower in these runs than in the runs without learning. This indicates that while genetic assimilation did uccur to some extent, it did not result in an optimal balance between plasticity and instinct. Again, this is probably connected to a lack of genetic variation caused by the low population size.

5.1.3 Learning and its costs

In my series of runs, learning turned out to benefit evolution only with low or medium learning costs. Obviously, the choise of learning costs is somewhat arbitrary and made to illustrate a point; there is no way of estimating to what degree these costs, or the way of computing them, corresponds to the costs of learning in nature, which of course are vastly more varied and complex.

Regarding the computation of the learning costs, two measures were used to penalize learning: 1) he number of learning trials used; and 2) the number of plastic alleles. In fact 1) would implicitly also measure 2), because the more plastic alleles, the more learning trials an individual would be likely to use before improvement siezed. Maybe a smaller penalty of learning would have lead to a more marked increase of plasticity early in the evolutionary run.

The continual assessment scheme turned out to be too costly in this model, and lead to decrease in plasticity early in the evolutionary run.

5.1.4 Neighbourhood correlation

As explained in section 2.4.2, for learning to have an effect on evolution there need to be a neighbourhood correlation between the phenotype and the genotype. In this experiment the phenotype and genotype had the same representation, bit strings.

The neighbourhood correlation therefore is the Hamming distance between a phenotype changed by its moves in the phenotypic space versus a genotype moving in genotypic space. Initially a couple of test runs with an initial learning rate of 50% was tested (i.e. 50% of the phenotype was plastic) as opposed to the 20% plasticity I ended up using, but even with the lowest learning cost, plasticity very soon decreased in this scenario. This is probably because the neighbourhood correlation became too weak, i.e. the phenotype could change so much that it to very little extent could inform the evolutionary search.

5.1.5 The importance of mutation

In these experiments, two different mutation rates were tested, 0.05 and 0.25 (meaning that 5% or 25% the population was mutated). When an individual is selected for mutation, 20% of the bits in the genome (both the ordinary genome and the platicty genome) are flipped. Also, for medium cost learning with K = 4 and K = 14, runs with no mutation were conducted; these simulations showed that the lack of mutation lead to lower changes in the degree of plasticity, and also to worse fitness in the population.

To summarize the results with regards to mutation rates:

- For low and medium cost learning high mutation rates tends to increase fitness.
- Higher fitness is reached in fewer generations when high mutation rates are used.
- Higher mutation rates leads to a more marked tendency to the 'rise-and-fall'-curve of plasticity associated with the Baldwin effect.

The above indicates that the higher fitness achieved with higher mutation rates is connected to the degree of learning. To test this, high versus low mutation rates were tested for evolution without learning on a fitness landscape of K = 14, and as mentioned in section 4.3.3, in this case the run with a high mutation rate performed slighlty lower than the run with a low mutation rate (0.729 for low mutation; 0.713 for high mutation).

Higher mutation rates made the most difference in landscapes with a K value of 9 and a medium learning cost. Here, the maximum fitness of the high-mutation run was 0.845 versus 0.731 for the low-mutation run.

There is much discussion about the role of mutation in genetic algorithms, not to say of the role of mutation in nature (where its actual rate and impact is often hard to estimate). An evolutionary search needs to maintain a balance between on the one hand retaining good solutions, and on the other coming up with novel solutions. Mutation can enhance the latter by increasing the genetic variation in the population (Seymour, 1994).

In evolution with small population sizes genetic drift, random fluctuations, and hitch-hiking are common problems. Hitch-hiking is the tendency of one or a few individuals initially receiving high fitness scores to dominate the population for a number of generations. The 1s and 0s in the fittest individual's plasticity genomes will have a tendency to dominate the gene pool when the selection preassure is high: their genes so to spek hitch-hike into later generations. Later, if the selection preassure decreases (because

of full or partial convergence on a local optimum), this decreases the genetic variation the population (Harvey, 1993). Mutation can counteract this tendency.

As shown in section 4.3.4, high mutation rates tended to yield less plasticity in the end of runs for high values of K than did low mutation rates, and overall higher mutation lead to both higher phenotypic fitness and innate plasticity. One explanation could be that in fitness landscapes with a higher degree of epistatic interaction random learning would take more trials to find a better phenotype, thus yielding a higher learning cost. Also, because of the epistasis there is a greater distance between phenotypic and gentoypic space, so that a good solution found by learning is less likely to have smoothed the fitness landscape. Higher mutation rates might aid evolution in discovering these factors as a result of more genetic variation in the population.

In summary, it seems that with higher mutation rates the evolutionary search was better able to identify the effect of learning, as higher mutation rates could counteract the lack of genetic variation in the population.

5.1.6 Evaluation

In fitness landscapes with epistasis, learning turned out to benefit evolution to some degree: higher fitness was reached earlier and best fitness at the end of the run tended to outperform the result of evolution alone. However, the average innate fitness in the learning-runs were usually lower than in the runs without learning.

