

Computational aspects of feedback in neural circuits

Wolfgang Maass*, Prashant Joshi*,
and Eduardo D. Sontag[†]

Corresponding author: Wolfgang Maass (maass@igi.tugraz.at)

*Institute for Theoretical Computer Science, Technische Universitaet Graz, Inffeldgasse 16b, A-8010
Graz, Austria, Tel. +43 316 873 5822, Fax. +43 316 873 5805

[†]Department of Mathematics, Rutgers, The State University of New Jersey, USA

Abstract

It had previously been shown that generic cortical microcircuit models can perform complex real-time computations on continuous input streams, provided that these computations can be carried out with a rapidly fading memory. We investigate in this article the computational capability of such circuits in the more realistic case where not only readout neurons, but in addition a few neurons *within* the circuit have been trained for specific tasks. This is essentially equivalent to the case where the output of trained readout neurons is fed back into the circuit. We show that this new model overcomes the limitation of a rapidly fading memory. In fact, we prove that in the idealized case without noise it can carry out any conceivable digital or analog computation on time-varying inputs. But even with noise the resulting computational model can perform a large class of biologically relevant real-time computations that require a non-fading memory. We demonstrate these computational implications of feedback both theoretically, and through computer simulations of detailed cortical microcircuit models that are subject to noise and have a complex inherent dynamics. We show that the application of simple learning procedures (such as linear regression or perceptron learning) to a few neurons enables such circuits to represent time over behaviorally relevant long time spans, to integrate evidence from incoming spike trains over longer periods of time, and to process new information contained in such spike trains in diverse ways according to the current internal state of the circuit. In particular we show that such generic cortical microcircuits with feedback provide a new model for working memory that is consistent with a large set of biological constraints.

Although this article examines primarily the computational role of feedback in circuits of neurons, the mathematical principles on which its analysis is based apply to a variety of dynamical systems. Hence they may also throw new light on the computational role of feedback in other complex biological dynamical systems, such as for example genetic regulatory networks.

Synopsis

Circuits of neurons in the brain have an abundance of feedback connections, both on the level of local microcircuits and on the level of synaptic connections between brain areas. But the functional role of these feedback connections is largely unknown. The authors present a computational theory which characterizes the gain in computational power that feedback can provide in such circuits. It shows that feedback endows standard models for neural circuits with the capability to emulate arbitrary Turing machines. In fact, with a suitable feedback they can simulate any dynamical system, in particular any conceivable analog computer. Under realistic noise conditions the computational power of these circuits is necessarily reduced. But the authors demonstrate through computer simulations that feedback also provides a significant gain in computational power for quite detailed models of cortical microcircuits with in-vivo-like high levels of noise . In particular it enables generic cortical microcircuits to carry out computations that combine information from working memory and persistent internal states in real-time with new information from online input streams.

1 Introduction

The neocortex performs a large variety of complex computations in real-time. It is conjectured that these computations are carried out by a network of cortical microcircuits, where each microcircuit is a rather stereotypical circuit of neurons within a cortical column. A characteristic property of these circuits and networks is an abundance of feedback connections. But the computational function of these feedback connections is largely unknown. Two lines of research have been engaged in order to solve this problem. In one approach, which one might call the constructive approach, one builds hypothetical circuits of neurons and shows that (under some conditions on the response behavior of its neurons and synapses) such circuits can perform specific computations. In another research strategy, which one might call the analytical approach, one starts with data-based models for actual cortical microcircuits, and analyses which computational operations such “given” circuits can perform under the assumption that a learning process assigns suitable values to some of their parameters (e.g. synaptic efficacies of readout neurons). An underlying assumption of the analytical approach is that complex recurrent circuits, such as cortical microcircuits, cannot be fully understood in terms of the usually considered properties of their components. Rather, system level approaches that address directly the dynamics of the resulting recurrent neural circuits are needed to complement the bottom-up analysis. This line of research started with the identification and investigation of so-called canonical microcircuits [1]. Several issues related to cortical microcircuits have also been addressed in the work of Grossberg; see [2] and the references therein. Subsequently it was shown that quite complex real-time computations on spike trains can be carried out by such “given” models for cortical microcircuits ([3–6], see [7] for a review). A fundamental limitation of this approach was that only those computations could be modeled which can be carried out with a fading memory, more precisely only those computations that only require to integrate information over a time span of 200 or 300 ms (its maximal length depends on the amount of noise in the circuit and the complexity of the input spike trains

[8]). In particular, computational tasks that require a representation of elapsed time between salient sensory events or motor actions [9], or an internal representation of expected rewards [10],[11],[12], working memory [13], accumulation of sensory evidence for decision making [14], the updating and holding of analog variables such as for example the desired eye position [15], and differential processing of sensory input streams according to attentional or other internal states of the neural system [16] could not be modeled in this way. Previous work on concrete examples of artificial neural networks [17] and cortical microcircuit models [18] had already indicated that these shortcomings of the model might arise only if one assumes that learning affects exclusively the synapses of readout neurons that project the results of computations to other circuits or areas, without giving feedback into the circuit from which they extract information. This scenario is in fact rather unrealistic from a biological perspective, since pyramidal neurons in the cortex typically have in addition to their long projecting axon a large number of axon collaterals that provide feedback to the local circuit [19]. Abundant feedback connections also exist on the network level between different brain areas [20]. We show in this article that if one takes feedback connections from readout neurons (that are trained for specific tasks) into account, generic cortical microcircuit models can solve all of the previously listed computational tasks. In fact, one can demonstrate this also for circuits whose underlying noise levels and models for neurons and synapses are substantially more realistic than those which had previously been considered in models for working memory and related tasks.

We show in section 2.1 that the significance of feedback for the computational power of neural circuits and other dynamical systems can be explained on the basis of general principles. Theorem 1 implies that a large class of dynamical systems, in particular systems of differential equations which are commonly used to describe the dynamics of firing activity in neural circuits, gain universal computational capabilities for digital and analog computation as soon as one considers them in combination with feedback. A further math-

emtical result (Theorem 2) implies that the capability to process online input streams in the light of non-fading (or slowly fading) internal states is preserved in the presence of fairly large levels of internal noise. On the basis of this theoretical foundation one can explain why the computer models of generic cortical microcircuits, which are considered in section 2.2, are able to solve the previously mentioned benchmark tasks. These results suggest a new computational model for cortical microcircuits, which includes the capability to process online input streams in diverse ways according to different “instructions” that are implemented through high-dimensional attractors of the underlying dynamical system. The high-dimensionality of these attractors results from the fact that only a small fraction of synapses need to be modified for their creation. In comparison with the commonly considered low dimensional attractors, such high-dimensional attractors have additional attractive properties such as compositionality (the intersection of several of them is in general non-empty), and compatibility with real-time computing on online input streams within the same circuit.

The presentation of theoretical results for abstract circuit models in section 2.1 is complemented by mathematical details in sections 4.1 and 4.2 of the Appendix. Details to the computer simulations of more detailed cortical microcircuit models (discussed in section 2.2) can be found in section 4.3 of the Appendix. A discussion of the results of this paper is given in section 3.

2 Results

We consider two types of models for neural circuits:

1. Mean field models, such as those defined by equation (6) in section 2.1, which model the dynamics of firing rates of neurons in neural circuits. These models have the advantage that they are theoretically tractable, but they have the disadvantage that they do not reflect many known details of cortical microcircuits. However we show

that the theoretical results that are proven in section 2.1 hold for fairly large classes of dynamical systems. Hence they potentially also hold for some more detailed models of neural circuits.

2. In section 2.2 we consider quite detailed models of cortical microcircuits consisting of spiking neurons (see the description in section 2.2 and 4.3). At present these models cannot be analyzed directly by theoretical methods, hence we can only present statistical data from computer simulations. Our simulation results show that feedback has in these more detailed models a variety of computational consequences that we have derived analytically for the simpler models of section 2.1. This is not totally surprising insofar, as the computations that we consider in the more detailed models can be approximately described in terms of time-varying firing rates for individual neurons.

In both types of models we focus on computations that transform time-varying input streams into time-varying output streams. The input streams are modeled in section 2.1 by time-varying analog functions $u(t)$ (that might for example represent time-varying firing rates of neurons that provide afferent inputs), and in section 2.2 by spike trains generated by Poisson processes with time-varying rates. Output streams are analogously modeled by time-varying firing rates, or directly by spike trains. We believe that such online computations, which transform time-varying inputs into time-varying outputs, provide a better framework for modeling cortical processing of information than computations that transform a static vector of numbers (i.e., a batch input) into a static output. Mappings from time-varying inputs to time-varying outputs are referred to as filters (or operators) in mathematics and engineering. A frequently discussed reference class of linear and nonlinear filters are those which can be described by Volterra- or Wiener series (see e.g. [21]). These filters can equivalently be characterized as those filters which are time-invariant (i.e., they are input-driven and have no “internal clock”) and have a fading memory (see [5]). Fading memory (which is formally defined in section 4.2.1) means

intuitively that the influence of any specific segment of the input stream on later parts of the output stream becomes negligible when the length of the intervening time interval is sufficiently large. We show in the next two subsections that feedback endows a circuit, that by itself can only carry out computations with fading memory, with flexible ways of combining fading-memory-computations on time varying inputs with computational operations on selected pieces of information in a non-fading memory.

2.1 Theoretical Analysis

The dynamics of firing rates in recurrent circuits of neurons is commonly modeled by systems of nonlinear differential equations of the form

$$x'_i(t) = -\lambda_i x_i(t) + \sigma \left(\sum_{j=1}^n a_{ij} x_j(t) + b_i \cdot v(t) \right), \quad i = 1, \dots, n, \quad (1)$$

or

$$x'_i(t) = -\lambda_i x_i(t) + \sigma \left(\sum_{j=1}^n a_{ij} x_j(t) \right) + b_i \cdot \sigma(v(t)), \quad i = 1, \dots, n \quad (2)$$

([22–25]). Here each $x_i, i = 1, \dots, n$, is a real-valued variable which represents the current firing rate of the i^{th} neuron or population of neurons in a recurrent neural circuit, and $v(t)$ is an external input stream. The coefficients a_{ij}, b_i denote the strengths of synaptic connections, and the $\lambda_i > 0$ denote time constants. The function σ is some sigmoidal activation function (nondecreasing, with bounded range). In most models of neural circuits, the parameters are chosen so that the resulting dynamical system has a fading memory for preceding inputs. If one makes the synaptic connection strengths a_{ij} in (1) or (2) so large that recurrent activity does not dissipate, the neural circuit tends to exhibit persistent memory. But it is usually quite difficult to control the content of this persistent memory, since it tends to be swamped with minor details of external inputs (or initial conditions) from the distant past. Hence this chaotic regime of recurrent neural circuits (see [26] for a review) is apparently also not suitable for biologically realistic online computations

that combine new information from the current input with selected (e.g., behaviorally relevant) aspects of external or internal inputs from the past.

Recurrent circuits of neurons (e.g. those described by equations (1) or (2)) are from a mathematical perspective special cases of dynamical systems. The subsequent mathematical results show that a large variety of dynamical systems, in particular also fading memory systems of type (1) or (2), can overcome in the presence of feedback the computational limitations of a fading memory without necessarily falling into the chaotic regime. In fact, feedback endows them with *universal* capabilities for *analog computing*, in a sense that can be made precise in the following way (see Fig. 1A-C for an illustration):

Theorem 1 *A large class \mathcal{S}_n of systems of differential equations of the form*

$$x'_i(t) = f_i(x_1(t), \dots, x_n(t)) + g_i(x_1(t), \dots, x_n(t)) \cdot v(t), \quad i = 1, \dots, n \quad (3)$$

are in the following sense universal for analog computing:

This system (3) can respond to an external input $u(t)$ with the dynamics of any n^{th} order differential equation of the form

$$z^{(n)}(t) = G(z(t), z'(t), z''(t), \dots, z^{(n-1)}(t)) + u(t) \quad (4)$$

(for arbitrary smooth functions $G : \mathbb{R}^n \rightarrow \mathbb{R}$) if the input term $v(t)$ is replaced in (3) by a suitable memoryless feedback function $K(x_1(t), \dots, x_n(t), u(t))$, and if a suitable memoryless readout function $h(\mathbf{x}(t))$ is applied to its internal state $\mathbf{x}(t) = \langle x_1(t), \dots, x_n(t) \rangle$: one can achieve then that $h(\mathbf{x}(t)) = z(t)$ for any solution $z(t)$ of (4).

Also the dynamic responses of all systems consisting of several higher order differential equations of the form (4) can be simulated by fixed systems of the form (3) with a corresponding number of feedbacks.

This result says more precisely that for any n^{th} order differential equation (4) there exists a (memory-free) feedback function $K : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$ and a memory-free readout

function $h : \mathbb{R}^n \rightarrow \mathbb{R}$ (which can both be chosen to be smooth, in particular continuous) so that, for every external input $u(t), t \geq 0$, and each solution $z(t)$ of the forced system (4) there is an input $u_0(t)$ with $u_0(t) \equiv 0$ for all $t \geq 1$, so that the solution $\mathbf{x}(t) = \langle x_1(t), \dots, x_n(t) \rangle$ of the fixed system (3)

$$\mathbf{x}'(t) = f(\mathbf{x}(t)) + g(\mathbf{x}(t))K(\mathbf{x}(t), u(t) + u_0(t)), \quad \mathbf{x}(0) = \mathbf{0} \quad (5)$$

(for $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ consisting of $\langle f_1, \dots, f_n \rangle$ and $g : \mathbb{R}^n \rightarrow \mathbb{R}^n$ consisting of $\langle g_1, \dots, g_n \rangle$) is such that

$$h(\mathbf{x}(t)) = z(t) \quad \text{for all } t \geq 1.$$

Note that the function $u_0(t)$, that is added to the input for $t < 1$ (whereas $u_0(t) = 0$ for $t \geq 1$), allows the system (3) (and (5)) to simulate with a standardized initial condition $\mathbf{x}(0) = \mathbf{0}$ any solution of (4) with arbitrary initial conditions.

Theorem 1 implies that even if some fixed dynamical system (3) from the class \mathcal{S}_n has fading memory, a suitable feedback K and readout function h will enable it to carry out specific computations with persistent memory. In fact, it can carry out *any* computation with persistent memory which could possibly be carried out by *any* dynamical system (4). To get a clear understanding of this universality property, one should note that the feedback function K and the readout function h depend only on the function G that characterizes the simulated system (4), but not on the external input $u(t)$ or the particular solution $z(t)$ of (4) that it simulates. Hence Theorem 1 implies in particular that any system (3) that belongs to the class \mathcal{S}_n has in conjunction with several feedbacks the computational power of a universal Turing machine (see [27] or [28] for relevant concepts from computation theory). This follows from the fact that every Turing machine (hence any conceivable digital computation, most of which require a persistent memory) can be simulated by systems of equations of the form (4) (this was shown in [29] for the case with continuous time, and in [30, 31] for recurrent neural networks with discrete time; see [32] for a review). But possibly more relevant for applications to biological systems is the fact that any fixed system (3) that belongs to the class \mathcal{S}_n is able to emulate any conceivable

continuous dynamic response to an input stream $u(t)$ if it receives a suitable feedback $K(\mathbf{x}(t), u(t))$, where K can always be chosen to be continuous. Hence one may argue that these systems (3) are also universal for *analog* computing on time-varying inputs.

The class \mathcal{S}_n of dynamical systems that become through feedback universal for analog computing subsumes systems of the form

$$x'_i(t) = -\lambda_i x_i(t) + \sigma \left(\sum_{j=1}^n a_{ij} \cdot x_j(t) \right) + b_i \cdot v(t), \quad i = 1, \dots, n; \quad (6)$$

for example if the λ_i are pairwise different and $a_{ij} = 0$ for all i, j , and all b_i are nonzero. Fewer restrictions are needed if more than one feedback to the system (6) can be used. Systems of the form (1) or (2) are of a slightly different form, since there the activation function σ (that has a bounded range) is applied to the term $v(t)$. But such systems (1), (2) can still be universal for all *bounded* analog responses of arbitrary dynamical systems (4), which are arguably the only ones of interest in a biological context. This follows from the fact that if the external input $u(t)$ of the system (4), as well as the resulting solution $z(t)$ and its derivatives $z^{(i)}(t)$ for $i \leq n - 1$, stay within some bounded range, then the values of the feedback $v(t)$ that is needed for the simulation of (4) by (3) will also stay within a bounded range. More precisely, one has that:

For each constant $c > 0$ there is a constant $C > 0$ such that: for every external input $u(t), t \geq 0$, and each solution $z(t)$ of the forced system (4) such that

$$|u(t)| \leq c \text{ and } |z^{(i)}(t)| \leq c \quad \text{for all } i = 0, \dots, n - 1, \text{ for all } t \geq 0$$

the input u_0 can be picked so that the feedback

$$v(t) = K(\mathbf{x}(t), u(t) + u_0(t)) \quad t \geq 0$$

to (1) or (2) satisfies:

$$|v(t)| \leq C \quad \text{for all } t \geq 0.$$

Thus, if we know *a priori* that we will only deal with solutions of the differential equation (4) that are bounded by c , and inputs are similarly bounded, we could also consider

instead of (3) a system such as $\mathbf{x}'(t) = f(\mathbf{x}(t)) + g(\mathbf{x}(t))\sigma(v(t))$ with $f, g : \mathbb{R}^n \rightarrow \mathbb{R}^n$, where some bounded activation function $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ (e.g. $q \cdot \tanh(v)$, for a suitable constant q) is applied to the term $v(t)$ (like in (2)). The resulting feedback term $\sigma(K(\mathbf{x}(t), u(t) + u_0(t)))$ is then of a mathematical form which is adequate for modeling feedback in neural circuits.

