Associative Learning on a Continuum in Evolved Dynamical Neural Networks

Eduardo Izquierdo, Inman Harvey
Dept. of Informatics
Centre for Computational Neuroscience and Robotics
University of Sussex
Brighton, BN1 9QG, UK

Randall D. Beer
Cognitive Science Program
Dept. of Computer Science
Dept. of Informatics
Indiana University
Bloomington, IN 47406, USA

This is a preprint. Article to appear in Adaptive Behavior (2008).

This paper extends previous work on evolving learning without synaptic plasticity from discrete tasks to continuous tasks. Continuous-time recurrent neural networks without synaptic plasticity are artificially evolved on an associative learning task. The task consists in associating paired stimuli: temperature and food. The temperature to be associated can be either drawn from a discrete set or allowed to range over a continuum of values. We address two questions: can the learning without synaptic plasticity approach be extended to continuous tasks? And if so, how does learning without synaptic plasticity work in the evolved circuits? Analysis of the most successful circuits to learn discrete stimuli reveal finite state machine (FSM) like internal dynamics. However, when the task is modified to require learning stimuli on the full continuum range, it is not possible to extract a FSM from the internal dynamics. In this case, a *continuous state machine* is extracted instead.

Keywords: associative learning, evolutionary robotics, dynamical systems theory, synaptic plasticity, evolution of learning.

1 Introduction

Learning is one of the most fundamental aspects of adaptive behavior for living organisms. Although there is no one agreed definition, it generally refers to changes in the behavior of the organism, such that its performance on some task improves with experience.

Almost all studies of the mechanisms underlying learning and memory have focused on the activity dependence of synaptic efficacy (see, for example, Kandel, 2000). In fact, synaptic plasticity is conventionally thought to be both necessary and sufficient to account for learning and memory. This is reflected in most models of learning (Abbott & Nelson, 2000). Ultimately, this has helped cement a perspective on learning where the behavior-producing and the learning mechanisms are neatly separated into neural activity and its synaptic modulation, respectively.

This traditional perspective has become less useful in the face of more recent theoretical and experimental work in neuroscience. First, it is now well accepted that neuronal activity itself modifies not only synaptic efficacy but also the intrinsic membrane properties of neurons, and changes in these properties also serve to modify the circuit's dynamics (Marder et al., 1996). This includes long-term potentiation of intrinsic excitability (Cudmore & Turrigiano, 2004). Second, studies have uncovered bewildering diversity in the timescales of activation and inactivation of neurons (Llinas, 1988), in postsynaptic signaling properties (Toledo-Rodriguez et al., 2005), as well as dendritic signaling (Hausser et al., 2000). Finally, long time delays before potentiation or depression reach stable levels have been discovered. This means that such slow processes cannot quantitatively account for fast memory formation. Therefore, other cellular, synaptic, or network mechanisms must fill in the gap between spike timing events occurring over very fast time scales (Bi & Rubin, 2005). As a result, the understanding of the mechanisms underlying learning is changing towards a dynamical process occurring over multiple timescales.

In modeling work, learning behavior and plasticity has also been traditionally associated with the modification of a neural network's parameters, especially involving changes in the synaptic connections or weights of the neural network during the lifetime of the individual (Churchland & Sejnowski, 1992). These assumptions have been carried into the studies of learning in Evolutionary Robotics (ER) (Nolfi & Floreano, 1999; Floreano & Urzelai, 2000, 2001, for example). However, one of the main strengths of this approach is to minimize building in preconceptions relevant to the phenomena of interest; allowing artificial evolution to 'figure it out' on its own instead. The concept of plasticity is basically no more or less than change on different timescales. Whereas in conventional feedforward neural networks it is traditionally fast changes of node-activations, and slow changes of synaptic-plasticity, there is no need for the differential timescales to be split up in that particular way. In the work presented here the plasticity is non-synaptic: in dynamical recurrent neural networks, activity occurring over multiple timescales is practically inevitable.

A more integrated view of learning, as a dynamical process occurring over multiple timescales, was first made concrete in a set of ER experiments (Yamauchi & Beer, 1994a,b). Agents were evolved in tasks where learning the relationship between a landmark and the goal improved the agent's performance in finding that goal. Agents were also evolved to learn sequential decision-making. Continuous-time recurrent neural networks (CTRNNs) without synaptic plasticity were successfully evolved. Similar experiments have been performed by others that have included actual robots and comparisons between controllers with and without synaptic plasticity (Tuci et al., 2002; Blynel & Floreano, 2003). More recently, an in-depth analysis of the internal dynamics of evolved CTRNNs without synaptic plasticity in an associative learning task for discrete stimuli is given (Phattanasri et al., 2007).

All of this work has focused on evolving agents that can behave differently in a discrete number of different environments, in practice two. The learning, in this case, corresponds to swapping between two different modes of interaction; depending on which environment the agent finds itself in (e.g. going towards the landmark in landmark-near environments or going away from it in landmark-far ones). A different, but arguably more common, form of learning requires the ability to adapt to changes in the environment ranging over a continuum of values. Temperature preferences in the *C. elegans*, song learning and parental imprinting in birds, and face recognition in humans, are a few examples of such learning of continuous stimuli. Izquierdo & Harvey (2006) report the first experiments on learning from a continuous range using circuits without synaptic plasticity. In that work, the analysis of evolved circuits illustrated how the rich environmental regularities arising from the agent's situatedness provided a way to 'offload' the plasticity to the agent-environment interaction.

The central contribution of this paper is the extension of previous work on evolving learning without synaptic plasticity from discrete tasks to continuous tasks, where the plasticity arises from the neural network controller. As such, there are two major questions it will address. First, *can* this approach be extended to continuous tasks? Second, if so, *how* does learning without synaptic plasticity work in the evolved circuits?

The particular associative learning task, neural models, and evolutionary algorithm that we employ are described in Section 2. Section 3 presents a 'baseline' for the continuous work by reproducing the results in Phattanasri et al. (2007) with a different training paradigm and evolutionary technique, but similar task and shaping protocol, and the same neural network model. This section shows that a finite state machine can be extracted from the functioning of the circuit's internal dynamics. In Section 4, we demonstrate that CTRNNs lacking synaptic plasticity can be successfully evolved to exhibit associative learning for a continuous version of the task. The dynamical operation of the best such circuit is then analyzed in detail in Section 5. Finally, Section 6 concludes with a discussion of the broader implications of our results and directions for future work.

2 Methods

We use evolutionary techniques to synthesize dynamical system circuits for an associative learning task abstracted from one of several learning behaviors studied in the nematode worm *Caenorhabditis elegans*. The behavior is known as temperature preference[1] (Hedgecock & Russell, 1975). It consists in associating paired stimuli: temperature and food. This paradigm was chosen to be the simplest possible scenario requiring associative learning, yet sufficiently sophisticated to allow for the 'remembered stimulus' to be either discrete or continuous.

2.1 Temperature preference task

As we are interested in the broader set of possible mechanisms that can give rise to such behavior, we do not want to explicitly specify the task in terms of the internal mechanisms. This would carry built-in assumptions about the mechanisms and architecture required to do so. We would like to define the task at a 'higher' behavioral level, instead.

C. elegans can sense temperature at the tip of their head, and (although less directly) they can also sense food (primarily bacteria)[2]. In the thermal preference behavioral paradigm, animals placed onto a thermal gradient will migrate to the temperature that they had been previously cultivated at. The behavior has been described as one of the most complex in the *C. elegans* repertoire (Hobert, 2003). They will only 'remember' their cultivation temperature if that temperature was paired with the presence of ample food supply. In contrast, if the previous cultivation temperature

[1] Although the behavior has been observed by several others (Mori & Ohshima, 1995, 1997; Mori, 1999; Ryu & Samuel, 2002; Zariwala et al., 2003; Mohri et al., 2005; Murakami et al., 2005; Biron et al., 2006; Luo et al., 2006; Clark et al., 2006) since it was first discovered by Hedgecock & Russell (1975), a more recent study has challenged the validity of the phenomena due to possible effects of body temperature on movement (Anderson et al., 2007). As a result, temperature preference is currently a debated topic of investigation in *C. elegans*.

[2] The main thermosensory neuron is called AFD, another neuron called ASH is known to be involved in sensing food. It is important to note that the sensory function of neurons is often determined using behavioral experiments on worms where the neuron has been laser-ablated. See De Bono & Maricq (2005) for more detail.

was paired with starvation (an aversive stimulus), animals will avoid that temperature. They are also known to learn new preferred temperatures: if starved in their current temperature they will move until they find bacteria and remember that temperature for the future (Mohri et al., 2005). In summary, the nematode worm modifies its behavior in relation to previous experience based on two distinct, yet paired, sensory inputs. The behavior involves memory formation as well as acute sensory input comparison to a reference value. Although ablation studies have been performed to identify the circuit of neurons responsible for the behavior (Mori & Ohshima, 1995), the underlying mechanisms are still poorly understood.

In our abstracted model, the task involves pairing the agent's food with a particular temperature (chosen at random). After some time has passed, the agent is presented with a temperature stimulus for testing. This can be similar or different from the originally paired signal. A successful agent is required to open its 'mouth' when it is the *same* stimulus and close it otherwise. The agent is required to identify the *pairing signal* for several consecutive tests separated by random delays. Also, the agent can be paired with a different temperature signal from the original at any point during its lifetime. Successful agents are required to re-learn new associations and identify subsequent test signals based on their last pairing.

2.2 Agent and structure of a trial

An agent is modeled as a circuit with two sensors and one output (see Figure 1). The 'thermal' sensor (*T*) provides the local temperature in the agent's environment. The 'food/reward' sensor (*F/R*) provides a binary signal corresponding to the presence or absence of food. This sensor can also act as a positive or negative reward signal, which can be loosely interpreted as coming from a gut sensor that signals the consequences of the agent's previous action. The only action available to the agent is to open or close its 'mouth' via a continuous effector output (*M*). Both temperature and food signal inputs can be perceived by any of the nodes in the network via a set of connections.

A single trial is structured into phases (see Figure 2). There are two kinds of phases the circuit can be exposed to: 'pairing' and 'testing'. During pairing, first the circuit is exposed to food and a particular temperature simultaneously. This lasts a fixed 20 units of time. Next, both the food and the temperature signals are removed (signals return to 0). The duration of the delay is random, lasting anywhere between 16 and 24 units of time. During testing, the circuit is first exposed to a particular temperature that may or may not be the same as the one applied during the pairing phase, but in the absence of the food signal. This lasts 10 units of time. The temperature signal is then removed and the circuit evaluated for another 10 units of time by observing the state of the 'mouth'. Ideally, it should be 'open' if the recent and original pairing temperature are the same, and 'closed' if they are not. Then a delay of random duration is applied, lasting between 8 and 12 units of time. After this, the agent receives either a positive or a negative reward. This is based on the correctness of its previous action (mouth open or closed) in relation to the recently experienced temperature and its environment, as determined by the original temperature. The testing phase ends with another delay of random duration in the same range as the previous one.

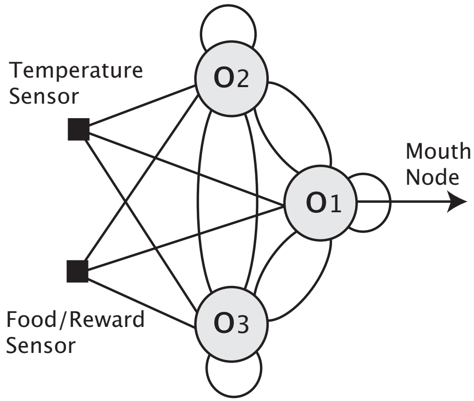

Figure 1. Circuit architecture. Example three-node fully-connected network, including self-connections. The 'thermal' and 'food/reward' sensors are also fully connected with all nodes. The strengths of all connections are evolved, but stay fixed within the lifetime of the agent. One node is specified as the 'mouth' neuron.

The training paradigm is given by the structure of a trial. For any single trial, there must always be at least one pairing and one testing phase, as illustrated in Figure 2. However, trials can be comprised of more than one pairing and testing phases. The pairing phase indicates the 'environment' the agent is in. The particular temperature where food is found defines an environment. In different environments food can be found in different temperatures. Thus, multiple pairing phases represent changes of environment. A circuit can also be subject to several subsequent testing phases within a particular environment. This requires the circuit to remember the paired temperature for longer. Thus, multiple testing phases represent subsequent evaluations of the agent's performance within an environment. For brevity, a length-*LK* trial corresponds to one with L different environments (i.e. $L-1$ changes of environment) and up to K testing phases per environment. During a trial, the state of the circuit is never altered. Only at the start of a trial is the state of the circuit 'reset' (more detail in the next section).

