Online Computing in Dynamical Systems as a Possible Paradigm for Auditory Processing

Wolfgang Maass

Institut für Grundlagen der Informationsverarbeitung
Graz University of Technology, Austria

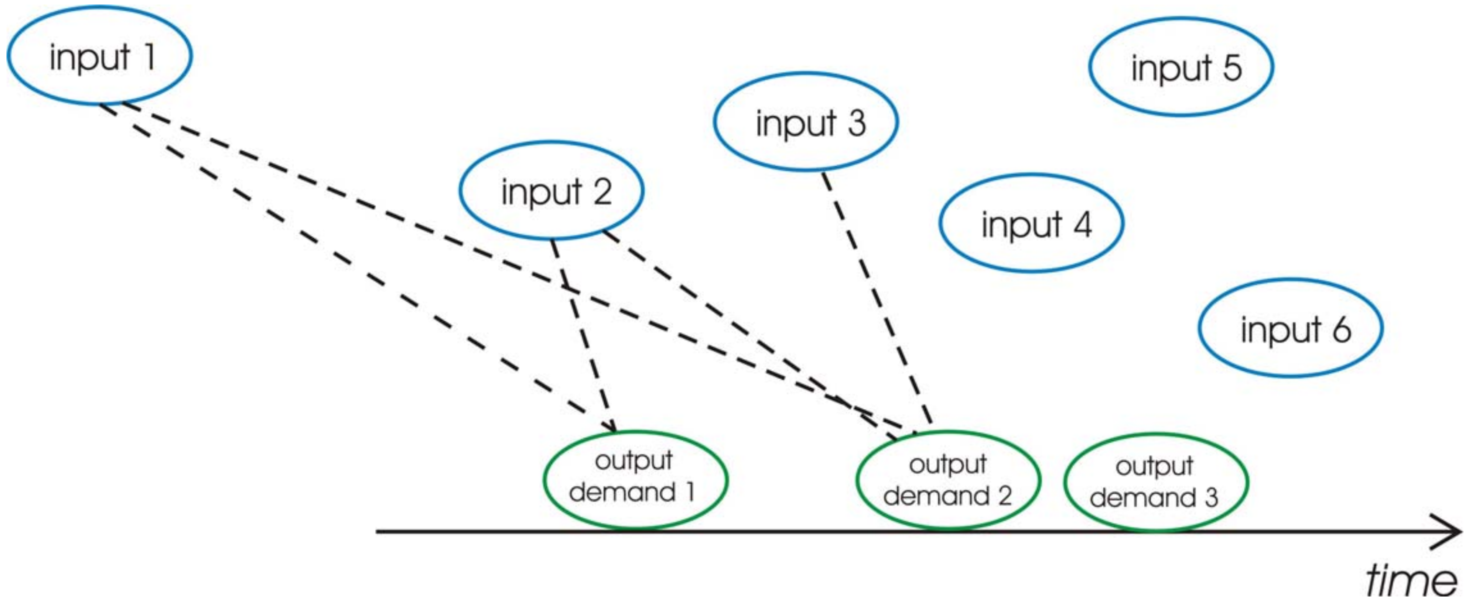
Table of Content

1. Why online computing in dynamical systems ?

Why „online computing“ ?

Classical models for computation (Turing machines, attractor neural networks) do not capture well the actual computational tasks that a biological organism has to perform in order to survive:

It receives continuously new pieces of information, and demands for results of computations may arise at any time (“online computing”, "anytime algorithm", "real-time computing"):



Hence from a mathematical point of view, neural readouts have to implement filters (operators), i.e. they map input streams to output streams (rather than implementing a static function, such as multiplication).

Why online computing in dynamical systems ?

Apparently the first circuits of neurons were already highly recurrent dynamical systems…

The nervous system of C-elegans consists of 302 neurons.

Shown here is the (highly recurrent) subcircuit for sensory processing in the head of C-elegans.

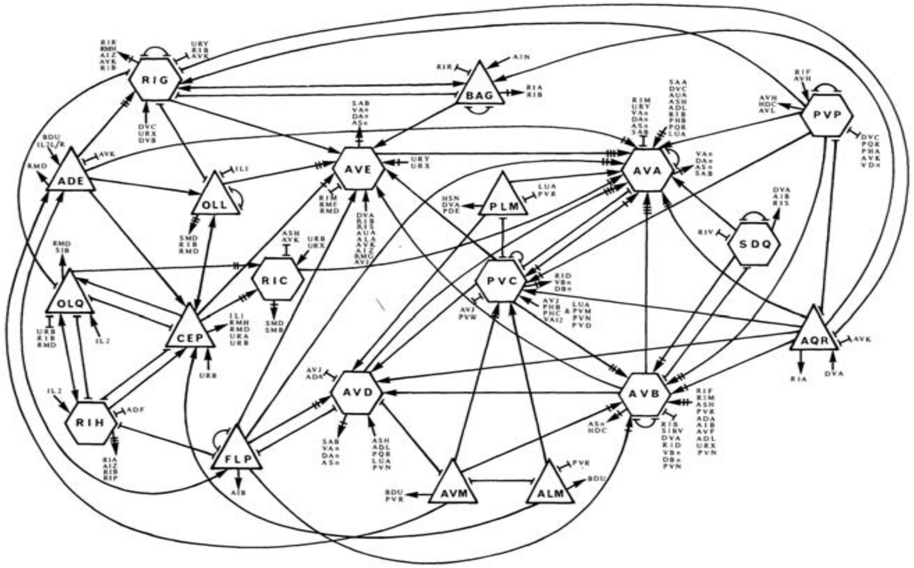


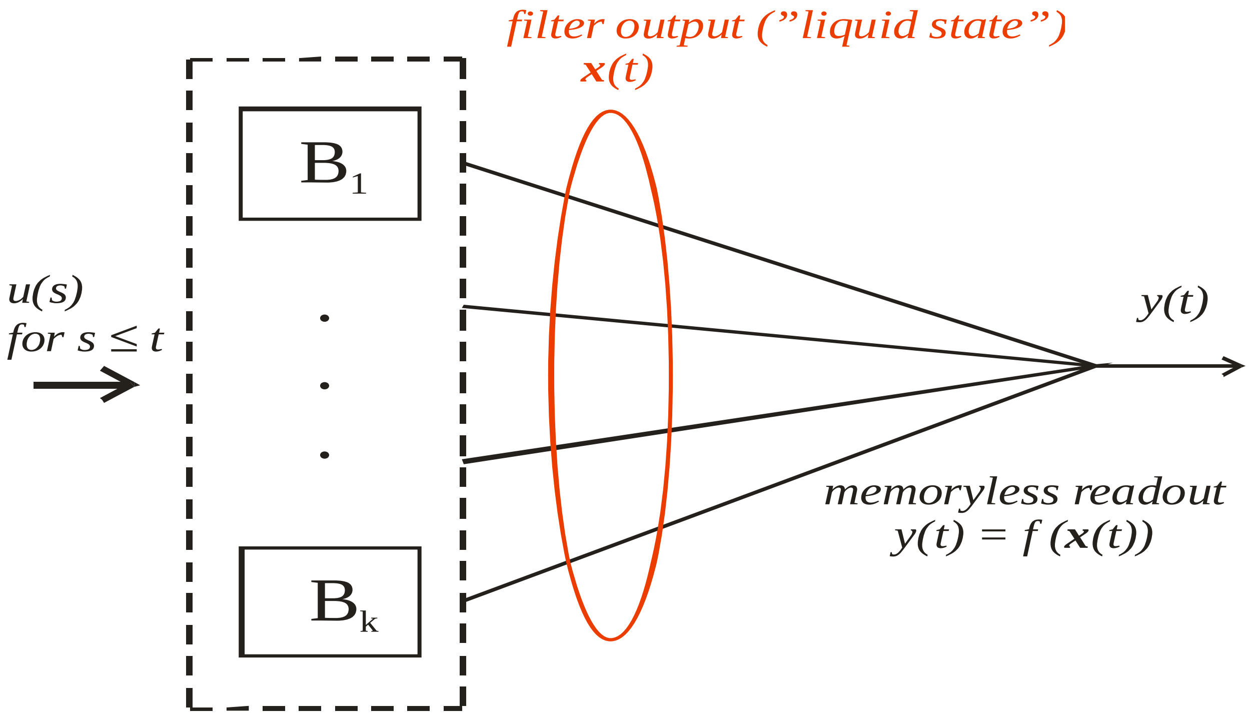
FIGURE 21. (b) Circuitry associated with other sensory receptors in the head.

…and evolution continued to improve the performance of such dynamical systems for more specialized tasks (e.g. for auditory processing).