This indicates that while the first step of the Baldwin effect, an enhancement of the evolutionary search took place, the population tended to rely on learning, and the degree of genetic assimilation was low. The persistence of plasticity in the evolution was a symptom of this.

In a comparable experiment Suzuki and Arita (2004) observed the same persistence of plasticity, but they propose a different explanation, that learning functions as a barrier against genetic vulnerability in the population caused by mutation. I suggest a different explanation: That the persistence of learning is caused by a lack of genetic variability in the population, probably to a large extent due to the small population size (50 in my experiment and 20 in Suzuki and Aritas (2004) experiment). As I have outlined in the previous section, mutation can mitigate this effect by introducing greater genetic variation in the population.

5.1.7 The layers of abstraction

I want to stress that this simulation of the interaction of learning and evolution is highly abstract on several levels. Perhaps it is not so much an improvement on other models as it is an illustration of the arbitrariness with regards to the choice of genotypic/phenotypic representation, operators, learning rules and parameter setting. In particular the genotype to phenotype mapping is very simple, and far from the complexities of nature. Also, the fitness measure is abstract and not endogenous to the individuals.

In my view the model is not grounded enough in biology to serve as any form of empirical evidence – or even suggestion – regarding the Baldwin effect in nature. I do believe though, that the model does illustrate some points with regards to the relationship between learning and evolution in search landscapes of different complexity (i.e. different degrees of epistatic interaction), and it sheds some light of the importance of mutation in learning-evolution models.

5.2 Choise of operators and parameter tuning

There are considerable differences between the implementations I have discussed in this thesis. The choice of genome representation, and recombination and mutation operators varies. In some experiments, like Hinton and Nowlan's (1987), recombination is the only genetic operator used; while in others, like Parisi et al.'s (1992), only mutation is used[12].

The parameter settings for the evolutionary algorithm and the learning rule also vary considerably. Hinton and Nowlan (1987) use a population size of 1000 and a maximum number of learning trials of 1000, and observe an increase of innate fitness after about 15 generations; while for instance Suzuki (2004) use a population size as small as 20, a maximum number of learning trials of 5, and observe an increase in innate fitness only after approaching 10,000 generations. Likevise the complexity of the landscape varies.

Of particular importance is the balance between the search of the genetic algorithm and the learning rule. Given a learning rule that is more powerfull than the genetic search (relative to a certain time periode), learning is obviously more likely to be selected for and a pronounced Baldwin effect more likely to occur. This was the situation in Hinton and Nowlan's (1987) experiment. Also, the topology of the search landscape might be such that the local search of learning will be more of a fine tuning of an approximate optimum which the more gross evolutionary search is not able to adjust to.

[12]In (Nolfi and Floreano, 2004) the autors recommend using only mutation when evolving ANNs because they find it to be the most efficient.

When evaluating an experiment it is important to take into consideration the different parameter settings used. As Mayley (1996b) has pointed out, it is of particular importance to consider the correlation between genotypic and phenotypic space. Also of importance is considering the relative powerfullness of learning versus genetic operators.

In my series of experiments, the degree of mutation turned out to be an important factor with regards to the effect of learning on evolution; but this still might be related to other implementation factors such as population size, selective preassure, recombination method, etc. Distinguishing the effect of the different operators and parameters is difficult because of their interaction.

There is a possibility that the Baldwin effect can be observed after extensive parameter-tuning, and there is seldom much reference to the similarities with natural evolution in nature (often because too little is known, for instance about the role of mutation). Therefore there is a danger of the models being fine-tuned to demonstrate the effect, while other configurations might have proven less convincing results regarding the Baldwin effect.

5.3 Models, simulations and the scientific method

5.3.1 Modelling nature

As I have argued in section 2, it is often unclear whether simulations of the Baldwin effect are intended to illustrate general search properties, or whether they are intended as empirical claims about evolution and ontogeny[13] in nature. References to biological phenomena are common, but the mapping between nature and the model is seldom discussed in detail.

In an assessment of the scientific status of artificial life, (Noble, 1997) argue for a distinction between an analytic and a synthetic approach in ALife. The analytic approach is the method of using computer programs to prove the logical implications of a set of assumptions; the synthetic approach claims that the assumptions in the system have empirical relevance. Noble (1997) argues that an important methodological problem in ALife is the confusion between analytic and synthetic approaches.

Since ALife simulations are computer programs, and since computer programs are conjunctions of mathematical and logical statements, the result of ALife simulations has a primarily analytic quality. Computer programs are by definition deterministic, and the fact that we cannot predict their result without actually running them does not preclude this (Noble, 1997). Among certain AI/ALife enthusiasts there exists a tendency to assume that

[13]The developmental history of an individual.

using biologically inspired methods, such as artificial neural networks or genetic algorithms, makes computer systems more biological, or even life-like, in a qualitative way. This of course is not the case. In his refutation of the various attempts at smouldering his Chinese Room, Searle clearly demonstrates the absurdity of such an idea (Searle, 1992, 1997).