The stimuli to be remembered can be drawn from a discrete set or from a range on a continuum. Consider a situation where only n types of environment exist for a particular agent. For the simplest $n = 2$ case, this would correspond to a 'world' where food could be found in a particularly 'cold' temperature, t_1, in some environments, and in a particularly 'hot' temperature, t_2, in the second environment. A successful agent would have to find out which of the two environments it finds itself in, and act accordingly. This is the discrete case. For the continuous case, the number of possible environments is, in principle, infinite. Pairing temperatures are chosen at random uniformly from the range $[1,2]$. Test temperatures are chosen to be equal or different from

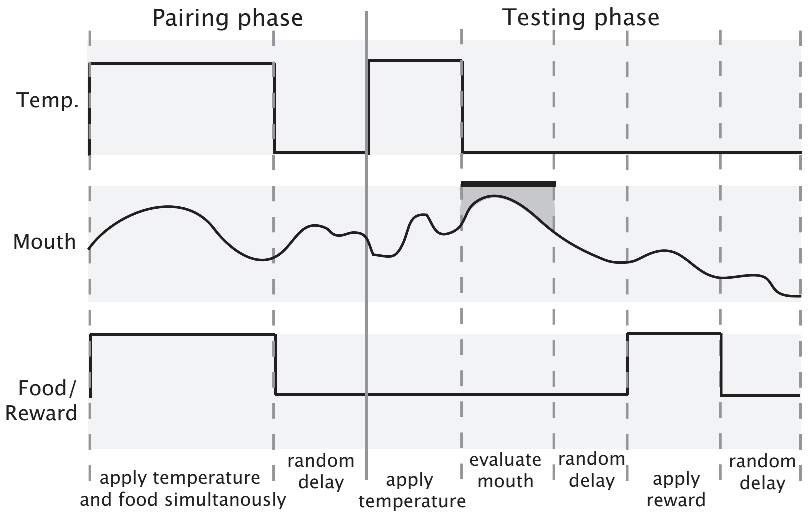

Figure 2. Structure of an individual trial. A trial can consist of a combination of the following two phases: pairing and testing (vertical gray line). Each of these phases is composed of distinct events (vertical dashed lines). During pairing, food and temperature signals are applied simultaneously. This is followed by a variable random delay. During testing, first a temperature signal is applied. Second, the state of the mouth is evaluated relative to the correct action (bold line). The error between the agent's action and the desired response is shaded in dark gray. Third, there is a variable random delay. Fourth, a reward signal is applied based on whether the previous action was correct or not. Finally, there is another variable random delay before the next trial begins. Multiple pairing and testing phases can occur during a single trial. The state of the network is never reset within a trial, only between different trials.

the last paired temperature with 50% chance. Test temperatures, when different, are also chosen at random uniformly from the full range. A successful agent must remember the particular temperature where food is found in each particular environment. In practice, there will be some differences that may be too small to detect given some precision. For this reason, although pairing and testing temperatures are drawn at random, the performance on test signals that are different from the pairing temperature by less than 0.1 do not count towards fitness.

The aim is to study the difference in the evolved internal dynamics of agents evolved on stimuli drawn from a discrete set ($n = 2$) versus a continuum.

2.3 Dynamical neural network

We use continuous-time recurrent neural networks (CTRNNs) as a model of the agent's internal dynamics. These are continuous-time nonlinear dynamical systems that can, in principle, approximate any dynamics with an arbitrary precision, given enough components (Funahashi & Nakamura, 1993). For this reason, our model does not include any form of explicit *synaptic* plasticity mechanisms (such as weight-changing or any other parameter-changing rules). Each component in the network is governed by the following state equation (Beer, 1995):

$$\tau_i \dot{y}_i = -y_i + \sum_{j=i}^{N} w_{ji}\sigma\left(y_j + \theta_j\right) + tw_i T(t) + fw_i F(t) \tag{1}$$

where y is the activation of each node; τ is its time constant; w_{ji} is the strength of the connection from the j^{th} to the i^{th} node; θ is a bias term; $\sigma(x) = 1/(1 + e^{-x})$ is the standard logistic activation function; and N represents the number of nodes in the network. All nodes can sense the external environment via a set of extra connection weights: tw_i is the weight of the connection from the 'thermal sensor', $T(t)$, to node i; fw_i is the weight of the connection from the 'food sensor', $F(t)$, to node i. The activation of all nodes is 'reset' to 0 at the beginning of each trial. Remember that each trial comprises a sequence of pairing and testing phases. In simulation, node activations are calculated forward through time by straightforward time-slicing using Euler integration with a time-step of 0.1. The network is fully connected.

2.4 Evolutionary algorithm

The parameters of each circuit (i.e. biases, time-constants, inter-node and sensor-node weights for each node) are evolved using a version of the Microbial genetic algorithm (Harvey, 2001). There are $N^2 + 4N$ parameters in total. These are encoded in a genotype as a vector of real numbers over the range [0, 1]. Offspring of microbial tournaments are generated as a mutation of the winner of the tournament (i.e. no recombination). The mutation is implemented as a random displacement on every gene drawn uniformly from a Gaussian distribution with mean 0 and variance 0.01. Each gene is forced to be in [0, 1]: when a mutation takes a gene out of this range it is reflected back. The offspring replace the loser of the tournament. Genes are mapped to network parameters linearly between [-10, 10] for biases and inter-node and sensory weights. Time constants are mapped exponentially between [e^0, e^5]. The size of the population used is 50. We define a generation as the time it takes to generate 50 new individuals. The evolutionary algorithm uses a 'geographical' method to allow different subpopulations to evolve semi-independently in 'demes' within the population. A minimal 1D wrap-around geography with demes of size 10 was used: such that only individuals 10, or less than 10, positions away from each other could compete in tournaments (Spector & Klein, 2005, for details). In principle, this allows for genotypic diversity to be maintained for longer in the population at almost no extra computational cost. Finally, because the fitness is noisy (described below), agents are reevaluated every time they participate in a tournament.

2.5 Fitness evaluation

A successful circuit must maximize its consumption of edible food and minimize the consumption of inedible food, regardless of the environment it is in. Each individual is evaluated over a set of R length-LK trial sequences. The fitness of a circuit is given by one minus the average consumption error over each testing phase:

$$F = 1 - \frac{\sum_{r=1}^{R}\sum_{l=1}^{L}\sum_{k=1}^{K} E_{rlk}}{RLK} \qquad (2)$$

where E_{rlk} stands for the error of an agent's action on the k^{th} test, as part of the l^{th} environment, and on the r^{th} trial repetition. The error is depicted in Figure 2 as the dark gray region and it is calculated by integrating the difference between the desired output and the agent's action during the evaluation phase:

$$E_{rlk} = \int_{T_{rlk}+10}^{T_{rlk}+20} |A_{rlk} - M(t)|\ \psi(T_{rlk} + 10, t)\ dt \qquad (3)$$

where T_{rlk} is the time the k^{th} trial begins as part of the l^{th} environment and r^{th} trial; A_{rlk} is the correct motor output given the temperature applied during the k^{th} testing phase and the pairing temperature corresponding to the l^{th} environment type, during the r^{th} trial; $M(t)$ is the agent's actual motor output during the evaluation period; and $\psi(t_0, t) = exp((-t-t_0-5)^2/5.12)/4.0034$ is a Gaussian weighting function normalized so that E_{rlk} runs between 0 and 1. Gaussian weighting assigns maximum importance to the error of the agent's action at the center of the evaluation period, with the importance smoothly falling off at earlier and later times. This weighting function was implemented following Phattanasri et al. (2007). There are two components that make the fitness evaluation of a single trial noisy: (1) the random values of the pairing and testing temperatures and (2) the random duration of the delays.

2.6 Incremental evolution

An incremental shaping protocol was employed during evolution. The strategy is straightforward. Evolution starts with the most basic form of the task. As the population succeeds, the level of difficulty of the task increases. This not only saves important evolutionary computation time, but in some situations it can also increase the chances of evolutionary success (Phattanasri, 2002).

The shaping protocol includes four stages. During the first stage, one pairing and one testing phase are applied. This is the most basic form the task can take ($L = 1$, $K = 1$), and it requires learning one association and remembering it for at least one test. During the second stage, agents are still exposed to only one environment (one pairing phase), but the number of testing phases is increased to three ($L = 1$, $K = 3$). The change from the first to the second stage serves to increase the selection pressure on the agent's memory. During the third stage, a change of environment is introduced for the first time. A trial consists of a pairing phase, followed by three testing phases, followed by another pairing phase, and finally followed by another three testing phases ($L = 2$, $K = 3$). This change of stage introduces new selection pressure on the agent's ability to re-learn and remain plastic, while still retaining selection pressure on the agent's memory. The fourth and last stage consists of three environments, each with a variable number of testing phases ($L = 3$, $K = [0, 5]$). A change of environment can occur directly after another (i.e. two consecutive pairing phases). Trial sequences without a single testing phase do not count towards fitness. Finally, given that the fitness evaluation of a single trial is noisy, agents are assessed on 100 trials per fitness evaluation.

There are some differences in the shaping protocols for the discrete and continuous scenarios. Transitions between the first three stages of the shaping protocol are triggered whenever the fitness of the best agent in the population exceeds a certain threshold (95% for the discrete case and 90% for the continuum case). The last stage is applied only after a certain number of generations and only for the most successful populations. How many generations and which populations are regarded as successful was also different for the discrete and the continuous versions of the task. This was based on two observations from preliminary experiments. First, populations evolving on the continuous task required more time to achieve appropriate performance than the discrete case. Second, while it was possible for successful circuits to achieve near perfect fitness (99.9%) on the discrete task, the fitness of circuits evolved for the continuous version of the task never reached similar levels of performance. Thus, for the discrete version, the last stage was applied after 10000 generations only to those populations whose best fitness reached greater than 99% at the third stage of the shaping protocol. For the continuous version, the last stage was applied after 20000 generations and only to those populations whose best fitness reached greater than 91% at the third stage of the shaping protocol. Finally, the variance of the mutation was halved during this last stage, to allow fine-tuning of the evolved parameters.

3 Learning Discrete Stimuli

Our first set of experiments examines the ability of CTRNNs to solve the discrete version of the associative learning task.

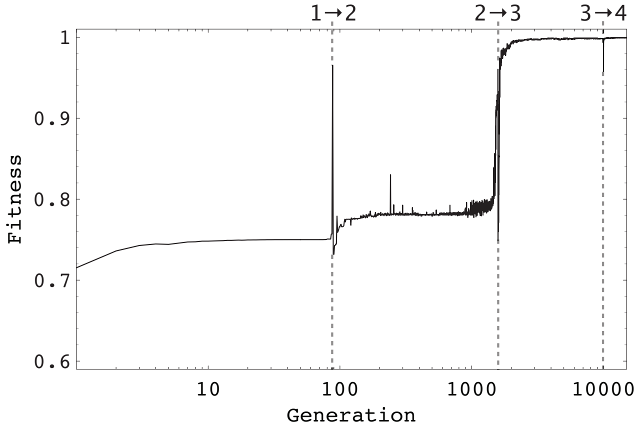

Figure 3. Plot of best fitness vs. generation for the best evolved 3-node circuit for the discrete learning task. The *x*-axis is plotted with a log scale, as the first transitions occur early during the evolutionary run. Transitions between stages of the incremental evolutionary technique are marked with dashed lines and labeled accordingly. First two transitions occur when the best fitness exceeds 0.95, the last transition occurs after 10^4 generations. The fitness drops sharply during the first two transitions, before the circuit can generalize to sequences of arbitrary length.

The main purpose of this section is to form a 'baseline' for the continuous work (Section 4) by reproducing the results in Phattanasri et al. (2007) with a different training paradigm and evolutionary technique, but similar task and shaping protocol, and the same neural network model. Another important role of this section is to show how the dynamical analysis leads to the finite state machine.