2. What makes a dynamical system powerful for online computing ?

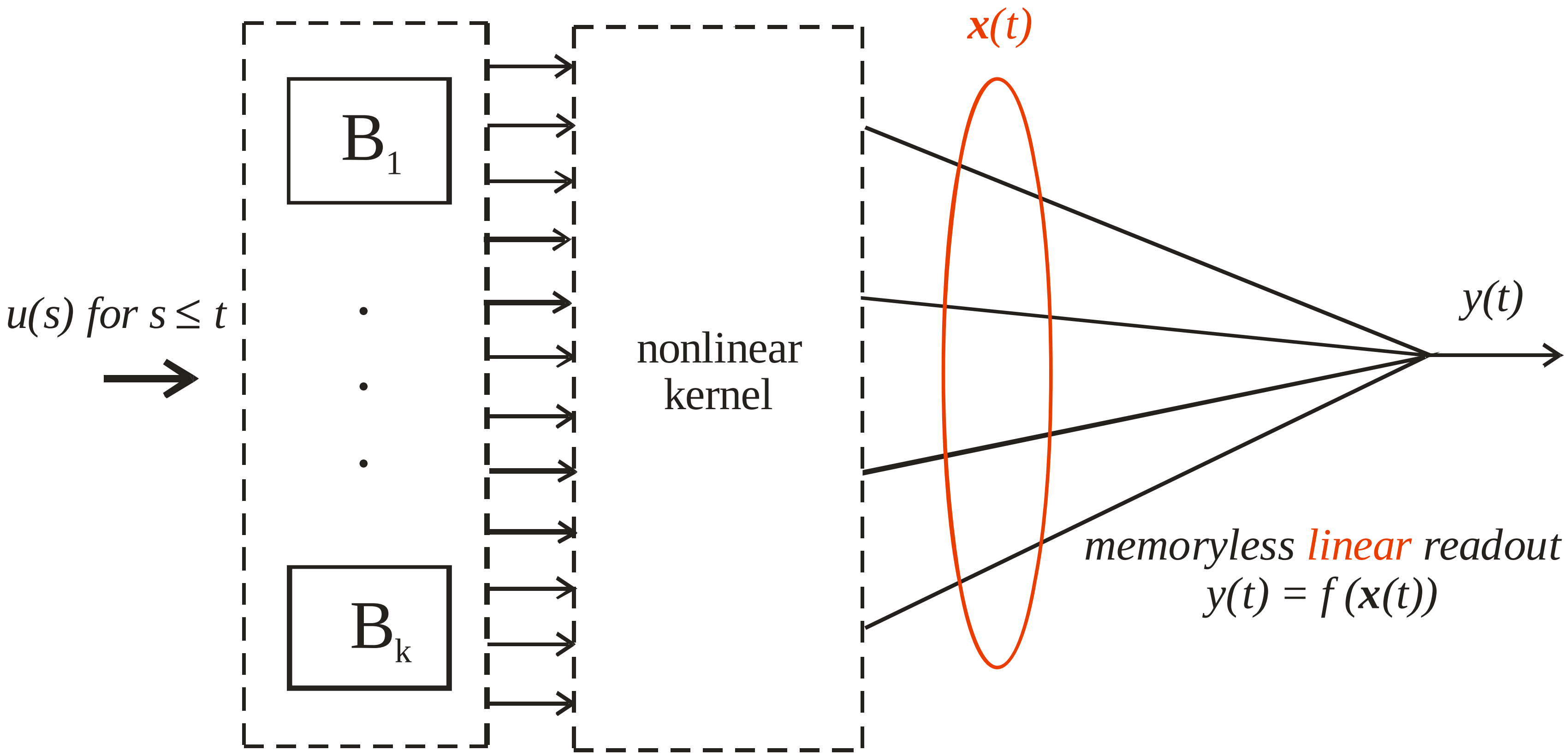
What computational operations does a dynamical system has to perform in order to approximate arbitrary Volterra-series (i.e., fading memory filters) ?

Consider very simple computational models ("liquid state machines") that only consist of a filterbank and a readout unit.



filterbank, consisting of basis filters $B_1, \ldots B_k$

Hence the following functionalities of cortical microcircuits would suffice in order to approximate any filter F that can be defined by Volterra series



Insert: What is a kernel (in the terminology of machine learning) ?

A kernel provides many nonlinear combinations of input variables, in order to boost the expressive power of any subsequent linear readout.

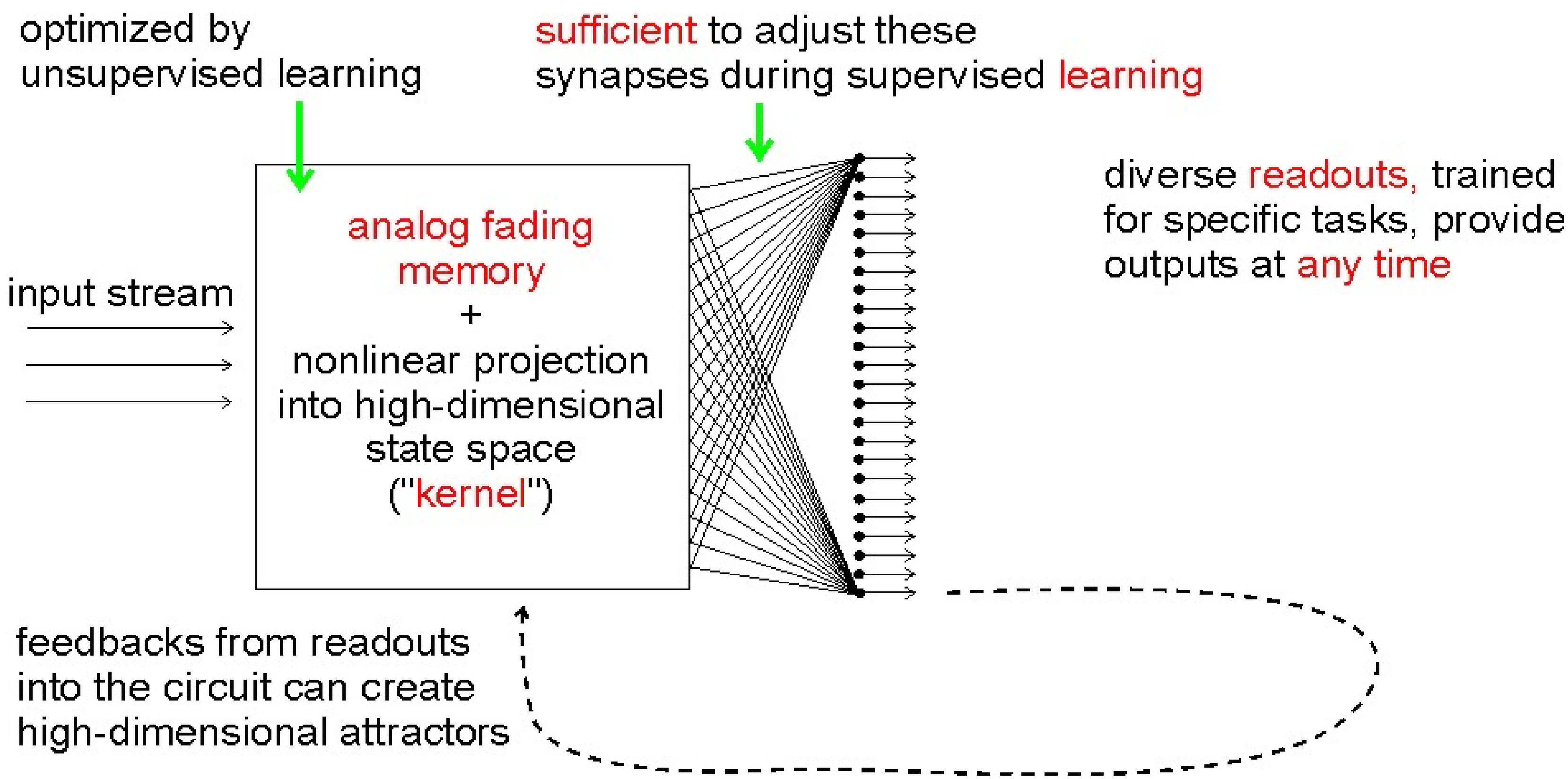
Example: If a circuit precomputes all products $x_i \cdot x_j$ *of n input variables* $x_1,\ldots,x_n$*, then every subsequent linear readout can compute any quadratic function of the original input variables* $x_1,\ldots,x_n$*.*

Remark 1: A clear theoretical advantage of linear readouts: their learning cannot get stuck in local minima of the error function.

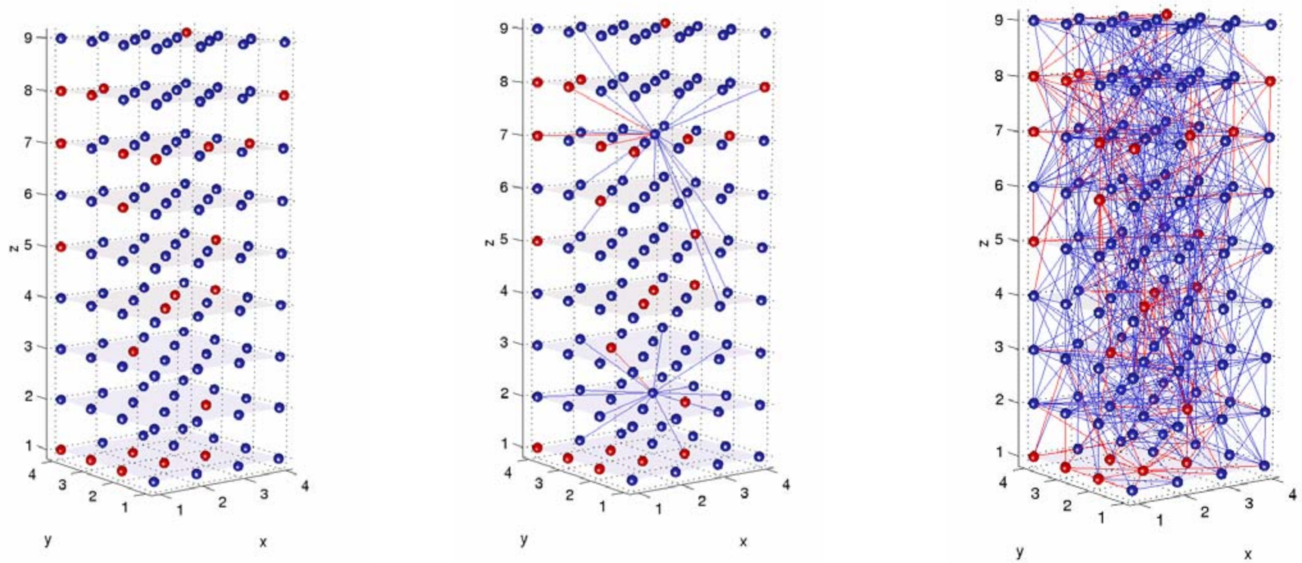
This fact suggests that it is advantageous for nature to restrict learning to linear devices.

