ON THE EVOLUTION OF IRRATIONAL BEHAVIOUR

JOHN A. BULLINARIA
School of Computer Science, The University of Birmingham
Edgbaston, Birmingham, B15 2TT, UK
j.bullinaria@physics.org

Many aspects of human and animal behaviour require individuals to learn quickly how to classify the patterns they encounter. One might imagine that evolution by natural selection would result in neural systems emerging that are very good at learning things like this. Explicit simulations of the evolution of simple developmental neural systems confirm that such rational behaviour can indeed emerge quite easily. However, the same simulations also reveal that there are situations in which evolution seems to let the species down, and populations emerge that appear to perform rather irrationally. There are actually many reasons why this might happen. I shall present the results from a selection of my simulations that begin to explore the issues involved.

1. Introduction

A standard definition of *irrational behaviour* is the performance in a manner which goes against one's objectives [10]. This can range from minor but persistent miscalculation of probabilities resulting in poor decisions, through to apparently totally random actions. Understanding why individuals should ever act in such a way is an interesting psychological research area in its own right [5], and there are many other fields in which irrational behaviour is studied in detail, e.g. finance [7] and law [8]. In this paper I shall begin to explore how such behaviour might evolve in simple neural network systems.

Many aspects of human and animal behaviour require individuals to learn quickly how to classify the patterns they encounter, and how to act on those classifications. For example, which foods are good or safe to eat, which other animals should be feared, which environments should be avoided, when to sell your stocks and shares, and so on. One might imagine that evolution by natural selection would result in neural systems emerging that are very good at learning things like this. This would certainly fit in well with modern ideas of evolutionary psychology as an explanation of human behaviour [6]. The question that still needs addressing is: why do individuals so often apparently get things wrong and act in an irrational manner (e.g. losing money by selling their shares too early [7], or by breaching a contract [8])? If we can understand such behaviour, and perhaps predict that behaviour, then, depending on our own motives, we can maybe help, or take advantage of, that behaviour.

Over the past few years I have been performing increasingly biologically realistic simulations of the evolution of neural systems with view to better understanding how various aspects of the human information processing system works. These have included the evolution of systems that perform adaptive control [4] and general binary mappings [3], and also explorations into the evolution of modularity in these systems [2]. Here I shall concentrate on simulations of the evolution of simple developmental neural systems that are required to classify various types of sensory information. We shall see that rational behaviour, in the form of learning to classify quickly and reliably, can emerge quite easily. However, in the same simulations we often find that there are situations in which evolution seems to lets the species down, and populations emerge that appear to perform rather irrationally. There are actually many reasons why this might happen, and often the seemingly irrational behaviour does actually make good sense. One can then attempt to discover which aspects of human behaviour this might correspond to [5, 7, 8, 10]. In the remainder of this paper I shall present the results from a small selection of my simulations that begin to explore the issues involved.

2. The Evolutionary Models

The field of Evolutionary Neural Networks is already quite mature [11], but here we are interested in simulations that are more biologically realistic than is usually the case. Two aspects are particularly important:

1. *The Evolutionary Population*. Biological populations usually contain individuals of all ages, who learn from their environment, compete amongst themselves for the opportunity to live and procreate, but eventually die of old age. We want to avoid a traditional generational approach [11] in which whole populations are created, taught, and tested one generation at a time.
2. *The Environment*. Biological individuals learn by sampling a continuous range of environmental conditions, and must generalize to cope with sensory information they have never seen before. We want to avoid the traditional training data split into fixed training and testing sets [1].

Naturally, we shall still have to make an enormous number of simplifications and abstractions, but an effort will be made to accommodate these two aspects.