Evolutionary searches with 3- and 4-node circuits were performed. Successful agents were reliably evolved using such small circuits. For the most part, successful circuits had evolved to a fitness of 99% after the first 3000 generations. After 10^4 generations, we found 4/20 evolutionary runs using 3-node circuits and 12/20 using 4-node circuits that achieved fitness greater than 99% at the third stage of the shaping protocol. The most successful populations (those that reached fitness greater than 99% at the third stage of the shaping protocol) were further evolved on the last stage of the shaping protocol for an additional 5000 generations. Figure 3 shows the fitness versus generation plot for the best 3-node population. The evolutionary runs from successful populations are all relatively similar. Periods of fairly steady fitness values are punctuated by sudden jumps to regions of higher fitness. Also, the fitness tends to drop significantly following changes to the difficulty of the task from the shaping protocol (dashed lines). In the remainder of this section we describe the performance and internal dynamics of the best evolved three-node circuit in some detail (see Table 1 in the Appendix for the evolved parameters of this circuit).

The best 3-node circuit attained a fitness of 99.99%. To verify that this circuit had truly generalized to longer sequences, we tested it on 10^6 trials with 5 changes of environment and up to 10 tests per environment during its lifetime. Pairing and testing temperatures were chosen at random (from the two available). The time-step was made an order of magnitude smaller (0.01) to avoid possible integration errors. The circuit performed correctly on 98.36% of evaluations for this set, demonstrating that it does indeed represent a general solution to the discrete temperature preference task.

The behavior of this circuit on a typical sequence of trials is shown in Figure 4. As there are only two possible temperatures (1 and 2) that food can be associated with in this version of the task, we will refer to them as environments *A* and *B*, respectively. During the sequence shown, the environment-type switches from *A* to *B* and then back to *A* again at the points indicated by the dashed vertical lines. We can observe that the activity of nodes o_1 and o_3 are nearly inverted during environment *B*, in relation to *A*. For example, the 'mouth' node (*M*) is open (highly activated) throughout most of the duration of environment *A*, closing only during the presentation of the 'wrong' signal. Exactly the opposite is the case throughout environment *B*. The same node is now mostly closed, opening only during the presentation of the wrong signal in this environment. For both environments, the presentation of the different signal serves to change the current state of the 'mouth', while the reward serves to replace the former state. Similarly, all other nodes are, in part, keeping track of the environment the agent finds itself in. Also important to note is that the timescale of activity of the second node (o_2) is relatively slower than that of nodes o_1 and o_3.

How does this circuit work? In order to visualize the overall structure of this circuit's operation we apply a similar technique as that used in Phattanasri et al. (2007) by 'strobing' the state of the system at selected times during a trial. In particular, we observe the state of the system at the end of the pairing signal, the resting time, the testing signal and the reward signal. Given that the evolved system has a three dimensional state space we can visualize the entire space directly. The 'strobes' fall into relatively distinct clusters (Figure 5). Each of the clusters can be labeled according to the previous environmental interaction that the circuit had undergone. The labels *A* and *B* for the states represent one of the two temperatures the circuit has been initially paired with (i.e. the environment type). Clusters *A1* and *B1* denote the state of the system after a pairing. Clusters *A2* and *B2* represent the state of the system after a random delay. Cluster *B2* is also the default starting state of the system. Clusters *A3* and *B3* represent the state of the system after the presentation of a test temperature. The presentation can be of one of two temperatures; these are sub-labeled with a further *A* or *B*, accordingly. For a successful circuit this means that clusters *A3A* and *B3B* correspond to states where the mouth is open, while clusters *A3B* and *B3A* correspond to closed-mouth states. Finally, clusters *A4* and *B4* correspond to the state of the system after a positive reward.

We can consider the dynamics of the circuit when decoupled from the environment. For each of the different combinations of input we can determine the limit sets of the circuit. Furthermore, we can compare the relation of the circuit's asymptotic behavior with the clusters. Although a number of these clusters are centered on the equilibrium points in

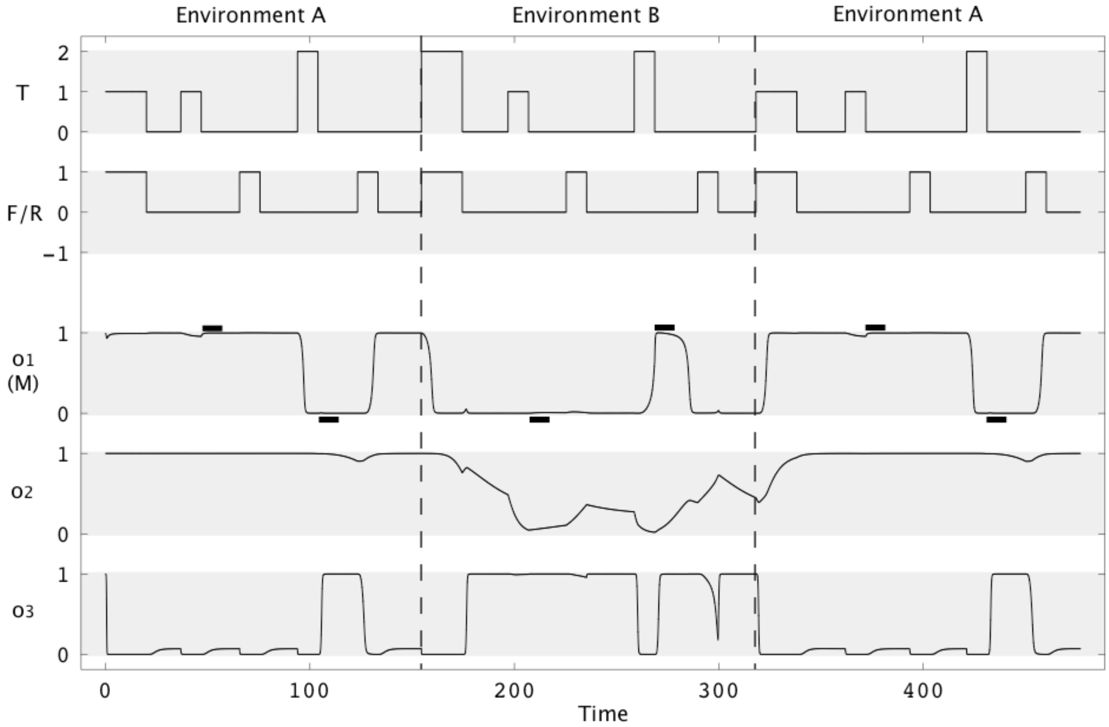

Figure 4. Activity of the best 3-node circuit on a typical trial sequence. From top to bottom the traces correspond to the temperature signal (*T*), the food/reward signal (*F/R*), the mouth state (*M*) and the outputs of the remaining nodes (o_2 and o_3). Small rectangles mark the time when the mouth state is evaluated and the state that the mouth should be during this time. Dashed lines mark transitions between environments.

the distinct phase-portraits of this system, many are not (data not shown). Also some of the clusters are not entirely contiguous in state space. This highlights the importance of the transients. Some of the non-contiguous clusters can be further subdivided according to the system's previous state. The labels in parenthesis denote the particular cluster from which the system departed to form those subclusters.

The strobed circuit dynamics from Figure 5 can be interpreted as implementing a finite state machine (FSM) with input. The strobed states correspond to the FSM states. Although not shown, transitions between strobed states correspond to input-driven transitions of the FSM. The FSM extracted from this circuit is shown in Figure 6. States *A1* through *A4* and *B1* through *B4* represent the states of the system described previously. There are 4 different types of transitions that can occur. First, the application of one of two possible temperatures $T = 1$ or $T = 2$, labeled ↓ and ↑, respectively. Second, the application of a positive or negative reward, labeled + or −, respectively. Third, a pairing that involves the application of a temperature and food simultaneously, denoted as ↑+ or ↓+ depending on the temperature. Finally, it is useful to treat the lack of stimuli as a transition that the circuit is exposed to, because it directs the state of the system to a relevant state in the machine. Thus, the last transition that can occur is a delay, corresponding to the absence of stimuli and denoted by o. Transitions are shown as arrows with labels in Figure 6. The start state is shown by an arrow pointing at it from s. Although this is not a situation encountered during evolution, at the start of a trial, if no pairing is applied, the system will move towards state *B2*. Notice subclusters (e.g. *A4(A3A)* and *A4(A3B)*) are grouped together. A more detailed FSM could be provided, but the extra detail does not add to our understanding of the operation of the state machine, as all of the incoming and outgoing transitions remain the same.

In our temperature preference task, an agent discovers which of the two environments it is in when there is a pairing. This means that the circuit doesn't have to wait to get a negative reward to learn. In fact, a successful circuit will never have to experience a negative reward. This is the case for the best evolved 3-node circuit. For this reason, we can ignore the punishment transitions and states altogether from our analysis. We can, however, artificially induce a negative reward. This makes sense only for this two-environment task, where being maladapted in one environment means inevitably that the circuit is in the only other possible environment. Interestingly, application of a negative reward while in environment *A* does switch the circuit's behavior to what it would be if it were in the opposite environment, however this switch does not occur the other way around. Thus, negative reward drives the state of the system to *B2*'s basin of attraction. This is linked to its role as the default initial state.

Before we move on to the continuum task, one question that we can ask is how this circuit deals with stimuli in between the discrete. We can test the performance of this system for the full range of possible combinations of pairing and

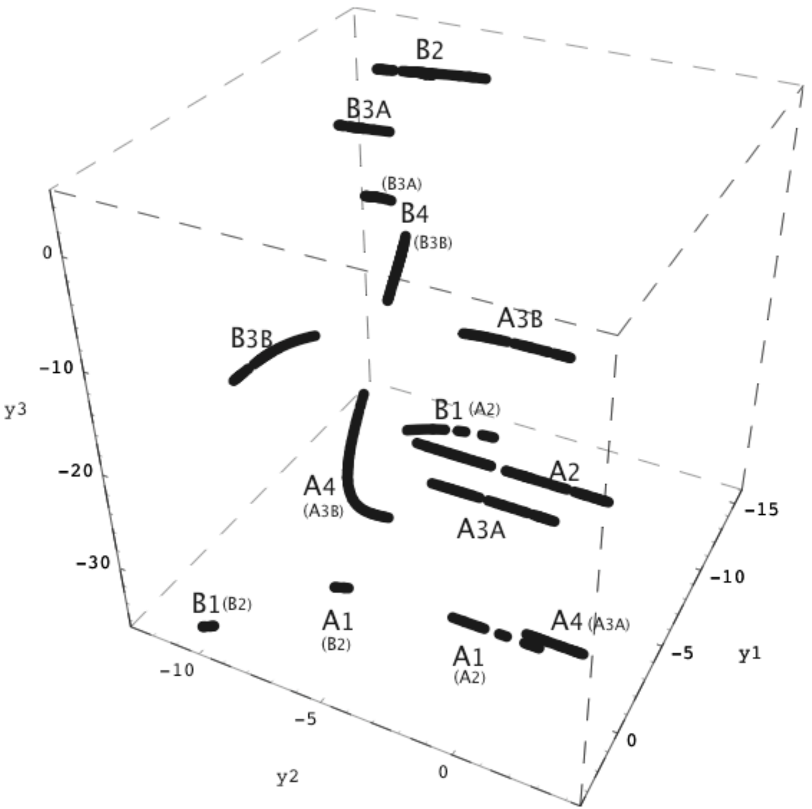

Figure 5. Strobed circuit dynamics in the best 3-node circuit at selected times during a trial. See the main text for the meaning of the labels.

testing temperatures. Figure 7 shows the generalization performance of the best evolved circuit across all combinations. The vertical axis corresponds to the pairing temperature signal. The horizontal axis corresponds to the test temperature signal. The shading represents how well the circuit performs: lighter shades correspond to better performance. What the circuit is expected to do changes depending on the pairing signal. For test temperatures equal to the pairing temperature the circuit is required to open its mouth. This corresponds to the line on the diagonal of the figure. For any test temperature that is different from the pairing temperature the circuit is required to close its mouth. This corresponds to all regions not directly on diagonal. For the discrete version of the task, the circuit is only evolved on the highest and lowest possible temperatures (denoted by circles in the figure). For this reason it is not expected to generalize to signals in-between. What we observe, instead, is an example of a binary categorization. When paired with 'cold' temperatures (below 1.6), the agent opens its mouth to all test temperatures below 1.3. When paired with 'hotter' temperatures (above 1.6), the agent opens its mouth to any test temperature above 1.3. Temperatures fall into one of two broad categories: 'cold' or 'hot'; with little or no generalization to temperatures in-between.

Finally, it is important to mention that other successfully evolved 3-node circuits displayed overall similar properties to the one analyzed here. Namely: (1) equivalent FSMs could be extracted from the evolved internal dynamics; (2) strobed states would not necessarily correspond to equilibrium points of the non-autonomous system with transients playing an equally important role; (3) categorization into two behavioral groups was observed without generalization within previously unseen environments; and (4) evolved time-constants would consistently fall into at least two relatively different time-scales: fastest possible (near 1.0) and slower by an order of magnitude.

4 Learning Continuous Stimuli

Can the same 'learning as dynamics' approach used in our previous section be extended to the continuous task? Our second set of experiments examine the ability of CTRNNs to solve the same associative learning task when the stimuli can be anywhere along a continuum. The main motivation for these experiments is to evaluate the similarities and differences between agents evolved for this task and the previous version.

4.1 Evolutionary Performance

Evolutionary searches were performed using 3- to 6-node circuits. We carried out 20 evolutionary runs with different seeds per group. After 2x10^4 generations, we found that none of the 20 evolutionary runs using 3- and 4-node circuits reached the third stage with a fitness greater than

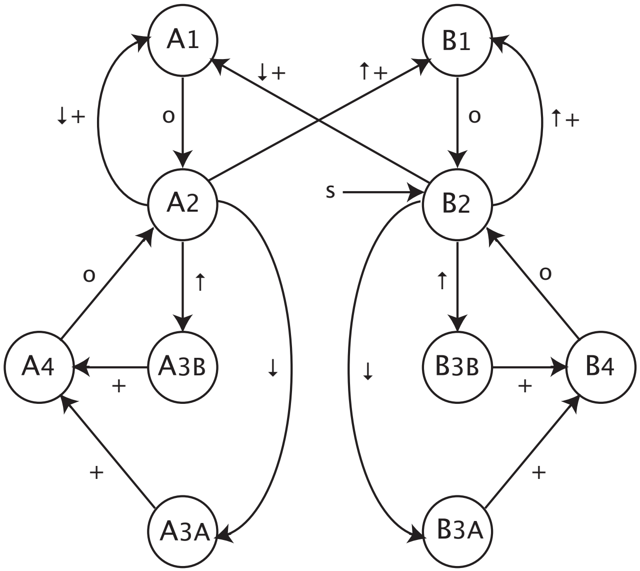

Figure 6. Extracted finite state machine from best 3-node circuit. Each state corresponds to a cluster of strobe states from Figure 5. Transitions between states are induced by the application of some combination of food, temperature or delays.

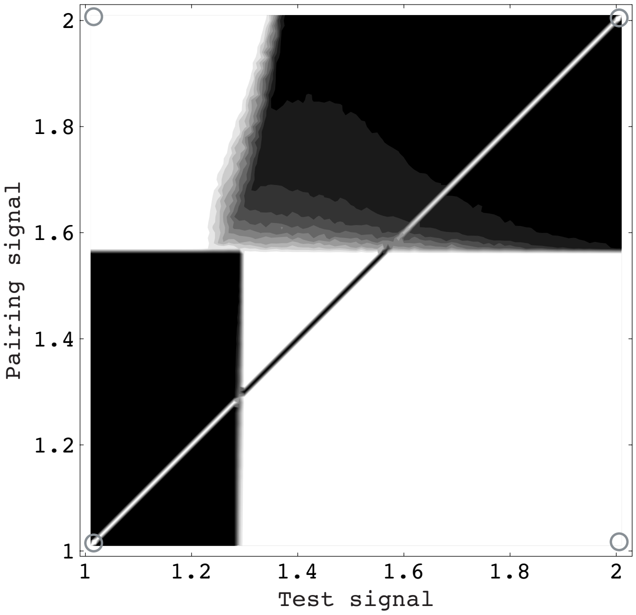

Figure 7. Generalization performance for the best-evolved 3-node circuit on the first testing phase after a pairing. The circles represent the situations this circuit was exposed to during evolution on the discrete (n = 2) version of the temperature preference task.

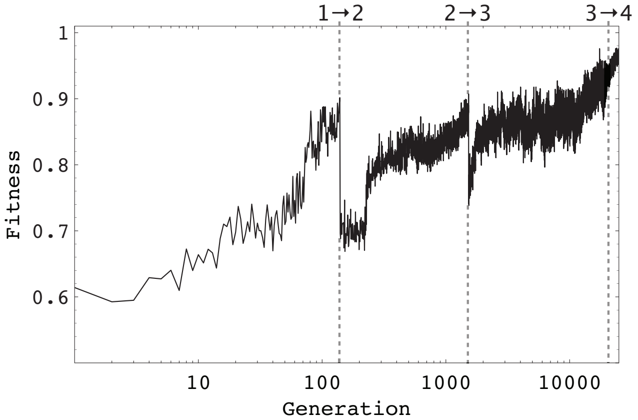

Figure 8. Plot of best fitness vs. generation for the best evolved 5-node circuit for the continuous learning task. Labeling conventions are the same as in Figure 3. The x-axis is plotted with a log scale, as the first transitions occur early during the evolutionary run. The first two transitions occur when the best fitness exceeds 0.90, and the last transition occurs after 2x10^4 generations. Similar to the discrete case, the fitness drops after the first two transitions, before the circuit can generalize to longer sequences.

91%, while 12/20 5-node and 13/20 6-node populations did. The most successful populations (those that reached fitness greater than 91% at the third stage of the shaping protocol) were further evolved for 5000 generations on the fourth stage of the shaping protocol, with the most successful 5-node population reaching a 96.85% best fitness. Figure 8 shows the best fitness versus generation plot for this population. Although success is achieved early on the first stages of the task, it takes many more generations to achieve sufficiently high scoring individuals. This was similar for most other successful evolutionary runs. Also, similar to evolution on the discrete version of the task, the fitnesses of the best individuals in the population drop after the difficulty of the task is increased.In the remainder of this section we describe the performance and internal dynamics of the best evolved five-node circuit in some detail (see Table 2 in the Appendix for the evolved parameters of this circuit).

4.2 Learning and Memory Performance

How well can this circuit learn and remember on trials involving more changes of environment and more tests than it was evolved for? To answer this question we tested it on 10^6 trials with 5 changes of environment and up to 10 tests per environment during its lifetime. Pairing and testing temperatures were chosen at random from the full range, and the time-step was an order of magnitude smaller (0.01). The circuit performs correctly on 95.77% of the trials on this set of experiments, indicating that it does indeed represent a sufficiently general solution to the associative learning task for stimuli that can range over a continuum of values.

The behavior of this circuit on a typical sequence of trials is shown in Figure 9. During this sequence, the environment type switches from A to B, and then to an in-between environment C (temp=1.5) at the points indicated by the dashed vertical lines. From Figure 7 we know that switching to an in-between environment (such as C) for the 3-node network discussed previously would have led to poor performance. In contrast, as depicted in Figure 9, the 5-node circuit does manage to remember the in-between signal correctly. One thing to note is the range of time-scales of activity displayed by the components in the circuit: o_1, o_3 and o_5 are fast act-

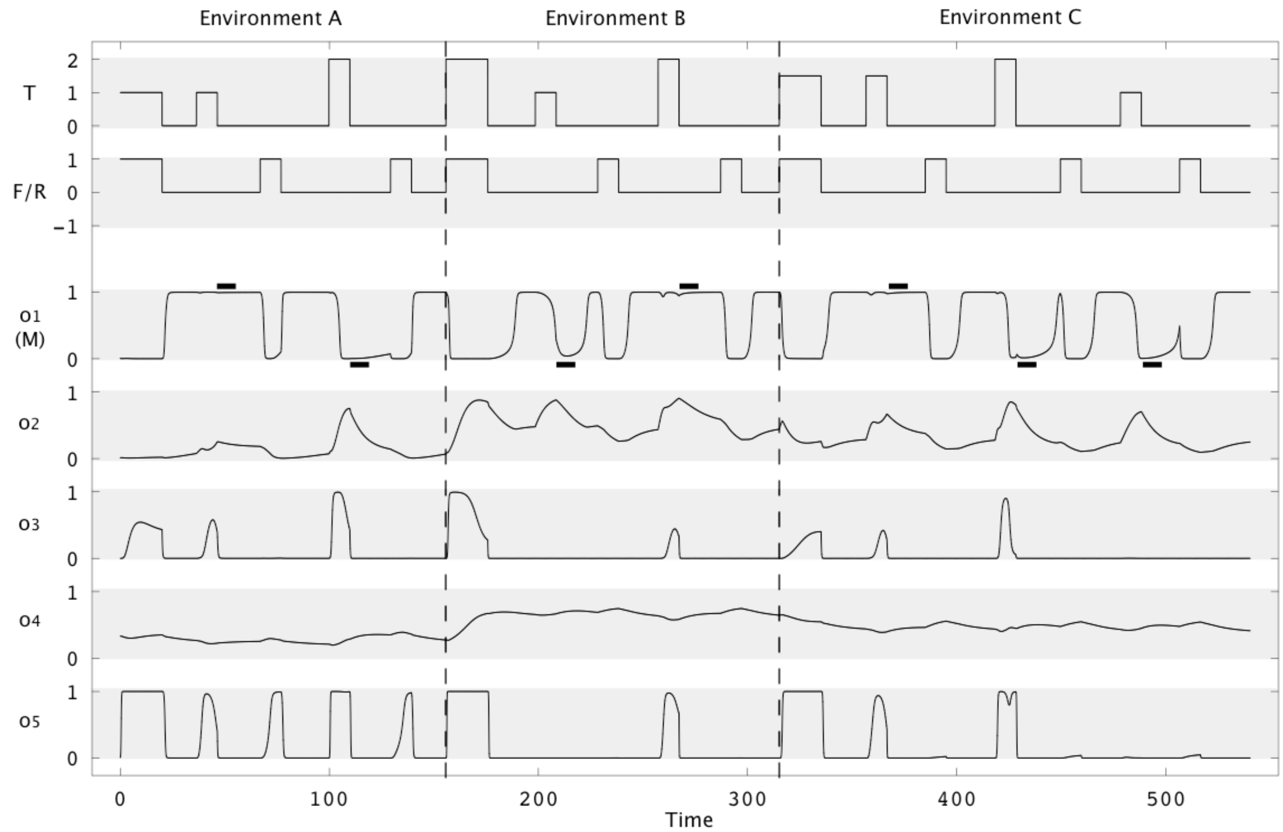

Figure 9. Activity of the best 5-node circuit on a typical trial sequence. From top to bottom the traces correspond to the temperature signal (*T*), the food/reward signal (*F/R*), the mouth state (*M*) and the outputs of the remaining nodes (o_2 to o_5). Labeling conventions are the same as in Figure 4.

ing with time-constants in the range between [1.06, 1.76], o_2 is somewhat slower than those (τ=16.9) and o_4 is much slower acting than all others (τ=73.9). Although this bears some resemblance with the different time-scales evolved for the smaller circuit, the differences between these ranges are much larger.

How well does this circuit generalize to signals in-between the border cases? To answer this we can study the learning map for the best-evolved 5-node circuit on the complete range of pairing and test temperatures. Remember from the circuit evolved for the discrete task (Figure 7) that two types of behavior dominate its performance, since any signal below a certain threshold is treated as ‘cold’ and anything above is treated as ‘hot’. In Figure 10A we show the learning map for the first test after a pairing systematically covering the full-spectrum of combinations. The dominance of the white shade reflects the good generalization performance. The shades of black around the white diagonal line depict the precision of the system’s evolved memory. Any testing temperature that is different from the pairing temperature by less than some variable amount is regarded as the same. Although one way to see this is as a ‘lack of precision’, it is more interesting to treat it as a behavioral generalization: treating temperatures similar to the paired temperature as correct. This is also a direct reflection on the evolutionary conditions, avoiding too similar signals when different. The dashed diagonal lines in the figure depict the range within which there was no selection pressure (except for when the pairing and test temperatures are exactly the same).

The first test is performed directly after the circuit has experienced the to-be-remembered temperature (i.e. the last paired temperature). How does the memory decay over multiple tests? Figure 10B shows the remembering performance for the second test after a random delay. Although there is some degradation in the memory of the original signal, as can be seen by the shades of gray around the diagonal line, it is still mostly appropriate. Figure 10C shows the remembering performance on the 10th test signal. As can be seen, the performance continues to degrade slowly as more and more tests and random delays are applied consecutively after the original pairing.

How fast does memory decay over many more presentations? How does memory depend on the circuit’s experience? Why does it decay, and can it be preserved for longer? In Figure 10D we show the circuit’s ‘forgetting curve’: the remembering performance as a function of the number of test phases experienced. Shaded in gray we show the number of trials the circuit was evolved for. The solid line represents the remembering performance when pairing and testing temperatures are chosen at random. As is expected, the memory

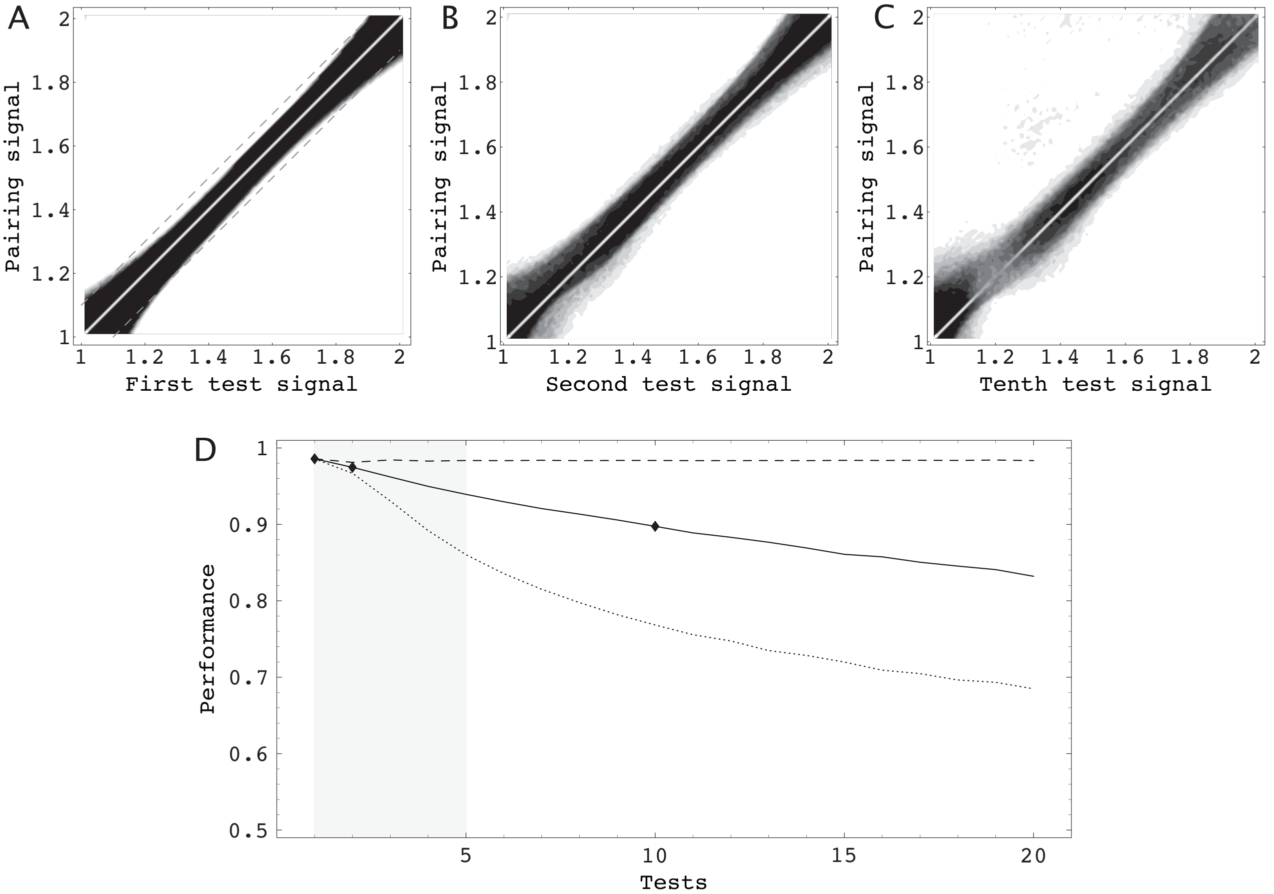

Figure 10. Remembering performance of the best 5-node circuit. Generalization map on the first [A], second [B] and tenth [C] test signal. Labeling conventions are the same as in Figure 5. [D] A measure of the circuit's performance on the i^{th} test: when the signals for all of the previous tests where randomly chosen to be equal or different than the pairing signal with 50% chance (solid line), when the previous test signals are always the same as the pairing signal (dashed line), and when the previous test signals are random but always different from the pairing signal (dotted line). Each point corresponds to the average over 10^5 random runs. The points marked with diamonds correspond to the performance maps shown for parts [A], [B] and [C], respectively. The gray shaded region corresponds to the range of conditions the circuits were evolved in.

of the originally paired temperature decays with the number of tests. But does how fast it decays depend on what the temperatures of the tests are? The dotted line represents the remembering performance when all of the testing temperatures are random but different from the paired temperature. As can be seen, the remembering performance falls much more dramatically, in a classical exponential decay curve. This corresponds well with the literature in experimental psychology, where memory retention is known to decay exponentially as a function of time, in the absence of revision of the learned material (Ebbinghaus, 1885). This memory decay also suggests that not re-experiencing the original temperature decreases the chances of remembering it correctly. Thus, we should expect good remembering performance if every one of the testing temperatures are the same as the one to-be-remembered. The dashed line shows the performance when all previous test signals are equal to the original pairing signal. Indeed, the continued presentation of the to-be-remembered signal strengthens the circuit's memory of it, while long absences result in the degradation of the original memory.

Does this circuit remain sufficiently plastic to re-learn new associations between temperatures and food throughout its lifetime? Or does the plasticity decay after some time or usage? We can test the agent's ability to learn new temperatures by changing the environment multiple times. In Figure 11 we observe the long-term performance for several different re-learning environments. We again observe satisfactory performance. It is as if re-learning resets the state of the circuit entirely. This shows that the agent remains fully plastic outside of the ranges that it was evolved for (i.e. only two changes of environment). In fact, the circuit shows no sign of losing its plasticity with time.

Finally, we can ask about the robustness of the performance in relation to the time delays between tests. Although transients play a role in the dynamics, ideally memory should be more permanent. In other words, it is important to know how stable the memory of the evolved circuits is. Figure 12

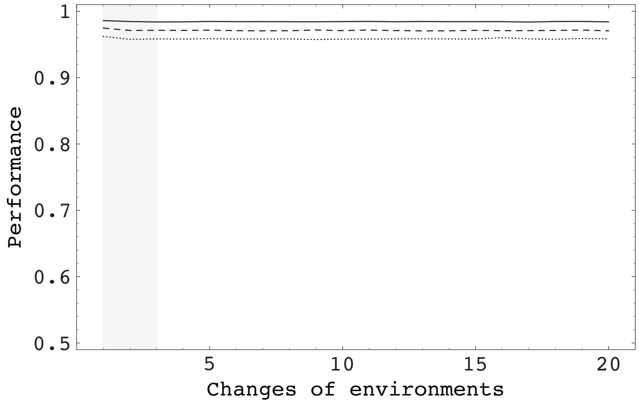

Figure 11. Learning performance of the best 5-node circuit. A measure of the circuit's performance for many changes of environments: on the first (solid line), second (dashed line) and third (dotted line) test after the *i*-th change of environment. Each point corresponds to the average over 10^5 random runs. The gray shaded region corresponds to the range of conditions the circuits were evolved in.

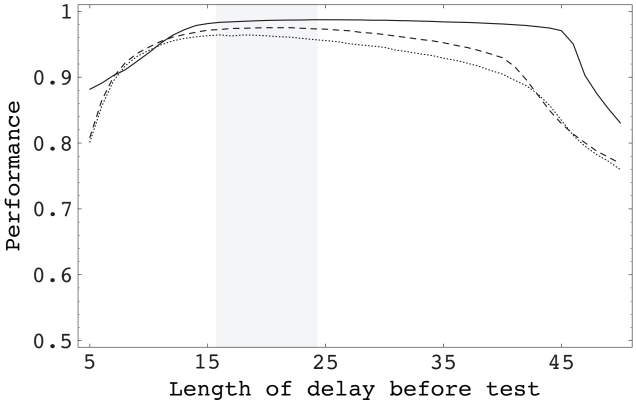

Figure 12. Robustness of the best 5-node circuit to changes in the time delay before the first (solid line), second (dashed line) and third (dotted line) testing phase. Each point corresponds to the average over 10^5 random runs. The gray shaded region corresponds to the range of conditions the circuits were evolved in.

shows the robustness of the circuit's learning performance as a function of the length of the time delays. The circuit manages to be quite robust to time-delays shorter and longer than the range that it was evolved for. Although this was not selected for during evolution, it is a relevant feature of this circuit.

In summary, evolutionary runs for the continuous task were successful only with circuits of size 5 and larger[3]. The behavior and performance of the most successful and smallest circuit was studied. The circuit manages to learn and generalize over the full range of signals on the continuum that it was evolved for. Interestingly, it can remember paired temperatures for longer than it was evolved for, as long as it continues to experience that temperature during tests. The circuit can also remain plastic enough to re-learn new associations within its lifetime. Furthermore, we observed no degradation in the circuit's plasticity over time. Finally, the circuit's memory was relatively robust to longer time delays than those experienced by its ancestors.

5 The Dynamics of Continuous Learning

How does this circuit work? How do the evolved circuit's mechanisms differ from those evolved for the discrete version of the task? Can a finite state machine be extracted to capture the workings of the dynamics of the best-evolved circuit? In order to answer these questions, we have to visualize the overall structure of this circuit's operation, and for this we will use a similar approach to that developed for the discrete version. We 'strobe' the state of the system at selected times during a trial. The main difficulty that arises in this case is that the internal state of the evolved system is composed of more components. Therefore, part of the work in analyzing the internal dynamics of this evolved circuit will involve: (a) looking at several different 3 dimensional slices of this 5 dimensional space, (b) building up our intuitions about the *structure* of the manifolds of activity, and (c) choosing the variables and perspectives that provide the most useful insights. In the figures to follow we look at some of these 3-dimensional slices of the space of activations of the evolved circuit. In particular, we look at slices from the two slowest nodes, y_2 and y_4, and the fastest one, y_5.

In Figure 13, we visualize the state of the system when strobed on the full range of stimuli. We observe that each of the states that correspond to the same activity form a 'stretched out' cluster. Each of the clusters represents the state of the system after a pairing, a rest, a test or a positive or negative reward (labeled *Q1* through *Q5*, respectively). Interestingly, each different state remains relatively well connected and separately clustered. We can also observe that each of the clusters forms a one-dimensional 'tube-like' structure, except for cluster *Q3* that forms something that looks more like a two-dimensional 'wing' structure. We will come back to this point later in the analysis.

An extended behavioral sequence such as the one shown in Figure 9 can then be understood as a set of trajectories between these strobe clusters (see Figure 14). A question of interest is: do these stretched out clusters and the trajectories between them have any further internal structure to them? We can visualize the trajectories in relation to the environment the circuit is in. In Figure 14, trajectories are coded in shades of gray according to their original paired temperature: lighter shades corresponding to hotter temperatures. What can be observed is that the transitions have a relatively structured pattern. The trajectories are arranged from top to bottom according to their paired temperature: with 'hotter'

[3] We do not investigate in this paper whether the ability to generalize is related to the size of the network. The task studied requires that successful circuits generalize, but it also requires: (1) that they remember the original signal on subsequent tests, and (2) that they remain sufficiently plastic to re-learn during their lifetime.

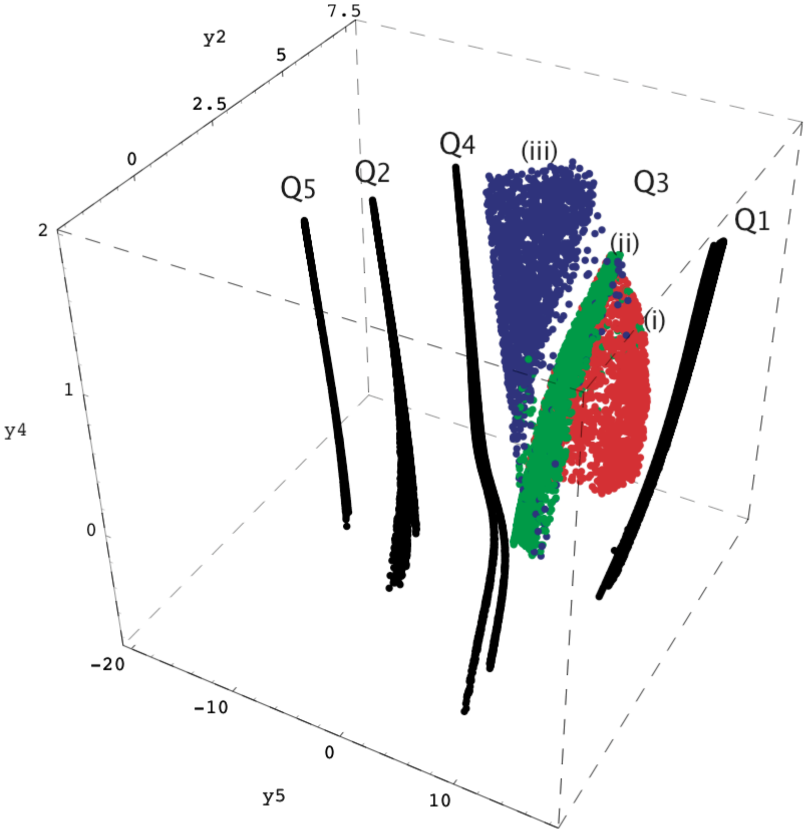

Figure 13. Internal dynamics of the best evolved 5-node circuit. Strobes while tested on the continuum of signals after a pairing (*Q1*), a rest (*Q2*), a test (*Q3*), or a positive (*Q4*) or negative reward (*Q5*). The *Q3* strobed region can be further subdivided according to the relation between the pairing and testing temperature. The pairing temperature can be 'hotter' *(i)*, the same *(ii)*, or 'colder' *(iii)* than the original.

ones at the top and 'colder' ones towards the bottom. Most importantly, this pattern is maintained as the state of the system flows between each of the different clusters. This corresponds to the 'memory' of the environment.

A crucial aspect to the learning behavior under study is the circuit's ability to make different decisions depending on its experience. One question of interest is, how does the decision arise internally? To answer this, it will be useful to take a closer look at the transitions between the resting state (*Q2*) and the testing state (*Q3*) (plots E, F and G in Figure 14). During the resting state the agent can be tested with the same temperature as it was paired with originally, or it can be tested with a different temperature. In the latter case, the temperature can be either 'colder' than the original or 'hotter'. Figure 14 shows the state of the system as it transitions from its resting state to being tested on any of the possible signals. The state of the system moves to the middle part of the cluster *Q3(ii)* for signals that are similar to the original (Figure 14F). What this means is that the original temperature is known from the level of the *Q2* cluster, where the system is operating. When the temperature is different than the original, the state of the system falls away from the middle into one of the two outer 'wings' of the structure. Falling to the top left 'wing' *Q3(iii)* when the testing temperature is 'colder' than the original pairing temperature (Figure 14E) and to the bottom right 'wing' *Q3(i)* when it is 'hotter' (Figure 14G).

As we have seen, the circuit discovers the environment it is in through the simultaneous pairing of food and temperature. Yet, unlike the discrete scenario (in particular for $n = 2$), receiving a negative reward is not enough to modify the state of the agent such that it 'finds out' which temperature is the right one. Furthermore, a successful circuit could simply never receive negative reward during its lifetime. This is not, however, the case for the circuit under analysis. From Figure 14C, we know that some signals end up near the border between the 'wings' and the middle part of the *Q3* cluster, which then receive a negative reward moving the state of the system to *Q5*. From Figure 10A, we know that these correspond to test signals that are very similar (but different) from the paired temperature. The negative reward, however, does not 'correct' these borderline cases. We examined this by artificially inducing a negative reward after the agent opens its mouth when tested on the paired temperature (for which it usually receives a positive reward). When tested again using the same temperature the agent would still open its mouth. Thus, the negative reward cannot override the original pairing memory in this circuit. Similarly, we examined whether the positive reward could trigger the circuit to relearn a new association. We artificially induced a positive reward after

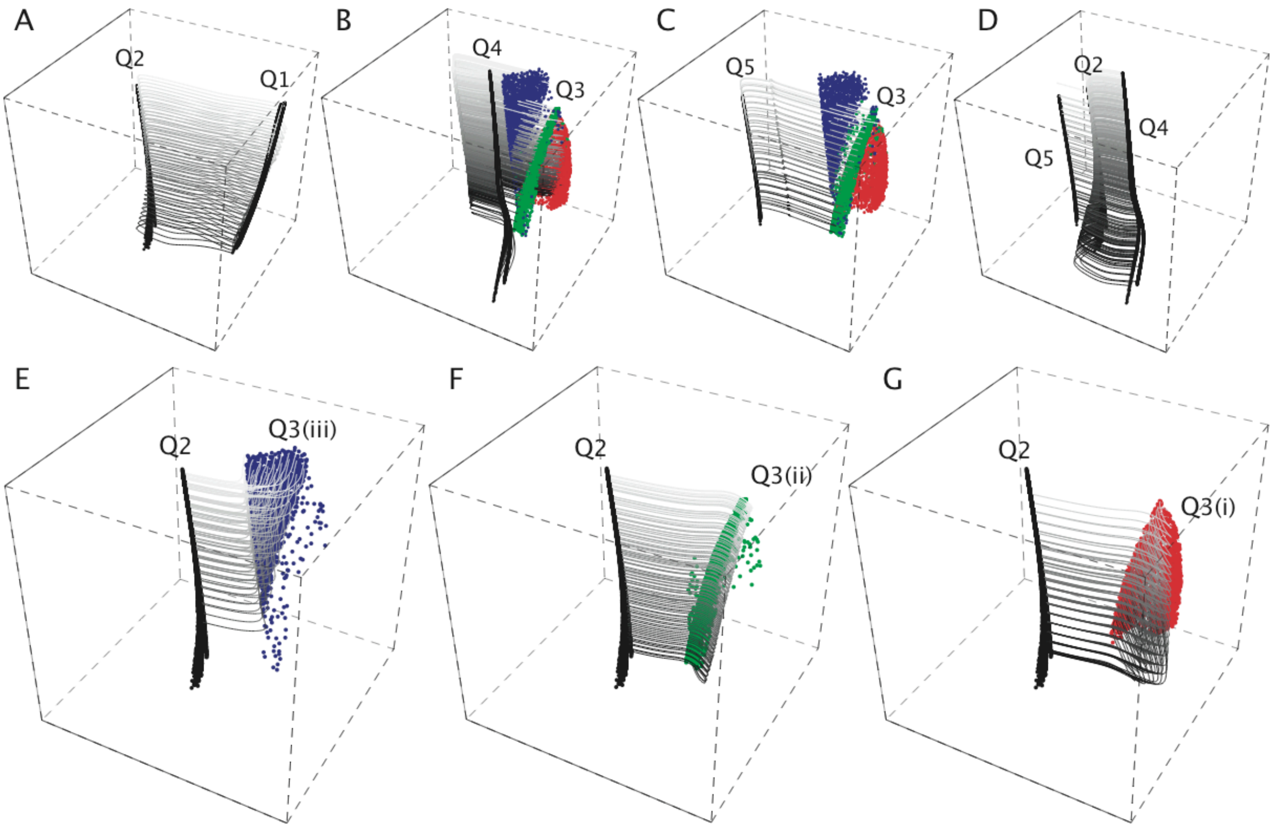

Figure 14. Transitions between each of the states superimposed over the strobed states. Each transition is shaded in gray according to the original pairing temperature, with lighter shades representing hotter temperatures and darker shades colder temperatures. [A] Transitions between pairing and resting. [B] Transitions while positive reward is applied. [C] Transitions while negative reward is applied. Notice that negative reward occurs naturally only when the test temperature is different but too similar to the pairing temperature. The circuit makes the mistake of classifying these as the same, despite the small difference. Thus, the transitions depart from *Q3(ii)* mostly. [D] Transitions during resting after a positive or negative reward. The bottom three figures depict the transitions between the resting state *Q2* and the presentation of test temperatures over the whole continuous range: [E] shows the transitions when the test temperature is lower ('colder') than the pairing temperature; [F] when the test temperature is the same as the pairing temperature; and [G] when the test temperature is higher ('hotter') than the original pairing temperature. As can be seen by the predominance of lighter-shaded transitions in [E], when the original pairing is of a 'hotter' (lighter gray trajectories) temperature, the number of test temperatures that are classified as 'colder' is greater than when the original pairing is of a 'colder' (darker gray) temperature. The inverse is true for [G], where there is a predominance of darker-shaded transitions.

the agent closes its mouth when tested on a different signal to the paired temperature (for which it usually receives a negative reward). When tested again using this new temperature (which provided a positive reward despite not being the original paired temperature), the response of the circuit was still to close its mouth. Thus, the circuit *only* learns new associations through the simultaneous pairing of food and temperature; not through the reward signal. The most likely reason for this is that no changes of environment were experienced of the latter form during evolution.

A key question that we would like to ask, then, is whether we can extract an FSM from its internal dynamics, such that it explains the learning behavior? Given that the network has to remember a continuous signal, a 'machine' is required that will allow for a continuum of states to represent the environment. No *finite* state machine can represent such internal mechanisms. A richer structure is needed: a machine that includes for each of the discrete states an inner (relatively independent) continuous state. We are calling this set of machines, continuous state machines (CSM). One way to think of these is as a *continuous manifold of finite state machines*. Accordingly, we can think of a FSM as a CSM with only one inner 'level'. Figure 15 shows two of the FSMs, on top of each other with a transition from one to the next. The dynamics are just like an FSM but with stretched-out regions for each state. We can think of each of the states as containing a real-value register. This inner state is continuous and is instantiated as the level within the extended strobe clusters.

At the behavioral level, the CSM denotes two seemingly distinct processes operating at two different scales. While the discrete states resemble states of an FSM, the continuous regions inside each of the discrete states resemble something more like an infinite tape. We can illustrate this idea using an example sequence trial in our evolved circuit. In Figure 16, we show the trajectory of the state of three of the nodes during an example sequence trial where the agent is first paired

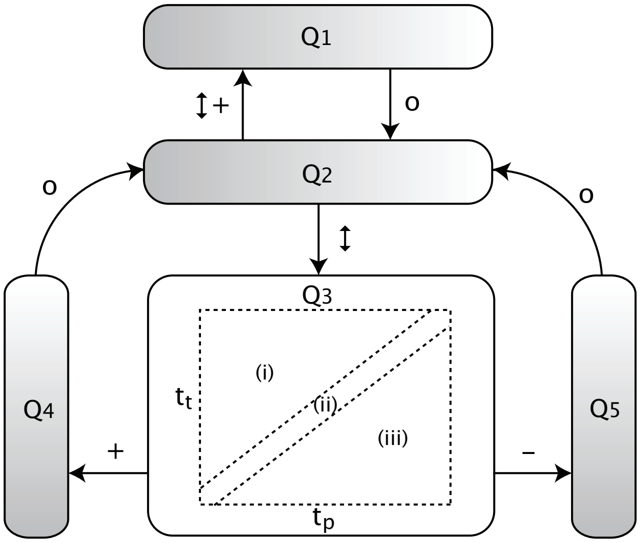

Figure 15. A ‘continuous’ state machine embedded in the best 5-node circuit. The states are labeled according to the strobed states from the previous figures. The shade of gray for each state represents the real-valued register. Where the agent’s state ‘lands’ inside the *Q3* region is determined by the pairing temperature (t_p) along the horizontal axis and the testing temperature (t_t) along the vertical axis: when the test temperature is ‘hotter’ than the original pairing temperature ($t_t > t_p$) the state falls into the *Q3(i)* region; when they are the same ($t_t = t_p$) it falls into the *Q3(ii)* region; finally, when the test temperature is ‘colder’ than the original pairing temperature $t_t < t_p$ it falls into the *Q3(iii)* region.

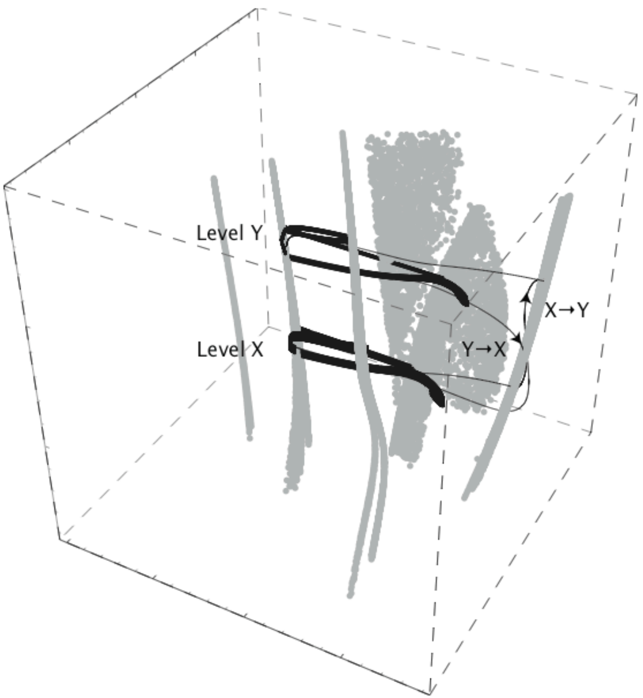

Figure 16. Example sequence trial of a continuous state machine as a manifold of finite state machines. An example finite state machine (level 1) is shown at the bottom of the figure and another one (level 2) is shown at the top of the figure. The continuity arises from the transitions between any two finite state machines. The example shows transitions from FSMs *X* (temp=1.3) and *Y* (temp=1.7).

and tested with temperature *X* (1.3), then paired with and tested with a different temperature *Y* (1.7), and vice-versa, several times. The trajectory is placed within the context of the ‘strobed’ states (in gray). This illustrates the notion of the system’s operation at different levels within the manifold of FSMs (*X* and *Y*), as well as the transitions between these ($X \to Y$ and $Y \to X$) corresponding to re-learning new associations. It is important to point out that this evolved circuit requires ongoing interaction with an environment in order to maintain a given operational level within the manifold of FSMs. This can be observed best from the forgetting curve shown in Figure 10D. Without this ongoing interaction, the state will eventually decay to some fixed level.

Finally, we would like to know how the state machine is related to the circuit’s evolved components. The best 5-node circuits taken from the five best evolutionary runs (using different seeds) show a distinct distribution of time-constant parameters: the majority of the components are as fast-acting as is allowed but a few are much slower. Is there a functional relation between the discrete states and the fast nodes, and between the continuous internal state and the slow nodes? We can answer this for the case of the best-evolved circuit. Although the full circuit is responsible (and necessary) for the learning phenomena, we can test the correlation between the paired temperature and the state of the system, at different times during a trial and for every component in the circuit. We do this using Pearson’s product-moment correlation coefficient (Moore, 2006):

$$r(p,y) = \frac{\sum_{t=1}^{2}(p_t - \bar{p})(y_t - \bar{y})}{\sqrt{\sum_{t=1}^{2}(p_t - \bar{p})^2}\sqrt{\sum_{t=1}^{2}(y_t - \bar{y})^2}} \qquad (4)$$

where p_t is the original paired temperature and y_t is the state of one of the nodes at a particular stage during the trial (i.e. *Q1* through *Q5*) in an environment t; $\bar{p}$ and $\bar{y}$ are the averages of p and y, respectively. Temperatures, t, in the full range $[1,2]$ (incremented in steps of 0.01), were used to record the state of each of the nodes during each of the different stages.

As can be seen from Figure 17, all of the nodes are highly correlated with the remembered temperature (either positively or negatively) at most of the stages of a sequence trial. However, while the correlation among the fast set of components (gray) varies within a trial, the correlation of the slow components (black) remains remarkably stable. This suggests that the role they play in the maintenance of the ‘memory trace’ is stronger than the faster subset of nodes.

In summary, to understand how the most successful and smallest circuit works we strobed the state of the system at selected times during a trial. We observed separate clusters that stretched out with a relatively structured inner dimension. The different clusters corresponded to the different events in the trial (i.e. pairing, rests, tests, rewards). Interestingly, the attractors of the circuit did not always correspond

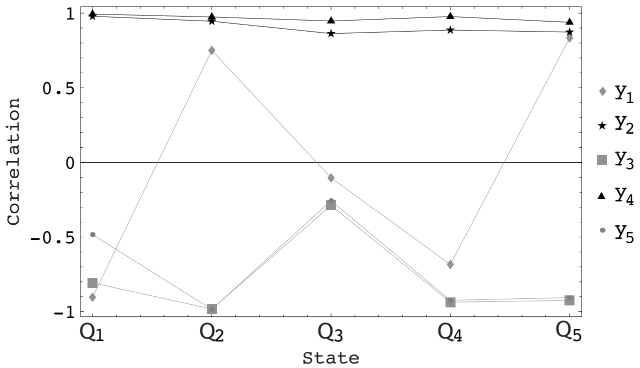

Figure 17. Pearson's correlation coefficient between the activation state of each node in the circuit and the paired temperature (corresponding to the to-be-remembered signal) at different times during a trial. The fast-acting nodes are shown in gray and the slow-acting nodes are shown in black.

to the clusters. The inner dimension, on the other hand, corresponded to the to-be-remembered signal. As the trial proceeds, the state of the system transitions from cluster to cluster, while maintaining the structure of the inner dimension. The decision process was shown to involve a more complex two-dimensional-like cluster, where a relational categorization process was observed: with 'hotter', 'similar', and 'colder' test temperatures neatly separated. We described the evolved learning mechanism in terms of a continuous manifold of finite state machines. Finally, although all of the components in the network are involved in the generation of the learning behavior, we observed a stronger maintenance of the correlation between the to-be-remembered signal in the slower acting components of the network compared to the faster ones.

6 General Discussion

In this work we have extended previous work on evolving learning without synaptic plasticity from discrete (in practice 2-choice) tasks to continuous tasks. We address two main questions.

First, can this approach be extended to continuous tasks? We show that continuous-time recurrent neural networks without synaptic plasticity are successfully evolved on an associative learning task abstracted from a temperature preference behavior observed in *Caenorhabditis elegans*. The behavioral task studied in this paper is, of course, not exclusive to *C. elegans*. Broadly, it involves learning an environmental feature that can range over a continuum of values and remembering it for later as a preference. It also involves the ability to change this preference when appropriate. This is a rather common ability amongst living organisms, including humans.

Second, how does learning without synaptic plasticity work in the evolved circuits? In this work we have shown how the evolved internal dynamics differ in an associative learning task when the stimuli to-be-associated is on a continuum as opposed to a discrete set. The analysis of evolved agents for associative learning, where the stimuli to-be-remembered are discrete signals, display finite state machine like internal mechanisms. This agrees with recent results presented in Phattanasri et al. (2007). A different and richer type of state machine is found when analyzing agents evolved to remember and discriminate between signals from a continuum. Because of the ability of the evolved circuit to use a *continuous* state inside a set of finite states, we have come to consider it as a different class of automata that we call a *continuous state machine*.

It has been known for some time that artificial neural networks have the capacity to act as finite state machines (McCulloch & Pitts, 1943; Minsky, 1967). In particular, the relation between recurrent neural networks and automata has been treated by several authors (Cleeremans et al., 1989; Servan-Schreiber et al., 1991; Pollack, 1991; Giles et al., 1992; Casey, 1996). None of this work has discussed the notion of a manifold of finite state machines or a continuous state machine, nor have they been observed to arise in neural networks. A relation between FSMs and the state space representation of continuous control theory has been indicated in Elgerd (1967). A related notion has been developed in the context of grammar recognition using recurrent networks in Servan-Schreiber et al. (1991) called graded state machines. The notion of continuous state and graded state machines is different in two important ways. First, the infinite and graded states of a CSM are clustered around discrete and separate finite states. Second, there is a relevant relationship between the continuous dimension across the separate clusters of finite states. Only an intuitive notion of continuous state machines has been provided in this work. Developing a formal account in the context of automata theory may be of interest in the future.

What is the role of transients over multiple timescales? First of all, it is important to note that the strobed points are not, in general, attractors of the evolved circuit. Rather, the system is always being pulled from one attractor to the next by the changing sensory input. Thus the evolutionary algorithm has shaped the *transient* dynamics of the circuits to solve the task at hand, not its attractor structure. It is also important to note that the best 5-node circuits taken from the 5 best evolutionary runs (using different seeds) all consistently showed at least two different time-scales in their evolved internal components. Although the majority of neural components evolved to be as fast-acting as possible (with time-constants near 1.0), for each circuit at least one (but in some cases two) of the neural components evolved to be much slower-acting (by at least an order of magnitude). This points to the importance of developing the tools and language to understand dynamical systems with components interacting over multiple timescales[4].

[4] Classical examples of multiple timescale systems (e.g. weakly-coupled and relaxation oscillations) are covered in most introductory dynamical systems textbooks (see, for example, Strogatz, 1994). For an example of multiple timescale techniques to analyze bursting in neurons see Izhikevich (2000). For applied dynamical

Learning has typically been associated with lifetime synaptic change. Here we contribute to the literature demonstrating that learning can occur in the absence of this type of change (Yamauchi & Beer, 1994a,b; Tuci et al., 2002; Phattanasri et al., 2007; Izquierdo & Harvey, 2007). But can circuits with fixed weights *really* learn? To be clear, the evolutionary algorithm does change the weights of the network over generations. But the learning behavior that we study in this paper occurs over a much shorter timescale: the lifetime of the agent. As a consequence, evolution does not operate during the temperature preference learning phenomena. Thus, the weights of the network (as well as all other parameters) remain fixed. But, doesn't that mean that the "learning" is just fixed memorization behavior tuned as a consequence of weight changes during evolution? No, it is not. The network must react differently to stimuli from the environment depending on events that occur during its lifetime. In our particular example, evolution cannot 'know' *a priori* whether the network will have to open or close its mouth for cold or hot temperatures. This the network must learn during its lifetime.

There are in fact multiple ways in which a CTRNN with fixed weights can exhibit learning behavior. During the lifetime of a network, the activation of some CTRNN nodes may change very slowly compared to other nodes in the network. A network of slow nodes and fast nodes might be understood to resemble a network of slow weights and fast nodes. The slow nodes might change in a way that modulates behavior, and responds to feedback in much the same way as weight changes are brought about by traditional "learning rules". These nodes just don't happen to be labeled as 'weights'. It would perhaps be possible to identify the nodes with slower time-parameters and arbitrarily label them 'synaptic-weight-equivalent nodes'. However, 'nodes acting as synaptic weights' is only one possibility. A CTRNN can exhibit dynamics on a range of timescales even if the time constants of all the nodes are fixed at unity, due to the interactions between the nodes. Notwithstanding this possibility, for the associative learning task studied here, artificial evolution exploited predominately the ability to use components with inherently different timescales of activity. Thus, while memory could have arisen from, for example, reverberatory dynamics (Lau & Bi, 2005), here it was the dynamics of the slower-acting components that instantiated the 'memory trace'. This is evidenced by the maintenance of a high correlation between their levels of activity and the continuous signal to-be-remembered. This demonstrates that continuous-time recurrent neural networks with fixed weights can produce genuine learning behaviors that go beyond switching between two modes of interaction.

The circuits evolved for the continuous version of this task required constant interaction with their environment to maintain their temperature memory. This highlights the situated nature of the learning task. Would it be 'better' if the circuits could maintain their memory indefinitely and in the absence of environmental interaction? Traditionally in robotics, 'memory mechanisms' are designed to remember everything indefinitely, regardless of how old or new, main or secondary the information is. This is not necessarily the case for living organisms. What accounts for appropriate memorization behavior when an agent needs to interact with its environment is likely to be very different than the memory required by information processors. Issues of context and time-sensitivity become relevant. For living organisms, remembering recent experiences is usually more important than older ones. For example, you would like to remember where you parked your car this morning, not necessarily all of the locations on previous days! Similarly, remembering highly recalled memories is also more important than recalling less frequently needed ones. For example, you would like to remember the names of people you interact with on a daily basis at work better than those of whom you haven't seen in a really long time. In fact, several studies have found consistency in forgetting curves across tasks, measurement metrics and even species (Wixted & Ebbesen, 1991). These studies suggest that memory declines as a power function of time. This is the first example known to the authors where similar forgetting curves are observed in artificially evolved circuits for learning behavior. It is important to note that this is not a limitation of the evolved circuit, but a consequence of its situated nature.

Continuous-time recurrent neural networks without synaptic plasticity have now been demonstrated to be capable of associative learning on both discrete and continuous stimulus spaces. How much further can this approach be taken? The most obvious next step could be to study second and higher-order conditioning, where the initially associated stimulus can consequently be used to learn about some new stimulus. Another useful next step would be to study the blocking effect, a phenomenon observed whereby conditioning to a stimulus is blocked if the stimulus has been reinforced in compound with a previously conditioned stimulus. Both phenomena are discussed in most textbooks on learning. Finally, it is important to note that one of the major differences between our task and the behavior performed by the nematodes is the agent's embodiment. In the case of the worm, it influences the sensory stimuli that it receives next by moving up or down the thermal gradient. In our task, the situation is more akin to traditional psychology experiments, where the experimenter immobilizes the subject (e.g. glues the worm to a petri dish) while applying different stimuli to it and studying its responses in a highly structured manner. One important direction of future work will be to analyse evolved circuits using more *ecological* learning scenarios.

Acknowledgements

We thank Seth Bullock, Peter Fine and Sara Mitri for their feedback on drafts of this article. We also thank helpful comments from anonymous reviewers.

systems describing the analysis of 3-timescale dynamics see Krupa et al. (2008).

References

Abbott, L., & Nelson, S. B. (2000). Synaptic plasticity: taming the beast. *Nature Neuroscience*, *3*, 1178 –1183.

Anderson, J., Albergotti, L., Proulx, S., Peden, C., Huey, R., & Philips, P. (2007). Thermal preference of *Caenorhabditis elegans*: a null model and empirical tests. *The Journal of Experimental Biology*, *210*, 3107–3116.

Beer, R. D. (1995). On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior*, *3*(4), 469–509.

Bi, G.-Q., & Rubin, J. (2005). Timing in synaptic plasticity: From detection to integration. *Trends in neuroscience*, *28*, 222–228.

Biron, D., Shibuya, M., Gabel, C., Wasserman, S., Clark, D., Brown, A., et al. (2006). A diacylglycerol kinase modulates long-term thermotactic behavioral plasticity in *C. elegans*. *Nature Neuroscience*, *9*, 1499–1505.

Blynel, J., & Floreano, D. (2003). Exploring the T-maze: Evolving learning-like robot behaviors using CTRNNs. In C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, & E. Costa (Eds.), *Applications of evolutionary computing* (pp. 593–604). Berlin: Springer-Verlag.

Casey, M. (1996). The dynamics of discrete-time computation, with application to recurrent neural networks and finite state machine extraction. *Neural Computation*, *8*, 1135–1178.

Churchland, P., & Sejnowski, T. (1992). *The computational brain*. Cambridge, MA: MIT Press.

Clark, D., Biron, D., Sengupta, P., & Samuel, A. (2006). The AFD sensory neurons enconde multiple functions underlying thermotactic behavior in *Caenorhabditis elegans*. *The Journal of Neuroscience*, *26*(28), 7444-7451.

Cleeremans, A., Servan-Schreiber, D., & McClelland, J. (1989). Finite state automata and simple recurrent networks. *Neural Computation*, *1*(3), 372–381.

Cudmore, R., & Turrigiano, G. (2004). Long-term potentiation of intrinsic excitability in LV visual cortical neurons. *Journal of Neurophysiology*, *92*, 341–348.

De Bono, M., & Maricq, A. (2005). Neuronal substrates of complex behaviors in *C. elegans*. *Annual Review of Neuroscience*, *28*, 451–501.

Ebbinghaus, H. (1885). *Memory: A contribution to experimental psychology*. New York: Columbia University.

Elgerd, O. (1967). *Control systems theory*. New York: McGraw Hill.

Floreano, D., & Urzelai, J. (2000). Evolutionary robots with online self-organization and behavioral fitness. *Neural Networks*, *13*(4–5), 431–443.

Floreano, D., & Urzelai, J. (2001). Evolution of plastic control networks. *Autonomous Robots*, *11*(3), 311–317.

Funahashi, K., & Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, *6*(6), 801–806.

Giles, C., Miller, C., Chen, D., Chen, H., Sun, G., & Lee, Y. (1992). Learning and extracting finite state automata with second-order recurrent neural networks. *Neural Computation*, *4*(3), 393–405.

Harvey, I. (2001). Artificial evolution: a continuing SAGA. In T. Gomi (Ed.), *Evolutionary robotics: From intelligent robots to artificial life* (Vol. 2217, p. 94-109). Berlin: Springer-Verlag.

Hausser, M., Spruston, N., & Stuart, G. (2000). Diversity and dynamics of dendritic signaling. *Science*, *290*, 739–744.

Hedgecock, E., & Russell, R. (1975). Normal and mutant thermotaxis in the nematode *Caenorhabditis elegans*. *Proceedings of the National Academy of Science of the USA*, *72*(10), 4061–4065.

Hobert, O. (2003). Behavioral plasticity in the *C. elegans*: Paradigms, circuits, genes. *Journal of Neurobiology*, *54*, 203–223.

Izhikevich, E. M. (2000). Neural excitability, spiking and bursting. *International Journal of Bifurcation and Chaos*, *10*(6), 1171–1266.

Izquierdo, E., & Harvey, I. (2006). Learning on a continuum in evolved dynamical node networks. In L. M. Rocha, M. Bedau, D. Floreano, R. Goldstone, A. Vespignani, & L. Yaeger (Eds.), *Proceedings of the 10th international conference on the simulation and synthesis of living systems* (pp. 507–512). Cambridge, MA: MIT Press.

Izquierdo, E., & Harvey, I. (2007). Hebbian learning using fixed weight evolved dynamical 'neural' networks. In H. A. Abbass, M. Bedau, S. Nolfi, & J. Wiles (Eds.), *Proceedings of the 1st ieee symposium on artificial life*. Piscataway, NJ: IEEE Press.

Kandel, E. (2000). Cellular mechanisms of learning and the biological basis of individuality. In *Principles of neural science* (Fourth ed., pp. 1247–1277). New York: McGraw-Hill.

Krupa, M., Popović, N., & Kopell, N. (2008). Mixed-mode oscillations in three time-scale systems: A prototypical example. *Journal on Applied Dynamical Systems*, *7*(2), 361-420.

Lau, P.-M., & Bi, G.-Q. (2005). Synaptic mechanisms of persistent reverberatory activity in neuronal networks. *Proceedings of the National Academy of Science of the USA*, *102*(29), 10333–10338.

Llinas, R. (1988). The intrinsic electrophysiological properties of mammalian neurons: Insights into central nervous system function. *Science*, *242*(4886), 1654–1664.

Luo, L., Clark, D., Biron, D., Mahadevan, L., & Samuel, A. (2006). Sensorimotor control during isothermal tracking in *Caernohabditis elegans*. *The Journal of Experimental Biology*, *209*, 4652–4662.

Marder, E., Abbott, L., Turrigiano, G., Liu, Z., & Golowasch, J. (1996). Memory from the dynamics of intrinsic membrane currents. *Proceedings of the National Academy of Science of the USA*, *93*, 13481–13486.

McCulloch, W., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, *5*, 115–133.

Minsky, M. (1967). *Computation: Finite and infinite machines*. New Jersey: Prentice-Hall.

Mohri, A., Kodama, E., Kimura, K., Koike, M., Mizuno, T., & Mori, I. (2005). Genetic control of temperature preference in the nematode *Caenorhabditis elegans*. *Genetics*, *169*(3), 1437-1450.

Moore, D. (2006). *The basic practice of statistics* (4th ed.). New York: W.H. Freeman.

Mori, I. (1999). Genetics of chemotaxis and thermotaxis in the nematode *Caenorhabditis elegans*. *Annual Review of Genetics*, *33*, 399–422.

Mori, I., & Ohshima, Y. (1995). Neural regulation of thermotaxis in *Caenorhabditis elegans*. *Nature*, *376*, 344–348.

Mori, I., & Ohshima, Y. (1997). Molecular neurogenetics of chemotaxis and thermotaxis in the nematode *Caenorhabditis elegans*. *BioEssays*, *19*(12), 1055–1064.

Murakami, H., Bessinger, K., Hellmann, J., & Murakami, S. (2005). Aging-dependent and independent modulation of associative learning behavior by insulin/insulin-like growth factor-1 signal in *Caenorhabditis elegans*. *Journal of Neuroscience*, *25*(47), 10894-10904.

Nolfi, S., & Floreano, D. (1999). Learning and evolution. *Autonomous Robots*, *7*(1), 89–113.

Phattanasri, P. (2002). *Associative learning in evolved dynamical neural networks*. Unpublished doctoral dissertation, Case Western University.

Phattanasri, P., Chiel, H., & Beer, R. D. (2007). The dynamics of associative learning in evolved model circuits. *Adaptive Behavior*, *15*(4), 377–396.

Pollack, J. (1991). The induction of dynamical recognizers. *Machine Learning*, *7*, 227–252.

Ryu, W., & Samuel, A. (2002). Thermotaxis in *Caenorhabditis elegans* analysed by measuring responses to defined thermal stimuli. *The Journal of Neuroscience*, *22*(13), 5727–5733.

Servan-Schreiber, D., Cleeremans, A., & McClelland, J. (1991). Graded state machines: The representation of temporal contingencies in simple recurrent networks. *Machine Learning*, *7*(2–3), 161–193.

Spector, L., & Klein, J. (2005). Trivial geography in genetic programming. In T. Yu, R. Riolo, & B. Worzel (Eds.), *Genetic programming theory and practice iii* (pp. 109–124). New York: Kluwer Academic Publishers.

Strogatz, S. (1994). *Nonlinear dynamics and chaos: With applications to physics, biology, chemistry and engineering*. Reading, MA: Addison-Wesley.

Toledo-Rodriguez, M., El Manira, A., Wallen, P., Svirskis, G., & Hounsgaard, J. (2005). Cellular signalling properties in microcircuits. *Trends in Neurosciences*, *28*(10), 534–540.

Tuci, E., Quinn, M., & Harvey, I. (2002). An evolutionary ecological approach to the study of learning behavior using a robot-based model. *Adaptive Behavior*, *10*(3–4), 201–221.

Wixted, J., & Ebbesen, E. (1991). On the form of forgetting. *American Psychological Society*, *2*(6), 409–415.

Yamauchi, B., & Beer, R. D. (1994a). Integrating reactive, sequential and learning behavior using dynamical neural networks. In D. Cliff, P. Husbands, J. Meyer, & S. Wilson (Eds.), *From animals to animats 3: Proceedings of the 3rd international conference on simulation of adaptive behavior* (pp. 382–391). Cambridge, MA: MIT Press.

Yamauchi, B., & Beer, R. D. (1994b). Sequential behavior and learning in evolved dynamical neural networks. *Adaptive Behavior*, *2*(3), 219–246.

Zariwala, H. A., Miller, A. C., Faumont, S., & Lockery, S. R. (2003). Step response analysis of thermotaxis in *Caenorhabditis elegans*. *Journal of Neuroscience*, *23*(10), 4369-4377.

7 Appendix: Evolved Parameters

Table 1
Best 3-node circuit for the discrete version of the temperature preference task

	y_1	y_2	y_3
y_1	9.7529	1.1023	-9.1226
y_2	-5.2143	2.2904	-1.7911
y_3	-9.7031	-9.2973	5.5012
T	-2.6291	-8.7298	-7.7723
F	0.4719	4.0520	-9.9976
θ	2.9464	6.4924	7.3431
τ	1.2145	27.8472	1.0256

Table 2
Best 5-node circuit for the continuum version of the temperature preference task

	y_1	y_2	y_3	y_4	y_5
y_1	9.6712	1.8562	4.3859	-0.6227	-0.7912
y_2	-5.1801	-2.4098	-3.5321	8.8119	-9.9244
y_3	-6.5417	9.3668	1.3050	9.7333	-5.9810
y_4	7.0174	5.7888	-9.9125	-4.5742	-8.5561
y_5	8.3435	-9.2429	2.8769	-4.9563	5.2326
T	-2.8401	5.5182	5.5025	-1.3114	9.3223
F	-7.6829	-1.5416	4.2153	3.9677	7.6002
θ	-4.8178	-4.4765	-9.9440	-0.7085	-5.2138
τ	1.7630	16.9439	1.5663	73.9571	1.0663