The **proof** of Theorem 1 builds on results from control theory. One important technique in nonlinear control is *feedback linearization* ([33], [34]). With this technique a large class of nonlinear dynamical systems can be transformed through suitable feedback into a linear system (which is then much easier to control). It should be pointed out that this feedback linearization is not a standard linearization method that only yields approximation results, but a method that yields an exact transformation. More generally, one can show in various cases that two dynamical systems D_1 and D_2 are *feedback equivalent*. The notion of “feedback equivalence” (see section 4.1.1 for a definition), which is in fact an equivalence relation, expresses that two systems of differential equations can be transformed into each other through application of a suitable feedback and a change of basis in the state space. Such change of basis can be achieved through readout functions $h(\mathbf{x}(t))$ as considered in the claim of Theorem 1. Thus, in order to show that a fixed system D_1 has the universality property that is specified in the claim of Theorem 1, it suffices to show that D_1 is feedback equivalent to all systems of the form (4). Known results about feedback linearization (see [34], Lemma 5.3.5) imply that the following linear system (7) is an example of a system D_1 (consisting of n differential equations) which has this universality property:

$$\mathbf{x}'(t) = \mathbf{A}_n \mathbf{x}(t) + \mathbf{b}_n v(t) \tag{7}$$

with

$$\mathbf{A}_n := \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix} \quad \mathbf{b}_n := \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

It is in fact very easy to see that any system (4) can be transformed into the system (7) with the help of feedback: Set $x_1(t) = z(t)$, $x_{i+1}(t) = z^{(i)}$ for $i = 1, \dots, n-1$, and use the feedback $v(t) = G(\mathbf{x}(t)) + u(t)$ in (7). In order to prove that many other dynamical systems have the same universality property as this system (7), it suffices to observe that feedback equivalence preserves this universality property.

We define the class \mathcal{S}_n in the claim of Theorem 1 as the class of *feedback linearizable* systems, that is, the class of dynamical systems (3) that are feedback equivalent to some generic linear system. It can be proved (see Lemma in section 4.1.2) that every feedback linearizable system (3) is also feedback equivalent to (7), and hence has the the same universality property as (7).

We give in section 4.1.2 a precise definition of the class \mathcal{S}_n in terms of feedback equivalence (which is formally defined in section 4.1.1). We present in section 4.1.3 a formal proof of the simulation result that is claimed in Theorem 1 (taking also initial conditions into account). In addition we formulate in section 4.1.5 an equivalent criterion for a system (3) to belong to the class \mathcal{S}_n , which can be more easily tested for concrete cases of dynamical systems. This criterion makes use of the Lie bracket formalism that is briefly reviewed in section 4.1.4. Applications of this criterion to neural network equations are discussed in section 4.1.6. In particular, we use this criterion to show that some dynamical systems (6) that are defined by standard equations for recurrent neural circuits belong to the class \mathcal{S}_n . We also show in section 4.1.6 that not all systems of the form (6) belong to the class \mathcal{S}_n , rather it depends on the particular choice of parameters a_{ij} and b_i in (6).

Theorem 1 implies that a generic neural circuit may become through feedback a universal computational device, which can not only simulate any Turing machine, but also any conceivable model for analog computing with bounded dynamic responses. The “program” of such arbitrary simulated computing machine gets encapsulated in the static functions K that characterize the memoryless computational operations that are required from feedback units, and the static readout functions h . Since these functions are static, i.e. time-invariant, and continuous, they provide suitable targets for *learning*. More precisely, in order to train a generic neural circuit to simulate the dynamic response of an arbitrary dynamical system, it suffices to train - apart from readout neurons - a few neurons within the circuit (or within some external loop) to transform the vector $\mathbf{x}(t)$, that represents the current firing activity of its neurons, and the current external input $u(t)$ into a suitable feedback value $K(\mathbf{x}(t), u(t))$. This could for example be carried out by training a suitable feedforward neural network within the larger circuit, which can approximate any continuous feedback function K [35]. Furthermore we will show in section 2.2 that these feedback functions K can in many biologically relevant cases be chosen to be linear, so that it would in principle suffice to train a single neuron to compute K .

It is known that the memory capacity of such circuit is reduced to some finite number of bits if these feedback functions K are not learnt perfectly, or if there are other sources of noise in the system. More generally, no analog circuit with noise can simulate arbitrary Turing machines [36]. But the subsequent Theorem 2 shows that fading memory systems with noise and imperfect feedback can still achieve the maximal possible computational power within this a-priori limitation: they can simulate any given finite state machine (FSM). Note that any Turing machine with tapes of finite length is a special case of a FSM. Furthermore any existing digital computer is a FSM, hence the computational capability of FSM’s is actually quite large.

In order to avoid the cumbersome mathematical difficulties that arise when one analyses differential equations with noise, we formulate and prove Theorem 2 on a more

abstract level, resorting to the notion of fading memory filters with noise (see section 4.2 for details). We assume here that the input-output behavior of those dynamical systems with noise, for which we want to determine the computational impact of (imprecise) state feedback, can be modeled by fading memory filters with additive noise on their output. The assumption that the amplitude of this noise is bounded is a necessary assumption according to [37]. We refer to [4], [5], [38] for further discussions of the relationship between models for neural circuits and fading memory filters. In particular it was shown in [38] that every time-invariant fading memory filter can be approximated by models for neural circuits, provided that these models reflect the empirically found diversity of time constants of neurons and synapses.

Theorem 2 *Feedback allows linear and nonlinear fading memory systems, even in the presence of additive noise with bounded amplitude, to employ for real-time processing of time-varying inputs the computational capability and non-fading states of any given FSM (see Fig. 1D-E).*

A precise formalization of this result is formulated as Theorem 5 in section 4.2.3, and a formal proof of Theorem 5 is given in section 4.2.4. The external input $u(t)$ can in this case be injected directly into the fading memory system, so that the feedback $K(\mathbf{x}(t))$ depends only on the internal state $\mathbf{x}(t)$ (see Fig. 1E). One essential ingredient of the proof is a method for making sure that noise does not get amplified through feedback: the functions K that provide feedback values $K(\mathbf{x}(t))$ can be chosen in such a way that they cancel the impact of imprecision in the values $K(\mathbf{x}(s))$ for immediately preceding time steps $s < t$.

2.2 Applications to Generic Cortical Microcircuit Models

We examine in this section computational aspects of feedback in recurrent circuits of spiking neurons that are based on data from cortical microcircuits. The dynamics of

these circuits is substantially more complex than the dynamics of circuits described by equ. (6), since it is based on action potentials (spikes) rather than firing rates. Hence one can expect at best that the temporal dynamics of firing rates in these circuits of spiking neuron is qualitatively similar to that of circuits described by (6).

The preceding theoretical results imply that it is possible for dynamical systems to carry out computations with persistent memory without acquiring all the computational disadvantages of the chaotic regime, where the memory capacity of the system is dominated by noise. Feedback units can create selective “loopholes” into the fading memory dynamics of a dissipative system, that can only be activated by specific patterns in the input or circuit dynamics. In this way the potential content of persistent memory can be controlled by feedback units that have been trained to recognize such patterns. This feedback may arise from a few neurons within the circuit, or from neurons within a larger feedback loop. The task to approximate a suitable feedback function K is less difficult than it may appear on first sight, since it suffices in many cases to approximate a *linear* feedback function. The reason is that sufficiently large generic cortical microcircuit models have an inherent kernel property [8], in the sense of machine learning [39]. This means that a large reservoir of diverse nonlinear responses to current and recent input patterns is automatically produced within the recurrent circuit. In particular, nonlinear combinations of variables a, b, c, \dots (that may result from the circuit input or internal activity) are automatically computed at internal nodes of the circuit. Consequently numerous low degree polynomials in these variables a, b, c, \dots can be approximated by *linear* combinations of outputs of neurons from the recurrent circuit. An example of this effect is demonstrated in Fig. 3G, where it is shown that the product of firing rates $r_3(t)$ and $r_4(t)$ of two independently varying afferent spike train inputs can be approximated quite well by a linear readout neuron. The kernel property of biologically realistic cortical microcircuit models is apparently supported by the fact that these circuits have many additional nonlinearities besides those that appear in the equations (1), (2), (6).

One formal difference between neurons in the mean field model (6) and more realistic models for spiking neurons is that the input to a neuron of the latter type consists of postsynaptic potentials, rather than of firing rates. Hence the time-varying input $\mathbf{x}(t)$ to a readout neuron is in this section not a vector of time-varying firing rates, but a smoothed version of the spike trains of all presynaptic neurons. This smoothing is achieved through application of a linear filter with an exponentially decaying kernel, whose time constant of 30 ms models time constants of receptors and postsynaptic membrane of a readout neuron in a qualitative fashion. Thus, if \mathbf{w} is a vector of synaptic weights, then $\mathbf{w} \cdot \mathbf{x}(t)$ models the impact of the firing activity of presynaptic neurons on the membrane potential of a readout neuron.

We refer in the following to those neurons where the weights of synaptic connections from neurons within the circuit are adapted for a specific computational task (rather than chosen randomly from distributions that are based on biological data, like for all other synapses in the circuit) as *readout neurons*. The output of a readout neuron was modeled in most of our simulations simply by a weighted sum $\mathbf{w} \cdot \mathbf{x}(t)$ of the previously described vector $\mathbf{x}(t)$. Such output can be interpreted as the time-varying firing rate of a readout neuron. However we show in Fig. 3 that these readout neurons can (with a moderate loss in performance) also be modeled by spiking neurons, exactly like the other neurons in the simulated circuit. This demonstrates that not only those circuits that receive feedback from external readout neurons, but also generic recurrent circuits in which a few neurons have been trained for a specific task, acquire computational capabilities for real-time processing that are not restricted to computations with fading memory.

Theorem 2 suggests that the training of a few of its neurons enables generic neural circuits to employ persistent internal states for state-dependent processing of online input streams. Previous models for non-fading memory in neural circuits [13, 40–42] proposed that it is implemented through low-dimensional attractors in the circuit dynamics. These attractors tend to freeze or entrain the whole state of the circuit, and thereby shut it

off from the online input stream (although independent local attractors could emerge in local subcircuits under some conditions [41]). In contrast, the generation of non-fading memory through a few trained neurons does not entail that the dynamics of the circuit is dominated by their persistent memory states. For example, when a readout neuron gives during some time interval a constant feedback $K(\mathbf{x}(t)) = c$, this only constrains the circuit state $\mathbf{x}(t)$ to remain in the sub-manifold $\{\mathbf{x} : K(\mathbf{x}) = c\}$ of its high-dimensional state space. This sub-manifold is in general high-dimensional. In particular, if $K(\mathbf{x})$ is a linear function $\mathbf{w} \cdot \mathbf{x}$, which often suffices as we will show, the dimensionality of the sub-manifold $\{\mathbf{x} : K(\mathbf{x}) = c\}$ differs from the dimension of the full state space only by 1. Hence several such sub-manifolds have in general a high-dimensional intersection, and their intersection still leaves sufficiently many degrees of freedom for the circuit state $\mathbf{x}(t)$ to also absorb continuously new information from online input streams. These sub-manifolds are in general not attractors in a strict mathematical sense. Rather, their effective attraction property (or noise-robustness) results from the subsequently described training process (“teacher forcing”). This training process produces weights \mathbf{w} which have the property that the resulting feedback $\mathbf{w} \cdot \tilde{\mathbf{x}}(t)$ moves a trajectory of circuit states that goes through states $\tilde{\mathbf{x}}(t)$ in the neighborhood of the sub-manifold $\{\mathbf{x} : K(\mathbf{x}) = c\}$ closer to this sub-manifold.

We simulated generic cortical microcircuit models consisting of 600 integrate-and-fire (I&F) neurons (for Fig. 3, 4), and circuits consisting of 600 Hodgkin-Huxley (HH) neurons (for Fig. 5), in either case with a rather high level of noise that reflects experimental data on the high conductance state in vivo [43]. These circuits were not constructed for any particular computational task. In particular, sparse synaptic connectivity between neurons was generated (with a biologically realistic bias towards short connections) by a probabilistic rule. Synaptic parameters were chosen randomly from distributions that depend on the type of pre- and postsynaptic neurons (in accordance with empirical data from [44], [45]). More precisely, we used biologically realistic models for dynamic synapses

whose individual mixture of paired-pulse depression and facilitation (depending on the type of pre- and postsynaptic neuron) was based on these data. It has previously been shown in [8],[6] that the presence of such dynamic synapses extends the time span of the inherent fading memory of the circuit. However the computational tasks that are considered in this paper require, apart from a non-fading memory, only a fading memory with a rather short time span (in order to make the estimation of the current firing rate of input spike trains feasible). Therefore the biologically more realistic dynamic synapses could be replaced in this model by simple static synapses, without a change in the performance of the circuit for the subsequently considered tasks. All details of the simulated microcircuit models can be found in section 4.3. Details of the subsequently discussed computer experiments are given in sections 4.4 - 4.6.

We tested 3 different types of computational tasks for generic neural circuits with feedback. The same neural circuit can be used for each task, only the organization of input- and output streams needs to be chosen individually (see Fig. 2). The following procedure was applied to train readout neurons, i.e. to adjust the weights of synaptic connections from neurons in the circuit to readout neurons for specific computational tasks (while leaving all other parameters of the generic microcircuit model unchanged):

- First those readout neurons were trained that provide feedback, then the other readout neurons.
- During the training of readout neurons that provide feedback, their actual feedback was replaced by a *noisy* version of their target output (“teacher forcing”).
- Each readout neuron was trained by linear regression to output at any time t a particular target value $f(t)$. Linear regression was applied to a set of data points of the form $\langle \mathbf{x}(t), f(t) \rangle$ for many time points t , where $\mathbf{x}(t)$ is a smoothed version of the spike trains of presynaptic neurons (as defined before).

Note that teacher forcing with noisy versions of target feedback values trains these read-

outs to correct errors resulting from imprecision in their preceding feedback (rather than amplifying errors). This training procedure is responsible for the robustness of the dynamics of the resulting closed-loop circuits, in particular for the “attractor” properties of the effectively resulting high-dimensional attractors.

In our first computer experiment, readout neurons were trained to turn a high-dimensional attractor on or off (Fig. 3D), in response to bursts in 2 of the 4 independent input spike trains. More precisely, 8 neurons were trained to represent in their firing activity at any time the information in which of the input streams 1 or 2 a burst had most recently occurred. If it had occurred most recently in stream 1, they were trained to fire at 40 Hz, and if a burst had occurred most recently in input stream 2, they were trained not to fire. Hence these neurons were required to represent the non-fading state of a simple FSM, demonstrating in an example the computational capabilities predicted by Theorem 2. Fig. 3G demonstrates that the circuit retains its kernel property in spite of the feedback injected into the circuit by these readouts. But beyond the emulation of a simple FSM, the resulting generic cortical microcircuit is able to combine information stored in the current state of the FSM with new information from the online circuit input. For example, Fig. 3E shows that other readouts from the same circuit can be trained to amplify their response to specific inputs if the high-dimensional attractor is in the “on”-state. Readouts can also be trained to change the function that they compute if the high-dimensional attractor is in the on-state (Fig. 3F). This provides an example for an online reconfigurable circuit. The readout neurons that provide feedback had been modeled in this computer simulation like the other neurons in the circuit: by I&F neurons with in-vivo like background noise. Hence they can be viewed equivalently as neurons *within* an otherwise generic circuit.

Another difficult problem in computational neuroscience is to explain how neural circuits can implement a parametric memory, i.e. how they can hold and update an *analog* value, that may represent for example an intended eye-position that a neural integra-

tor computes from a sequence of eye-movement commands [46], an estimate of elapsed time [9], or accumulated sensory evidence [14]. Various designs have been proposed for parametric memory in recurrent circuits, where continuous attractors (also referred to as line attractors) hold and update an analog value. But these approaches are inherently brittle [42], and have problems in dealing with high noise or online circuit inputs. On the other hand Fig. 4 shows that dedicated circuit constructions are not necessary, since feedback from readout neurons in *generic* cortical microcircuits models can also create high-dimensional attractors that hold and update an *analog* value for behaviorally relevant time spans. In fact, due to the high-dimensional character of the resulting high-dimensional attractors, two such analog values can be stored and updated independently (Fig. 4C,D), even within a fairly small circuit. In this example the readouts that provide feedback were simply trained to increase or reduce their feedback at each time point. Note that the resulting circuit activity is qualitatively consistent with recordings from neurons in cortex and striatum during reward expectation [10],[11],[12]. A similar ramp-like rise and fall of activity as shown in panels C, D, F has also been recorded in neurons of posterior parietal cortex of the macaque in experiments where the monkey had been trained to classify the duration of elapsed time [9]. The high-dimensionality of the continuous attractors in this model makes it feasible to constrain the circuit state to stay simultaneously in more than one continuous attractor, thereby making it in principle possible to encode complex movement plans that require specific temporal relationships between individual motor commands.