Noble (1997) argues that while ALife simulations can be used to post empirical hypothesises, they cannot at the same time be used as proving ground for these claims – they must refer back to the world. Also, ALife experiments can be used to assess the logical coherence of pre-existing empirical theories. Good empirical ALife depends upon the generation of good hypothesis; the researcher must be able to identify the similarities between real-world phenomena and the result of a simulation. Knowledge of the real world domain is important also in selecting the analytical assumptions; AL hypothesis must be relevant with regards to the real-world domain (Noble, 1997).

The problem with simulating the Baldwin effect is that there is no real-world empirical data to use as validation, as the Baldwin effect is still a theoretical hypothesis in evolutionary biology. Of course, the lack of real-world empirical data, and the difficulty of obtaining it, is in many ways the very reason for using computer models. But had empirical data been available, a computer model could have been validated against the real world observations, and then elaborated on to make set up experiments one were not able to observe in the real world. In the lack of such verification, the best one can do using computer models is to account for the resemblance between the model and the real world phenomenon; but to quantify that resemblance is often hard. Genetic algorithms, though biologically inspired, necessarily strip away a lot of biophysical phenomena (transcription, protein synthesis, meiosis etc.) (Mitchell and Forrest, 1994). The challenge is to identify where the AL model is general enough to tell us something about certain conditions and the emergence of certain behaviours or other qualitative properties.

On a more philosophical level, ALife is of course prone to all the usual criticisms of computationalism. The emergent phenomena observed in computer simulations are dependent on our interpretations of them; they do not exist on an independent basis. Even more so, they exist because we created them. Risan (1997) points at an important paradox in ALife: When interpreting the experiments the researchers assumes the scientific objective position, while 'the nature they study [is] so obviously constructed' (Risan, 1997, p. 11).

5.3.2 Validity and reliability

Two well agreed upon measures of the soundness of science is validity and reliability. As I have shown, it is hard to validate the results of computer simulations of the Baldwin effect, as there is scant empirical evidence of the phenomenon in nature.

However, it is possible to measure the degree of reliability between different simulations, though an important prerequisite is that the details of the set-up and the implementation and parameter settings are clearly stated. The degree of consistency between simulations, or the lack thereof, is an important indicator of the soundness of the research conducted.

Likewise, it is essential to discuss the consistency between the interpretation of results. At the time being, there seems to be quite a bit of disagreement between different researchers as to how to interpret the results. The different interpretations of the relationship between the learning task and the evolutionary task, described in section 2.3.4, serves as an example.

In my interpretation of the experiments implemented in this thesis, I observed much the same results regarding the persistence of learning as (Suzuki and Arita, 2004), but my interpretation and explanation of the results differ from theirs. To achieve sound research I think it is important to highlight such differences when they occur. This way, they can be critically discussed and compared, which I think is an important part of staking out a way for a more unified research paradigm on the Baldwin effect.

6 Conclusion

In this thesis I have given an account of the Baldwin effect as it is perceived in both evolutionary biology and in computer science/ALife. I have shown that there is a difference in the perception of the phenomenon: In evolutionary biology there is much discussion about what the Baldwin effect actually is, or whether it exists at all; while in ALife research the Baldwin effect tends to be perceived as less problematic, which is supported by numerous experiments confirming the positive effect of learning on evolution.

However, as my literature survey has shown, there is confusion as to how to define the Baldwin effect and what elements constitutes it. Early experiments tended to focus on the benefits of learning. The set-ups in the experiments are varied, with quite different choices regarding genetic operators, learning rules and evolutionary parameters. These are all factors which affect the outcome of the experiments and should be clearly accounted for and critically discussed.

The level of abstraction in the computer simulations is obviously very high. This makes it hard to measure the explanatory value of the models for the Baldwin effect in nature, though investigating the Baldwin effect as a computational search problem in itself is also of interest. Still, the intention of the experiments should be clearly stated, in particular with regards to whether the approach is intended to be analytical or empirical.

The results of the computer simulations are often interpreted differently or even contradictory, which is one of the symptoms of the confusion surrounding the Baldwin effect also in ALife; but this problem area is often under-communicated in the literature. It would probably be healthy for the field if a stronger focus was placed on the differing interpretations and understandings, which should be critically discussed.

A simulation of the effect of learning on evolution was implemented in a fitness landscape of tunable complexity. An extra plasticity genome was used for the evolution of plasticity itself. The experiment was run with several different parameter settings. In some areas the results were in accordance with earlier observations; in particular, learning proved advantageous in more complex fitness landscapes and a higher fitness was reached in fewer generations. While genetic assimilation, which is an important second step in the Baldwin effect, was observed, its effect was not strong. The simulations using learning tended to outperform no-learning trials if the learning costs were set relatively low, but the average *innate* fitness of the population was often worse than was the case for the non-learning trials.