We shall work with a population of individual neural networks, each specified by a number of 'innate' parameters that are recorded in its genotype. For simplicity, we restrict ourselves to simple feed-forward networks with one hidden layer trained by gradient descent with momentum [1]. The training data

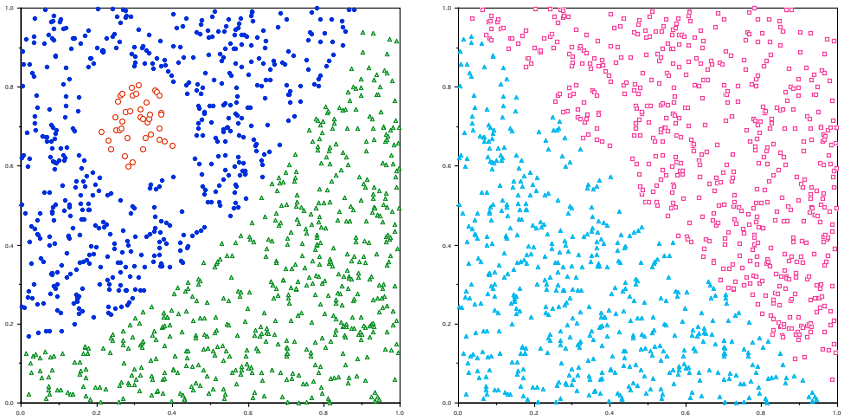

Figure 1: Samples from the dual classification training data distributions. One case (left) has three classes (two large separated classes and one small surrounded class), while the other (right) has two classes (two different large separated classes).

will specify the number of inputs and outputs. The genotype will specify the network architecture, the initial weight distributions, and the learning rates. During each simulated year, the least fit individuals and the oldest individuals will die (i.e. be removed from the population) to be replaced by the children of the fittest individuals. The genotype of each child will depend on the genotypes of its two parents, plus random mutations. We can then expect evolution by natural selection to produce increasingly fit populations.

3. Simulation Details

Here it is convenient to consider a particularly simple environment in which each individual has just two continuous valued sensor inputs, and must perform two distinct classifications on them. Figure 1 shows samples from the distributions used in all the simulations. The inputs might, for example, correspond to particular components of smell and colour, and the two output cases could be classes of usefulness, such as {good tasting, bad tasting, poisonous} and {easy to get, hard to get}. Alternatively, the inputs could be financial indicators, and the outputs could be trading strategies.

During each individual's life, they will experience a stream of samples from the input distribution, and must learn quickly how to perform well on any new samples they come across. The individual neural networks need to perform two distinct classifications based on data in a single input space. We shall therefore

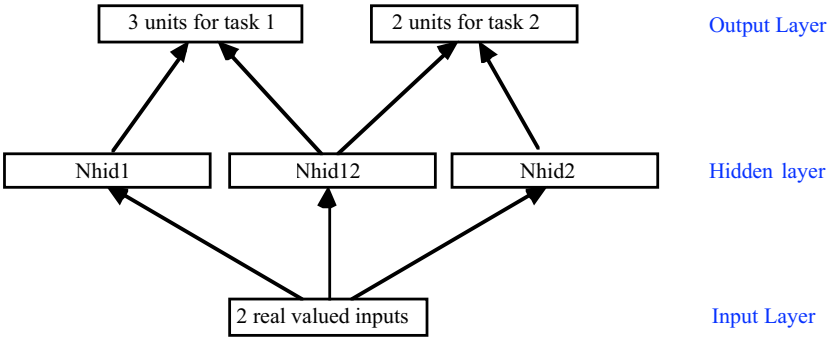

Figure 2: Architecture of the individual neural networks.

parameterize the networks in such a way that they may evolve either a modular, or non-modular, architecture to do this [2]. At the same time, they are also allowed to evolve their own associated learning algorithm and parameters [3].

The basic evolutionary implementation has been described in some detail previously [2, 3, 4], so I shall simply summarize the main details here. Each individual neural network will have the same basic architecture as shown in Figure 2, with the same total number of hidden units $Nhid = 36$. Then two parameters $Con1 = Nhid1 + Nhid12$ and $Con2 = Nhid12 + Nhid2$ are sufficient to specify how many hidden units connect to each output block, and how many connect to both. If *Nhid12* takes on a zero value, we have a totally modular architecture, and if both *Nhid1* and *Nhid2* become zero, we have a totally non-modular architecture [2]. Each new network is created with random initial weights drawn from innately specified distributions, and learns using a learning algorithm with innately specified parameters [3]. For each network it proves appropriate to have four uniform initial random weight distributions $[-iw_L, +iw_L]$, four learning rates η_L, and a momentum parameter α, in which L refers to the network layer (input to hidden, hidden biases, hidden to output, output biases). It is likely that the optimal parameters will depend on which gradient descent cost function is used. Here we shall compare using a sum squared error cost function with targets of 0.1 and 0.9 as used in the original modularity study of Rueckl et al. [9], and the cross-entropy cost function that is now known to be better for classification problems [1, 3].