Our model for parametric memory in cortical circuits is consistent with high noise: Fig. 5G shows the typical trial-to-trial variability of a neuron in our simulated circuit of HH neurons with in-vivo like background noise. It qualitatively matches the “wide diversity of neural firing drift patterns in individual fish at all states of tuning” that was observed in the horizontal oculomotor neural integrator in goldfish [15], and the large trial-to-trial variability of neurons in prefrontal cortex of monkeys reported in [10]. In

addition, this model is consistent with the surprising plasticity that has been observed even in quite specialized neural integrators [15], since continuous attractors can be created or modified in this model by changing just a few synaptic weights of neurons that are immediately involved. It does not require the presence of long-lasting postsynaptic potentials, NMDA-receptors, or other specialized details of biological neurons or synapses, although their inclusion in the model is likely to provide additional temporal stability [13]. Rather it points to complementary organizational mechanisms on the circuit level, that are likely to enhance the controllability and robustness of continuous attractors in neural circuits. The robustness of this learning-based model can be traced back to the fact that readout neurons can be trained to correct undesired circuit responses resulting from errors in their previous feedback. Furthermore such error correction is not restricted to linear computational operations, since the previously demonstrated kernel property of these generic circuits allows even linear neurons to implement complex nonlinear control strategies through their feedback. As an example we demonstrate in Fig. 5 that even under biologically realistic high noise conditions a linear readout can be trained to update a continuous attractor (Fig. 5D), to filter out input activity during certain time intervals in dependence of the current state of the continuous attractor (Fig. 5E), or to combine the time-varying analog variable encoded by the current state $CA(t)$ of the continuous attractor with a time-varying variable $r_1(t)$ that is delivered by an online spike input. Hence intention-based information processing [16] and other tasks that involve a merging of external inputs and internal state information can be implemented in this way. Fig. 5C shows that a high-dimensional attractor need not entrain the firing activity of neurons in a drastic way, since it just restricts the high-dimensional circuit dynamics $\mathbf{x}(t)$ to a slightly lower dimensional manifold of circuit states $\mathbf{x}(t)$ that satisfy $\mathbf{w} \cdot \mathbf{x}(t) = f(t)$ for the current target output $f(t)$ of the corresponding linear readout. On the other hand Fig. 5E shows that the activity level $CA(t)$ of the high-dimensional attractor can nevertheless be detected by other linear readouts, and can simultaneously be combined in a nonlinear

manner with a time-varying variable $r_2(t)$ from one afferent circuit input stream, while remaining invariant to the other afferent input stream.

Finally, the same generic circuit also provides a model for the integration of evidence for decision making that is compatible with in-vivo like high noise conditions. Fig. 5H depicts the time course of the same neural integrator as in panel D, but here for the case where the rates r_1, r_2 of the 2 input streams assume in 8 trials 8 different constant values after the first 100 ms (while assuming a common value of 65 Hz during the first 100 ms). The resulting time course of the continuous attractor is qualitatively similar to the meandering path towards a decision threshold that has been recorded from neurons in area LIP where firing rates represent temporally integrated evidence concerning the dominating direction of random dot movements (see Fig. 5A in [14]).

3 Discussion

We have presented a theoretically founded model for real-time computations on complex input streams with persistent internal states in generic cortical microcircuits. This model does not require a handcrafted circuit structure or biologically unrealistic assumptions such as symmetric weight distributions, static synapses that do not exhibit pair-pulsed depression or facilitation, or neuron models with low levels of noise that are not consistent with data on in-vivo conditions. Our model only requires the assumption that adaptive procedures (synaptic plasticity) in generic neural circuits can approximate linear regression. Furthermore, in contrast to classical learning paradigms for attractor neural networks, it is here not required that a large fraction of synaptic parameters in the circuit are changed when a new computational task is introduced, or a new item is stored in working memory. Rather, it suffices if those neurons that provide the circuit output and a few neurons that provide feedback are subject to synaptic plasticity. Such minimal circuit modifications have the advantage that thereby created attractors of the circuit dynamics are high-dimensional. We have shown that the circuit state can be simultaneously

in several of such high-dimensional attractors, and still retain sufficiently many degrees of freedom to absorb and process new information from online input streams. In particular, we have shown in Fig. 3 and 5 how bottom-up processing can be reconfigured in dependence of discrete internal states (implemented through high-dimensional attractors) by turning certain input channels on or off, and by changing the computational operations that are applied to input variables. Furthermore we have shown in Fig. 5 that analog variables, which are extracted from an online input stream, can be combined in real-time computations with analog variables that are stored in high-dimensional continuous attractors. This provides in particular a model for the implementation of intention-based information processing [16] in cortical microcircuits.

It remains open how learning signals can induce neurons in a biological organism to compute specific linear feedback functions. But at least we have reduced this problem to the feasibility of perceptron-like learning (or more abstractly: to linear regression) for single neurons. Subsequent research will have to determine whether these learning requirements (which can partially be reduced to spike-timing dependent plasticity [47]) can be justified on the basis of results on unsupervised learning and reinforcement learning [48] in biological organisms.

Whereas it was previously already known that one can construct specific circuits that have universal computational capabilities for real-time computing on analog input streams, Theorems 1 and 2 of this article imply that a large variety of dynamical systems (in particular generic cortical microcircuits) can acquire through feedback such universal capabilities for computations that map time-varying inputs to time-varying outputs. It should be noted that these universal computational capabilities differ from the well known but much weaker universal approximation property of feedforward neural networks (see [35]), since not only the static output of an arbitrary continuous static function is approximated, but the dynamic response of arbitrary differential equations of higher order to time-varying inputs.

The theoretical results of this article also provide an explanation for the astounding computational capability and flexibility of echo state networks [17]. In addition they can be used to analyze computational aspects of feedback in other biological dynamical systems besides neural circuits. Several such systems, for example genetic regulatory networks, are known to implement complex maps from time-varying input streams (e.g. external signals) onto time-varying outputs (e.g. transcription rates). But little is known about the way in which these maps are implemented. Whereas feedback in biological dynamical systems is usually only analyzed and modeled from the perspective of control, we propose that an analysis of its computational aspects is likely to yield a better understanding of the computational capabilities of such systems.

4 Appendix

4.1 Mathematical Definitions, Details to the Proof of Theorem 1, and Examples

4.1.1 Definition of Feedback Equivalence

We recall that a *smooth* mapping is one for which derivatives of all orders exist (infinite differentiability), and that a *diffeomorphism* $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a smooth mapping for which there exists a well-defined smooth inverse $T^{-1} : \mathbb{R}^n \rightarrow \mathbb{R}^n$.

Definition (see [34], Def. 5.3.1): Two n -dimensional systems $\mathbf{x}' = f(\mathbf{x}) + g(\mathbf{x})v$ and $\mathbf{x}' = \tilde{f}(\mathbf{x}) + \tilde{g}(\mathbf{x})v$ (with smooth vector fields $f = \langle f_1, \dots, f_n \rangle, g = \langle g_1, \dots, g_n \rangle, \tilde{f} = \langle \tilde{f}_1, \dots, \tilde{f}_n \rangle, \tilde{g} = \langle \tilde{g}_1, \dots, \tilde{g}_n \rangle$) are called *feedback equivalent* (over the state space \mathbb{R}^n) if there exists

- a diffeomorphism $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$, and
- smooth maps $\alpha, \beta : \mathbb{R}^n \rightarrow \mathbb{R}$ with $\beta(\mathbf{x}) \neq 0$ for all $\mathbf{x} \in \mathbb{R}^n$,

such that, for each $\mathbf{x} \in \mathbb{R}^n$

$$T_*(\mathbf{x})(f(\mathbf{x}) + g(\mathbf{x})\alpha(\mathbf{x})) = \tilde{f}(T(\mathbf{x}))$$

and

$$\beta(\mathbf{x})T_*(\mathbf{x})g(\mathbf{x}) = \tilde{g}(T(\mathbf{x}))$$

(where T_* denotes the Jacobian of T).

4.1.2 Definition of the class \mathcal{S}_n

Recall that a linear system $\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{b}u$ is *controllable* if it is possible to drive any state \mathbf{x}_0 to any other state \mathbf{x}_1 using an input (see [34], Def. 3.1.6). Controllability is a generic property of systems, and amounts to the requirement that the matrix $(\mathbf{b}, \mathbf{A}\mathbf{b}, \dots, \mathbf{A}^{n-1}\mathbf{b})$ has full rank, where n is the dimension of the system (see [34], Theorem 2). Note that the linear system (7) satisfies this requirement, and hence is controllable.

We take \mathcal{S}_n to be the class of n -dimensional systems (3) that are (*globally*) *feedback linearizable*, that is to say, the systems (3) for which there exists some linear controllable system that is feedback equivalent to (3) (see [34], Def. 5.3.2).

An n -dimensional system is feedback linearizable if and only if it is feedback equivalent to the system (7) (see [34], Lemma 5.3.5). Therefore, we have the following:

Lemma: *A system (3), with smooth vector fields $f = \langle f_1, \dots, f_n \rangle$ and $g = \langle g_1, \dots, g_n \rangle$, belongs to \mathcal{S}_n if and only if there exists a diffeomorphism $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and two smooth maps $\alpha, \beta : \mathbb{R}^n \rightarrow \mathbb{R}$, with $\beta(\mathbf{x}) \neq 0$ for all $\mathbf{x} \in \mathbb{R}^n$, such that, for each $\mathbf{x} \in \mathbb{R}^n$:*

$$T_*(\mathbf{x})f(\mathbf{x}) = \mathbf{A}_n T(\mathbf{x}) - \frac{\alpha(\mathbf{x})}{\beta(\mathbf{x})}\mathbf{b}_n \tag{8}$$

and

$$\beta(\mathbf{x})T_*(\mathbf{x})g(\mathbf{x}) = \mathbf{b}_n, \tag{9}$$

where T_* denotes the Jacobian of T and

$$\mathbf{A}_n := \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix} \quad \mathbf{b}_n := \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

An interpretation of the property given in the above Lemma, that will be used in the proof of Theorem 1 in section 4.1.3, is as follows (see [34], Chapter 5, for more discussion): For each input $\mu(t)$ and each solution $z(t)$ of

$$z^{(n)} = \mu,$$

the vector function $\mathbf{x}(t) = T^{-1}(Z(t))$ satisfies (3) with the input $v(t) = \alpha(\mathbf{x}(t)) + \beta(\mathbf{x}(t))\mu(t)$, where

$$Z(t) = (z(t), z'(t), z''(t), \dots, z^{(n-1)}(t)).$$

4.1.3 Details to the Proof of Theorem 1

In this section, we prove the simulation result that is claimed in Theorem 1.

Take any system (3) in \mathcal{S}_n and any system (4) to be simulated. Using T, α, β as in the Lemma in section 4.1.2 that characterizes the class \mathcal{S}_n , we define:

$$K(\mathbf{x}, w) := \alpha(\mathbf{x}) + \beta(\mathbf{x}) [G(T(\mathbf{x})) + w]$$

and we let $h(\mathbf{x})$ be the first coordinate of $T(\mathbf{x})$. In the special case where (3) describes the dynamics of a circuit according to (6), α is a linear function, β is a constant, and T is an invertible linear map from \mathbb{R}^n to \mathbb{R}^n .

Next, pick an external input $u(t), t \geq 0$, and a solution $z(t)$ of the forced system (4).

From the interpretation of feedback linearization given earlier (in the last part of section 4.1.2), it follows that for any inputs $u(t)$ and $u_0(t)$ (in particular, one could take $u_0 \equiv 0$), and each solution $z(t)$ of

$$z^{(n)}(t) = G(z(t), z'(t), z''(t), \dots, z^{(n-1)}(t)) + u(t) + u_0(t)$$

(that is, we use $\mu(t) = G(z(t), z'(t), z''(t), \dots, z^{(n-1)}(t)) + u(t) + u_0(t)$ as the input to $z^{(n)} = \mu$), the vector function $\mathbf{x}(t) = T^{-1}(Z(t))$ satisfies (3) with input

$$v(t) = \alpha(\mathbf{x}(t)) + \beta(\mathbf{x}(t))\mu(t) = K(\mathbf{x}(t), u(t) + u_0(t)) .$$

Furthermore, $Z(t) = T(\mathbf{x}(t))$ means that $z(t) = h(\mathbf{x}(t))$, as required for the notion of simulation.

This almost proves the simulation result, except for the fact that there is no reason for the initial value $\mathbf{x}(0) = T^{-1}(Z(0))$ to be zero, since $z(t)$ is an arbitrary trajectory. This is where the input u_0 plays a role. Let $\xi := T(0)$. We will show that, given any solution $z(t)$ and any input $u(t)$, there is some input $u_0(t)$, with $u_0(t) \equiv 0$ for all $t \geq 1$, so that the solution of

$$y^{(n)}(t) = G(y(t), y'(t), y''(t), \dots, y^{(n-1)}(t)) + u(t) + u_0(t) \tag{10}$$

with $y(0) = \xi$ has the property that $y(t) = z(t)$ for all $t \geq 1$. (Where $z(t)$ is the desired trajectory to be simulated, with $u_0 \equiv 0$.) Then letting $\mathbf{x}(t) = T^{-1}(Y(t))$ instead of $T^{-1}(Z(t))$ means that $\mathbf{x}(0) = 0$ and still $h(\mathbf{x}(t)) = y(t) = z(t)$ for all $t \geq 1$.

Consider now an arbitrary solution $z(t)$ of the equation (4) and let ζ be the vector with entries

$$\zeta_{i+1} := z^{(i)}(1), \quad i = 0, \dots, n-1.$$

We next pick a scalar differentiable function ϕ such that $\phi^{(i)}(0) = \xi_{i+1}$ and $\phi^{(i)}(1) = \zeta_{i+1}$ for $i = 0, \dots, n-1$. (It is easy to see that such functions exist. For example, one may simply consider the linear system $\mathbf{p}' = \mathbf{A}_n \mathbf{p} + \mathbf{b}_n q$ with states \mathbf{p} and input q . This is a

completely controllable linear system, cf. [34], Chapter 3, so we just pick an input $q(t)$ which steers ξ into ζ , and finally let $\phi(t)$ be the first coordinate of $\mathbf{p}(t)$.) Now we let

$$u_0(t) := \phi^{(n)}(t) - G(\phi(t), \dots, \phi^{(n-1)}(t)) - u(t)$$

for $t < 1$, and $u_0(t) \equiv 0$ for $t \geq 1$, and claim that the solution of (10) with $y(0) = \xi$ has the property that $y(t) = z(t)$ for all $t \geq 1$. Since $u(t) + u_0(t) = u(t)$ for all $t \geq 1$, we only need to show that $y^{(i)}(1) = z^{(i)}(1)$ for every $i = 0, \dots, n-1$. To see this, in turn, and using uniqueness of solutions of differential equations, it is enough to show that $y(t) := \phi(t)$ satisfies

$$\phi^{(n)}(t) = G(\phi(t), \phi'(t), \phi''(t), \dots, \phi^{(n-1)}(t)) + u(t)$$

on the interval $[0, 1]$ and has derivatives at $t = 0$ as specified by the vector ξ . But this is indeed true by construction.

Finally, we remark that if $|u(t)| \leq c$ and $|z^{(i)}(t)| \leq c$ for all $t \geq 0$ then $\mathbf{x}(t) = T^{-1}(Z(t))$ is bounded in norm by a constant that only depends on c (since T^{-1} is continuous, by definition of diffeomorphism), and the numbers $b_i := z^{(i)}(1)$ are also bounded by a constant that depends only on c , so $K(\mathbf{x}(t), u(t) + u_0(t))$ also is.

Corollary 3 *Analogous results can be shown for the simulation of systems consisting of any number k of higher order differential equations as in (4). In this case fixed systems of first order differential equations of a form as in (3), but with k memoryless feedback functions K_1, \dots, K_k that depend on the simulated higher order system, can be shown to be able to simulate the dynamic response of arbitrary higher order systems of differential equations.*

4.1.4 Lie Brackets

The study of controllability and other properties of nonlinear systems is based upon the use of Lie bracket formalism and theory ([34], Chapter 4). We need this formalism to

show in section 4.1.6 that the class \mathcal{S}_n includes some neural networks of the form (6). For any two vector fields f and g ,

$$[f, g] = g_*f - f_*g$$

denotes the *Lie bracket* of f and g . Recall that the Lie bracket of two vector fields is a vector field that characterizes the effective direction of movement obtained by performing this “commutator” motion: follow the vector field f for t time steps, then g for t time steps, then f backward in time for t time steps, and finally g backward in time for t time steps, for small $t > 0$. To be more precise, denote formally by e^{tf} the flow associated to f , and similarly for g . Consider the following curve, for any initial state x_0 :

$$\gamma(t) := e^{-\sqrt{t}g}e^{-\sqrt{t}f}e^{\sqrt{t}g}e^{\sqrt{t}f}x_0.$$

Applying repeatedly this expansion:

$$e^{tf}x_0 = x(0) + tx'(0) + \frac{t^2}{2}x''(0) + o(t^2) = x_0 + tf(x_0) + \frac{t^2}{2}f_*(x_0)f(x_0) + o(t^2)$$

(and similarly for g), we obtain that

$$e^{-\sqrt{t}g}e^{-\sqrt{t}f}e^{\sqrt{t}g}e^{\sqrt{t}f}x_0 = e^{t[f,g]}x_0 + o(t)$$

as $t \rightarrow 0$, from which it follows that $\gamma'(0) = [f, g](x_0)$, which means that the direction of $[f, g]$ is followed when performing the commutator motions. Using the possible non-commutativity of the vector fields, one generates in this manner genuinely new directions of movement in addition to those provided by the linear combinations of f and g . Well-known examples are provided by the Lie bracket of two rotations around orthogonal axes, which is a rotation around the remaining axis (see for example [34], page 150), or the motions involved in parking an automobile (see for example [34], Example 4.3.13).

Iterations of Lie brackets play a key role. Let us introduce, for any given vector field f , the operator ad_f , which maps vector fields into vector fields, by means of the formula $\text{ad}_f(g) := [f, g]$. Iterations of the operator ad_f are defined in the obvious way: $\text{ad}_f^0(g) = g$ and $\text{ad}_f^{k+1}(g) = \text{ad}_f(\text{ad}_f^k(g))$.

It is also useful to consider an operator L_f that acts on scalar functions. We use the notation $L_f\phi$, for any (smooth) vector field f and (smooth) function ϕ , to denote the Lie derivative of ϕ along f , that is, $\nabla\phi \cdot f$. The function $L_f\phi$, which is again smooth, is nothing more than the *directional derivative* of the function ϕ in the direction of the vector field f , in the sense of elementary calculus. One can also consider iterated applications of the operator L_f .

4.1.5 A Characterization of \mathcal{S}_n via Lie Brackets

With these notations, we are ready to present a Lie geometric characterization of the class \mathcal{S}_n . The next theorem follows by combining the proofs of Proposition 5.3.9 and of Theorem 15 in [34] (with $\mathcal{O} = \mathcal{X} = \mathbb{R}^n$ in the notations of that text).