Remark 2: Because of Vapnik's "kernel trick" one can use in machine learning big kernels without additional computational cost. This is different for neural circuits that have to implement a kernel explicitly !

Resulting computational model for a generic cortical microcircuit



3. Testing our hypotheses through computer simulations of generic cortical microcircuits



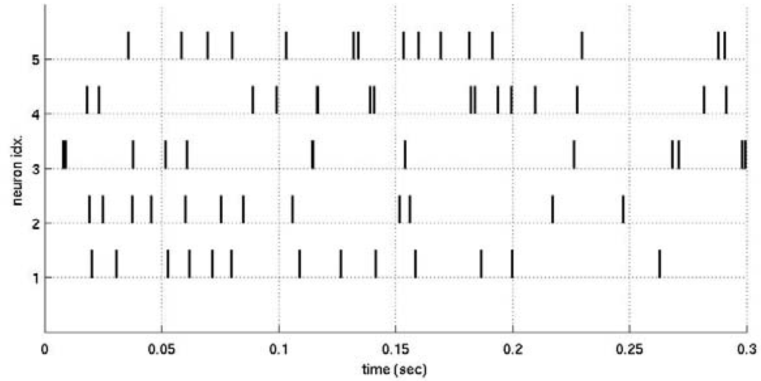
neurons: leaky integrate-and-fire neurons, 20% of them inhibitory, neuron a is synaptically connected to neuron b with probability $C \cdot \exp(-D^2(a,b)/\lambda^2)$

synapses: dynamic synapses with fixed parameters w, U, D, F chosen from distributions based on empirical data from the Lab of Markram

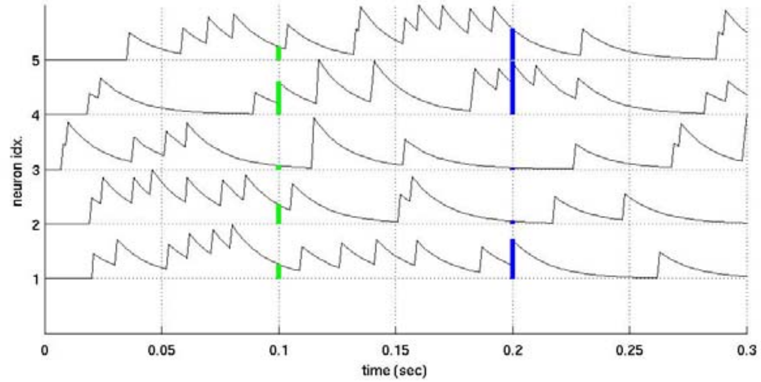
input spike trains injected into 30% randomly chosen neurons, with fixed randomly chosen amplitudes

A simple model for a neural readout:
a linear weighted sum with adaptive weights **w**

Each readout neuron receives as input a vector x(t), which has as many components as it has presynaptic neurons in the circuit.



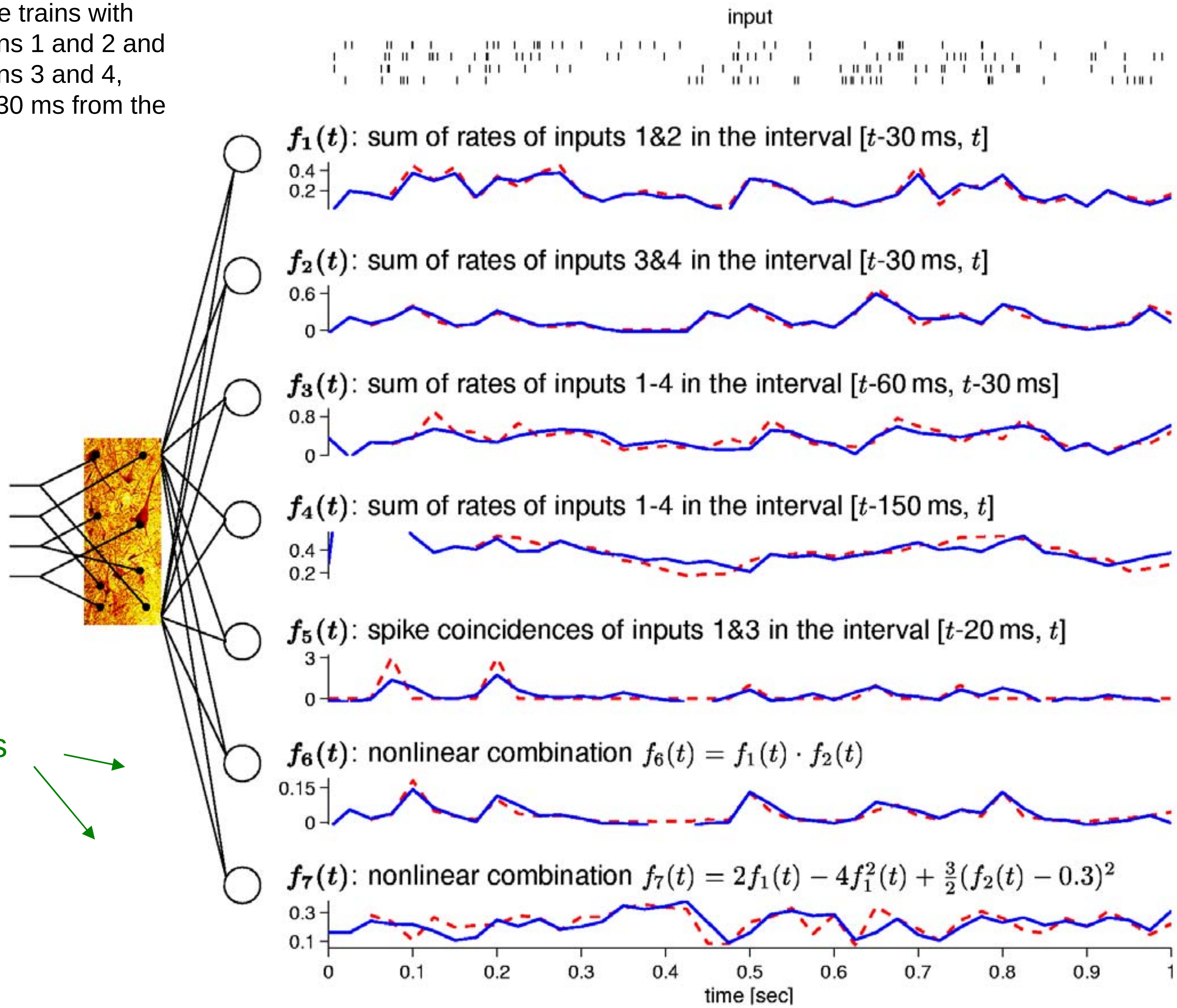
The i-th component of x(t) results from the spike train of the i-th presynaptic neuron by applying a low-pass filter, which models the low-pass filtering properties of receptors and membrane of the readout neuron.



We assume that a readout neuron has at time t a firing rate proportional to $\mathbf{w} \cdot x(t)$.

What can a generic cortical microcircuit compute in this way ?

Circuit input: 4 Poisson spike trains with firing rates $f_1(t)$ for spike trains 1 and 2 and firing rates $f_2(t)$ for spike trains 3 and 4, drawn independently every 30 ms from the interval [0, 80] Hz



7 linear readouts with adjustable weights

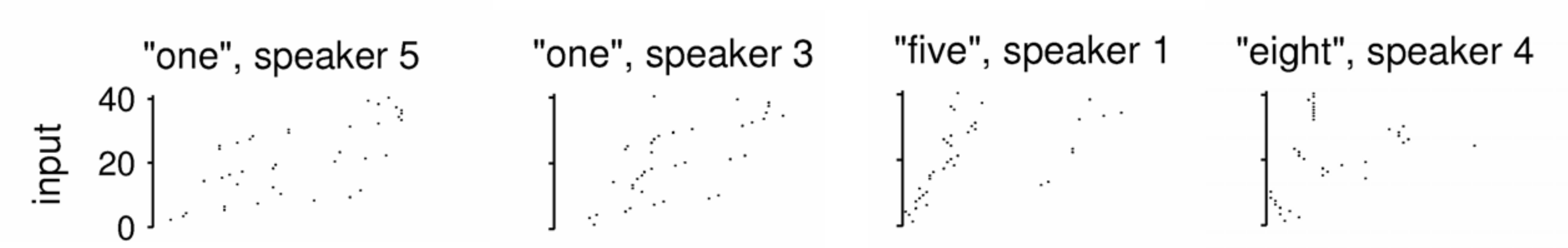
Testing our approach on a popular benchmark task: speech recognition

We consider the task considered by Hopfield and Brody in PNAS 2000 and 2001

(with the same transcription of speech into spike trains):

- recognition of spoken words "zero", "one", ... "nine", each spoken 10 times by 5 different speakers, each spoken word encoded into 40 spike trains by Hopfield and Brody

  (we used 300 examples for training, 200 for testing; note that the circuit constructed by H&B did not require any training)



Comparing the performance of generic diverse circuits and constructed circuits for this task:

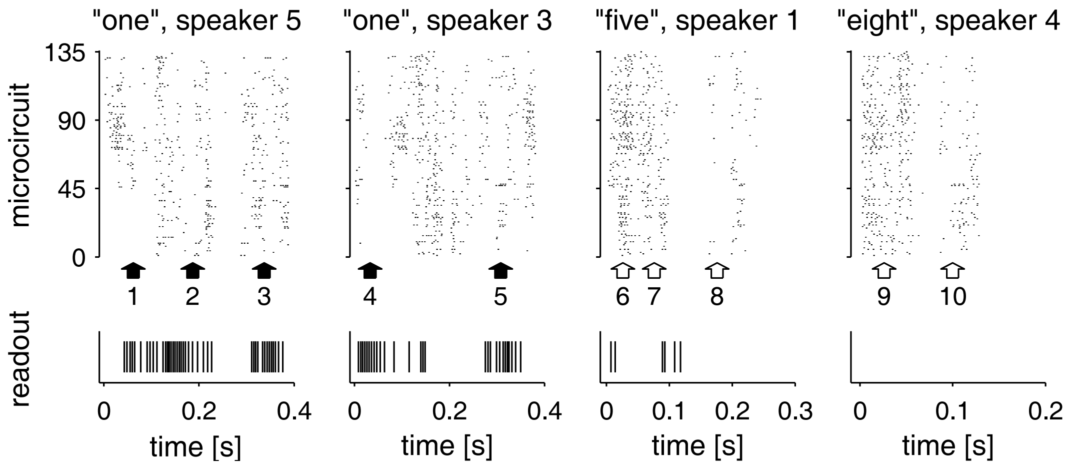
- linear readouts from a generic neural microcircuit model (consisting of 135 neurons) recognize after training spoken test-words as well as the ingenious circuit consisting of >> 6000 I&F neurons constructed especially for this task by Hopfield and Brody

- the generic neural microcircuit model can handle linear time warps in the input at least as well as the circuit constructed to achieve that

  (and it can also handle nonlinear time warps)

- the generic neural microcircuit model classifies the spoken word instantly when the word ends (i.e., in real-time), rather than 300 – 500 ms later

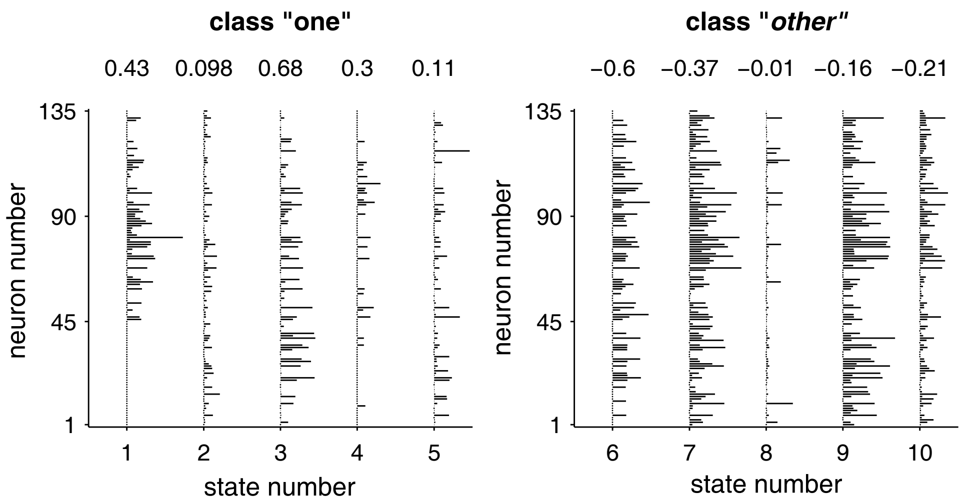
Consider a readout neuron that is trained to fire whenever „one“ is currently spoken

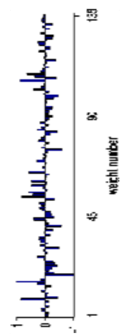


resulting values of $\boldsymbol{w\, x(t)}$

various $\mathbf{x}(t)$

$\mathbf{w}$



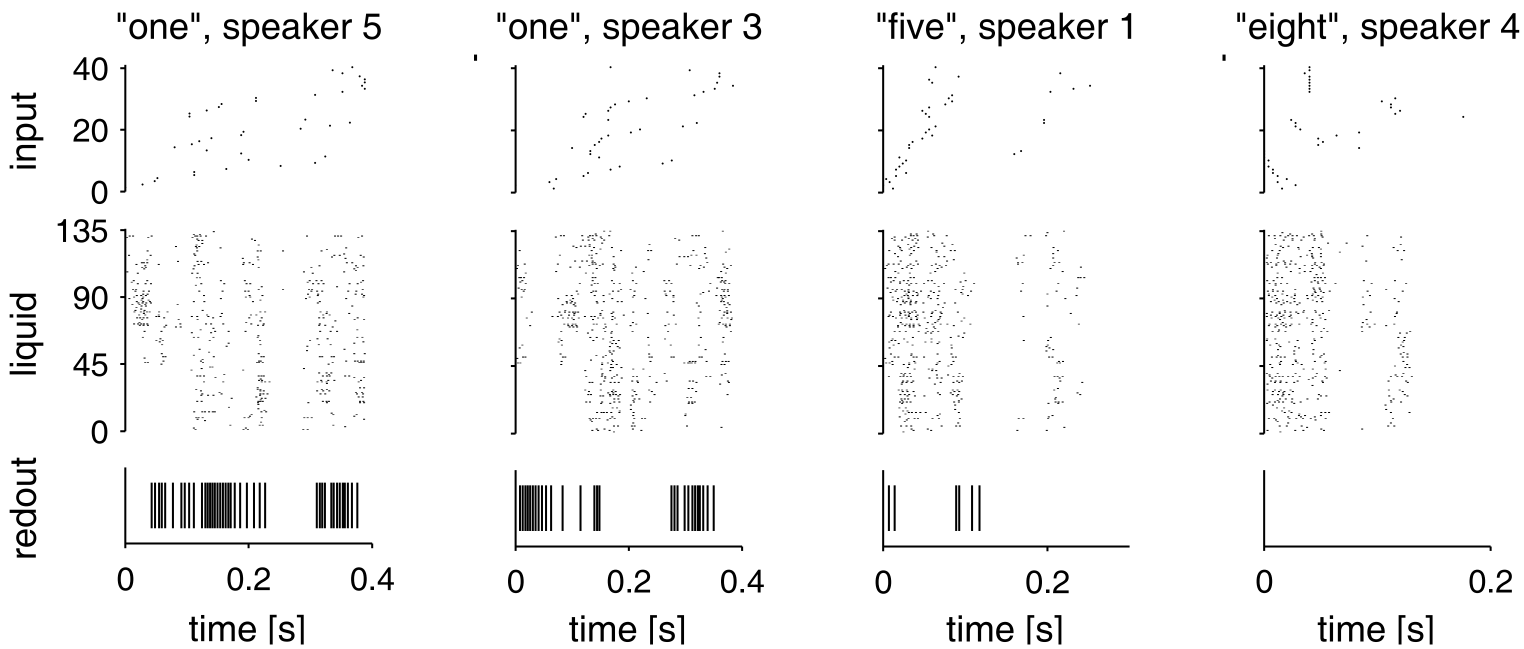


Thus:
linear readouts can form complex equivalence classes

of circuit states $\boldsymbol{x(t)}$

Linear readouts from a generic microcircuit model can also be trained to classify a spoken word (encoded by spike trains), even before the spoken word ends. Hence generic circuit models can implement "anytime algorithms".

Example: anytime recognition of "one" :