The evolutionary survival/procreation fitness is inversely and logarithmically related to the total number of classification errors during the last simulated year as $1/\log(1+ErrorCount)$. Each individual experiences 125 randomly selected data samples per simulated week. The population size is fixed at 100, with

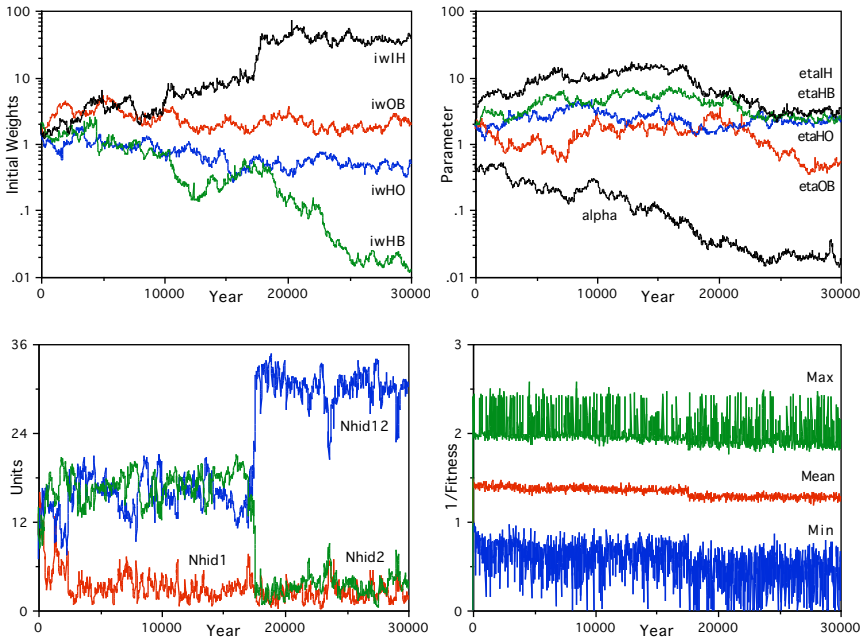

Figure 3: Typical evolution of the mean initial weight distributions, learning rates, architecture parameters, and population fitnesses for the sum squared error model.

death rates chosen to give realistic age distributions, with life expectancies of about thirty years. The procreation and mutation parameters are chosen to maintain genetic diversity without introducing too much noise. The resulting simulations are rather course, but good enough to provide reliable results.

4. Simulation Results

The complexity of our tasks were chosen so that our simple neural networks could evolve to perform essentially perfectly on them. If populations emerge that perform in a manner which does not achieve their objective, then this can be classed as an example of irrational behaviour, and we will wish to investigate further, and perhaps relate it to corresponding patterns of human behaviour.

Long simulation runs indicate that the evolving populations usually stabilize by about 30,000 simulated years. Figure 3 shows the typical evolutionary pattern when the sum squared error cost function is used for learning. We find that similar values emerge for the four learning rate parameters, but there are very large variations between the evolved initial weight

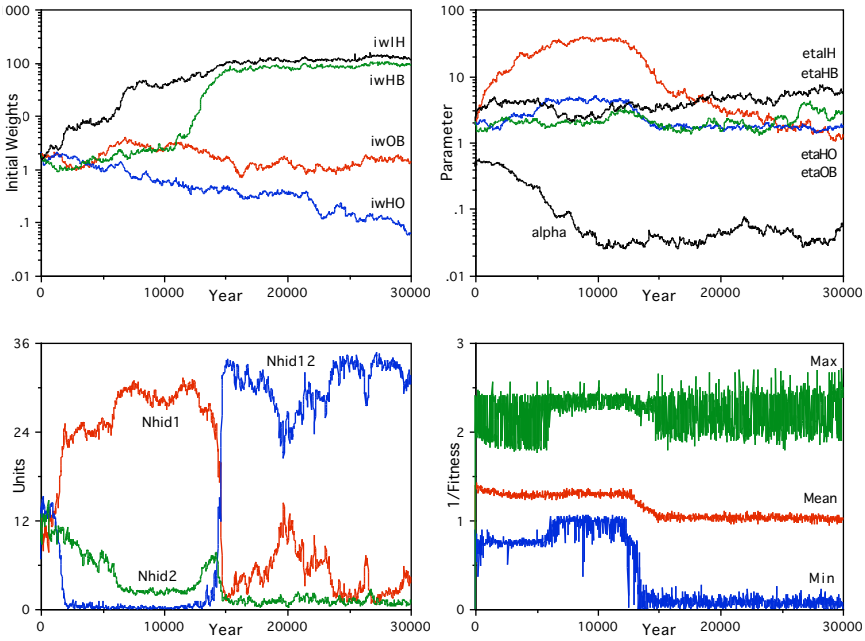

Figure 4: An unusual evolutionary pattern for the mean initial weight distributions, learning rates, architecture parameters, and population fitnesses, for the same underlying system as that of Figure 3.

distributions. We see that a strange equilibrium state exists for about 18,000 years, before the rapid emergence of a totally non-modular architecture and a step change in the mean fitness level of the population. Notice that even the best performing individuals rarely manage to learn the task to perfection.

As always, it is important to test the robustness of the simulation results. Numerous runs of the same system, with different random initial conditions and training data, reveal that the populations can end up in at least two different local maxima of evolutionary fitness. Figure 4 shows an evolutionary run that ends up with a final population that has significantly better mean fitness and best individual fitnesses than the typical run shown in Figure 3. The final non-modular architecture and learning rates are much the same as before, but the initial weight distributions are very different, with a much closer correspondence here between the input-hidden initial weights and the initial hidden unit biases. Interestingly, between years 6,000 and 12,000 the populations go through a pure modular phase in which the best individual performances are worse than before, but without making the mean fitness worse. The individuals there have evolved

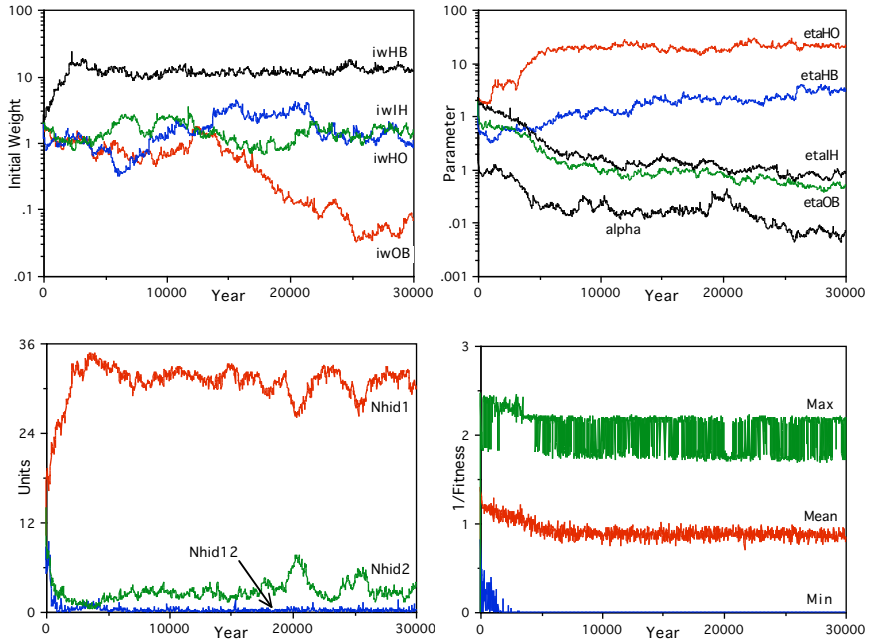

Figure 5: Typical evolution of the mean initial weight distributions, learning rates, architecture parameters, and population fitnesses for the cross-entropy model.

to learn faster early on, at the expense of their final performances. At the end of this phase the architecture and initial hidden unit biases drift into the better non-modular regime, and the learning rates soon follow suit, ending up with a population of individuals that perform better throughout their lives.

My earlier study on modularity evolution [2] indicated a strong dependence on the gradient descent learning cost function, so the above simulations were repeated using the cross-entropy cost function [1, 3]. Figure 5 shows the patterns of evolution typically observed in this case. As one might expect, the evolved initial weight distributions and learning rates are rather different to the sum-squared error case. Moreover, the population mean fitnesses are better, and individuals regularly learn the tasks to perfection. The individuals achieve this by evolving a modular architecture, with the numbers of hidden units assigned to each of the two tasks in agreement with their relative difficulty, and in line with the modular phase observed in Figure 4.

In my previous study of modularity using the simpler fixed "What-Where,, training data [2], it was the sum squared error systems that evolved modular architectures, and the better performing cross-entropy systems evolved non-

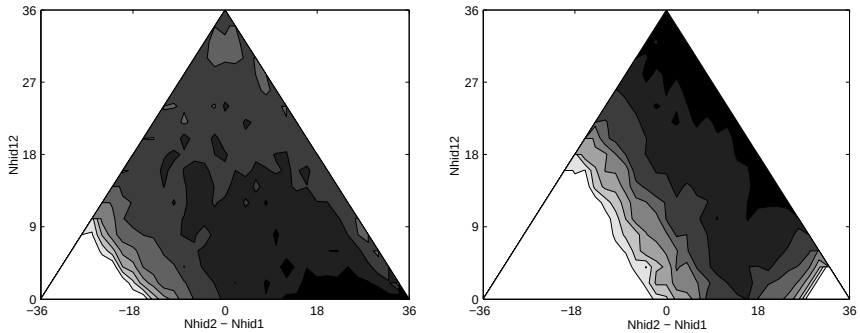

Figure 6: Mean learning times for the cross-entropy model as a function of architecture using the evolved learning parameters (left) and parameters evolved for a fixed non-modular architecture (right). Darker shading means faster learning.

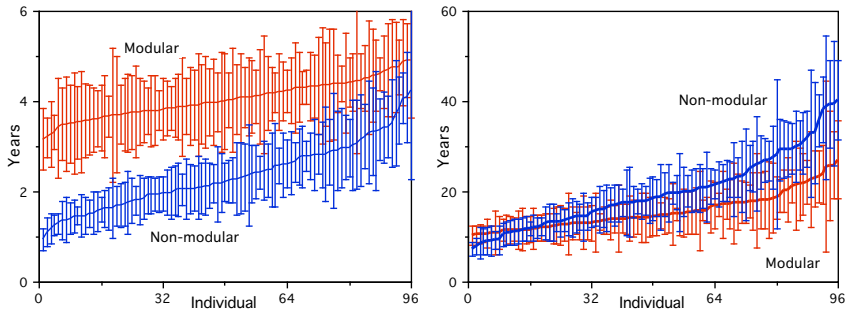

Figure 7: Ages at first full week of perfect performance (left) and first full year of perfect performance (right) for the modular and non-modular individuals.

modular architectures. We can investigate the fitness dependence on architecture by training a series of networks using the evolved initial weight distributions and learning rates, and plotting as a function of architecture the mean times taken to reach the first full week of perfect performance. This is shown for the cross-entropy case on the left of Figure 6. The fastest learners do indeed occur in the modular regime ($Nhid12 = 0$) with more hidden units dedicated to the harder task ($Nhid2 - Nhid1 > 0$) as the evolutionary simulations suggest.

The question remains as to what architecture comes out best if we were to optimize the learning parameters for a totally non-modular architecture? It is easy to fix the architecture and evolve those parameters, and then test the learning times again. This is shown on the right of Figure 6. Now the fastest learners are in the non-modular ($Nhid12 = 36$, $Nhid1 = Nhid2 = 0$) regime.