Theorem 4 *The system $\mathbf{x}' = f(\mathbf{x}) + g(\mathbf{x})v$ is globally feedback linearizable if and only if there exists a smooth function*

$$\varphi : \mathbb{R}^n \rightarrow \mathbb{R}$$

having everywhere nonzero gradient and satisfying the following properties:

1. *for each $\mathbf{x} \in \mathbb{R}^n$, the vectors $g(\mathbf{x}), ad_f g(\mathbf{x}), \dots, ad_f^{n-1}g(\mathbf{x})$ are linearly independent;*
2. *for each $\mathbf{x} \in \mathbb{R}^n$ and each $j = 0, \dots, n-2$, $\nabla\varphi(\mathbf{x}) \cdot ad_f^j g(\mathbf{x}) = 0$;*
3. *the map $\mathbf{x} \mapsto (\varphi(\mathbf{x}), L_f\varphi(\mathbf{x}), \dots, L_f^{n-1}\varphi(\mathbf{x}))$ is a bijection $\mathbb{R}^n \rightarrow \mathbb{R}^n$.*

Observe that the conditions amount to the existence of a well-behaved solution φ of a set of first-order linear partial differential equations. Existence of a solution of this form is not trivial to verify. In order to study solvability, in control theory one considers the following conditions:

(LI) The set of vector fields $\{g(\mathbf{x}), ad_f g(\mathbf{x}), \dots, ad_f^{n-1}g(\mathbf{x})\}$ is linearly independent.

(INV) The distribution generated by $\{g, \text{ad}_f g, \dots, \text{ad}_f^{n-2} g\}$ is involutive.

This last condition means that the Lie bracket of any two of the vector fields $\text{ad}_f^i g$, for $i \in \{0, \dots, n-2\}$, should be, for each \mathbf{x} , a linear combination of these same $n-1$ vectors.

One then has the following result (see Theorem 15 in [34]), which is a consequence of Frobenius' Theorem in partial differential equation theory: *A system satisfies both conditions (LI) and (INV) at a state \mathbf{x} if and only if it is feedback linearizable in some open set containing \mathbf{x} .* This provides a useful and complete characterization of local feedback linearizability, and in particular a necessary condition for global feedback linearizability. In examples, often these conditions lead one to a globally defined solution, see e.g. example 5.3.10 in [34]).

4.1.6 Application to Neural Network Equations

Let us now show with the help of Theorem 4 that the class \mathcal{S}_n includes some fading memory systems of the form (6). Indeed, consider any system as follows:

$$\mathbf{x}' = -\text{diag}(\lambda_1, \dots, \lambda_n)\mathbf{x} + \mathbf{b} \cdot v \quad (11)$$

where the $\lambda_i \neq \lambda_j$ for each $i \neq j$ are all positive, $\text{diag}(\lambda_1, \dots, \lambda_n)$ is the resulting diagonal matrix, and the column vector $\mathbf{b} = \text{col}(b_1, \dots, b_n)$ has nonzero entries: $b_i \neq 0$ for all i . (Such a system, which has the form (6) with $\sigma(A\mathbf{x}) \equiv 0$, consists of n first order linear differential equations in parallel, and is obviously fading-memory.) It is easy to see that, up to signs $(-1)^i$, we have

$$\text{ad}_f^i g(\mathbf{x}) = \text{col}(\lambda_1^i b_1, \dots, \lambda_n^i b_n)$$

for $i > 0$, and the linear independence of $g(\mathbf{x}), \text{ad}_f g(\mathbf{x}), \dots, \text{ad}_f^{n-1} g(\mathbf{x})$ follows from the fact that these constant vectors form a Vandermonde matrix. Then we can pick $\varphi(\mathbf{x})$ as a linear map $\mathbf{x} \rightarrow \mathbf{a}\mathbf{x}$, where \mathbf{a} is any vector in \mathbb{R}^n which is orthogonal to all of the vectors

$$\text{col}(\lambda_1^i b_1, \dots, \lambda_n^i b_n), i = 0, 1, \dots, n-2.$$

The map $\mathbf{x} \mapsto (\varphi(\mathbf{x}), L_f\varphi(\mathbf{x}), \dots, L_f^{n-1}\varphi(\mathbf{x}))$ is represented then also by a Vandermonde matrix, so it is a bijection. Hence the conditions 1.-3. of Theorem 4 are satisfied, which implies that the system (11) belongs to the class \mathcal{S}_n .

As a further example, we now consider the following system, which also has the general form of the neural-network equations (6):

$$\begin{aligned}x'_1(t) &= -\lambda_1 x_1(t) + \sigma(x_2(t) + ax_3(t)) \\x'_2(t) &= -\lambda_2 x_2(t) + \sigma(x_2(t) + x_3(t)) \\x'_3(t) &= -\lambda_3 x_3(t) + v(t)\end{aligned}$$

where σ is a scalar function, smooth but otherwise arbitrary for now, and a as well as the λ_i 's are constants, also arbitrary for now. We will analyze this example using the Lie formalism described in section 4.1.4. The system has the form $\mathbf{x}' = f(\mathbf{x}) + g(\mathbf{x})v$, with $n = 3$, and f and g are the following vector fields:

$$f = \begin{pmatrix} -\lambda_1 x_1 + \sigma(x_2 + ax_3) \\ -\lambda_2 x_2 + \sigma(x_2 + x_3) \\ -\lambda_3 x_3 \end{pmatrix}, \quad g = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Note that the Jacobian g_* of g is identically zero, which simplifies the computation of Lie brackets. We calculate $\text{ad}_f g(\mathbf{x}) = [f, g](\mathbf{x}) = -f_*(\mathbf{x})g(\mathbf{x})$ and $[g, \text{ad}_f g] = (\text{ad}_f g)_*(\mathbf{x})g(\mathbf{x})$ as follows:

$$\text{ad}_f g(\mathbf{x}) = \begin{pmatrix} -a\sigma'(x_2 + ax_3) \\ -\sigma'(x_2 + x_3) \\ \lambda_3 \end{pmatrix}, \quad [g, \text{ad}_f g] = \begin{pmatrix} -a^2\sigma''(x_2 + ax_3) \\ -\sigma''(x_2 + x_3) \\ 0 \end{pmatrix}.$$

The involutivity condition says that the set of vector fields $\{g, \text{ad}_f g\}$ should be involutive, which means that $[g, \text{ad}_f g](\mathbf{x})$ should be in the span of $g(\mathbf{x})$ and $\text{ad}_f g(\mathbf{x})$ for all \mathbf{x} . Let us evaluate $g, \text{ad}_f g, [g, \text{ad}_f g]$ at the particular points for which $x_3 = 0$, so that we obtain,

respectively, three vectors v_1, v_2, v_3 that depend on x_2 only:

$$v_1(x_2) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad v_2(x_2) = \begin{pmatrix} -a\sigma'(x_2) \\ -\sigma'(x_2) \\ \lambda_3 \end{pmatrix}, \quad v_3(x_2) = \begin{pmatrix} -a^2\sigma''(x_2) \\ -\sigma''(x_2) \\ 0 \end{pmatrix}.$$

If $[g, \text{ad}_f g](\mathbf{x})$ is in the span of $g(\mathbf{x})$ and $\text{ad}_f g(\mathbf{x})$ for all vectors \mathbf{x} , then, in particular, $v_3(x_2)$ must belong to the span of $v_1(x_2)$ and $v_2(x_2)$ for all x_2 . This means that there is, for each x_2 , a scalar $r(x_2)$ such that

$$a^2\sigma''(x_2) = ar(x_2)\sigma'(x_2), \quad \sigma''(x_2) = r(x_2)\sigma'(x_2).$$

If $a \neq 0$, it follows that $\sigma''(x_2) = a\sigma''(x_2)$ for all x_2 . So, if also $a \neq 1$, we conclude that $\sigma''(x_2)$ must vanish for all x_2 . Thus, the system in our example (assuming $a \notin \{0, 1\}$) is feedback linearizable only if σ is a linear function.

On the other hand, consider now the cases $a = 0$ or $a = 1$. Then, the involutivity condition becomes the requirement that there should exist a scalar function r such that

$$\sigma''(x_2 + x_3) = r(\mathbf{x})\sigma'(x_2 + x_3),$$

which can be achieved provided only that the function σ' is everywhere nonzero (which is true if σ is, for example, a standard sigmoidal function), simply by taking $r(\mathbf{x}) = \sigma''(x_2 + x_3)/\sigma'(x_2 + x_3)$. The linear independence condition amounts to showing that the set of vectors $\{g, \text{ad}_f g, \text{ad}_f^2 g\}$ is linearly independent. Computing the determinant of the matrix that has these vectors as columns, when $a = 0$ we obtain $-\sigma'(x_2)[\sigma'(x_2 + x_3)]^2$, which is everywhere nonzero, provided that we again assume that σ has an everywhere nonzero derivative. Thus, the Lie-theoretic conditions for feedback linearization are satisfied, for any choice of λ_i 's, when $a = 0$. In the case $a = 1$, the same computation gives a determinant of $(\lambda_1 - \lambda_2)[\sigma'(x_2 + x_3)]^2$, so the Lie-theoretic conditions for feedback linearization are satisfied, for any choice of λ_i 's such that $\lambda_1 \neq \lambda_2$.

4.2 Mathematical Definitions and Details to the Proof of Theorem 2

4.2.1 Fading Memory Filters

A map (or filter) F from input- to output streams is defined to have *fading memory* if its current output at time t depends (up to some precision ε) only on values of the input \mathbf{u} during some finite time interval $[t - T, t]$. (We use in this section boldface letters to denote input streams, because they typically have a dimension larger than 1.) In formulas: F has fading memory if there exists for every $\varepsilon > 0$ some $\delta > 0$ and $T > 0$ so that $|(F\mathbf{u})(t) - (F\tilde{\mathbf{u}})(t)| < \varepsilon$ for any $t \in \mathbb{R}$ and any input functions $\mathbf{u}, \tilde{\mathbf{u}}$ with $\|\mathbf{u}(\tau) - \tilde{\mathbf{u}}(\tau)\| < \delta$ for all $\tau \in [t - T, t]$. This is a characteristic property of all filters that can be approximated by an integral over the input stream \mathbf{u} , or more generally by Volterra- or Wiener series. Note that non-trivial Turing machines and FSMs can *not* be approximated by filters with fading memory, since they require a persistent memory.

4.2.2 Finite State Machines

The deterministic *finite state machine* (FSM), also referred to as deterministic finite automaton, is a standard model for a digital computer, or more generally for any realistic computational device that operates in discrete time with a discrete set of inputs and internal states [27]. One assumes that a FSM is at any time in one of some finite number l of states, and that it receives at any (discrete) time step one input symbol from some alphabet $\{s_1, \dots, s_k\}$ that may consist of any finite number k of symbols. Its “program” may consist of any transition function $TR : \{s_1, \dots, s_k\} \times \{1, \dots, l\} \rightarrow \{1, \dots, l\}$, where $TR(s_i, j') = j$ denotes the new internal state j which the FSM assumes at the next time step after processing input symbol s_i in state j' .

4.2.3 Precise Statement of Theorem 2

We consider here a slight variation of the finite state machine model, which is more adequate for systems that operate in continuous time and receive analog inputs (for example trains of spikes in continuous time). We assume that the raw input is some arbitrary n -dimensional input stream \mathbf{u} (i.e., $\mathbf{u}(t) \in \mathbb{R}^n$ for every $t \in \mathbb{R}$). Furthermore we assume that there exist pattern detectors F_1, \dots, F_k that report the occurrence of spatio-temporal patterns in the input stream \mathbf{u} from k different classes C_1, \dots, C_k . In the case where the input \mathbf{u} consists of spike trains, these classes could consist for example of particular patterns of firing rates, of particular spike patterns, or particular correlation patterns among some of the input spike trains. It was shown in [5] that readouts from generic neural microcircuit models can easily be trained to approximate the role of such pattern detectors F_1, \dots, F_k . We assume that the detection of a pattern from class C_i by pattern detector F_i affects the state of the FSM according to its transition function TR in a way which corresponds to the presentation of input symbol s_i in the discrete-time version: if j' was its preceding state, then it changes now within some finite switching time to state $j = TR(s_i, j')$.

In order to make an implementation of such FSM by a noisy system feasible, we assume that the pattern detectors $(F_1\mathbf{u})(t), \dots, (F_k\mathbf{u})(t)$ always assume values ≤ 0 , except during a switching episode. During a switching episode exactly one of the pattern detectors $(F_i\mathbf{u})(t)$ assumes values > 0 . We assume that this $(F_i\mathbf{u})(t)$ reaches values ≥ 1 during this switching episode. We also assume that the length of each switching episode (i.e., the time during which some $(F_i\mathbf{u})(t)$ assumes values > 0) is bounded from above by some constant δ , and that the temporal distance between the beginnings of any two different switching episodes is at least $\Delta + 3\delta$ (where Δ is the assumed temporal delay of the feedback in the circuit). In order to avoid that the subsequent construction is based on unrealistic assumptions, we allow that each pattern detector F_i is replaced by some arbitrary filter \hat{F}_i so that $(\hat{F}_i\mathbf{u})(t)$ is a continuous function of time (with values in some arbitrary bounded range $[-B, B]$) with $|(\hat{F}_i\mathbf{u})(t) - (F_i\mathbf{u})(t)| \leq \frac{1}{4}$ for any input stream \mathbf{u} that is considered.

The informal statement of Theorem 2 is made precise by the subsequent Theorem 5 (see Fig. 6 for an illustration). It exhibits a simple construction method whereby fading memory filters with additive noise of bounded amplitude can be composed into a closed loop system C that emulates an arbitrary given FSM in a noise-robust manner. The resulting system C can be embedded into any other fading memory system, which receives the outputs $CL - \hat{H}_j(t)$ of C as additional inputs. In this way any given fading memory system can integrate the computational capability and non-fading states of the FSM that is emulated by C into its own real-time computation on time-varying input streams \mathbf{u} .

An essential aspect of the proof of Theorem 5 is that suitable fading memory filters H_j can prevent in the closed loop the accumulation of errors through feedback, even if the ideal fading memory filters H_j are subsequently replaced by imperfect approximations \hat{H}_j . One just has to construct the ideal fading memory filters H_j in such a way that they take into account that their previous outputs, that have been fed back into the system C , may have been corrupted by additive noise. As long as this additive noise of bounded amplitude has not been amplified in the closed loop, the filters H_j can still recover which of the finitely many states of the emulated FSM A was represented by that noise-corrupted feedback.

From the perspective of neural circuit models it is of interest to note that the construction of the system C can be replaced by an adaptive procedure, whereby readouts from generic cortical microcircuit models are trained to approximate the target filters H_j . General approximation results [4, 5, 38] imply that if the neural circuit is sufficiently large and contains sufficiently diverse components (for example dynamic synapses with slightly different parameter values), then the actual outputs \hat{H}_j of these readouts can approximate the target filters H_j uniformly up to any given maximal error $\varepsilon > 0$. Theorem 5 guarantees that the resulting neural circuit model with these (imperfectly) trained readouts can in the closed loop emulate the given FSM A in a reliable manner, provided that the neural circuit model is sufficiently large and diverse so that its readout can achieve an

approximation error ε not larger than $1/4$.

Theorem 5 *One can construct for any given finite state machine (FSM) A some time invariant fading memory filters H_1, \dots, H_l with the property that any approximating filters $\hat{H}_1, \dots, \hat{H}_l$ with $|H_j - \hat{H}_j| \leq 1/4$ provide in the closed loop with delay Δ (see Fig. 6) outputs $CL - \hat{H}_1, \dots, CL - \hat{H}_l$ that simulate the FSM A in the following sense:*

If $[t_1, t_2]$ is some arbitrary time interval between switching episodes of the FSM A with noise-free pattern detectors $(F_1 \mathbf{u})(t), \dots, (F_k \mathbf{u})(t)$ during which A is in state j , then the outputs $CL - \hat{H}_i(t)$ of the approximating filters \hat{H}_i in the closed loop with noisy pattern detectors $(\hat{F}_1 \mathbf{u})(t), \dots, (\hat{F}_k \mathbf{u})(t)$ satisfy $CL - \hat{H}_j(t) \geq \frac{3}{4}$ and $CL - \hat{H}_{j^}(t) \leq \frac{1}{4}$ for all $j^* \neq j$ and all $t \in [t_1, t_2]$.*

4.2.4 Proof of the Precise Statement of Theorem 2

We present here a proof of Theorem 5 (see section 4.2.3), which provides a formally precise version of Theorem 2.

In order to prove that the given FSM A can be implemented in a noise robust fashion, we construct suitable time invariant fading memory filters H_1, \dots, H_l . They receive as inputs the time-varying functions $(\hat{F}_1 \mathbf{u})(t), \dots, (\hat{F}_k \mathbf{u})(t)$. In addition they receive in the open loop inputs $v_1(t), \dots, v_l(t)$, where each $v_j(t)$ will be replaced by a delayed version of the output of H_j (or \hat{H}_j) in the closed loop (see Fig. 6). The filters H_j will be defined in such a way that $H_j(t) \geq 1$ signals in the closed loop that the FSM A is at time t in state j . To make this implementation noise robust, we make sure that even if one replaces the filters H_j by noisy approximations \hat{H}_j which satisfy in the open loop $|H_j(t) - \hat{H}_j(t)| \leq \frac{1}{4}$ (for all $t \in \mathbb{R}$ and any time varying inputs $(\hat{F}_1 \mathbf{u})(t), \dots, (\hat{F}_k \mathbf{u})(t)$ and $v_1(t), \dots, v_l(t)$), then the closed loop version of such imperfect approximations \hat{H}_j simulates the FSM A in such a way that $\hat{H}_j(t) \geq \frac{3}{4}$ implies that A is in state j at time t .