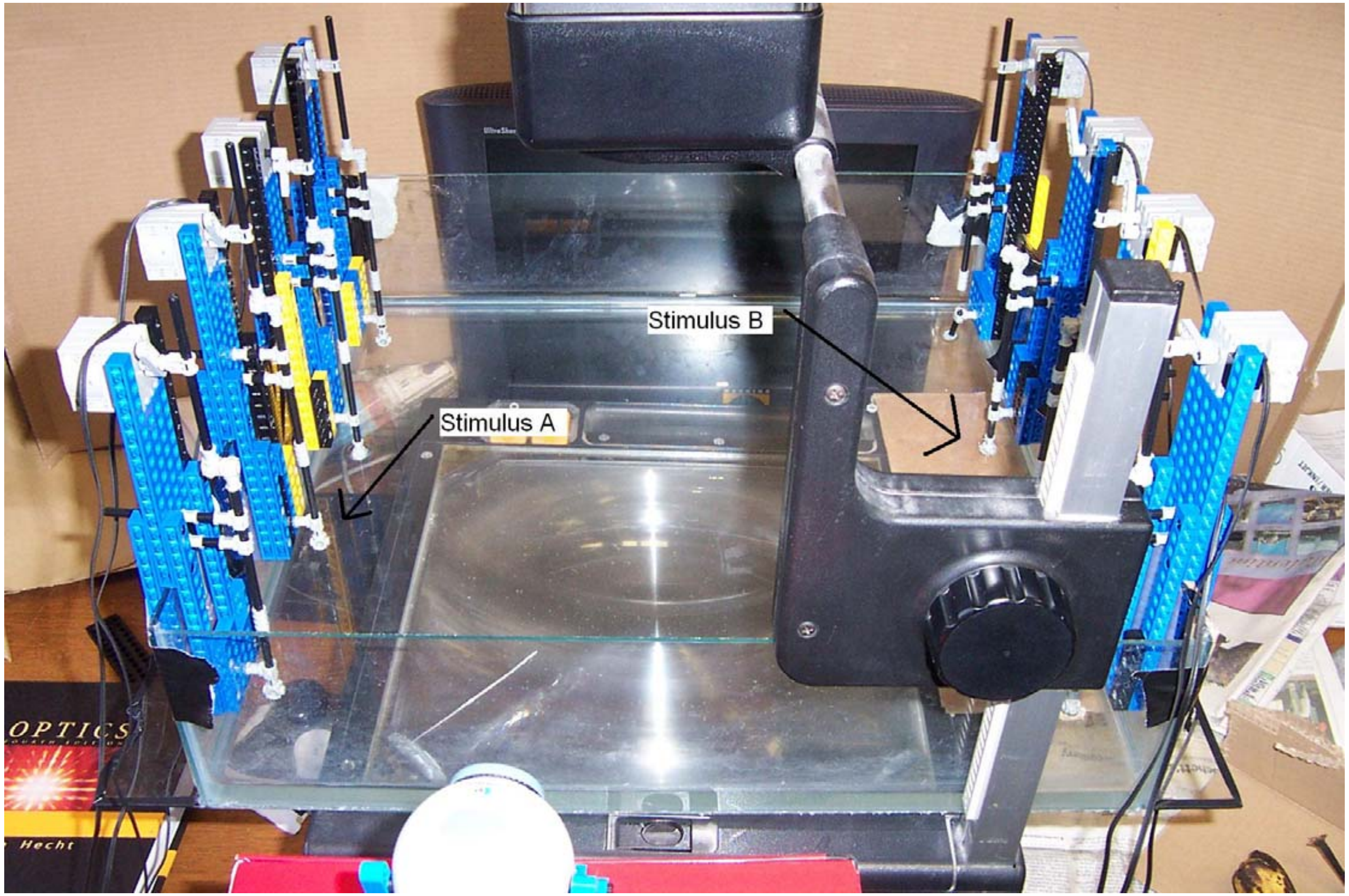


Experimental question:
Can your brain still recognize speech after the consumption of too much liquid?

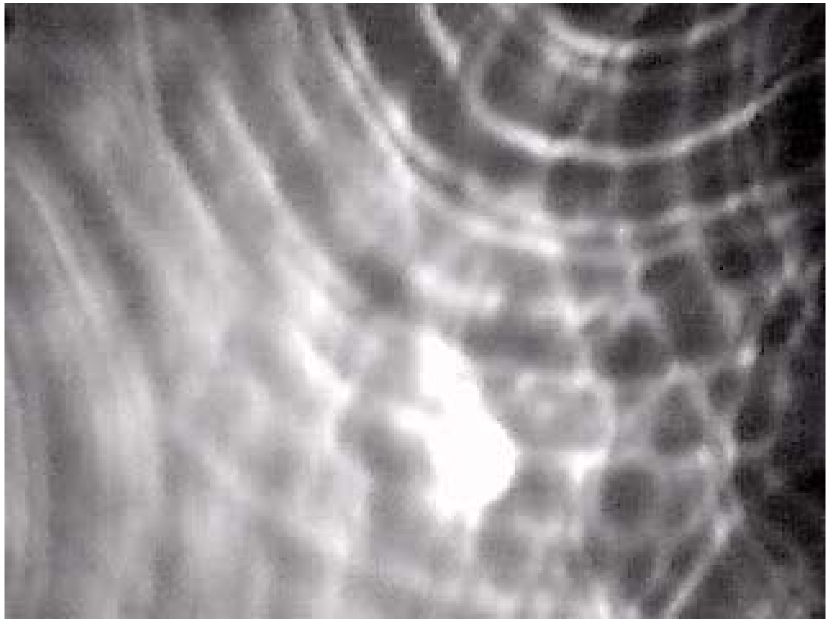
This has been tested by Fernando and Sojakka
(*Pattern recognition in a bucket: A real liquid brain*,
ECAL 2003):

“This paper demonstrates that the waves produced on the surface of water can be used as a medium for a “Liquid State Machine”. We made a bucket of water, vibrated it with lego motors, filmed the waves with a webcam and put it through a perceptron on matlab and got it to solve the XOR problem and do speech recognition.”

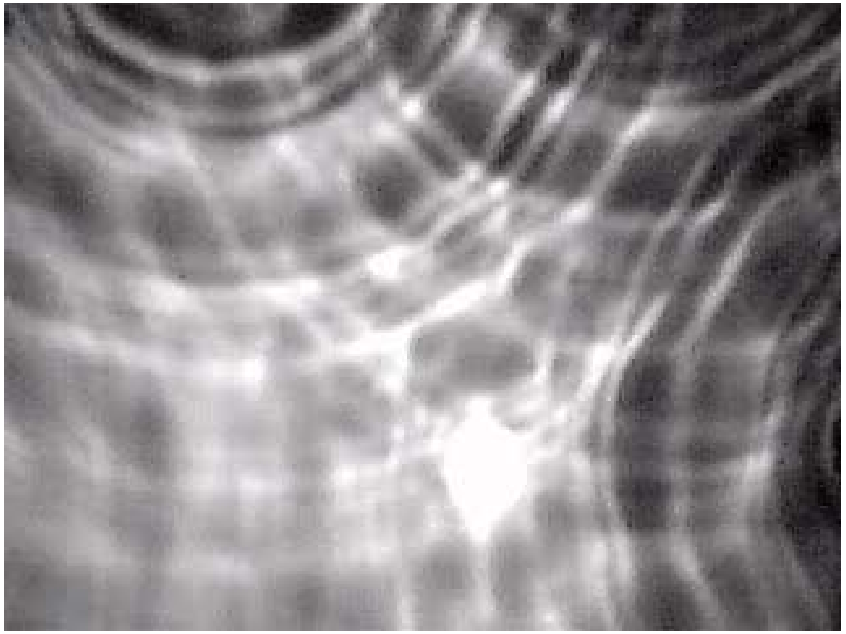
- They injected the same speech data as Hopfield and Brody into a bucket of water.



Examples for „liquid states“ in a bucket of water:



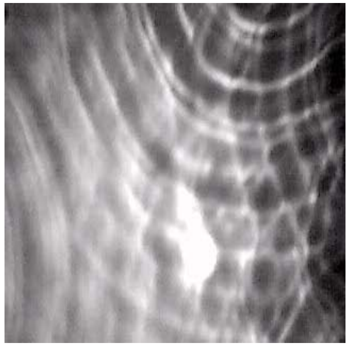
Zero

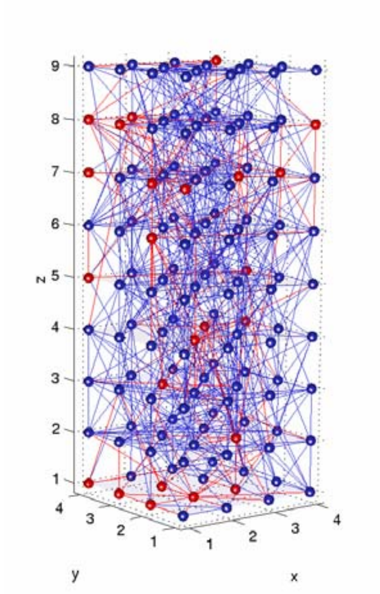


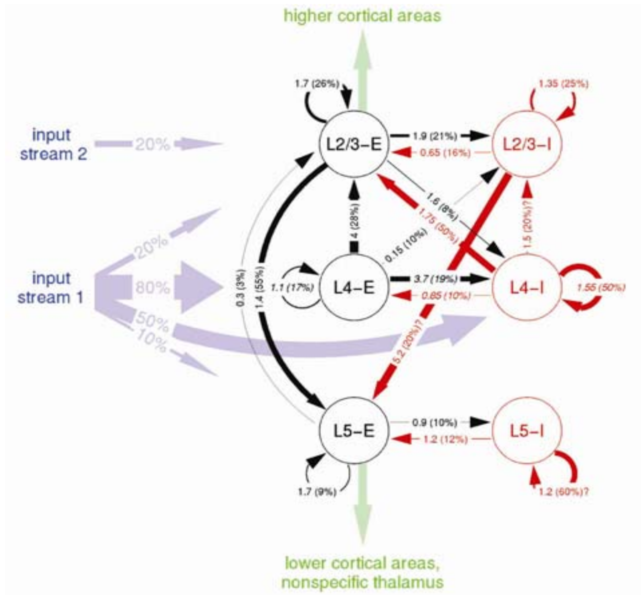
One

Food for thought: Both cortical microcircuits and buckets of water are high-dimensional dynamical systems. Can we figure out which inherent properties make the circuits in the auditory cortex better suited as preprocessors for speech recognition ?

Hypothesis: Neural circuits are particular dynamical systems, which are better suited for anytime computing (for a particular type of input stream) than most other dynamical systems







Analysis of laminar circuit model, based on data by Thomson et al, in [Haeusler, Maass, Cerebral Cortex 2006]

4. New results/ideas

a) Relevance of the dynamic mode of the circuit for ist computational capability

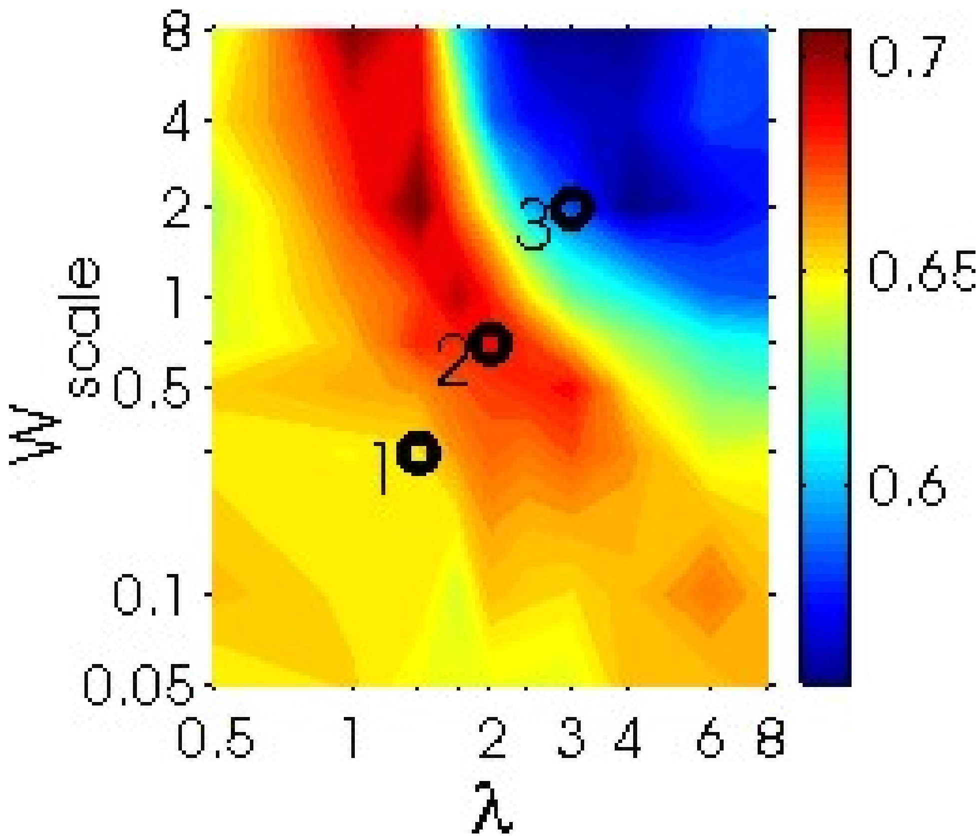
b) Predicting the computational power of a neural circuit

c) Possible implementation of Bayesian inference in generic cortical microcircuits

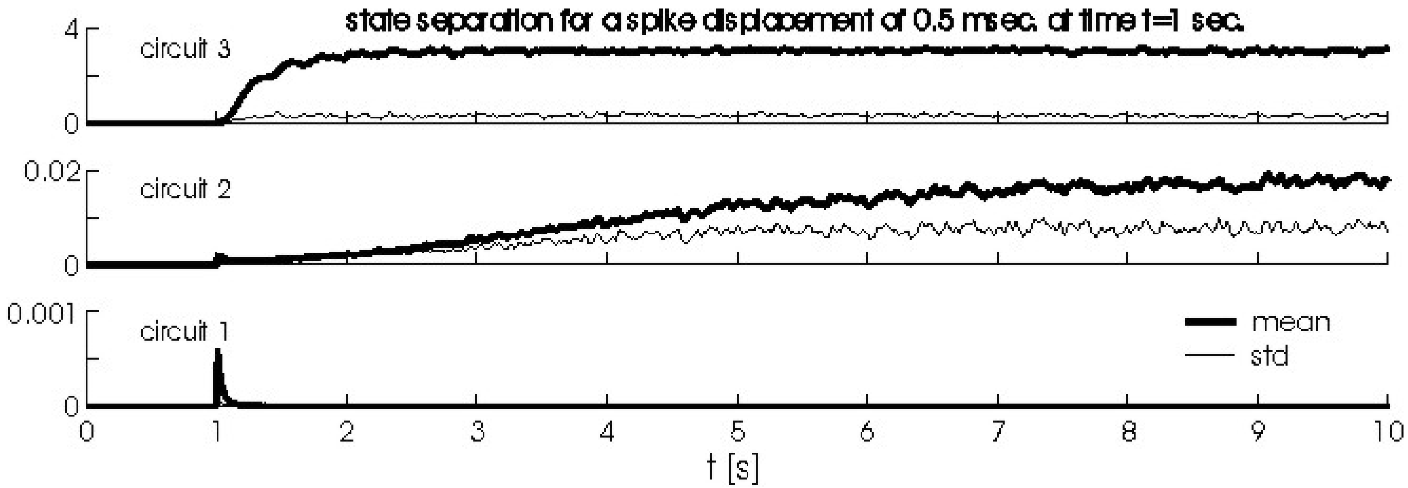
d) Extension of the computational power of generic neural circuits through feedback from trained neurons

a) Relevance of the dynamic mode of the circuit for ist computational capability

computational performance
of 90 types of circuits
(for randomly selected
classification tasks) in
different dynamic modes



Edge of chaos in cortical microcircuit models



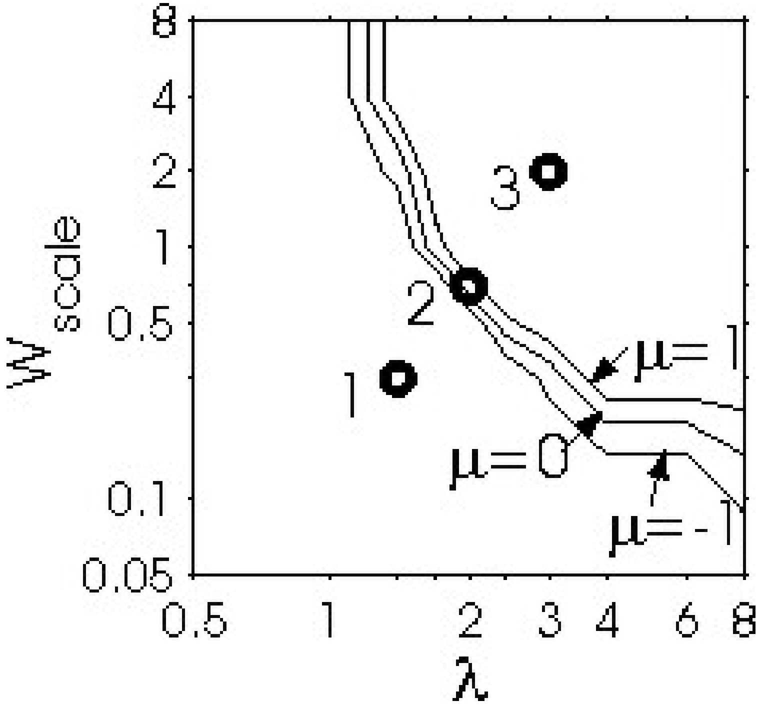
We define the Lyapunov exponent for this case with online input as that exponent $\mu \in \mathbb{R}$ which provides through the formula

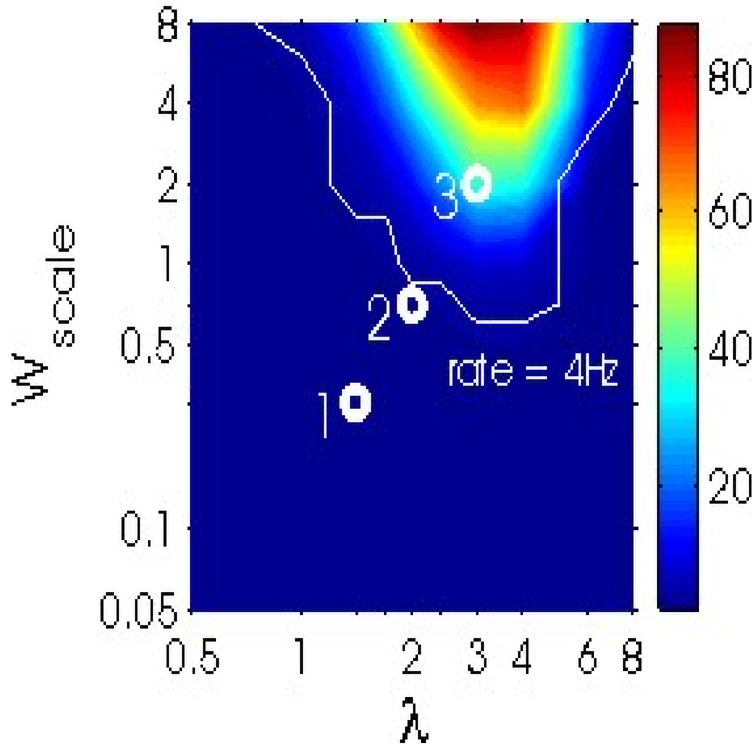
$$\delta_{\Delta T} \approx \delta_0 \cdot e^{\mu \Delta T}$$

the best estimate of the state separation $\delta_{\Delta T}$ at time ΔT after the computation was started in two trials with an “infinitesimal” initial state difference δ_0.

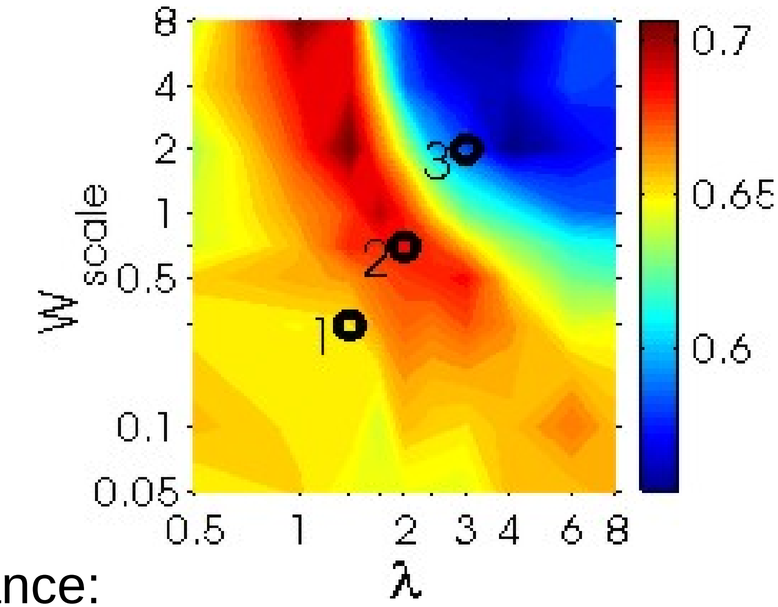
b) Predicting the computational power of a neural circuit

