Neural Plasticity and Minimal Topologies for Reward-based Learning

Andrea Soltoggio
University of Birmingham
School of Computer Science
Birmingham, B15 2TT, UK
a.soltoggio@cs.bham.ac.uk

Abstract

Artificial Neural Networks for online learning problems are often implemented with synaptic plasticity to achieve adaptive behaviour. A common problem is that the overall learning dynamics are emergent properties strongly dependent on the correct combination of neural architectures, plasticity rules and environmental features. Which complexity in architectures and learning rules is required to match specific control and learning problems is not clear. Here a set of homosynaptic plasticity rules is applied to topologically unconstrained neural controllers while operating and evolving in dynamic reward-based scenarios. Performances are monitored on simulations of bee foraging problems and T-maze navigation. Varying reward locations compel the neural controllers to adapt their foraging strategies over time, fostering online reward-based learning. In contrast to previous studies, the results here indicate that reward-based learning in complex dynamic scenarios can be achieved with basic plasticity rules and minimal topologies.

1. Introduction

Synaptic plasticity is considered a fundamental working mechanism of memory and learning in biological neural networks. Although growing experimental data support this hypothesis, the richness of neural dynamics involved with synaptic modification, and the various forms of plasticity found in brains result in a complex system to understand [8]. Various computational models of synaptic mechanisms have been formalised to describe specific neural dynamics and analyse their properties [11, 3, 7]. Aspects like synaptic stability, competitive growth, heterosynaptic growth, time-dependency and other have been included in computational models. The analysis of the computational features of a rule, however, – when considered on an open-loop single synapse or neuron – reveals properties at the cellular level, but does not disclose the potential of such rule at system level, i.e. when the rule acts in a closed-loop synergy with activations patterns, neural architectures and environmental stimuli. The system level dynamics of learning and memory is an emergent property that originates from a combination of plasticity rules, neural architectures and sensory-motor signals. Therefore, the growth of a single synapse when isolated might not be descriptive of the system dynamics that brings about learning and memory.

To overcome this problem, recent work in the fields of Robotics, Artificial Life and Neural Networks focus on the integration of learning rules into embodied neural controllers. Unfortunately, the emergent properties of learning and memory make it difficult to hand-design neural architectures for adaptive and learning robots and agents, and consequently evolutionary techniques are commonly used [6, 1, 5]. Even with the current limitation of evolutionary algorithms and scalability problems, experiments in simple settings have provided remarkable examples of learning networks that implement homosynaptic and heterosynaptic plasticity (or neuromodulation) [6, 12], resulting in various advantages of plastic networks as adaptivity, noise tolerance, better transferability from simulation to hardware, and learning capabilities.

Despite the cited examples of learning plastic networks, there is still a lack of understanding on which plasticity rule and neural architecture are required and can be coupled to solve certain learning problems. This is mainly due to the unclear mapping from local unsupervised plasticity mechanisms and system level effects. To address this issue, in this paper the analysis of both architectures and plasticity rules is carried out when networks operate and evolve in reward-based environments. The chosen simulated environments are inspired by natural foraging problems and are implemented here as bee-foraging uncertain scenarios and T-maze navigation and learning problems. In these scenarios, different plasticity rules are tested with evolving architectures. Plasticity rules do not include synaptic normalization, competitive growth or heterosynaptic mechanisms

(neuromodulation) [4] to reduce the complexity of models and find the minimal required mechanisms. Evolutionary runs are performed to observe the emergence of learning and memory dynamics that allow for high performing neural controllers. In contrast to previous studies [9, 10], the results indicate that complex learning problems in uncertain stochastic environments can be solved with simple homosynaptic plasticity rules and minimal architectures.

The next section outlines the plasticity rules used for this study and the learning problems employed to evolve and test learning networks. Section 3 describes the implementation details of the evolutionary algorithm. Sections 4 and 5 present the results and the analysis before the conclusion.

2. Synaptic Plasticity and Learning Scenarios

Recent studies have employed the following plasticity rule where the weight update δ is a function of pre- and postsynaptic activities

$$\delta_{ji} = \eta \cdot [Ao_j o_i + Bo_j + Co_i + D] \quad (1)$$

where o_j and o_i are the pre- and postsynaptic neuron outputs, η, A, B, C, and D are tunable parameters [10, 13, 12].

According to Equation 1, the update of a synaptic weight occurs as a function of 1) correlated activity weighted on parameter A, 2) presynaptic activity weighted on parameter B, 3) postsynaptic activity weighted on parameter C, 4) independently of pre- and postsynaptic activity based on parameter D. The last term (parameter D) is a constant weight decay or increase that, when combined with neuromodulation [12], results in pure heterosynaptic update. As heterosynaptic plasticity is not considered here, only A, B and C will be used to form the following 7 particular rules

$$\delta_{ji} = \eta \cdot Ao_j o_i \quad (2)$$
$$\delta_{ji} = \eta \cdot Bo_j \quad (3)$$
$$\delta_{ji} = \eta \cdot Co_i \quad (4)$$
$$\delta_{ji} = \eta \cdot [Ao_j o_i + Bo_j] \quad (5)$$
$$\delta_{ji} = \eta \cdot [Ao_j o_i + Co_i] \quad (6)$$
$$\delta_{ji} = \eta \cdot [Bo_j + Co_i] \quad (7)$$
$$\delta_{ji} = \eta \cdot [Ao_j o_i + Bo_j + Co_i] \quad (8)$$

The first three rules use correlation, pre- and postsynaptic mechanisms separately and independently. The next three rules are linear combinations of two of the previous ones. The last rule is a combination of all terms. These seven rules represent particular instances of the general rule of Equation 1 when some of the parameters are clamped to 0. The purpose is to test the minimal sufficient dynamics for solving the proposed problems. Equations 2-8 are unstable as growth in synaptic weight due to neural activity leads to a positive feedback. More complex models like the Oja rule [11] and the BCM rule [3] implement synaptic normalization and competitive growth. In this study, the synaptic growth is limited by a saturation value.

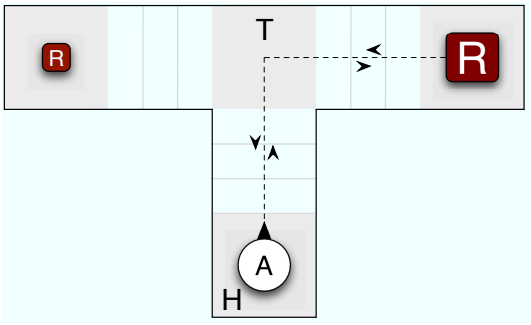

Figure 1. T-maze: An agent navigates the corridors, makes a choice at the turning point and reaches the end collecting a reward proportional to the size of the token.

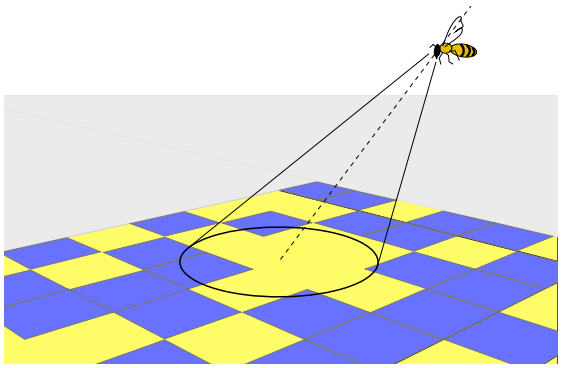

Figure 2. Simulated bee on a flower field.

2.1 Dynamic Reward-based Scenarios

Two learning problems have been utilized here. In the T-maze represented in Figure 1, an agent performs a number of trips (trials) from the home location to the maze-ends. Different maze-ends yield different amounts of rewards. The purpose of the agent is to visit repeatedly the maze-end with the higher reward in order to maximise the reward during a lifetime. To reach a maze-end, the agent encounters a turning point upon which a choice between left and right turn will be taken. An extended version of this problem can be specified by requiring to the agent to return home after the reward has been collected.

From an evolutionary perspective, if the position of the high reward was fixed to one location, say left, such information will be encoded in the genome, allowing the agent