Let Δ be the time delay in the feedback for the closed-loop. We now define the target

outputs $H_1(t), \dots, H_l(t)$ (for the open loop version, where the H_j receive in addition to $(\hat{F}_1 \mathbf{u})(t), \dots, (\hat{F}_k \mathbf{u})(t)$ some arbitrary time-varying variables $v_1(t), \dots, v_l(t)$ with values in $[-1, 2]$ as inputs). We define the target outputs of H_1, \dots, H_l as a stationary transformation of the time-varying inputs $v_j(t)$ and of the outputs of the following two other types of time invariant fading memory filters:

(i) $f_i(t) := \max\{(\hat{F}_i \mathbf{u})(\tau) : t - \Delta - \delta \leq \tau \leq t\}$ for $i = 1, \dots, k$

(ii) $v_j(t - 2\delta)$ for $j = 1, \dots, l$.

We will show below in Lemma 6 and Lemma 7 that both of these functions of time can be viewed as outputs of time invariant fading memory filters that receive as inputs the time-varying functions $(\hat{F}_i \mathbf{u})(t)$ (for some arbitrary input stream \mathbf{u}) and $v_j(t)$. On the basis of these two Lemmata it is clear that the H_j are time invariant fading memory filters if one can define $H_1(t), \dots, H_l(t)$ as (static) continuous functions of the variables $v_j(t)$ and the outputs of the filters (i) and (ii). In the following we sometimes refer to H_1, \dots, H_l as static functions of input vectors $(f_1(t), \dots, f_k(t), v_1(t), \dots, v_l(t), v_1(t - 2\delta), \dots, v_l(t - 2\delta))$ from \mathbb{R}^{k+2l} , and sometimes as filters with time-varying inputs $\hat{F}_i \mathbf{u}$ and v_j (if we view the filters (i) and (ii) as being part of the computation of H_j). In order to define such functions $H_j(t)$ we first define for each $j \in \{1, \dots, l\}$ two disjoint closed and bounded sets $S_{j,0}, S_{j,1} \subseteq \mathbb{R}^{k+2l}$, and we set $H_j(\mathbf{x}) = 0$ for $\mathbf{x} \in S_{j,0}$ and $H_j(\mathbf{x}) = 1$ for $\mathbf{x} \in S_{j,1}$. Since the sets $S_{j,0}$ and $S_{j,1}$ will have positive distance (i.e., $\inf\{\|\mathbf{x} - \mathbf{y}\| : \mathbf{x} \in S_{j,0} \text{ and } \mathbf{y} \in S_{j,1}\} > 0$), it follows from standard arguments of analysis that the definition of H_j can be continued outside of $S_{j,0}, S_{j,1}$ to yield a continuous function from \mathbb{R}^{k+2l} into \mathbb{R} .

In order to define the sets $S_{j,0}, S_{j,1}$ we consider the following two types of conditions:

(A_j) There exist $i \in \{1, \dots, k\}$ and $j' \in \{1, \dots, l\}$ so that $TR(i, j') = j$,

$$f_i(t) \geq \frac{3}{4} \text{ and } f_{i'}(t) \leq \frac{1}{4} \text{ for all } i' \neq i,$$

$$v_{j'}(t - 2\delta) \geq \frac{3}{4} \text{ and } v_{j^*}(t - 2\delta) \leq \frac{1}{4} \text{ for all } j^* \neq j'.$$

(B_j) $f_i(t) \leq \frac{1}{4}$ for $i = 1, \dots, k$, $v_j(t) \geq \frac{3}{4}$ and $v_{j^*}(t) \leq \frac{1}{4}$ for all $j^* \neq j$.

We say that a vector $(f_1(t), \dots, f_k(t), v_1(t), \dots, v_l(t), v_1(t - 2\delta), \dots, v_l(t - 2\delta)) \in [-B, B]^k \times [-1, 2]^{2l}$ belongs to set $S_{j,1}$ if the conditions A_j or B_j apply, and to set $S_{j,0}$ if there exists some $j^* \neq j$ so that the conditions A_{j^*} or B_{j^*} apply.

It follows immediately from the definition of the sets $S_{j,0}$ and $S_{j,1}$ that they are closed and bounded. One can also verify immediately that for any $j, j' \in \{1, \dots, l\}$ the conditions A_j and $B_{j'}$ can never be simultaneously satisfied (for any values of the variables $f_i(t), v_j(t), v_j(t - 2\delta)$). In addition the conditions A_j and $A_{j'}$ (B_j and $B_{j'}$) can never be simultaneously satisfied for any $j \neq j'$. This implies that the sets $S_{j,0}$ and $S_{j,1}$ are disjoint for each $j \in \{1, \dots, l\}$.

We define for each $j \in \{1, \dots, l\}$ a continuous function $H_j : \mathbb{R}^{k+2l} \rightarrow [0, 1]$ by setting

$$H_j(\mathbf{x}) := \begin{cases} 1 & , \text{ if } \text{dist}(\mathbf{x}, S_{j,0}) \geq \text{dist}(S_{j,0}, S_{j,1}) \\ \text{dist}(\mathbf{x}, S_{j,0}) / \text{dist}(S_{j,0}, S_{j,1}) & , \text{ otherwise } , \end{cases}$$

where $\text{dist}(\mathbf{x}, S) := \inf\{\|\mathbf{x} - \mathbf{y}\| : \mathbf{y} \in S\}$ for any set $S \subseteq \mathbb{R}^{k+2l}$. It is then obvious that H_j is a continuous function from \mathbb{R}^{k+2l} into $[0, 1]$ with $H_j(\mathbf{x}) = 0$ for all $\mathbf{x} \in S_{j,0}$ and $H_j(\mathbf{x}) = 1$ for all $\mathbf{x} \in S_{j,1}$. These functions H_j will prevent the amplification of noise in the closed loop, since they assume outputs 1 or 0 in all relevant situations, even if their inputs deviate by up to $\frac{1}{4}$ from their “ideal” values.

We consider some arbitrary imprecise and/or noisy versions \hat{H}_j of these filters H_j (with inputs $(\hat{F}_1 \mathbf{u})(t), \dots, (\hat{F}_k \mathbf{u})(t)$ and additional inputs $v_1(t), \dots, v_l(t)$) whose output differs at any time t by at most $\frac{1}{4}$ from that of H_j (of course in the closed loop these deviations could be accumulated and amplified to values $> \frac{1}{4}$). We want to show that for any such $\hat{H}_1, \dots, \hat{H}_l$ the closed loop version of the circuit implements the given FSM A . As initial condition we assume that the given FSM A is in state 1 for $t \leq 0$, and consequently also that $\hat{H}_1(t) \geq \frac{3}{4}$ and $\hat{H}_j(t) \leq \frac{1}{4}$ for $j = 2, \dots, l$, as well as $f_i(t) \leq \frac{1}{4}$ for all $t \leq 0$ and $i = 1, \dots, k$.

We will now prove the claim of Theorem 5 for arbitrary time intervals $[t_1, t_2]$ outside of switching episodes. We assume without loss of generality that t_2 marks the beginning

of the next switching episode $[t_2, t_3]$ for some $t_3 > t_2$ with $|t_3 - t_2| \leq \delta$. Furthermore we assume that either $t_1 = 0$ (Case 1), or t_1 is the endpoint of the preceding switching episode $[t_0, t_1]$ with $|t_1 - t_0| \leq \delta$ (Case 2). The formal proof is carried out by induction on the number of preceding switching episodes (and Case 2 represents the induction step). In both cases one just needs to analyze the outputs of the previously defined filters $\hat{H}_j(t)$ in the case where some of their inputs are delayed feedbacks of their previous outputs.

Case 1: $t_1 = 0$

We prove by a nested induction on $m \in \mathbb{N}$ that $CL - \hat{H}_1(t) \geq \frac{3}{4}$ and $CL - \hat{H}_j(t) \leq \frac{1}{4}$ for all $j > 1$ holds for all $t \in [m \cdot \Delta, (m + 1) \cdot \Delta) \cap [t_1, t_2]$. Since by assumption no switching episode occurs during $[t_1, t_2]$, one has $f_i(t) \leq \frac{1}{4}$ for $i = 1, \dots, k$ and for all $t \in [t_1, t_2]$. Furthermore by our assumption on the initial condition of the FSM A (for $m = 0$), or by the induction hypothesis of the nested induction (for $m > 0$) we can assume that the variables $v_j(t)$ of the open loop have now been assigned in the closed loop the values $CL - \hat{H}_j(t - \Delta)$, therefore they are $\geq \frac{3}{4}$ for $j = 1$ and $\leq \frac{1}{4}$ for all $j > 1$. Hence condition B_1 in the definition of the sets $S_{j,0}, S_{j,1}$ applies, and the current circuit input is therefore in $S_{1,1}$. Thus $H_1 = 1$ and $H_j = 0$ for $j > 1$, which implies $\hat{H}_1 \geq \frac{3}{4}$ and $\hat{H}_j \leq \frac{1}{4}$ for $j > 1$ in the open loop, hence $CL - \hat{H}_1(t) \geq \frac{3}{4}$ and $CL - \hat{H}_j(t) \leq \frac{1}{4}$ for $j > 1$ in the closed loop (since $v_j(t) = CL - \hat{H}_j(t - \Delta)$ in the closed loop).

Case 2: t_1 is the endpoint of a preceding switching episode $[t_0, t_1]$.

Assume that $(\hat{F}_i \mathbf{u})(t)$ is the (approximating) pattern detector that assumes a value $\geq \frac{3}{4}$ during the preceding switching episode $[t_0, t_1]$, while $(\hat{F}_{i'} \mathbf{u})(t) \leq \frac{1}{4}$ for all $i' \neq i$ during $[t_0, t_1]$. Let $t' \in [t_0, t_1]$ be the first time point where $(\hat{F}_i \mathbf{u})(t)$ reaches a value $\geq \frac{3}{4}$. Then $f_i(t) \geq \frac{3}{4}$ and $f_{i^*}(t) \leq \frac{1}{4}$ for all $i^* \neq i$ and for all $t \in [t', t' + \Delta + \delta]$ (by the definition of the filters $f_i(t)$). Furthermore one has by the induction hypothesis that for the state j' in which the FSM A was before the switching episode $[t_0, t_1]$ that $CL - \hat{H}_{j'}(t - \Delta - 2\delta) \geq \frac{3}{4}$ and $CL - \hat{H}_{j^*}(t - \Delta - 2\delta) \leq \frac{1}{4}$ for all $j^* \neq j'$ and all $t \in [t', t' + \Delta + 2\delta]$. We exploit here that

$t_0 \leq t' \leq t_1 \leq t_0 + \delta$, hence $t_0 - \Delta - 2\delta \leq t - \Delta - 2\delta \leq t_0$ for all $t \in [t', t' + \Delta + \delta]$. Furthermore we have assumed that the minimal distance between the beginnings of switching episodes is $\Delta + 3\delta$. Therefore the considered range $[t_0 - \Delta - 2\delta, t_0]$ for $t - \Delta - 2\delta$ is contained in the preceding time interval before the switching episode $[t_0, t_1]$ to which the induction hypothesis applies.

The previously listed conclusions imply that for $t \in [t', t' + \Delta + \delta]$ the current input to the open loop lies in the set $S_{j,1}$ for $j = TR(i, j')$, hence $H_j = 1$ and $\hat{H}_j \geq \frac{3}{4}$, while $H_{j^*} = 0$ and $\hat{H}_{j^*} \leq \frac{1}{4}$ for all other j^* . But if one chooses as inputs $v_1(t), \dots, v_l(t)$ to the open loop just those values which the circuit receives in the closed loop, one gets that $CL - \hat{H}_j(t) \geq \frac{3}{4}$ and $CL - \hat{H}_{j^*}(t) \leq \frac{1}{4}$ for all $j^* \neq j$ and all $t \in [t', t' + \Delta + \delta]$, in particular for all $t \in [t_1, t_1 + \Delta]$.

One can then prove by a nested induction on $m \in \mathbb{N}$ like in Case 1 that the outputs $CL - \hat{H}_{j^*}(t)$ for $j^* = 1, \dots, l$ have the desired values for $t \in [t_1 + m\Delta, t_1 + (m+1) \cdot \Delta] \cap [t_1, t_2]$. The preceding argument provides the verification of the claim for the initial step $m = 0$ of this nested induction.

In order to complete the proof of Theorem 5 it only remains to verify the following two simple facts about time invariant fading memory filters.

Lemma 6 *Assume that \hat{F}_i is some arbitrary time invariant fading memory filter, and Δ, δ are arbitrary positive constants. Then the map which assigns to an input stream \mathbf{u} the function $f_i(t) := \max\{(\hat{F}_i \mathbf{u})(\tau) : t - \Delta - \delta \leq \tau \leq t\}$ is also a time invariant fading memory filter.*

Proof of Lemma 6: Assume some $\varepsilon > 0$ is given. Fix δ' and $T > 0$ so that $|(\hat{F}_i \mathbf{u})(\tau) - (\hat{F}_i \mathbf{v})(\tau)| < \varepsilon$ for all $\tau \in [t - \Delta - \delta, t]$ and all \mathbf{u}, \mathbf{v} with $\|\mathbf{u}(s) - \mathbf{v}(s)\| < \delta'$ for all $s \in [t - \Delta - \delta - T, t]$. Then $|\max\{(\hat{F}_i \mathbf{u})(\tau) : t - \Delta - \delta \leq \tau \leq t\} - \max\{(\hat{F}_i \mathbf{v})(\tau) : t - \Delta - \delta \leq \tau \leq t\}| < \varepsilon$. ■

Lemma 7 *The filter which maps for some arbitrary fixed $\delta > 0$ the function $u(t)$ onto the function $u(t - 2\delta)$ is time invariant and has fading memory.*

Proof of Lemma 7: Follows immediately from the definitions (choose $T \geq 2\delta$ in the condition for fading memory). ■

This completes the proof of Theorem 5, which shows that any given FSM can be reliably implemented by fading memory filters with feedback even in the presence of noise. ■

Remark: In the application of this theory to cortical microcircuit models we train readouts from such circuits to *simultaneously* assume the role of the pattern detectors $\hat{F}_1, \dots, \hat{F}_k$, which become active if some pattern occurs in the input stream that may trigger a state change of the simulated FSM A , *and* the role of the fading memory filters $\hat{H}_1, \dots, \hat{H}_l$, that create high-dimensional attractors of the circuit dynamics that represent the current state of the FSM A .

4.3 Details of the Cortical Microcircuit Models

We complement in this section the general description of the simulated cortical microcircuit models from section 2.2, providing in particular all missing data that are needed to reproduce our simulation results. The original code that was used for these simulations is online available from <http://www.lsm.tugraz.at/research/index.html>.

Each circuit consisted of 600 neurons, which were placed on the integer grid points of a $5 \times 5 \times 24$ grid. 20% of these neurons were randomly chosen to be inhibitory. The probability of a synaptic connection from neuron a to neuron b (as well as that of a synaptic connection from neuron b to neuron a) was defined as $C \cdot \exp(-D^2(a, b)/\lambda^2)$, where $D(a, b)$ is the Euclidean distance between neurons a and b , and λ is a parameter

which controls both the average number of connections and the average distance between neurons that are synaptically connected (we set $\lambda = 3$). Depending on whether the pre- or postsynaptic neuron were excitatory (E) or inhibitory (I), the value of C was set according to [45] to 0.3 (EE), 0.2 (EI), 0.4 (IE), 0.1 (II), yielding an average of 10900 synapses for the chosen circuit size. External inputs and feedbacks from readouts were connected to populations of neurons in the circuit with randomly chosen connection strengths.

I&F neurons: A standard leaky-integrate-and-fire neuron model was used, where the membrane potential V_m of a neuron is given by:

$$\tau_m \frac{dV_m}{dt} = -(V_m - V_{resting}) + R_m \cdot (I_{syn} + I_{inject} + I_{noise}) \quad (12)$$

where t_m is the membrane time constant (30 ms), which subsumes the time constants of synaptic receptors as well as the time constant of the neuron membrane. Other parameters: absolute refractory period 3 ms (excitatory neurons), 2 ms (inhibitory neurons), threshold 15 mV (for a resting membrane potential $V_{resting}$, assumed to be 0), reset voltage drawn uniformly from the interval [13.8, 14.5 mV] for each neuron, input resistance R_m , 1 M Ω , constant non-specific background current I_{inject} uniformly drawn from the interval [13.5 nA, 14.5 nA] for each neuron, an additional time-varying noise input current I_{noise} was drawn every 5 ms from a Gaussian distribution with mean 0 and SD chosen for each neuron randomly from the uniform distribution over the interval [4.0 nA, 5.0 nA]. For each simulation, the initial condition of each I&F neuron, i.e., its membrane voltage at time $t = 0$, was drawn randomly (uniform distribution) from the interval [13.5 mV, 14.9 mV]. Finally, $I_{syn}(t)$ is the sum of input currents supplied by the explicitly modeled synapses.

HH-neurons: We used single compartment HH neuron models with passive and active properties modeled according to [49, 50]. The membrane potential was modeled by

$$C_m \frac{dV}{dt} = -g_l(V - E_l) - I_{Na} - I_{Kd} - I_M - \frac{1}{a} I_{noise} - I_{syn} , \quad (13)$$

where $C_m = 1 \mu F/cm^2$ is the specific membrane capacitance, $g_L = 0.045 mS/cm^2$ is the leak conductance density, $E_L = -80 mV$ is the leak reversal potential, and $I_{syn}(t)$ is the input current supplied by explicitly modeled synapses (see the definition below). The membrane area a of the neuron was set to be $34636 \mu m^2$ as in [49]. The term $I_{noise}(t)$ (see the precise definition below) models smaller background input currents from a large number of more distal neurons, causing a depolarization of the membrane potential and a lower input resistance commonly referred to as 'high conductance state' (for a review see [43]).