This was interpreted as a consequence of the population continuing to rely on learning because of a lack of genetic variability, probably caused by a low

population size resulting in genetic drift and hitch-hiking. I found that higher rates of mutation counteracted this tendency and contributed to a more pronounced Baldwin effect, with the classical rise-and-fall of the plasticity in the population. This interpretation differs from Suzuki and Aritas (2004), who also observed persistence of learning, but interpreted it as a mechanism against genetic vulnerability caused by mutation. I suggest that the problem is not so much genetic vulnerability as a lack of genetic variation caused by a small population size. This is in accordance with my observation that the persistence of plasticity was less pronounced when mutation rates were high.

The confusion surrounding the Baldwin effect notwithstanding, it is an important and indeed interesting theory in evolutionary biology. As it is hard to validate the theory in nature, computer simulations represents one source of information, but they should be interpreted with great care. The level of abstraction is very high on several levels, and the models are often implemented quite differently. One way of trying to improve the soundness of the research on the Baldwin effect in ALife is to focus on the reliabilty of the simulations and of the coherence of the interpretations.

References

Ackley, D. and Littman, M. (1992). Interaction between learning and evolution. In Langton, C., editor, *Artficial Life II*. Reading, Massachusetts: Addison-Wesley.

Back, T., Fogel, D., and Michalewicz, Z., editors (1997). *The Handbook of Evolutionary Computation*. Oxford University Press.

Baldwin, J. M. (1896). A new factor in evolution. *American Naturalist*, 30 (June):441–451.

Bryson, J. J. and Hauser, M. D. (2002). What monkeys see and don't do: Agent models of safe learning in primates. In Barley, M. and Guesgen, H. W., editors, *AAAI Spring Symposium on Safe Learning Agents*.

Depew, D. J. (2003). Baldwin and his many effects. In Depew, D. J. and Weber, B. H., editors, *Evolution and Learning – the Baldwin effect reconsidered*, pages 3–31. Cambridge, Massachusetts: MIT press.

Downes, S. M. (2003). Baldwin effects and the expansion of explanatory repertoire. In Depew, D. J. and Weber, B. H., editors, *Evolution and Learning – the Baldwin effect reconsidered*, pages 33–51. Cambridge, Massachusetts: MIT press.

Downing, K. (2004). Development and the baldwin effect. *Artificial Life*, 10:39–63.

Eiben, A. E. and Smith, J. E. (2003). *Introduction to evolutionary computing*. Springer-Verlag Berlin Heidelberg.

Godfrey-Smith, P. (2003). Baldwin sceptisism and baldwin boosterism. In Depew, D. J. and Weber, B. H., editors, *Evolution and Learning – the Baldwin effect reconsidered*, pages 53–67. Cambridge, Massachusetts: MIT press.

Harvey, I. (1993). The puzzle of the persistent question marks: a case study of genetic drift. In Forrest, S., editor, *Conf. on Genetic Algorithms*. Morgan Kaufmann.

Harvey, I. (1997). Is there another new factor in evolution? *Evolutionary Computation*, pages 313–329.

Hinton, G. E. and Nowlan, S. J. (1987). How learning can guide evolution. *Complex Systems*, 1:495–502.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA. 1st edition: 1975, The University of Michigan Press, Ann Arbour.

J. Sanchez-Velazco and J. A. Bullinaria (2003). *Sexual Selection with Competitive/Co-Operative Operators for Genetic Algorithms*. IASTED/ACTA Press.

Jones, T. (1995). *Evolutionary Algorithms, Fitness Landscapes and Search*. PhD thesis, The University of New Mexico, Albuquerque New Mexico.

Mayley, G. (1996a). The evolutionary cost of learning. In Maes, P., Mataric, M., Meyer, J. A., Pollack, J., and Wilson, S., editors, *From Animals to Animats 4: Proc. of the 4th Intern. Conf. on Simulation of Adaptive Behavio*. Cambridge, Massachusetts: MIT press.

Mayley, G. (1996b). Landscapes, learning costs and genetic assimilation. *Evolutionary Computation*, 4(3):213–234.

Mayley, G. (1997). Guiding or hiding: Explorations into the effects of learning on the rate of evolution. In Husbands, P. and Harvey, I., editors, *Fourth European Conference on Artificial Life*. Cambridge, Massachusetts: MIT press.

Maynard Smith, J. (1987). Natural selection: when learning guides evolution. *Nature*, 329:761–762.

McLeaod, P., Plunkett, K., and Rolls, E. T. (1998). *Introduction to Connectionist Modelling of Cognitive Processes*. Oxford University Press, New York.

Menczer, F. and Belew, R. K. (1994). Evolving sensors in environments of controlled complexity. pages 210–221. Cambridge, Massachusetts: MIT press.

Mitchell, M. and Forrest, S. (1994). Genetic algorithms and artificial life. *Artificial Life*, 1:267–289.

Noble, J. (1997). The scientific status of artificial life. In *Fourth European Conference on Artificial Life (ECAL'97)*.

Nolfi, S., Elman, J. L., and Parisi, D. (1994). Learning and evolution in neural networks. *Adaptive Behavior*, pages 5–28.