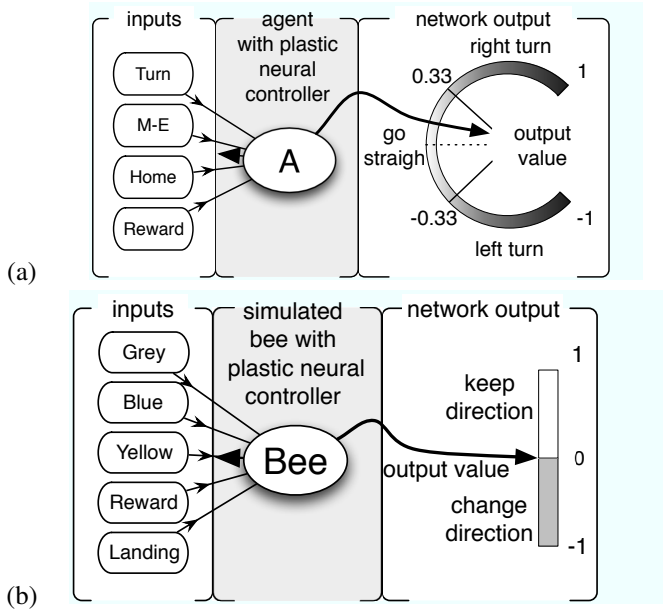

Figure 3. Inputs and output of neural networks. (a) T-maze agent: the *Turn* input is 1 when a turning point is encountered. Maze-End goes to 1 at the end of the maze. *Home* becomes 1 at the home location. The *Reward* input returns the amount of reward collected at the maze-end, it remains 0 during navigation. One output determines the actions of turning left (if less than -1/3), right (if greater than 1/3) or straight navigation otherwise. (b) The bee receives the percentages of the different colours seen under the cone view, where blue and yellow are flower-colours, grey is the colour outside the field. Upon landing, the reward input gives the amount of nectar (reward) contained in the flower. During the flight, one output neuron decides whether to maintain the current flying direction, or change to a new random direction. Inputs and internal neural transmission are affected by 2% noise.

to exploit the correct maze-end without need for learning or exploration. When, on the other hand, the location of the high reward is uncertain, and changes with a certain frequency during an agent's lifetime, the correct sequence of actions to achieve the high reward will change, compelling the agent to adapt its policy according to external reward information. This scenario requires the capability of learning new actions and retain them during the exploration and exploitation of the maze. Such environmental characteristics will result in better performance when an agent is capable

Table 1. Rewarding policies. P indicates the probability of the reward.

Scenario	Reward of the high rewarding flower	Reward of the low rewarding flower
1	1.0	0.2
2	0.7	1.0 with P=0.2 0 with P=0.8
3	1.6 with P=0.75 0 with P=0.25	0.8 with P=0.75 0 with P=0.25
4	0.8 with P=0.75 0 with P=0.25	0.8 with P=0.25 0 with P=0.75

of learning and memory.

Figure 2 illustrates a simulated flying bee on a flower field with uncertain reward conditions. The bee [10, 13] equipped with one cyclopean eye (10 degrees cone view) flies and repeatedly lands on a flower field with flowers of two different colours. One colour yields a high quantity of nectar, whilst the other colour yields a low quantity. During the bee's lifetime, the two flowers are inverted, compelling the bee to change colour preference in order to maximise the total reward (nectar) intake. Although one of the two flowers yield on average a higher reward, the content of each single flower can be either a deterministic value or a stochastic value. When flowers provide stochastic rewards, the higher rewarding flower can be identified only by averaging the samples on more trials. Here, two problems were devised, one with deterministic rewards (scenario 1), and a more complex problem with stochastic rewards (scenarios 2-4) as illustrated in Table 1. When the bee is tested on scenario 1, a lifetime is composed of 100 flights (or trials), when the bee is exposed to all 4 scenarios, each scenarios is presented consecutively to the bee whose lifetime is therefore extended to 400 trials. At the 50th flight $\pm$ 15 of each scenario, the high and low rewarding flowers switch colours. Switching points between scenarios are also affected by a variability of $\pm$ 15 trials. At each scenario switch, the high and low rewarding flower switch colour with probability 0.5. See [13] for further implementation details.

Figure 3 shows the input/output signals used by the neural controllers. The bee is simulated in a continuos 3D space, whereas the agent in the T-maze is in discrete space.

In conclusion, four experiments are carried out: two experiments with the agent in the T-maze, without and with homing behaviour, and two experiments with the foraging bee with deterministic and stochastic reward conditions.

3. Evolutionary Method

A basic Evolution Strategy [2] was employed here to evolve topologies, weights and plasticity rules of neural networks. A matrix of real-valued weights encoded the network weights w_{ij}. The parameters for the plasticity rule A, B, C and η were separately encoded and evolved in the range [-1,1] for A-C, and [-100,100] for η. Genes in the range [-1,1] are mapped into phenotypical values with a cubic function: this produces a bias towards small values initially. Phenotytical weights were in the range [-10,10]. Insertion, duplication and deletion of neurons were applied with probability 0.01, 0.01 and 0.02 respectively.

A Gaussian mutation with standard deviation 0.02 was applied to all genes, and an additional Gaussian mutation (with a larger standard deviation of 0.2) was applied with a small probability of 0.02. One point crossover on the weight matrix was applied with probability 0.1. A spatial tournament selection mechanism was implemented by dividing the array of the population in adjacent segments of size 5 (with random offset at each generation). A population of a 150 individuals was employed with 2000 generations as termination criterion. To foster the synthesis of minimal neural architectures, after generation 1000, the algorithm continued the evolutionary process with no insertion and duplication of neurons, but maintaining deletion.

4. Results

One set of experiments for each learning rule of Equations 2-8 and each problem was executed. To provide statistically significant data, each set included 30 independent evolutionary runs.

Figure 4(top) shows the median fitness progress over the 30 independent runs for the controllers in the T-maze without homing. Four rules out of 7 (C, AC, BC and ABC) allowed to solve the problem maximising the performance in the majority of runs. Rules A, B and AB alone did not allow the solution of the problem. Figure 4(bottom) shows the fitness progress in the T-maze with homing. In this case, the problem is more difficult because the agent needs to remember the way back home after collecting the reward, and failure to do so result in a penalty of 0.3. However, even in this problem, three rules AC, BC and ABC allowed to solve the problem. One rule (C) reached good performance with some difficulty, while rules A, B, and AB failed as in the previous problem.

Figure 5(top) shows the median of fitness values over the 30 independent runs for the bee controllers in scenario 1. Three rules (B,C and BC) failed to solve the problem, two rules (A and AB) achieved good performance. ABC and AC gave the best performance. Figure 5(bottom) shows the fitness progress when the bee performs continuously over

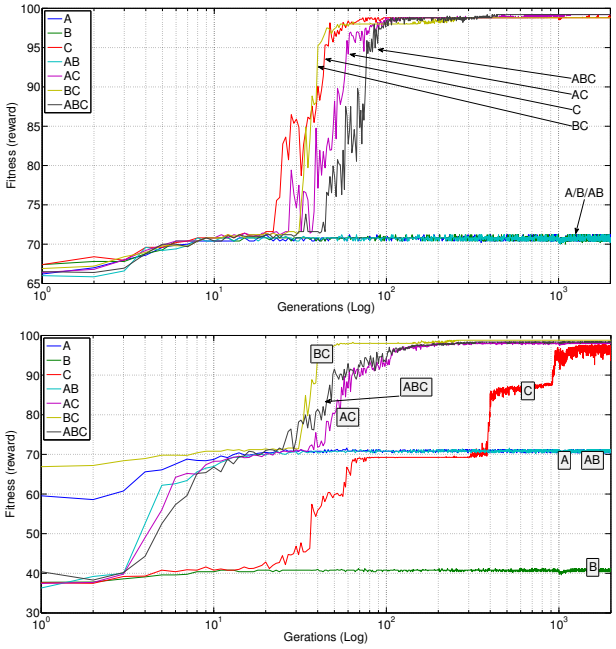

Figure 4. Fitness for each plastic rule with the agent in the maze (top graph) and maze with homing (bottom graph).

the all 4 scenarios. In this case, only the rule ABC appeared to maximise the performance.

5 Analysis

Generally, although different rules performed differently according to the problem, optimal solutions were discovered in the majority of runs. Not surprisingly, the general rule ABC allowed good performances, but interestingly the graphs show that other simpler rules (Equations 2-7) also allowed to solve some of the problems. The bee problems appear to benefit particularly from the correlation Hebbian term (A). This can be explained with the fact that the action of choosing a flower colour (i.e. keep the output signal high to maintain the flying direction) is continuous during the flight and landing. Upon landing the input for the flower colour can be correlated to the reward intake. The T-maze problems instead seem to benefit mainly from the postsynaptic rule C, but not from the correlation term A. This could be explained by the fact that the reward intake in the maze occurs with a certain delay from the action of choosing the maze-end. After the turning point, the agent needs to navigate straight and lowers the output to values less than 1/3 before reaching the maze-end and collecting the reward: as a consequence the action taken for turning cannot be directly correlated at the reward intake.

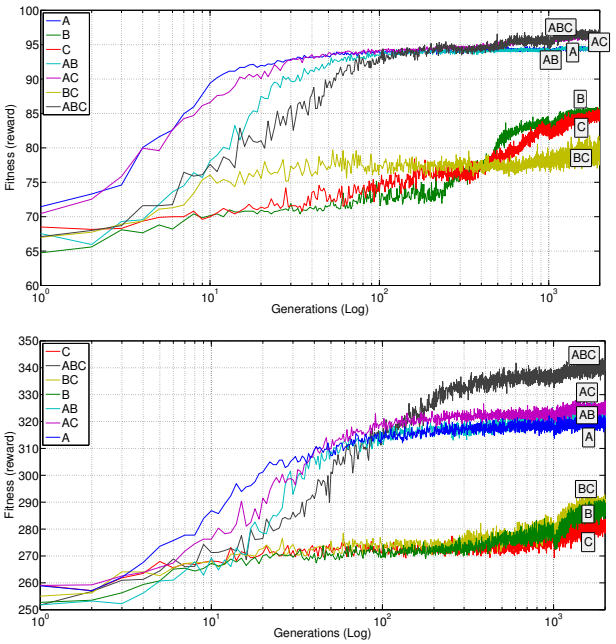

Figure 5. Fitness for each plastic rule with the foraging bee experiment in scenario 1 (top graph) and in scenarios 1-4 (bottom graph).

Although different problems seem to benefit differently from the proposed rules, a number of plasticity rules resulted in the evolution of successful controllers. This fact suggests that these kinds of reward-based learning problem do not necessitate more complex learning rules as it was suggested in previous studies [9, 10]. The hand designed neural architecture proposed in [10] employed the four-parameter rule of equation 1 with the addition of neuromodulatory plasticity, and solved scenario 1 and 2; on the other hand, the solutions that were discovered here achieve optimal performances in all 4 deterministic and stochastic scenarios with less complex rules and without neuromodulatory dynamics.

A possible explanation for that is that allowing the evolutionary search to exploit minimal rules and topologies resulted in the discovery of better solutions than the handcrafted modulatory architecture in [10]. Although neuromodulation was also employed on the bee foraging problem in [13] showing optimal performances, the results showed here indicate that neuromodulation is not an essential requirement. These results confirm the data shown in a recent study [12] where neuromodulation is shown not to bring an advantage in the single T-maze. The simulations show that this set of rules can implement operant reward learning without the use of a reinforcement learning algorithm [14].

		Nr of neurons		Nr of connections	
Problem	Rule	Mean	Std	Mean	Std
1)	C	1.10	0.30	3.36	0.76
	AC	1.03	0.18	1.70	0.91
	BC	1.07	0.25	2.26	0.74
	ABC	1.03	0.18	2.6	0.85
2)	C	1.83	0.64	5.46	2.36
	AC	1.36	0.49	2.83	0.98
	BC	1.07	0.25	2.23	0.78
	ABC	1.36	0.49	2.83	1.08
3)	ABC	1.2	0.61	6.37	5.68
4)	ABC	1	0	4.63	1.0

Table 2. Mean and standard deviation of number of neurons and connections in evolved networks that solved the proposed problems.

5.1 Neural Architectures

The topologies of networks that solved the problems were analysed to discover common features and minimal structures. The networks in the population after the first 1000 generations displayed a wide variety of topologies and varying number of neurons. However, the further 1000 generations without neuron insertion and duplication resulted in a considerable reduction of the number of neurons without decrement in performance as confirmed by the fitness graphs of Figures 4 and 5.

Surprisingly, the inspection of neural controllers revealed that all four problems could be solved with remarkably small neural networks of one output neuron and no hidden neurons. Table 2 shows the mean and average number of neurons and connections of the best networks from the 30 runs in each problem. Interestingly, a further analysis revealed that the number of neurons has a level of correlation with the specific rules being available during evolution. For example, in the T-maze with homing, rule C developed well performing networks with one inner neuron plus the output, contrary to other rules which required no inner neurons. In the bee foraging problem, rule B resulted in good performances only when deploying 2, 3 or even 4 inner neurons. The ABC rule which evolved the networks with best performances did not require inner neurons.