The edge of chaos is reached in these models at rather low firing rates (and cannot be characteri



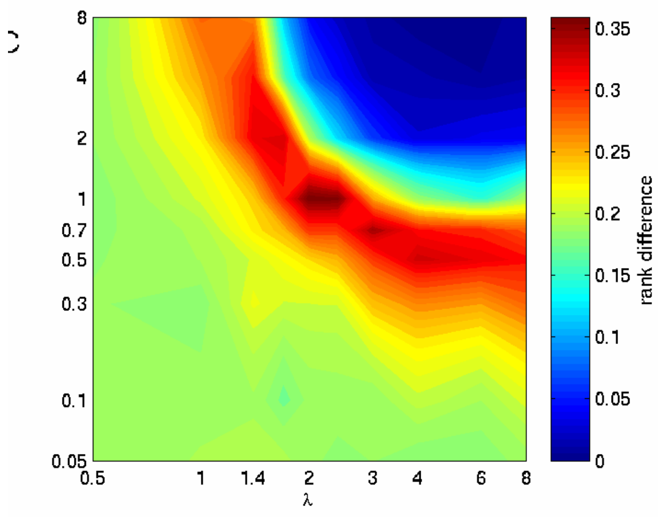


Direct evaluation of the
computational performance
of 90 types of circuits
(for randomly selected
classification tasks)

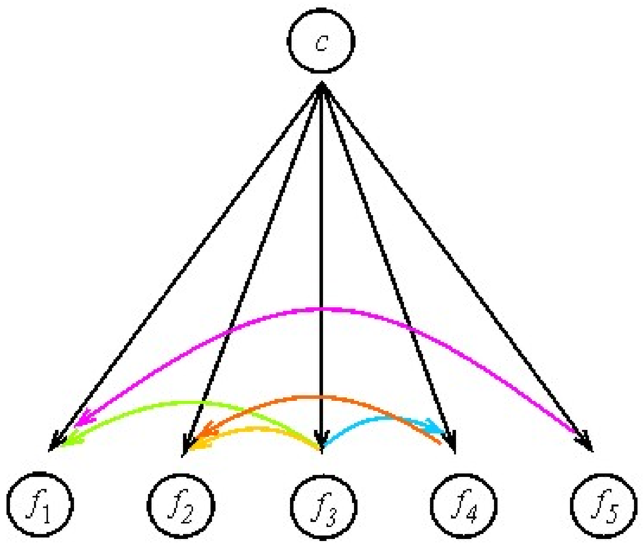


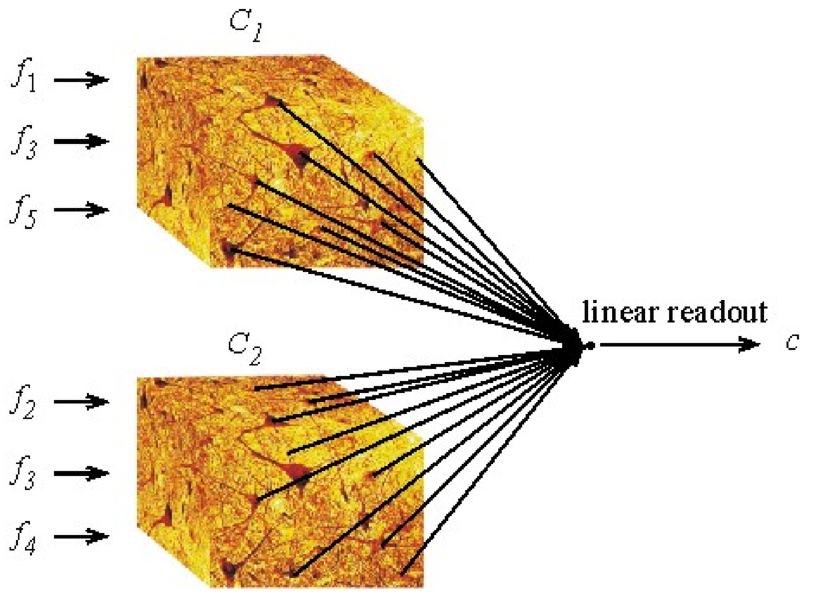
Prediction of computational performance:
Linear dimension („kernel measure“)
– VC-dimension
for the same
90 types of neural circuits:

The two terms of this difference
provide explanations
why a circuit has high/low
computational performance.



c) Possible implementation of Bayesian inference in generic cortical microcircuits

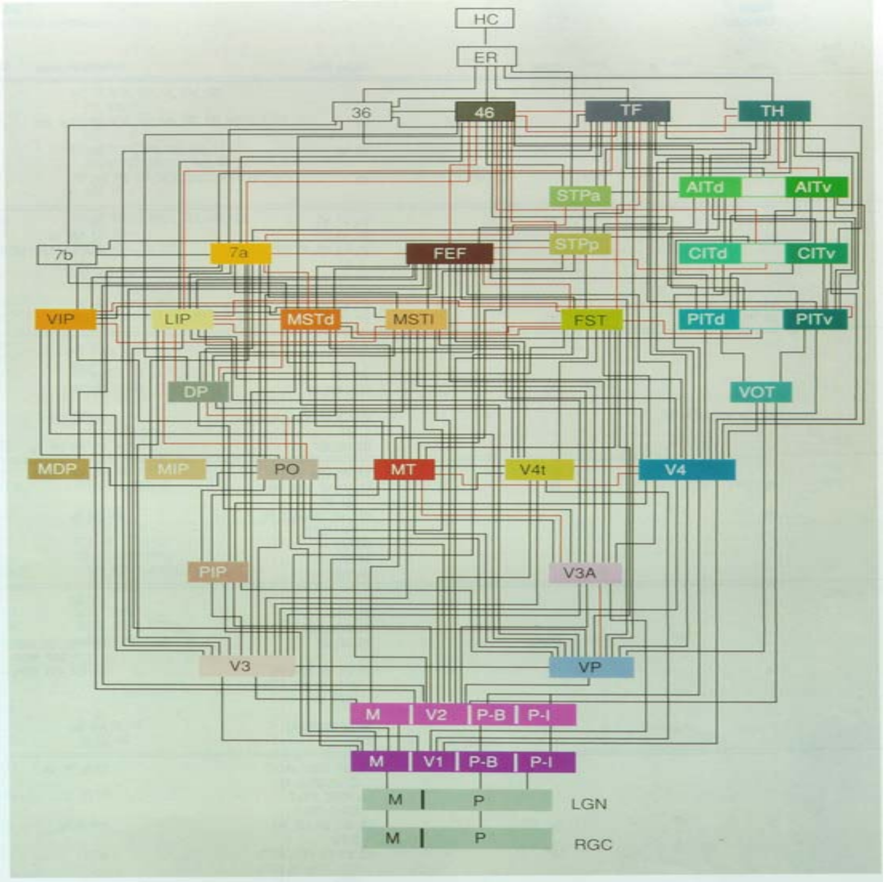




One can prove that the shown neural network can achieve the optimal classification performance of the Bayesian network if

- C_1 maps saliently different values of $\langle f_1, f_3, f_5 \rangle$ onto linearly independent liquid states, and
- C_2 maps saliently different values of $\langle f_2, f_3, f_4 \rangle$ onto linearly independent liquid states.

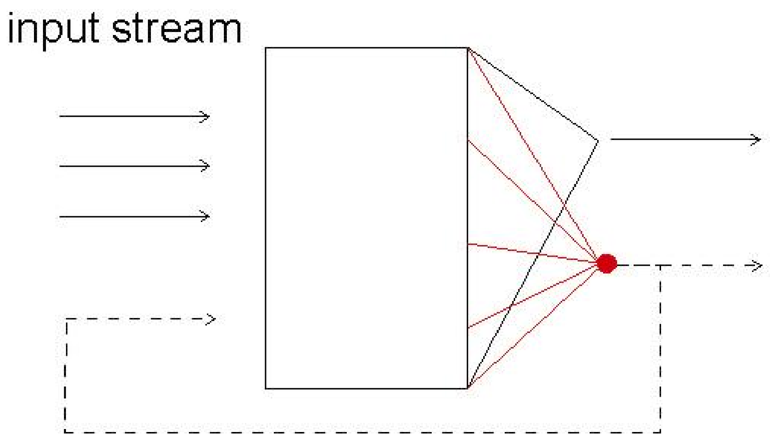
... perhaps there is a reason why there exist so many different brain areas (that receive different input combinations)

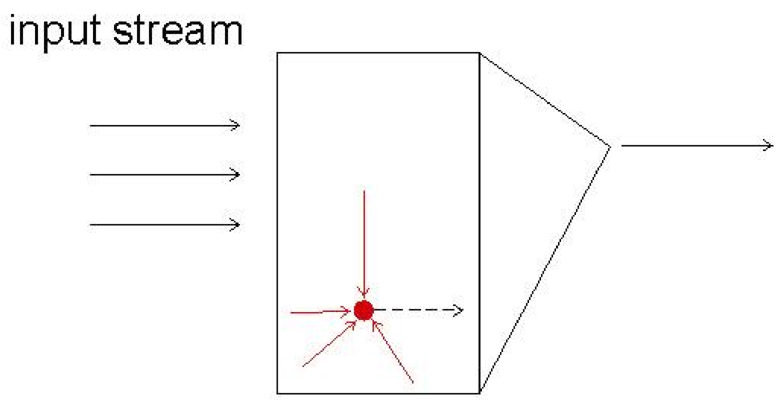


Shown: network of visual cortical areas in macaque monkey

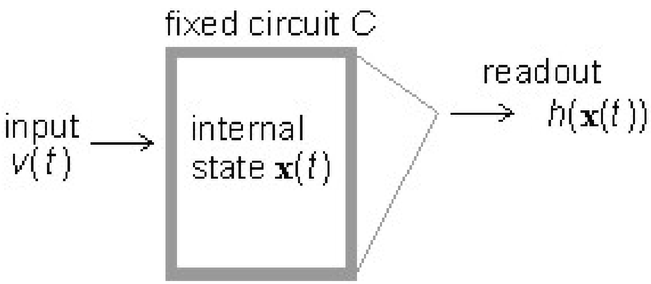
[Felleman, Van Essen, 1991]