In accordance with experimental data on neocortical and hippocampal pyramidal neurons ([51–54]) the active currents in the HH neuron model comprise a voltage dependent Na^+ current I_{Na} ([55]) and a delayed rectifier K^+ current I_{Kd} ([55]). For excitatory neurons a non-inactivating K^+ current I_M ([56]) responsible for spike frequency adaption was included in the model.

The voltage-dependent Na^+ current was modeled by:

$$I_{Na} = \bar{g}_{Na} m^3 h (V - E_{Na})$$

$$\frac{dm}{dt} = \alpha_m(V)(1 - m) - \beta_m(V)m$$

$$\frac{dh}{dt} = \alpha_h(V)(1 - h) - \beta_h(V)h$$

$$\alpha_m = \frac{-0.32(V - V_T - 13)}{\exp[-(V - V_T - 13)/4] - 1}$$

$$\beta_m = \frac{0.28(V - V_T - 40)}{\exp[(V - V_T - 40)/5] - 1}$$

$$\alpha_h = 0.128 \exp[-(V - V_T - V_S - 17)/18]$$

$$\beta_h = \frac{4}{1 + \exp[-(V - V_T - V_S - 40)/5]}$$

where $V_T = -63 \text{ mV}$, and the inactivation was shifted by 10 mV toward hyperpolarized values ($V_S = -10 \text{ mV}$) to reflect the voltage dependence of Na^+ currents in neocortical pyramidal cells [57]. The peak conductance densities for the I_{Na} current was chosen to be $500 \text{ pS}/\mu\text{m}^2$.

The delayed rectifier K^+ current was modeled by:

$$I_{Kd} = \bar{g}_{Kd} n^4 (V - E_K)$$

$$\frac{dn}{dt} = \alpha_n(V)(1 - n) - \beta_n(V)n$$

$$\alpha_n = \frac{-0.032(V - V_T - 15)}{\exp[-(V - V_T - 15)/5] - 1}$$

$$\beta_n = 0.5 \exp[-(V - V_T - 10)/40]$$

The peak conductance densities for the I_{Kd} current was chosen to be $100 \text{ pS}/\mu\text{m}^2$.

The noninactivating K^+ current was modeled by:

$$I_M = \bar{g}_M n (V - E_K)$$

$$\frac{dn}{dt} = \alpha_n(V)(1 - n) - \beta_n(V)n$$

$$\alpha_n = \frac{0.0001(V + 30)}{1 - \exp[-(V + 30)/9]}$$

$$\beta_n = \frac{-0.0001(V + 30)}{1 - \exp[(V + 30)/9]}$$

The peak conductance density for the I_M current was chosen to be $5pS/\mu m^2$.

For each simulation the initial condition of each neuron, i.e. the membrane voltage at time $t = 0$, was drawn randomly (uniform distribution) from the interval $[-70, -60]$ mV.

The total **synaptic background current**, $I_{noise}(t)$, was a sum of two independent currents:

$$I_{noise}(t) = g_e(t)(V - E_e) + g_i(t)(V - E_i) ,$$

where $g_e(t)$ and $g_i(t)$ are time-dependent excitatory and inhibitory conductances. The values of respective reversal potentials were $E_e = 0$ mV and $E_i = -75$ mV.

The conductances $g_e(t)$ and $g_i(t)$ were modeled according to [49] as a one-variable stochastic process similar to an Ornstein-Uhlenbeck process:

$$\frac{dg_e(t)}{dt} = -\frac{1}{\tau_e}[g_e(t) - g_{e0}] + \sqrt{D_e}\chi_1(t)$$

$$\frac{dg_i(t)}{dt} = -\frac{1}{\tau_i}[g_i(t) - g_{i0}] + \sqrt{D_i}\chi_2(t)$$

where $g_{e0} = 0.012 \mu S$ and $g_{i0} = 0.057 \mu S$ are average conductances, $\tau_e = 2.7$ ms and $\tau_i = 10.5$ ms are time constants, $D_e = 0.0067 \mu S^2/s$ and $D_i = 0.0083 \mu S^2/s$ are noise diffusion constants, $\chi_1(t)$ and $\chi_2(t)$ are Gaussian white noise of zero mean and unit standard deviation.

Since these stochastic processes are Gaussian, they can be integrated by an exact update rule:

$$g_e(t + \Delta t) = g_{e0} + [g_e(t) - g_{e0}] \exp(-\Delta t/\tau_e) + A_e N_1(0, 1)$$

$$g_i(t + \Delta t) = g_{i0} + [g_i(t) - g_{i0}] \exp(-\Delta t/\tau_i) + A_i N_2(0, 1)$$

where $N_1(0, 1)$ and $N_2(0, 1)$ are normal random numbers (zero mean, unit SD) and A_e and A_i are amplitude coefficients given by:

$$A_e = \sqrt{\frac{D_e \tau_e}{2} \left[1 - \exp\left(\frac{-2\Delta t}{\tau_e}\right) \right]}$$

$$A_i = \sqrt{\frac{D_i \tau_i}{2} \left[1 - \exp\left(\frac{-2\Delta t}{\tau_i}\right) \right]} .$$

According to [49], this model captures the spectral and amplitude characteristics of the input conductances of a detailed biophysical model of a neocortical pyramidal cell that was matched to intracellular recordings in cat parietal cortex *in vivo*. Furthermore the ratio of the average contributions of excitatory and inhibitory background conductances was chosen to be 5 in accordance with experimental studies during sensory responses [58–60]. The maximum conductances of the synapses were chosen from a Gaussian distribution with a SD of 70% of its mean (with negative values replaced by values chosen from a uniform distribution between 0 and two times the mean).

We modeled the (short term) **dynamics of synapses** according to the model proposed in [44], with the synaptic parameters U (use), D (time constant for depression), F (time constant for facilitation) randomly chosen from Gaussian distributions that model empirically found data for such connections (see supplementary information). This model predicts the amplitude A_k of the EPSC for the k^{th} spike in a spike train with interspike intervals $\Delta_1, \Delta_2, \dots, \Delta_{k-1}$ through the equations

$$\begin{aligned} A_k &= w \cdot u_k \cdot R_k \\ u_k &= U + u_{k-1}(1 - U)\exp(-\Delta_{k-1}/F) \\ R_k &= 1 + (R_{k-1} - u_{k-1}R_{k-1} - 1)\exp(-\Delta_{k-1}/D) \end{aligned}$$

with hidden dynamic variables $u \in [0, 1]$ and $R \in [0, 1]$ whose initial values for the first

spike are $u_1 = U$ and $R_1 = 1$ (see [61] for a justification of this version of the equations, which corrects a small error in [44]).

The postsynaptic current for the k^{th} spike in a presynaptic spike train, that had been generated at time t_k , is modeled for $t \geq t_k + \Delta$ (where Δ is the transmission delay) by $A_k \exp(-(t - t_k - \Delta)/\tau_s)$ with $\tau_s = 3$ ms ($\tau_s = 6$ ms) for excitatory (inhibitory) synapses. The transmission delays Δ between neurons were chosen uniformly to be 1.5 ms for EE-connections, and 0.8 ms for the other connections. The total **synaptic input current** $I_{syn}(t)$ was modeled by the sum of these currents for all synapses onto a neuron.

Synaptic parameters: Depending on whether a and b were excitatory (E) or inhibitory (I), the mean values of the three parameters U, D, F (with D, F expressed in seconds, s) were chosen according to [45] to be .5, 1.1, .05 (EE), .05, .125, 1.2 (EI), .25, .7, .02 (IE), .32, .144, .06 (II). The SD of each of these parameters was chosen to be 50% of its mean. The mean of the scaling parameter w (in nA) was chosen to be 70 (EE), 150 (EI), -47 (IE), -47 (II). The SD of the parameter w was chosen to be 70% of its mean and was drawn from a gamma distribution. In the case of input synapses the parameter w had a value of 70 nA if projecting onto a excitatory neuron and -47 nA if projecting onto an inhibitory neuron.

The synaptic weights \mathbf{w} of **readout neurons** were computed by linear regression to minimize the mean squared error $(\mathbf{w} \cdot \mathbf{x}(t) - f(t))^2$ with regard to a specific target output function $f(t)$ (which is described for each case in the text or figure legends) for a series of randomly generated time-varying circuit input streams $\mathbf{u}(t)$ of length up to 1 second. Up to 200 such time-varying input streams $\mathbf{u}(t)$ were used for training, amounting to at most 200 seconds of simulated biological time for training the readouts.

The performance of trained readouts was evaluated by measuring the correlation between $\mathbf{w} \cdot \mathbf{x}(t)$ and the target function $f(t)$ during separate testing episodes where the

circuit received new input streams $\mathbf{u}(t)$ (that were generated by the same random process as the training inputs).

All simulations were carried out with the software package CSIM [62], which is freely available from <http://www.lsm.tugraz.at>. It uses a C++-kernel with Matlab interfaces for input generation and data analysis. As simulation time step we chose 0.5 ms.

4.4 Technical Details to Figure 3

4 randomly generated test input streams, each consisting of 8 spike trains (see Fig. 3A), were injected into 4 disjoint (but interconnected) subsets of $5 \times 5 \times 5 = 125$ neurons in the circuit consisting of 600 neurons. Feedbacks from readouts were injected into the remaining 100 neurons of the circuit. The set of 100 neurons for which the firing activity is shown in Fig. 3C contained 20 neurons from each of the resulting 5 subsets of the circuit.

Generation of input streams for training and testing: The time-varying firing rate $r_i(t)$ of the 8 Poisson spike trains that represented input stream i was chosen as follows. The baseline firing rate for streams 1 and 2 (see the lower half of Fig. 3A) was chosen to be 5 Hz, with randomly distributed bursts of 120 Hz for 50 ms. The rates for the Poisson processes that generated the spike trains for input streams 3 and 4 were periodically drawn randomly from the two options 30 Hz and 90 Hz. The actual firing rates (i.e. spike counts within a 30 ms window) resulting from this procedure are plotted in Fig. 3B.

In order to demonstrate that readouts that send feedback into the circuit can just as well represent neurons *within* the circuit, we had chosen the readout neurons that send feedback to be I&F neurons with noise, like the other neurons in the circuit. Each of them received synaptic inputs from a slightly different randomly chosen subset of neurons within the circuit. Furthermore the signs of weights of these synaptic connections were restricted to be positive (negative) for excitatory (inhibitory) presynaptic neurons.

The 8 readout neurons that provided feedback were trained to represent in their firing activity at any time the information in which of input streams 1 or 2 a burst had most

recently occurred. If it occurred most recently in input stream 1, they were trained to fire at 40 Hz, and they were trained not to fire whenever a burst had occurred most recently in input stream 2. The training time was 200 s (of simulated biological time). After training, their output was correct 86% of the time (average over 50 s of input streams; counting the high-dimensional attractor as being in the on-state if the average firing rate of the 8 readout neurons was above 34 Hz). It was possible to train these readout neurons to acquire such persistent firing behavior, although they only received input from a circuit with fading memory, because they were actually trained to acquire the following behavior: fire whenever the rate in input stream 1 becomes higher than 30 Hz, or if one can detect in the current state $\mathbf{x}(t)$ of the circuit traces of recent high feedback values, provided the rate of input stream 2 stayed below 30 Hz. Obviously this definition of the learning target for readout neurons only requires a fading memory of the circuit.

The readouts for the other 3 tasks achieved in 50 tests for new inputs over 1 s (that had been generated by the same distribution as the training inputs, see the preceding description) the following average performance:

Task of panel E: Mean correlation: 0.85

Task of panel F: Mean correlation: 0.63

Task of panel G: Mean correlation: 0.86 .

4.5 Technical Details to Figure 4

The same circuit as for Fig. 3 was used. First 2 linear readouts with feedback were simultaneously trained to become highly active after the occurrence of the cue in the spike input, and then to linearly reduce their activity, but each within a different time span (400 versus 600 ms). Their feedback into the circuit consisted of 2 time-varying analog values (representing time-varying firing rates of 2 population of neurons), which were both injected (with randomly chosen amplitudes) into the same subset of 350 neurons in the circuit. Their weights \mathbf{w} were trained by linear regression for a total training time

of 120 s (of simulated biological time), consisting of 120 runs of length 1 s with randomly generated input-cues (a burst at 200 Hz for 50 ms) and noise inputs (5 spike trains at 10 Hz).

4.6 Technical Details to Figure 5

Time-varying firing rates for the two input streams (consisting each of 8 Poisson spike trains) were drawn randomly from values between 10 and 90 Hz. The 16 spike trains from the 2 input streams, as well as feedback from trained readouts were injected into randomly chosen subsets of neurons. In contrast to the experiment for Fig. 3, these circuit inputs were not injected into spatially concentrated clusters of neurons, but to a sparsely distributed subset of neurons scattered throughout the 3-dimensional circuit. As a consequence, the firing activity $CA(t)$ of the high-dimensional attractor (see Fig. 5D) cannot be readily detected from the spike raster in Fig. 5C. Both the linear readout that sends feedback, and subsequently the other two linear readouts (whose output for a test input to the circuit is shown in Fig. 5E,F), were trained by linear regression during 140 s of simulated biological time.

Average performance of linear readouts on 100 new test inputs of length 700 ms (that had been generated from the same distribution as the training inputs):

Task of panel D: Mean correlation: 0.82

Task of panel E: Mean correlation: 0.71

Task of panel F: Mean correlation: 0.79 .

Control experiments (see Fig. 7) show that the feedback is essential for the performance of the circuit for these computational tasks.

5 Acknowledgments

Comments from Wulfram Gerstner, Stefan Haeusler, Herbert Jaeger, Konrad Koerding, Henry Markram, Gordon Pipa, Misha Tsodyks, and Tony Zador are gratefully acknowledged. Our computer simulations used software written by Thomas Natschlaeger, Stefan Haeusler, and Michael Pfeiffer. This research was partially supported by the Austrian Science Fund FWF, # S9102-N04, and # P17229-N04, and PASCAL, project # IST2002-506778, of the European Union. The work of Eduardo D. Sontag was partially supported by NSF grants DMS-0504557 and DMS-0614371.

References

- [1] R. J. Douglas, C. Koch, M. Mahowald, K. Martin, and H. Suarez. Recurrent excitation in neocortical circuits. *Science*, 269(5226):981–985, 1995.
- [2] S. Grossberg. How does the cerebral cortex work? development, learning, attention, and 3D vision by laminar circuits of visual cortex. *Behavioral and Cognitive Neuroscience Reviews*, 2:47–76, 2003.
- [3] D. V. Buonomano and M. M. Merzenich. Temporal information transformed into a spatial code by a neural network with realistic properties. *Science*, 267:1028–1030, Feb. 1995.
- [4] W. Maass and E. D. Sontag. Neural systems as nonlinear filters. *Neural Computation*, 12(8):1743–1772, 2000.
- [5] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

- [6] S. Häusler and W. Maass. A statistical analysis of information processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, 2007. [Epub ahead of print].
- [7] A. Destexhe and E. Marder. Plasticity in single neuron and circuit computations. *Nature*, 431:789–795, 2004.
- [8] W. Maass, T. Natschläger, and H. Markram. Fading memory and kernel properties of generic cortical microcircuit models. *Journal of Physiology – Paris*, 98(4–6):315–330, 2004.
- [9] M. I. Leon and M. N. Shadlen. Representation of time by neurons in the posterior parietal cortex of the macaque. *Neuron*, 38(2):317–322, 2003.
- [10] K. Hikosaka and M. Watanabe. Delay activity of orbital and lateral prefrontal neurons of the monkey varying with different rewards. *Cerebral Cortex*, 10(3):263–267, 2000.
- [11] L. Tremblay and W. Schultz. Modifications of reward expectation-related neuronal activity during learning in primate orbitofrontal cortex. *J Neurophysiol*, 83(4):1877–1885, 2000.
- [12] W. Schultz, L. Tremblay, and J. R. Hollerman. Changes in behavior-related neuronal activity in the striatum during learning. *Trends Neurosci*, 26(6):321–328, 2003.
- [13] X. J. Wang. Synaptic reverberation underlying mnemonic persistent activity. *Trends Neurosci.*, 24(8):455–463, 2001.
- [14] M. E. Mazurek, J. D. Roitman, J. Ditterich, and M. N. Shadlen. A role for neural integrators in perceptual decision making. *Cerebral Cortex*, 13(11):1257–1269, 2003.

- [15] G. Major, R. Baker, E. Aksay, B. Mensh, H. S. Seung, and D. W. Tank. Plasticity and tuning by visual feedback of the stability of a neural integrator. *Proc Natl Acad Sci*, 101(20):7739–7744, 2004.
- [16] M.N. Shadlen and J.I. Gold. The neurophysiology of decision-making as a window on cognition. In M. S. Gazzaniga, editor, *The Cognitive Neurosciences*, pages 1229–1241. MIT Press, 3rd edition, 2005.
- [17] H. Jäger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, 304:78–80, 2004.
- [18] P. Joshi and W. Maass. Movement generation with circuits of spiking neurons. *Neural Computation*, 17(8):1715–1738, 2005.
- [19] E. L. White. *Cortical Circuits*. Birkhaeuser, Boston, 1989.
- [20] O. Sporns and R. Kötter. Motifs in brain networks. *PLOS Biology*, 2:1910–1918, 2004.
- [21] F. Rieke, D. Warland, R. R. D. van Steveninck, and W. Bialek. *SPIKES: Exploring the Neural Code*. MIT Press, Cambridge, MA, 1997.
- [22] J. D. Cowan. Statistical mechanics of neural nets. In E. R. Caianiello, editor, *Neural Networks*, pages 181–188. Springer, Berlin, 1968.
- [23] M. A. Cohen and S. Grossberg. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Trans. Systems, Man, and Cybernetics*, 13:815–826, 1983.
- [24] J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proc. Nat. Acad. Sci. USA*, 81:3088–3092, 1984.
- [25] P. Dayan and L. F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. MIT-Press, 2001.