Figures 6 and 7 provide examples of minimal architectures for learning networks in the T-maze with homing navigation and in the 4-scenario foraging bee problem. As indicated in Table 2, these surprisingly simple structures emerge constantly from evolutionary runs and solve the problems with optimal performance. Therefore, although neural networks are difficult to hand design, the small architectures that evolved in these experiments suggest that essential reward-based learning based on few sensory-motors

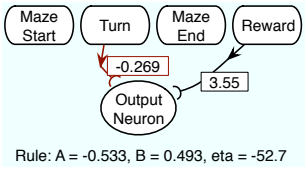

Figure 6. Example of a network that controls the agent in the T-maze. This network is capable of identifying the higher rewarding maze-end and adapt its preference when its location changes. Although the inputs "maze start" and "maze end" are available to the network, the algorithm performed feature selection by evolving 0-valued weights.

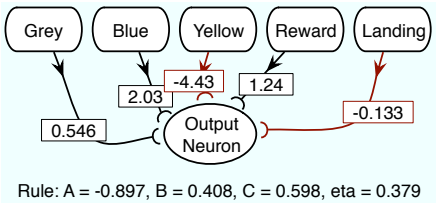

Figure 7. Example of a network that controls the bee. This network is capable of identifying the higher rewarding flower and adapt its preference according to the reward given in 4 different deterministic and stochastic scenarios.

signals can be implemented in very compact structures and can be evolved with basic search algorithms.

6. Conclusion

This work indicates that basic types of reward-based learning problem in dynamic scenarios can be solved with remarkably small neural architectures and simple plasticity rules. Although more complex problems involving a longer sequence of actions (like a double T-maze) might require additional heterosynaptic dynamics, basic operant conditioning as in the single T-mazes and in the simulated bee foraging problems can be achieved with basic rules and architectures. Consequently, although more analysis is required to uncover the precise neural and plasticity dynamics, the learning in the dynamic, reward-based scenarios simulated here does not appear to require neuromodulated plasticity or large neural topologies.

The methodology of testing different rules on freely evolvable neural architectures while operating in the required environment appeared to provide surprisingly simple solutions to apparently complex problems. The validation of learning rules and architectures finalised to learning was implicitly guaranteed by the coupled simulation of networks and uncertain environments. The methodology offers a valid tool to outline relations between a variety of learning problems and minimal plasticity rules and topologies to solve them.

Acknowledgement

The author would like to thank John Bullinaria and Peter Dürr for their comments on the draft of this paper.

References

[1] W. H. Alexander and O. Sporns. An Embodied Model of Learning, Plasticity, and Reward. *Adaptive Behavior*, 10:143, 2002.

[2] T. Bäck, D. B. Fogel, and Z. Michalevicz, editors. *Handbook of Evolutionary Computation*. Oxford University Press, Oxford, 1997.

[3] P. Dayan and L. F. Abbott. *Theoretical Neuroscience*. MIT Press Cambridge, MA, USA, 2001.

[4] J.-M. Fellous and C. Linster. Computational models of neuromodulation. *Neural Computation*, 10:771–805, 1998.

[5] D. Floreano, P. Dürr, and C. Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, 2008.

[6] D. Floreano and J. Urzelai. Evolution of plastic control networks. *Auton. Robots*, 11(3):311–317, 2001.

[7] W. Gerstner and M. W. Kistler. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, Cambridge, UK, August 2002.

[8] S. J. Martin, P. D. Grimwood, and R. G. M. Morris. Synaptic Plasticity and Memory: An Evaluation of the Hypothesis. *Annual Review of Neuroscience*, 23:649–711, 2000.

[9] P. R. Montague, P. Dayan, C. Person, and T. J. Sejnowski. Bee foraging in uncertain environments using predictive hebbian learning. *Nature*, 377:725–728, October 1995.

[10] Y. Niv, D. Joel, I. Meilijson, and E. Ruppin. Evolution of Reinforcement Learning in Uncertain Environments: A Simple Explanation for Complex Foraging Behaviours. *Adaptive Behaviour*, 10(1):5–24, 2002.

[11] E. Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273, November 1982.

[12] A. Soltoggio, J. A. Bullinaria, C. Mattiussi, P. Dürr, and D. Floreano. Evolutionary Advantages of Neuromodulated Plasticity in Dynamic, Reward-based Scenarios. In *Proceedings of the Artificial Life XI Conference 2008 (to appear)*, 2008.

[13] A. Soltoggio, P. Dürr, C. Mattiussi, and D. Floreano. Evolving Neuromodulatory Topologies for Reinforcement Learning-like Problems. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2007*, 2007.

[14] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 1998.