Moreover, if we run a full evolutionary simulation with evolvable architectures, starting with the non-modular population, the population stays non-modular, indicating that it is a stable local maximum of evolutionary fitness.

The natural question to ask is: does the non-modular population perform better than the evolved population? Individuals in both populations regularly learn the task to perfection, but it is the non-modular population that has the lower mean fitness, indicating that its individuals tend to learn faster. We can explore this more carefully by plotting the learning times from Figure 6 for each individual. The left graph of Figure 7 shows the means and standard deviations over ten sets of training data, with the individuals ordered for clarity. We see that the non-modular individuals are indeed better, which makes us wonder why the modular population evolved. The crucial fact is that our simulated evolution depends on the performance over a full year, not a single week. The right graph in Figure 7 shows the times taken to reach the first full year of perfect performance, and now the modular population (with mean 15.7) does out-perform the non-modular population (mean 20.1). This factor also provides a likely explanation of why using fixed training sets, rather than the more realistic continuous data sampling used here, led to different architectures evolving [2].

5. Discussion and Conclusions

I have outlined a framework for simulating the evolution of neural network systems that must learn to perform simple classification tasks, and presented a small selection of my simulations which show how behaviour can evolve which is less than optimal and might therefore be classed *irrational*. By making these simulations sufficiently realistic, one can hope to come to a better understanding of the emergence of irrational behaviour in real systems.

We have seen examples of several factors that can lead to what looks like irrational behaviour, but make sense when we investigate why they arise. Often it is simply the result of conflicting requirements on the system. We found:

1. Reliable good performance might require slower learning, and may even be more important than good average performance,
2. Fast initial learning might mean bad final performance, and good final performance might need long learning times.

In the psychology and economics literature, the term *bounded rationality* is often used to describe the results of such constrained optimization [5, 8]. Our simulations have also found purely evolutionary factors:

3. Optimal behaviour may simply be difficult or extremely slow to evolve,

4. Certain non-evolvable aspects of the learning mechanism can place limitations on what objectives are actually achievable,
5. Accidents of evolutionary history may leave populations in local maxima of fitness, even though they could in principle perform better.

Further simulations are likely to reveal other important factors, such as:

6. Robustness against mutations might limit performance levels,
7. The importance of co-evolution and coping with very noisy data.

It is clear that there is much scope for further research in this area. Perhaps this paper will give others an idea of the range of effects that can come out of simple evolutionary neural network simulations.

References

1. Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press.
2. Bullinaria, J. A. (2001). Simulating the Evolution of Modular Neural Systems. In *Proceedings of the Twenty-third Annual Conference of the Cognitive Science Society,* 146-151. Mahwah, NJ: Lawrence Erlbaum.
3. Bullinaria, J. A. (2003). Evolving Efficient Learning Algorithms for Binary Mappings. *Neural Networks*, **16**, 793-800.
4. Bullinaria, J. A. (2003). From Biological Models to the Evolution of Robot Control Systems. *Philosophical Transactions of the Royal Society of London* A, **361,** 2145-2164.
5. Chase, V. M., Hertwig, R. & Gigerenzer, G. (1998). Visions of Rationality. *Trends in Cognitive Science*, **2**, 206-214.
6. Mitchell, M. (1999). Can Evolution Explain How the Mind Works? A Review of the Evolutionary Psychology Debate. *Complexity,* **4**, 17-24.
7. Montier, J. (2002). *Behavioural Finance: Insights into Irrational Minds and Markets.* New York, NY: Wiley.
8. Parisi, F. & Smith, V. L. (2003). *The Law and Economics of Irrational Behavior*. Chicago, IL: Chicago University Press.
9. Rueckl, J. G., Cave, K. R. & Kosslyn, S. M. (1989). Why are "What,, and "Where,, Processed by Separate Cortical Visual Systems? A Computational Investigation. *Journal of Cognitive Neuroscience*, **1**, 171-186.
10. Tribus, M. (1969). *Rational Descriptions, Decisions and Designs*. New York, NY: Pergamon Press.
11. Yao, X. (1999). Evolving Artificial Neural Networks. *Proceedings of the IEEE,* **87**, 1423-1447.