Nolfi, S. and Floreano, D. (2004). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines (Intelligent Robotics and Autonomous Agents)*. MIT Press, Cambridge, MA.

Parisi, D., Nolfi, S., and Cecconi, F. (1992). Learning, behavior and evolution. In Varela, F. and Bourgine, P., editors, *Toward a Practice of Autonomous Systems*, pages 207–216. Cambridge, Massachusetts: MIT press.

Puentedura, R. R. (2003). The baldwin effect in the age of computation. In Depew, D. J. and Weber, B. H., editors, *Evolution and Learning – the Baldwin effect reconsidered*, pages 219–234. Cambridge, Massachusetts: MIT press.

Risan, L. C. (1997). Artificial life: A technoscience leaving modernity? an anthropology of subjects and objects. Master's thesis, TMV-senteret.

Searle, J. R. (1992). *The Rediscovery of the Mind*. MIT Press.

Searle, J. R. (1997). *The Mystery of Consciousness*. MIT Press.

Seymour, M. A. B. R. (1994). Adaptation of mutation rates in a simple model of evolution. In Stonier, J. R. and Yu, X. H., editors, *Complex SystemsMechanism of Adaptation*, pages 37–44. Amsterdam: IOS Press.

Stearns, S. C. and Hoekstra, R. F. (2000). *Evolution: an introduction*. Oxford University Press Inc., New York.

Sterelny, K. (2004). The baldwin effect and its significance: A review of Bruce Weber and David Depew (eds.) evolution and learning: The baldwin effect reconsidered. *Evolution and Development*, 6:295–300.

Suzuki, R. and Arita, T. (2004). Drastic changes in roles of learning in the course of evolution. In *Proceedings of Artificial Life IX*, pages 369–374.

Suzuki, R. and Arita, T. (2007a). The dynamic changes in roles of learning through the baldwin effect. *Artificial Life*, 13(1):31–43.

Suzuki, R. and Arita, T. (2007b). Repeated occurrences of the baldwin effect can guide evolution on rugged fitness landscapes. *Proceedings of the IEEE Symposium on Artificial Life (CI-ALife'07)*, pages 8–14.

Todd, P. M. and Miller, G. F. (1993). Parental guidance suggested: How parental imprinting evolves through sexual selection as an adaptive learning mechanism. *Adaptive Behavior*, pages 5–47.

Todd, P. and Miller, G. F. (1991). *Exploring adaptive agency II: Simulationg the evolution of associative learning.* Cambridge, Massachusetts: MIT press.

Turney, P. (1996). Myths and legends of the baldwin effect. In *In Proceedings Workshop on Evolutionary Computation and Machine Learning at the 13th International Conference on Machine Learning*, pages 135–142.

Waddington, C. H. (1953). Genetic assimilation of an acquired character. *Evolution*, pages 118–126.

7 Appendix

7.1 Source code

```
from random import *
from Individual import Individual
from NKLandscape import *

class Population(object):

    def __init__(self, genomeSize, populationSize, learningRate, learningCost):
        self.size = populationSize
        self.genomeSize = genomeSize
        self.population = []
        self.alleles = [1,0]
        self.learningCost = learningCost
        i = 0
        j = 0
        self.plasticityAlleles = []
        while i < self.genomeSize:
            while j < learningRate:
                self.plasticityAlleles.append(1)
                i += 1
                j += 1
            self.plasticityAlleles.append(0)
            i += 1

    def initialize(self):
        """
        Makes a population of self.size individuals. Two genomes are constructed,
        one genotype and one plasticity genome. The value of alleles are chosen randomly.
        """
        i = 0
        j = 0
        for i in range(self.size):
            genome = []
            plasticity = []
            for j in range(self.genomeSize):
                genome.append(choice(self.alleles))
                plasticity.append(choice(self.plasticityAlleles))
            self.population.append(Individual(genome, plasticity, self.learningCost))
        return self.population


from NKLandscape import *
from random import *
from Numeric import *
import copy


class Individual:

    def __init__(self, genome, plasticity, learningCost):
        self.genotype = genome
        self.phenotype = copy.deepcopy(genome)
        self.plasticity = plasticity
        self.trials = 0
        self.maxFitness = 1.0
        self.learningCost = learningCost
        self.fitness = gLandscape.fitness(genome)
```

```
    self.maxTrials = 100

def setPlasticityGenome(self, plasticity):
    self.plasticity = plasticity

def setGenotype(self, genotype):
    self.genotype = genotype

def getPlasticityGenome(self):
    return self.plasticity

def getGenotype(self):
    return self.genotype

def getPhenotype(self):
    return self.phenotype

def getGoal(self):
    return self.goal

def getFitness(self):
    return self.fitness

def getPlasticity(self):
    return self.plasticity.count(1)

def plasticityIndex(self):
    index = []
    for i in range(len(self.plasticity)):
        if self.plasticity[i] == 1: # is the genome plastic?
            index.append(i) # then add it to the index
    return index

def computeFitness(self, trials):
    originalFitness = gLandscape.fitness(self.genotype)
    currentFitness = gLandscape.fitness(self.phenotype)
    nPlasticityGenomes = self.getPlasticity()
    genomeSize = len(self.genotype)
    return originalFitness + ((currentFitness - originalFitness)
     * ((self.maxTrials - trials) / float(self.maxTrials))) - (self.learningCost *
     (nPlasticityGenomes / float(genomeSize)))

def learn(self): # learnRandom
    """
    Performs a random search in the space of the plastic alleles. A cost
    based on number of learning trials is extracted from the final fitness.
    """
    # First we have to see if the pehnotype is plastic at all.
    if self.getPlasticity() < 1:
        self.fitness = gLandscape.fitness(self.genotype)
    else:
        index = self.plasticityIndex()
        max = gLandscape.fitness(self.phenotype)
        lastChange = 0
        trials = 0
        best = copy.deepcopy(self.phenotype)
        test = copy.deepcopy(self.phenotype)
        while trials < self.maxTrials:
            for j in index:
                test[j] = choice([0,1])
                #self.phenotype[j] = choice([0,1])
            f = gLandscape.fitness(test)
```

```
                #f = gLandscape.fitness(self.phenotype)
                trials += 1
                if f > max:
                    max = f
                    lastChange = trials
                    best = copy.deepcopy(test)
            self.phenotype = copy.deepcopy(best)
            self.fitness = self.computeFitness(lastChange)

            return self.fitness

    def learnC(self): #learnContinualAssessment
        """
        Continual
        """
        # First we have to see if the pehnotype is plastic at all.
        if self.getPlasticity() < 1:
            self.fitness = gLandscape.fitness(self.genotype)
        else:
            self.fitness = gLandscape.fitness(self.genotype)
            # first find index of every plastic allele in genome
            index = self.plasticityIndex()
            max = gLandscape.fitness(self.phenotype)
            lastChange = 0
            fitnessScores = []
            trials = 0
            best = copy.deepcopy(self.phenotype)
            test = copy.deepcopy(self.phenotype)
            while trials < self.maxTrials:
                for j in index:
                    test[j] = choice([0,1])
                f = gLandscape.fitness(test)
                fitnessScores.append(f)
                trials += 1
                if f > max:
                    max = f
                    lastChange = trials
                    best = copy.deepcopy(test)
            self.phenotype = best

            searchFitness = 0
            count = 0
            for i in range(lastChange):
                searchFitness += fitnessScores[i]
            searchFitness = searchFitness / self.maxTrials

            optimalFitness = 0
            for i in range(lastChange, self.maxTrials):
                optimalFitness += fitnessScores[i]
            optimalFitness = optimalFitness / self.maxTrials
            self.fitness = optimalFitness + searchFitness
            return self.fitness


from random import *
from Numeric import *
from __future__ import division
import Singleton


class NKLandscape(Singleton.Singleton):
    """
```

```
Kaufman's NK fitness landscape.
"""
def __init__(self, n, k):
    """
    n (int) => length of genome
    k (int) => degree of epistasis
    """
    self.n = n
    self.k = k
    self.fitnessTable = []
    self.previousNeedle = []
    self.globalOptima = False

def makeLandscape(self):
    self.__makeTable()

def setGlobalOptima(self, g):
    self.globalOptima = g

def setN(self, n):
    self.n = n

def setK(self, k):
    self.k = k

def __epistasisTable(self):
    """
    Creates table of allele interaction of length 2^(K+1).
    Each possible alelle permutation is assigned a uniform random number
    in [0.0,1.0].

    <= (dict) epistasis
    """
    length = pow(2, (self.k + 1))
    epistasis = {}
    for i in range(length):
        epistasis[i] = uniform(0.0,1.0)
    return epistasis

def __makeTable(self):
    """
    Creates a table of length N which points to the table of epistasis for
    each allele.
    """
    table = []
    for i in range(self.n):
        table.append(self.__epistasisTable())
    self.fitnessTable = table

def makeIndex(self, individual):
    """
    Takes a genome and constructs the index of alleles for the genome.

    => genome (list) binary bitstring
    <= index (list) index for the alleles of the genome
    """
    length = len(individual)
    index = []
    for i in range(length):
        # combine i with its k neighbours
        string = str(individual[i])
        for j in range(self.k):
```

```
            pos = i + j + 1
            if pos < length:
                string += str(individual[pos])
            else:   # we need to wrap around the string
                pos = pos - length - 1
                string += str(individual[pos])
        index.append(string)
    return index

def fitness(self, individual):
    """
    Returns the fitness of an individual. We use ajacant neighborhoods.

    individual (array) => array of 1 and 0
    <= fitness (real)
    """
    # First we need to construct index of ajacant alleles
    index = self.makeIndex(individual)
    # Traverse index and get fitness
    sum = 0
    for i in range(len(index)):
        dec = int(index[i],2) # convert the bit string to decimal
        epistasis = self.fitnessTable[i]
        sum += epistasis[dec]
    if individual == self.globalOptima: # FIXME: Ugly hack to hide the needle
        return 1.0
    else:
        return sum / len(index) #self.n

def hideNeedle(self, needle):
        """
        Changes the NK lanscape so that genomes corresponding to needle
        will be rewarded with optimal fitness (1.0).

        => needle (list) bitstring of {1,0}
        <= void
        """
        # Is there already a needle in the landscape? Then romove it.
        if self.previousNeedle != []:
            self.removeNeedle(self.previousNeedle)
        self.previousNeedle = needle
        # Get the epistasis index for each gene in the genome
        index = self.makeIndex(needle)
        # for each table replace index with max fitness 1.0
        for i in range(len(index)):
            dec = int(index[i],2) # convert the bit string to decimal
            self.fitnessTable[i][dec] = 1.0

def removeNeedle(self, needle):
        index = self.makeIndex(needle)
        # for each table replace index with max fitness 1.0
        for i in range(len(index)):
            dec = int(index[i],2) # convert the bit string to decimal
            self.fitnessTable[i][dec] = uniform(0.0,1.0)

def saveLandscape(self):
    """ Saves the current landscape to file. """
    file = open('landscape.txt','w')
    metadata = "# N = " + str(self.n) + " K = " + str(self.k) + '\n'
    file.write(metadata)
    for epistasis in self.fitnessTable:
        line = ""
```

```
            for j in range(len(epistasis)):
                value = str(epistasis[j])
                line += value + ","
            file.write(line + '\n')
        file.close()

    def loadLandscape(self):
        """
        Loads the current landscape from file. If a landscape is already
        initialized, it will be replaced by the landscape from file.
        """
        file = open('landscape.txt','r')
        file.readline().strip() # disregard first line because its metadata
        newFitnessTable = []
        for line in file.readlines():
            values = line.split(',')
            values.pop() # remove last element because it's a newline sign
            epistasis = {}
            j = 0
            for v in values:
                v = float(v.strip())
                epistasis[j] = v
                j += 1
            newFitnessTable.append(epistasis)
        file.close()
        self.fitnessTable = newFitnessTable



class GA:
    """
    The genetic algoritm. Initializes a population, creates the fitness landscape
    and evolves.
    """

    def __init__(self, n=20):
        self.results = ''#[]
        self.population = []
        self.genomeSize = n
        self.learningCost = False
        self.tournamentSize = 0.10
        self.learning = int(self.genomeSize * 0.5)

    def setLearningRate(self, rate):
        self.learning = int(self.genomeSize * rate)

    def setTournamentSize(self, size):
        self.tournamentSize = size

    def setLearningCost(self, cost):
        self.learningCost = cost

    def setGenerations(self, nGenerations):
        self.generations = nGenerations

    def setPopulation(self, nPopulation):
        self.populationSize = nPopulation

    def setMutationRate(self, mRate):
        self.mutationRate = int(round(self.populationSize * mRate))
```

```
def setCrossoverRate(self, cRate):
    self.crossoverRate = cRate

def setGenomeSize(self, gSize):
    self.genomeSize = gSize

def initialize(self):
    """
    Generates populationSize number of individuals. Creates an NK
    fitness lanscape.
    """
    p = Population(self.genomeSize, self.populationSize, self.learning, self.learningCost)
    self.population = p.initialize()

def randomMutation(self, string):
    """
    0.2 of the bitstring is flipped.
    """
    def flip(bit):
        if bit == 1:
            return 0
        else:
            return 1

    length = len(string)
    nFlips = int(round(length * 0.2))
    index = range(length)
    shuffle(index)
    for i in range(nFlips):
        #newValue = choice([0, 1])
        #string[index[i]] = newValue
        string[index[i]] = flip(string[index[i]])
    return string

def onePointCrossover(self, p1, p2):
    """
    Creates two children.

    => p1, p2 individuals
    """
    def combine(g1, g2, point):
        g1 = g1[:point]
        g2 = g2[point:]