- [26] R. A. Legenstein and W. Maass. Edge of chaos and prediction of computational power for neural microcircuit models. *submitted for publication*, 2006.
- [27] J. E. Savage. *Models of Computation: Exploring the Power of Computing*. Addison-Wesley (Reading, MA), 1998.
- [28] W. Maass and H. Markram. Theory of the computational function of microcircuit dynamics. In S. Grillner and A. M. Graybiel, editors, *The Interface between Neurons and Global Brain Function*, Dahlem Workshop Report 93. MIT Press, 2006. in press.
- [29] M. S. Branicky. Universal computation and other capabilities of hybrid and continuous dynamical systems. *Theoretical Computer Science*, 138:67–100, 1995.
- [30] H. Siegelmann and E. D. Sontag. Analog computation via neural networks. *Theoretical Computer Science*, 131(2):331–360, 1994.
- [31] H. Siegelmann and E. D. Sontag. On the computational power of neural nets. *Journal of Computer and System Sciences*, 50:132–150, 1995.
- [32] P. Orponen. A survey of continuous-time computation theory. In D.-Z. Du and K.-I. Ko, editors, *Advances in Algorithms, Languages, and Complexity*, pages 209–224. Kluwer Academic Publishers, 1997.
- [33] J.-J. E. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, 1991.
- [34] E. D. Sontag. *Mathematical Control Theory*. Springer-Verlag, 1998.
- [35] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, New Jersey, 2nd edition, 1999.
- [36] W. Maass and P. Orponen. On the effect of analog noise in discrete-time analog computations. *Neural Computation*, 10:1071–1095, 1998.

- [37] W. Maass and E. Sontag. Analog neural nets with Gaussian or other common noise distribution cannot recognize arbitrary regular languages. *Neural Computation*, 11:771–782, 1999.
- [38] W. Maass and H. Markram. On the computational power of recurrent circuits of spiking neurons. *Journal of Computer and System Sciences*, 69(4):593–616, 2004.
- [39] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- [40] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA*, 79:2554–2558, 1982.
- [41] D. J. Amit and N. Brunel. Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cerebral Cortex*, 7(3):237–252, 1997.
- [42] C. D. Brody, R. Romo, and A. Kepecs. Basic mechanisms for graded persistent activity: discrete attractors, continuous attractors, and dynamic representations. *Curr Opin Neurobiol*, 13(2):204–211, 2003.
- [43] A. Destexhe, M. Rudolph, and D. Pare. The high-conductance state of neocortical neurons in vivo. *Nat. Rev. Neurosci.*, 4(9):739–751, 2003.
- [44] H. Markram, Y. Wang, and M. Tsodyks. Differential signaling via the same axon of neocortical pyramidal neurons. *PNAS*, 95:5323–5328, 1998.
- [45] A. Gupta, Y. Wang, and H. Markram. Organizing principles for a diversity of GABAergic interneurons and synapses in the neocortex. *Science*, 287:273–278, 2000.
- [46] G. Major, R. Baker, E. Aksay, H. S. Seung, and D. W. Tank. Plasticity and tuning of the time course of analog persistent firing in a neural integrator. *Proc Natl Acad Sci*, 101(20):7745–7750, 2004.

- [47] R. A. Legenstein, C. Näger, and W. Maass. What can a neuron learn with spike-timing-dependent plasticity? *Neural Computation*, 17(11):2337–2382, 2005.
- [48] J. Wickens and R. Kötter. Cellular models of reinforcement. In J. C. Houk, J. L. Davis, and D. G. Beiser, editors, *Models of Information Processing in the Basal Ganglia*. MIT Press, Cambridge, 1998.
- [49] A. Destexhe, M. Rudolph, J. M. Fellous, and T. J. Sejnowski. Fluctuating synaptic conductances recreate in vivo-like activity in neocortical neurons. *Neuroscience*, 107(1):13–24, 2001.
- [50] A. Destexhe and D. Pare. Impact of network activity on the integrative properties of neocortical pyramidal neurons in vivo. *J. Neurophysiol.*, 81(4):1531–1547, 1999.
- [51] D. A. Hoffman, J. C. Magee, C. M. Colbert, and D. Johnston. K⁺ channel regulation of signal propagation in dendrites of hippocampal pyramidal neurons. *Nature*, 387(6636):869–875, 1997.
- [52] J. C. Magee and D. Johnston. Characterization of single voltage-gated Na⁺ and Ca²⁺ channels in apical dendrites of rat CA1 pyramidal neurons. *J. Physiol.*, 487 (Pt 1):67–90, 1995.
- [53] J. Magee, D. Hoffman, C. Colbert, and D. Johnston. Electrical and calcium signaling in dendrites of hippocampal pyramidal neurons. *Annu. Rev. Physiol.*, 60:327–346, 1998.
- [54] G. J. Stuart and B. Sakmann. Active propagation of somatic action potentials into neocortical pyramidal cell dendrites. *Nature*, 367(6458):69–72, 1994.
- [55] R. D. Traub and R. Miles. *Neuronal Networks of the Hippocampus*. Cambridge Univ. Press, Cambridge, UK, 1991.

- [56] Z. T. Mainen, J. Joerges, J. R. Huguenard, and T. J. Sejnowski. A model of spike initiation in neocortical pyramidal neurons. *Neuron*, 15(6):1427–1439, 1995.
- [57] J. R. Huguenard, O. P. Hamill, and D. A. Prince. Developmental changes in Na^+ conductances in rat neocortical neurons: appearance of a slowly inactivating component. *J. Neurophysiol.*, 59:778–795, 1988.
- [58] J. Anderson, I. Lampl, I. Reichova, M. Carandini, and D. Ferster. Stimulus dependence of two-state fluctuations of membrane potential in cat visual cortex. *Nature Neuroscience*, 3(6):617–621, 2000.
- [59] L. J. Borg-Graham, C. Monier, and Y. Fregnac. Visual input evokes transient and strong shunting inhibition in visual cortical neurons. *Nature*, 393(6683):369–373, 1998.
- [60] J. A. Hirsch, J. M. Alonso, R. C. Reid, and L. M. Martinez. Synaptic integration in striate cortical simple cells. *J. Neurosci.*, 18(22):9517–9528, 1998.
- [61] W. Maass and H. Markram. Synapses as dynamic memory buffers. *Neural Networks*, 15:155–161, 2002.
- [62] T. Natschläger, H. Markram, and W. Maass. Computer models and analysis tools for neural microcircuits. In R. Kötter, editor, *Neuroscience Databases. A Practical Guide*, chapter 9, pages 123–138. Kluwer Academic Publishers (Boston), 2003.

Figure 1: Computational architectures considered in Theorems 1 and 2. **(A)** A fixed circuit C whose dynamics is described by the system (3). **(B)** An arbitrary given n^{th} order dynamical system (4) with external input $u(t)$. **(C)** If the input $v(t)$ to circuit C is replaced by a suitable feedback $K(\mathbf{x}(t), u(t))$, then this fixed circuit C can simulate the dynamic response $z(t)$ of the arbitrarily given system shown in B, for any input stream $u(t)$. **(D)** Arbitrary given finite state machine (FSM) A with l states. **(E)** A noisy fading memory system with feedback can reliably reproduce the current state $A(t)$ of the given FSM A , except for time points t shortly after A has switched its state.

Figure 2: Organization of input- and output streams for the 3 computational tasks considered in the computer simulations. Each input stream consisted of multiple spike trains, that provided synaptic inputs to individually chosen subsets of neurons in the recurrent circuit (which is indicated by a gray rectangle). In **(A)** and **(C)** these input streams consisted of multiple Poisson spike trains with a time-varying firing rate $r_i(t)$. In **(B)** the input consisted of a burst (“cue”) in one spike train (which marks the beginning of a time interval) and independent Poisson spike train (“noise”) in the other input channels. The actual outputs of the readouts (that were trained individually for each computational task) in panels **(A, B, C)** is shown in Figures 3, 4, 5.

Figure 3: State-dependent real-time processing of 4 independent input streams in a generic cortical microcircuit model. **(A)** 4 input streams, consisting each of 8 spike trains generated by Poisson processes with randomly varying rates $r_i(t), i = 1, \dots, 4$ (rates plotted in **(B)**; all rates are given in Hz). The 4 input streams and the feedback were injected into disjoint sets of neurons in the circuit. **(C)** Resulting firing activity of 100 out of the 600 I&F neurons in the circuit. Spikes from inhibitory neurons marked in red. **(D)** Target activation times of the high-dimensional attractor (blue shading), spike trains of 2 of the 8 I&F neurons that were trained to create the high-dimensional attractor by

sending their output spike trains back into the circuit, and average firing rate of all 8 neurons (lower trace). **(E and F)** Performance of linear readouts that were trained to switch their real-time computation task in dependence of the current state of the high-dimensional attractor: output $2 \cdot r_3(t)$ instead of $r_3(t)$ if the high-dimensional attractor is on (E), output $r_3(t) + r_4(t)$ instead of $|r_3(t) - r_4(t)|$ if the high-dimensional attractor is on (F). **(G)** Performance of linear readout that was trained to output $r_3(t) \cdot r_4(t)$, showing that another linear readout from the same circuit can simultaneously carry out nonlinear computations that are invariant to the current state of the high-dimensional attractor.

Figure 4: Representation of time for behaviorally relevant time spans in a generic cortical microcircuit model. **(A)** Afferent circuit input, consisting of a cue in one channel (red) and random spikes (freshly drawn for each trial) in the other channels. **(B)** Response of 100 neurons from the same circuit as in Fig. 3, which has here two co-existing high-dimensional attractors. The autonomously generated periodic bursts with a periodic frequency of about 8 Hz are not related to the task, and readouts were trained to become invariant to them. **(C and D)** Feedback from two linear readouts that were simultaneously trained to create and control two high-dimensional attractors. One of them was trained to decay in 400 ms (C), and the other in 600 ms (D) (scale in nA is the average current injected by feedback into a randomly chosen subset of neurons in the circuit). **(E)** Response of the same neurons as in (B), for the same circuit input, but with feedback from a different linear readout that was trained to create a high-dimensional attractor that increases its activity and reaches a plateau 600 ms after the occurrence of the cue in the input stream. **(F)** Feedback from the linear readout that creates this continuous high-dimensional attractor.

Figure 5: A model for analog real-time computation on external and internal variables in a generic cortical microcircuit (consisting of 600 conductance based HH-neurons). **(A)**

and B) Two input streams as in Fig. 3; their firing rates $r_1(t), r_2(t)$ are shown in (B). **(C)** Resulting firing activity of 100 neurons in the circuit. **(D)** Performance of a neural integrator, generated by feedback from a linear readout that was trained to output at any time t an approximation $CA(t)$ of the integral $\int_0^t (r_1(s) - r_2(s)) ds$ over the difference of both input rates. Feedback values were injected as input currents into a randomly chosen subset of neurons in the circuit. Scale in nA shows average strength of feedback currents (also in panel H). **(E)** Performance of linear readout that was trained to output 0 as long as $CA(t)$ stayed below 0.83 nA, and to output $r_2(t)$ once $CA(t)$ had crossed this threshold, as long as $CA(t)$ stayed above 0.66 nA (i.e., in this test run during the shaded time periods). **(F)** Performance of linear readout trained to output $r_1(t) - CA(t)$, i.e. a combination of external and internal variables, at any time t (both r_1 and CA normalized into the range $[0, 1]$). **(G)** Response of a randomly chosen neuron in the circuit for 10 repetitions of the same experiment (with input spike trains generated by Poisson processes with the same time-course of firing rates), showing biologically realistic trial-to-trial variability. **(H)** Activity traces of a continuous attractor as in (D), but in 8 different trials for 8 different fixed values of r_1 and r_2 (shown on the right).

Figure 6: Emulation of a finite state machine (FSM) by a noisy fading memory system with feedback according to Theorem 5. **(A)** Underlying open loop system with noisy pattern detectors $\hat{F}_1, \dots, \hat{F}_k$ and suitable fading memory readouts $\hat{H}_1, \dots, \hat{H}_l$ (which may also be subject to noise). **(B)** Resulting noise-robust emulation of an arbitrary given FSM by adding feedback to the system in panel A. The same readouts as in A (denoted $CL - \hat{H}_j(t)$ in the closed loop) now encode the current state of the simulated FSM.

Figure 7: Evaluation of the dependence of the performance of the circuit in Fig. 5 on the feedback strength (i.e., on the mean amplitude of current injection from the readout back into neurons in the circuit). For each feedback strength that was evaluated, the

readouts were trained and tested for this feedback strength like for the preceding experiments. Error bars in B-D denote standard error. These control experiments show that the feedback is essential for the performance of the circuit.

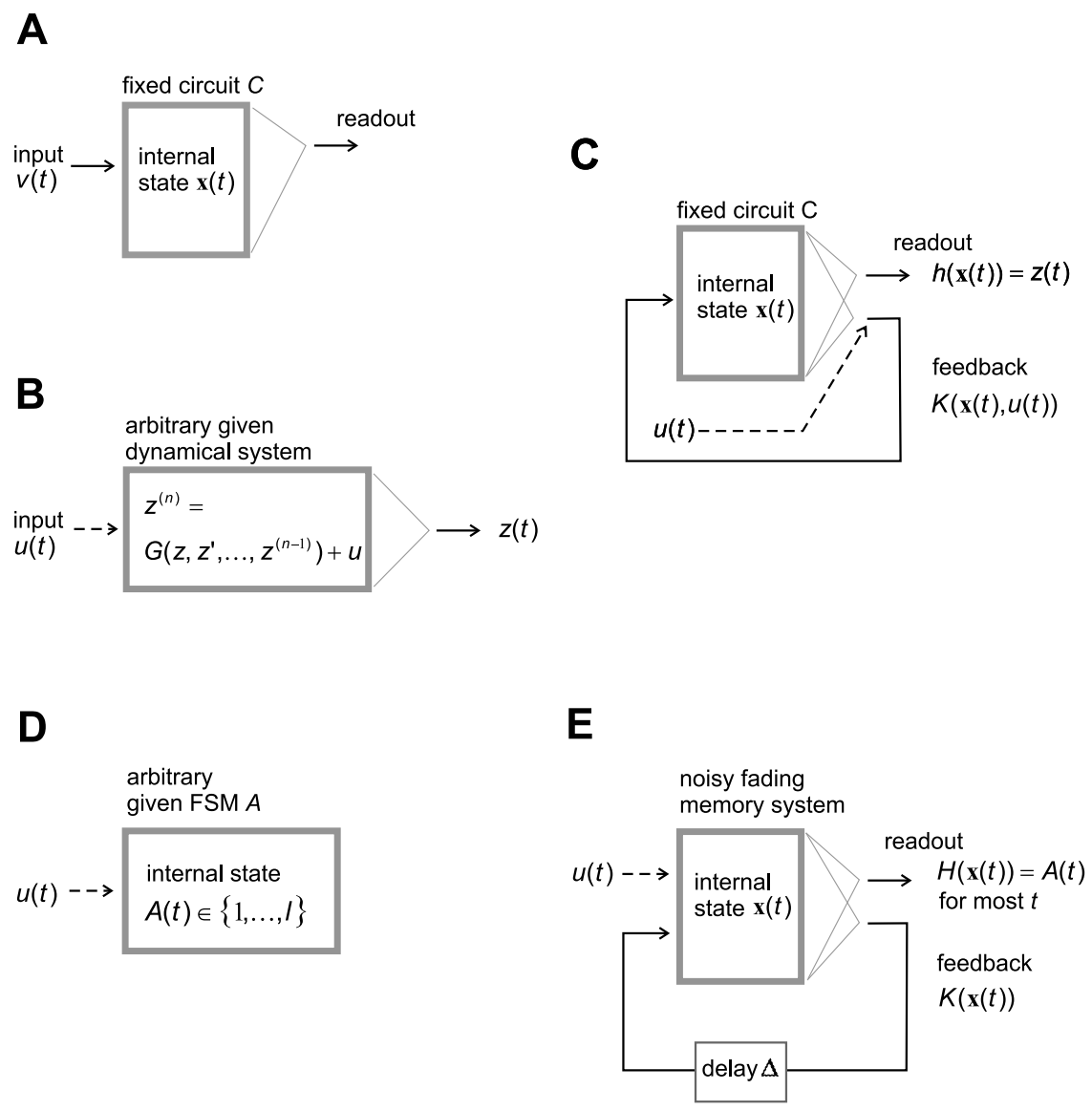
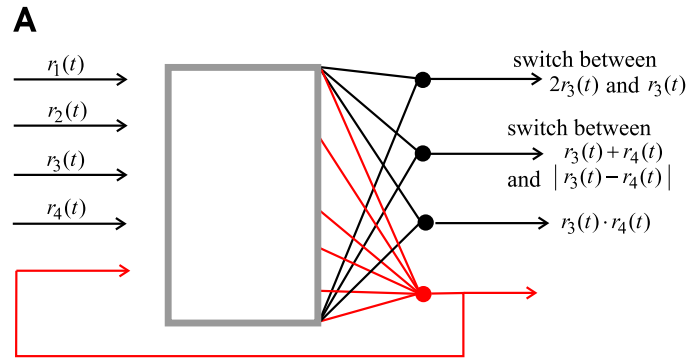


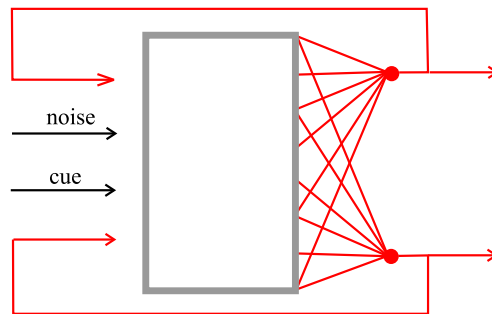
Figure 1:



feedback that creates a discrete attractor

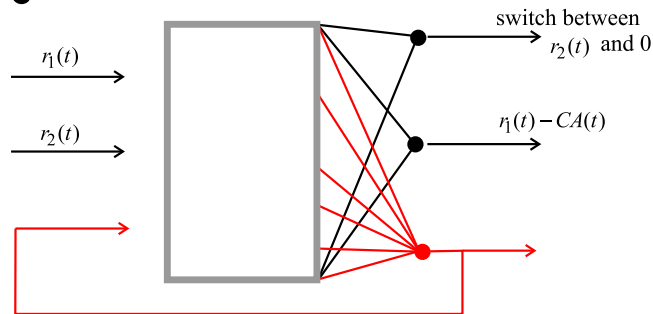
B

feedback that creates a continuous attractor for timing an interval of 600 ms



feedback that creates a continuous attractor for timing an interval of 400 ms

C



feedback that creates a continuous attractor $CA(t)$
 (neural integrator with target value $\int_0^t (r_1(s) - r_2(s)) ds$)