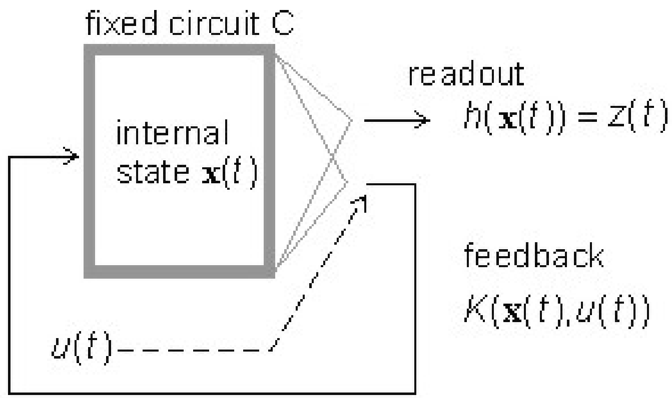
d) Extension of the computational power of generic neural circuits through feedback from trained neurons

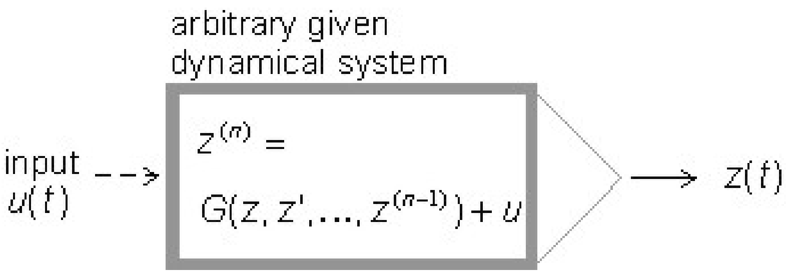




Underlying mathematical theory: There exists a large class S_n of analog circuits C with fading memory (described by systems of n first order differential equations) that gain through feedback universal computational capabilities for analog computing.







Note: Any Turing machine can be simulated by such dynamical system [Branicky, 1995],

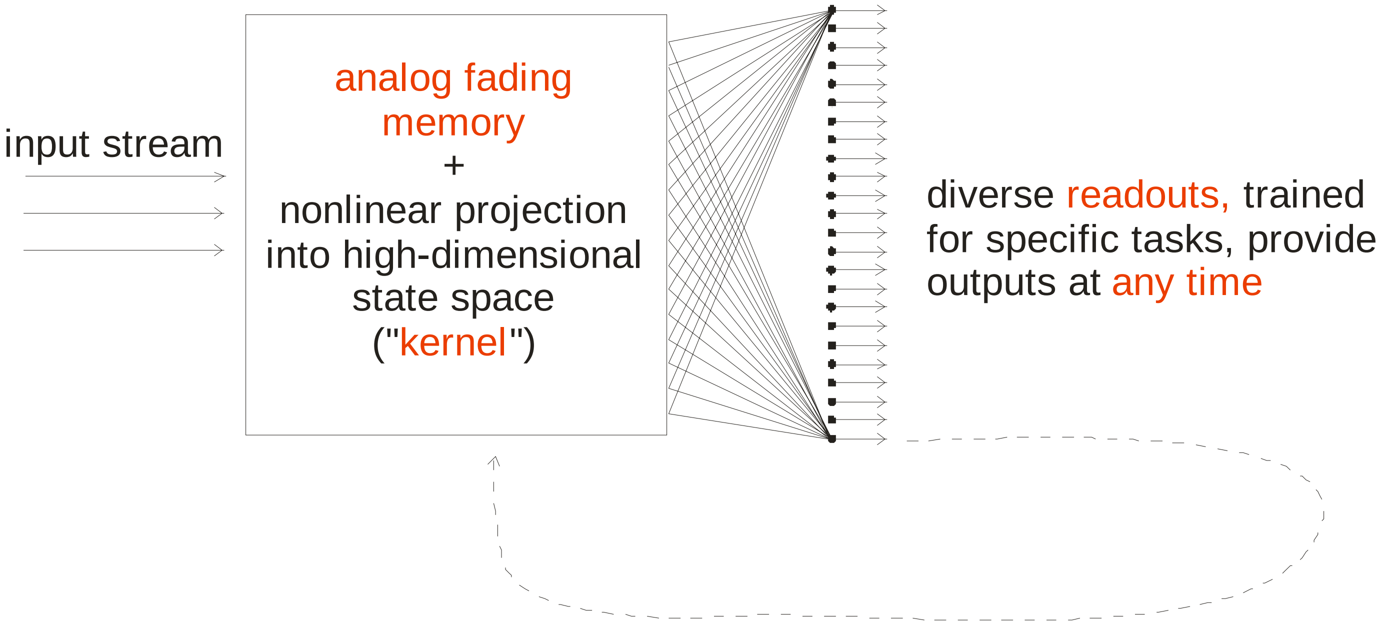
hence all digital computations (including those that require a non-fading memory).

This holds in particular for neural circuits *C* defined by DEs of the form

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

(under some conditions on the λ_i, a_{ij}, b_i).

5. Is our model for online computing in dynamical systems biologically realistic ?



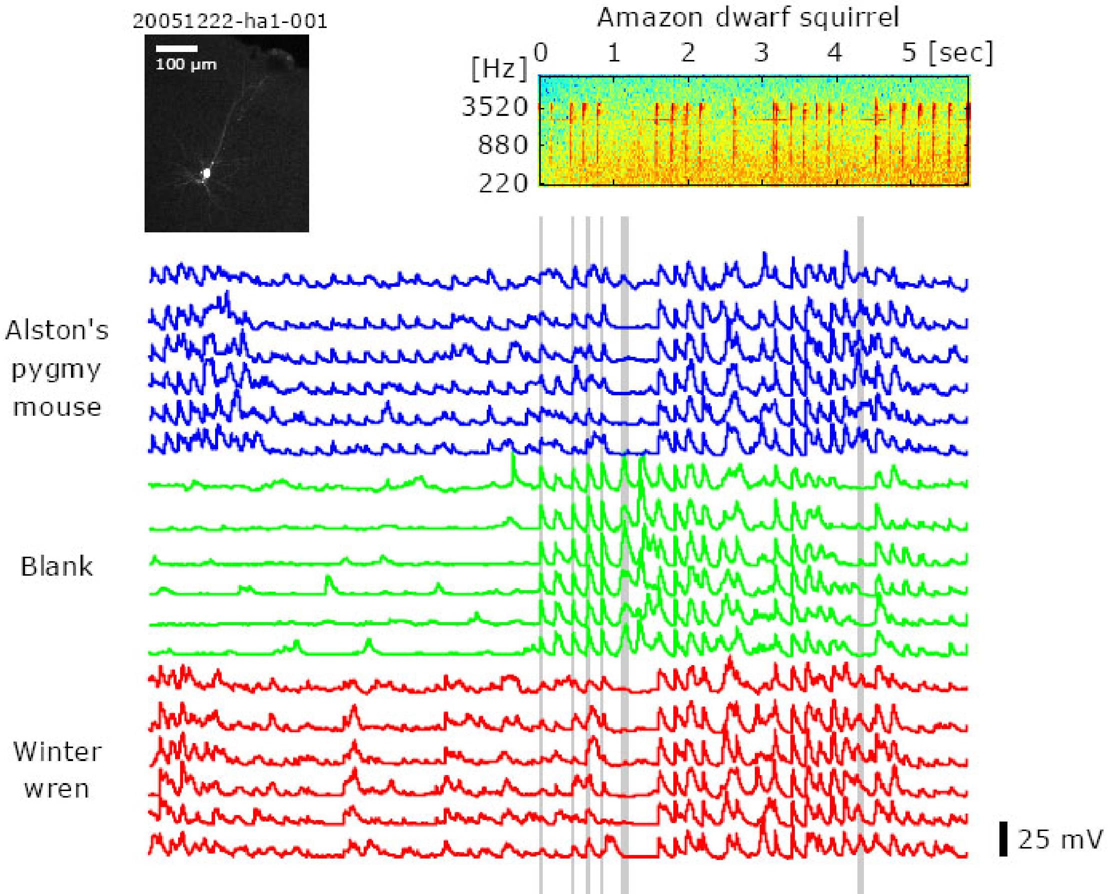
Predictions of this model: Generic cortical microcircuits exhibit

a) Temporal integration of information
b) General purpose nonlinear preprocessing (kernel-property)
c) Artefacts that result from keeping the system near the edge-of-chaos
d) Diversity of readouts

Biological evidence for temporal integration in the auditory cortex of rats [Asari, Oviedo, Zador, 2006]

3 different natural sounds (indicated by 3 different colors) precede a fixed natural sound that starts at time 0.

Each line shows the response of the same neuron at a different trial (in-vivo whole cell recording).



Result:

The response after time 0 conveys information both about the current AND the preceding stimulus („temporal integration“)

6. Open Problems

- Which circuit architectures and learning algorithms optimize neural circuits (or artificial devices) as preprocessors for auditory processing ?
- Measure the amount of information that various components of the auditory system contain about preceding auditory inputs
- How does the auditory system deal with adaptive processes on various time scales (and possible ongoing activity) ?
- Which learning algorithms train the „readouts“ at various stages of the auditory system ?