        return g1 + g2

    # choose two random corssoverpoints
    genotypeLength = len(p1.getGenotype())
    gPoint = choice(range(genotypeLength -1))
    pPoint = choice(range(genotypeLength -1))
    genotype1 = combine(p1.getGenotype(), p2.getGenotype(), gPoint)
    genotype2 = combine(p2.getGenotype(), p1.getGenotype(), gPoint)
    plasticity1 = combine(p1.getPlasticityGenome(), p2.getPlasticityGenome(), pPoint)
    plasticity2 = combine(p2.getPlasticityGenome(), p1.getPlasticityGenome(), pPoint)
    learningCost = p1.learningCost
    child1 = Individual(genotype1, plasticity1, learningCost)
    child2 = Individual(genotype2, plasticity2, learningCost)
    children = []
    children.append(child1)
    children.append(child2)
    return children

def uniformCrossover(self, p1, p2):
```

```
    """
    Each allele in child is randolmly choosen from one of the parents.

    => p1, p2 individuals
    <= child indivudal
    """
    def combine(s1, s2):
        """ Returns s1 and s2 combined randomly """
        comb = []
        for i in range(len(s1)):
            if choice([0,1]) == 0:
                comb.append(s1[i])
            else:
                comb.append(s2[i])
        return comb

    genotypeLength = len(p1.getGenotype())
    genotype1 = p1.getGenotype()
    genotype2 = p2.getGenotype()
    plasticity1 = p1.getPlasticityGenome()
    plasticity2 = p2.getPlasticityGenome()
    childPlasticity = combine(plasticity1, plasticity2)
    childGenotype = combine(genotype1, genotype2)
    learningCost = p1.learningCost
    child1 = Individual(childGenotype, childPlasticity, learningCost)
    childPlasticity = combine(plasticity2, plasticity1)
    childGenotype = combine(genotype2, genotype1)
    child2 = Individual(childGenotype, childPlasticity, learningCost)
    children = []
    children.append(child1)
    children.append(child2)
    return children


def tournamentSelection(self, tournamentSize):
    """
    Chooses one parent for crossover.
    Returns the index of the parent.
    """
    index = False

    def winner(disorder):
        """
        Returns index to the best individual.
        """
        max = 0
        best = False
        for i in disorder:
            #fitness = gLandscape.fitness(self.population[i].getPhenotype())
            fitness = self.population[i].fitness
            if fitness >= max:
                max = fitness
                best = i
        return best

    # pick k individuals randomly
    disorder = range(self.populationSize)
    shuffle(disorder)
    disorder = disorder[:tournamentSize]
    # pick the best individual
    return winner(disorder)
```

```
def sortPopulation(self):
    """
    Sorts the population, individuals with highest fitness first.
    """
    def numericCompare(x, y):
        """ Help-function for sorting."""
        if x>y:
            return 1
        elif x==y:
            return 0
        else:
            return -1

    self.population.sort(lambda x, y: numericCompare(y.fitness, x.fitness))

def test(self):
    """
    Tests the current generation's fitness.
    """
    bestInnate = 0.0
    best = 0.0
    sum = 0.0
    innateSum = 0.0
    worst = 1.0
    plasticity = 0.0
    for i in self.population:
        fitness = i.fitness
        if fitness > best: best = fitness
        if fitness < worst: worst = fitness
        innateSum += gLandscape.fitness(i.getGenotype())
        innateFitness = gLandscape.fitness(i.getGenotype())
        if innateFitness > bestInnate: bestInnate = innateFitness
        sum += fitness
        plasticity += i.getPlasticity()
    average = sum / self.populationSize
    innateAverage = innateSum / self.populationSize
    averagePlasticity = (plasticity / float(self.populationSize)) / float(self.genomeSize)
    if self.learning:
        self.results += str(best) + ' ' + str(average) + ' ' + str(worst) + ' ' + str(averagePlasticity) + ' ' +
    else:
        self.results += str(best) + ' ' + str(average) + ' ' + str(worst) + '\n'
    return best


def learn(self):
    """
    Makes every individual in the population perform learning.
    """
    for ind in self.population:
        ind.learn()

def nextGeneration(self):
    """
    Makes self.crossoverRate number of children.
    """
    children = []
    self.sortPopulation()
    for i in range(int(self.crossoverRate / 2)): # Denne slik mens vi tester ut
        # choose two parents
        tSize = int(self.populationSize * self.tournamentSize)
        #self.population = self.operators.linearRank(self.population)
        index1 = self.tournamentSelection(tSize)
```

```
            index2 = self.tournamentSelection(tSize)
            p1 = self.population[index1]
            p2 = self.population[index2]
            # mate them
            #child = self.onePointCrossover(p1, p2)
            child = self.uniformCrossover(p1, p2)
            children.append(child[0])
            children.append(child[1])
        self.population[(self.populationSize - self.crossoverRate):] = children # population is sorted lowest first,
        self.sortPopulation()
        # mutate
        disorder = range(self.populationSize)
        shuffle(disorder)
        disorder = disorder[:self.mutationRate]
        for i in disorder:
            self.population[i].setGenotype(self.randomMutation(self.population[i].getGenotype()))
            if self.learning:
                self.population[i].setPlasticityGenome(self.randomMutation(self.population[i].getPlasticityGenome()))


    def replaceWorst(self, children):
        """
        Replaces
        """
        True

    def evolve(self):
        """
        Runs the GA for generations generations.
        Returns string of results.
        """
        i = 0
        while i <= self.generations:
            if self.learning:
                self.learn()
            self.nextGeneration()
            best = self.test()
            i += 1
        print best
        return self.results


def writeResults(results):
    file = open('result.txt','w')
    file.write("#Best,Average,Worst,Plasticity,InnateAverage\n")
    for line in results:
        file.write(line)
    file.close()
```