Figure 2:

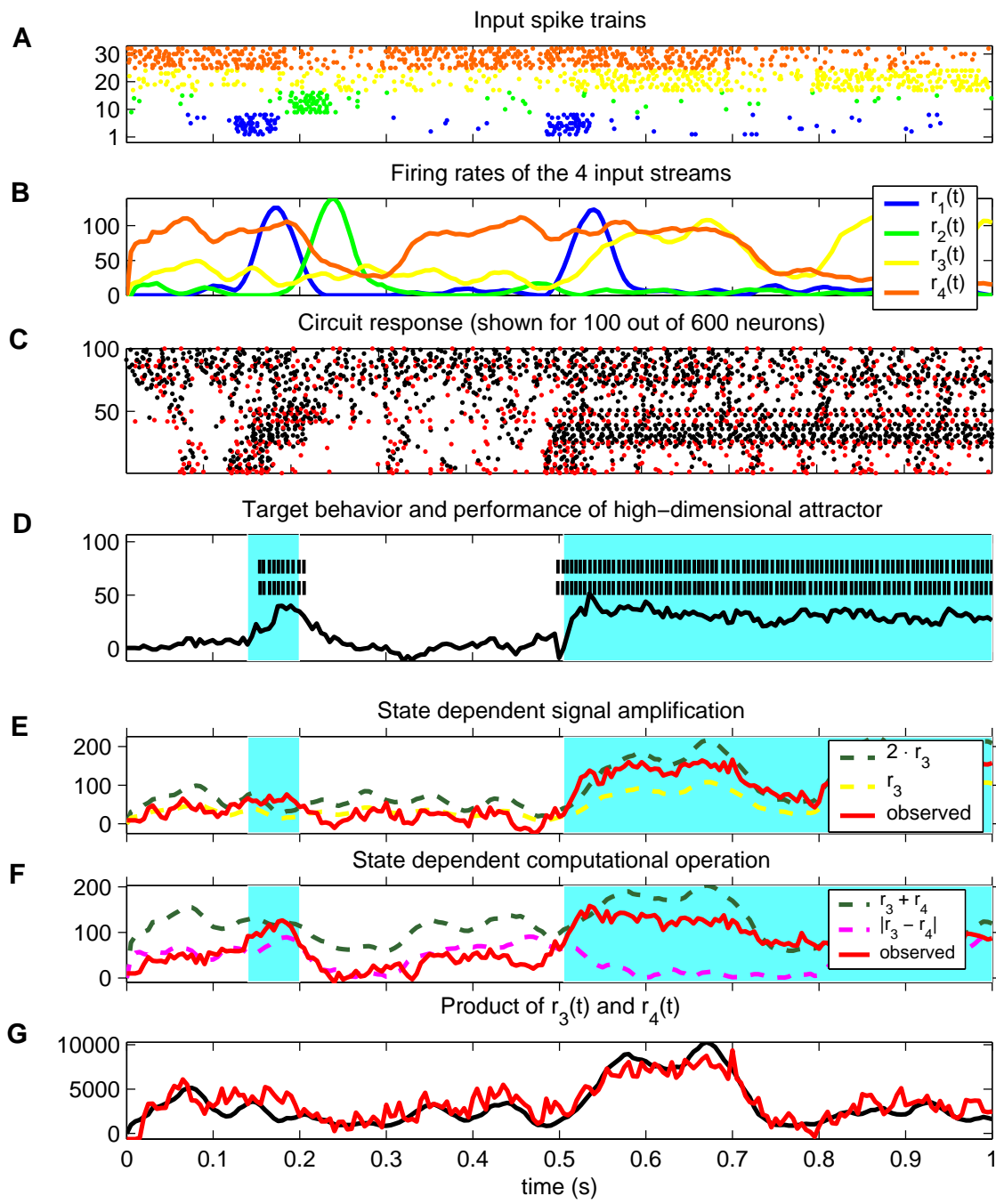


Figure 3:

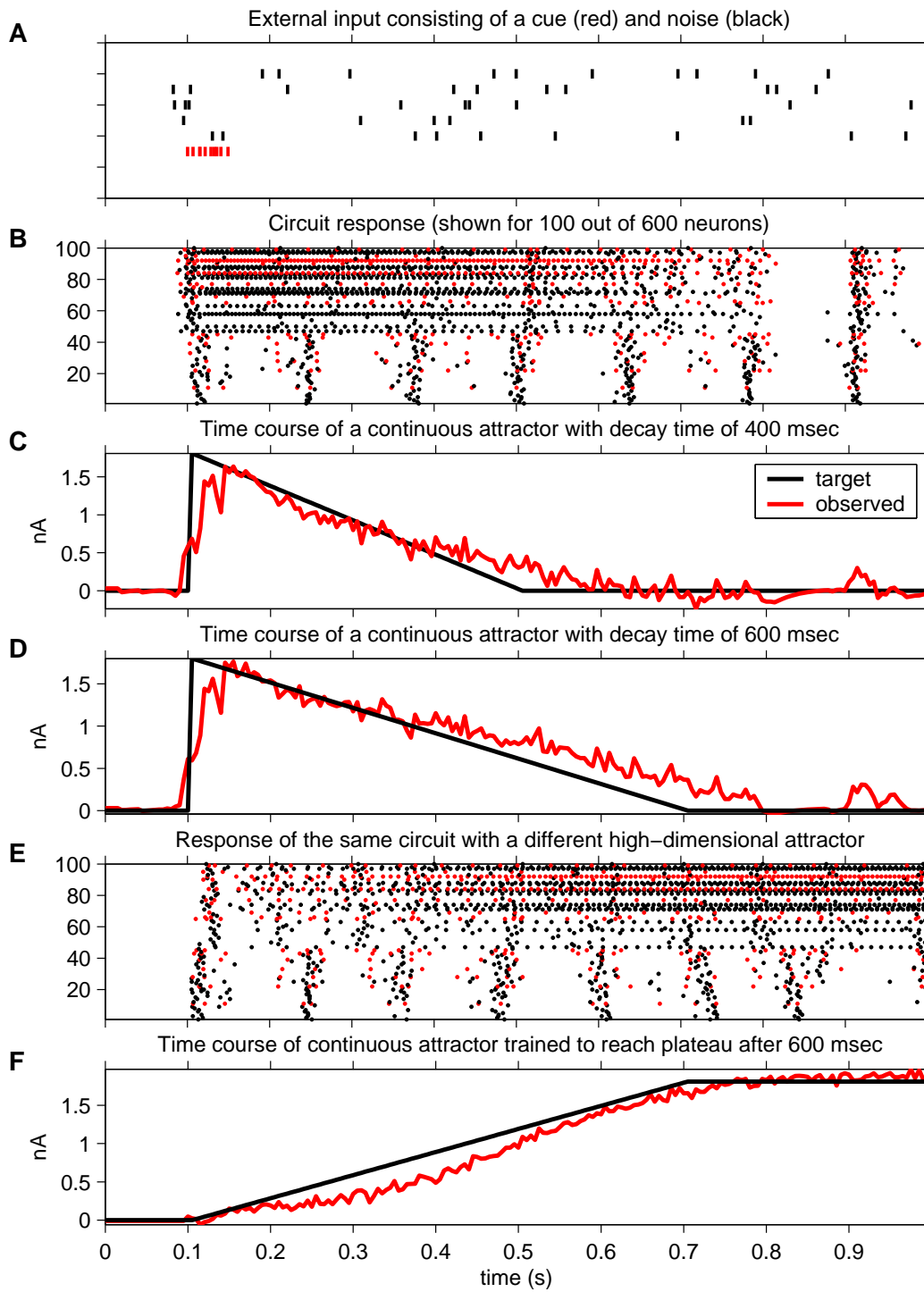


Figure 4:

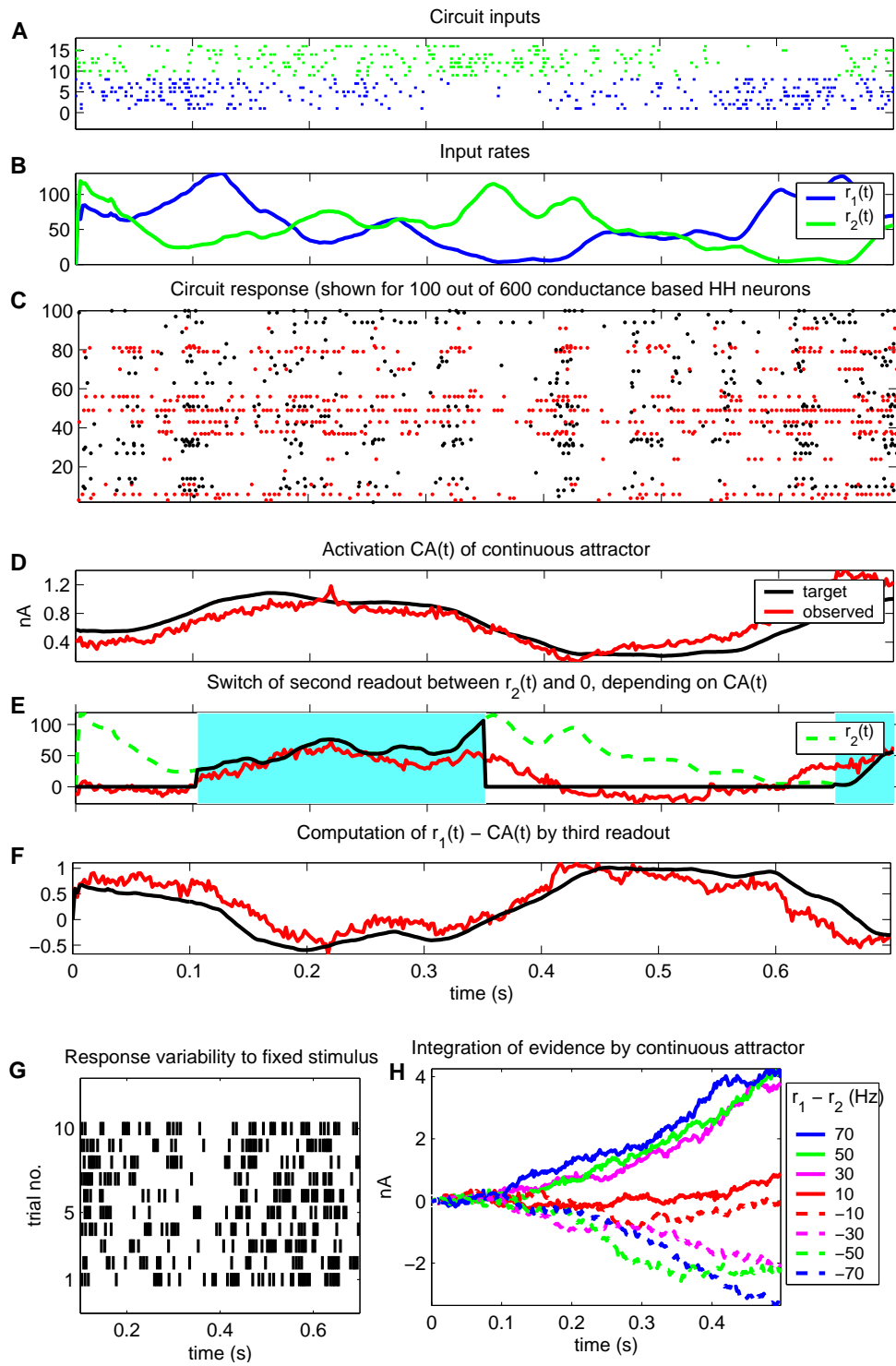


Figure 5:

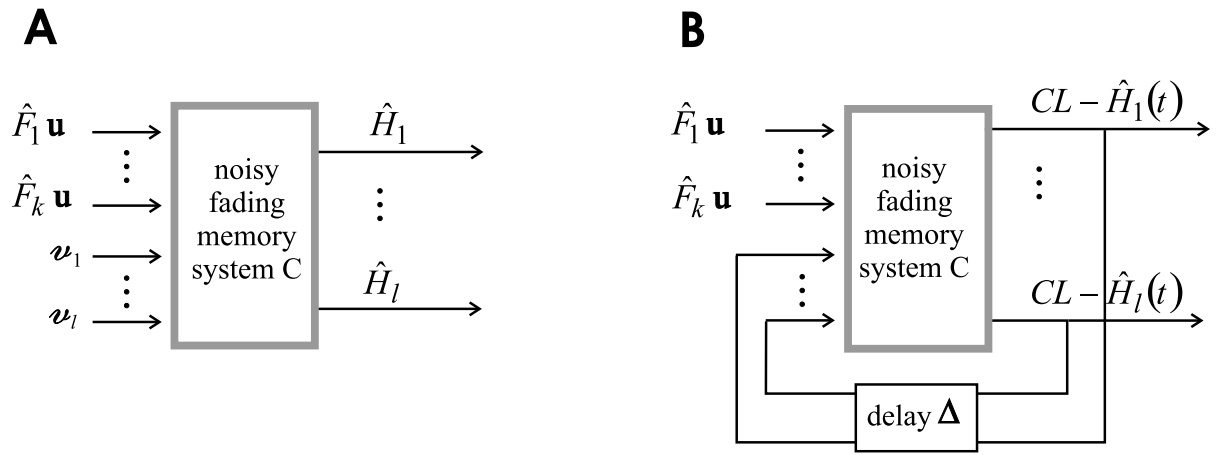


Figure 6:

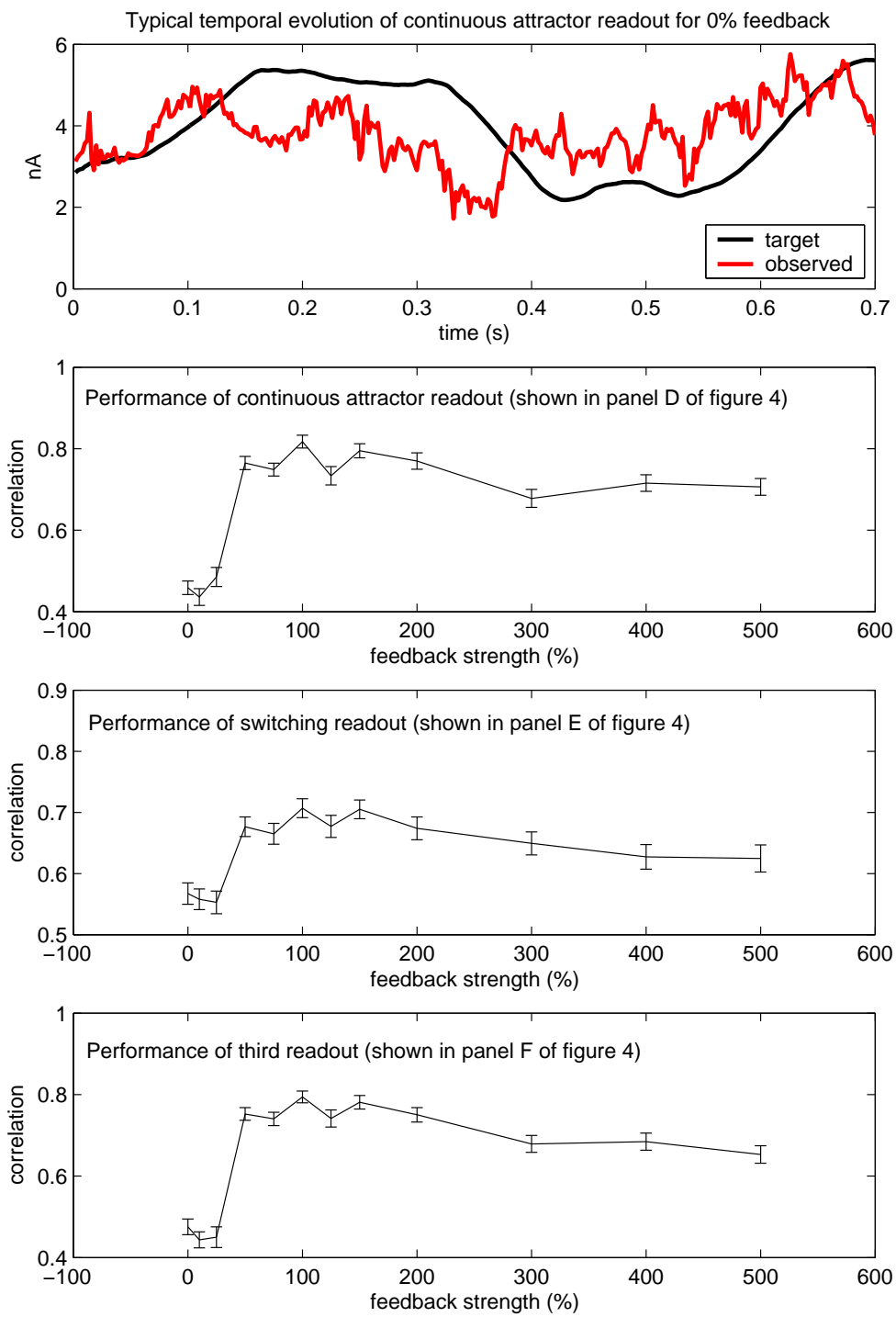


Figure 7: