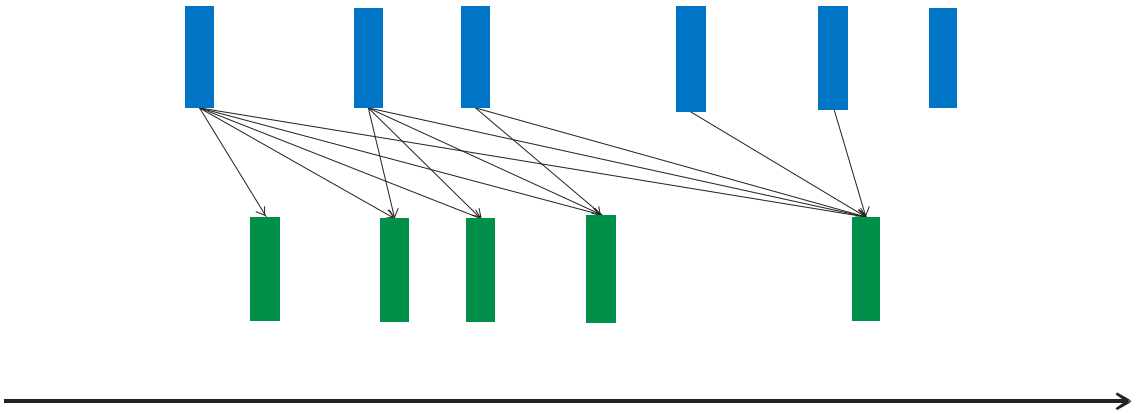
The Liquid-State-Machine Approach

What types of computation does the brain perform ?

Anytime-computation:



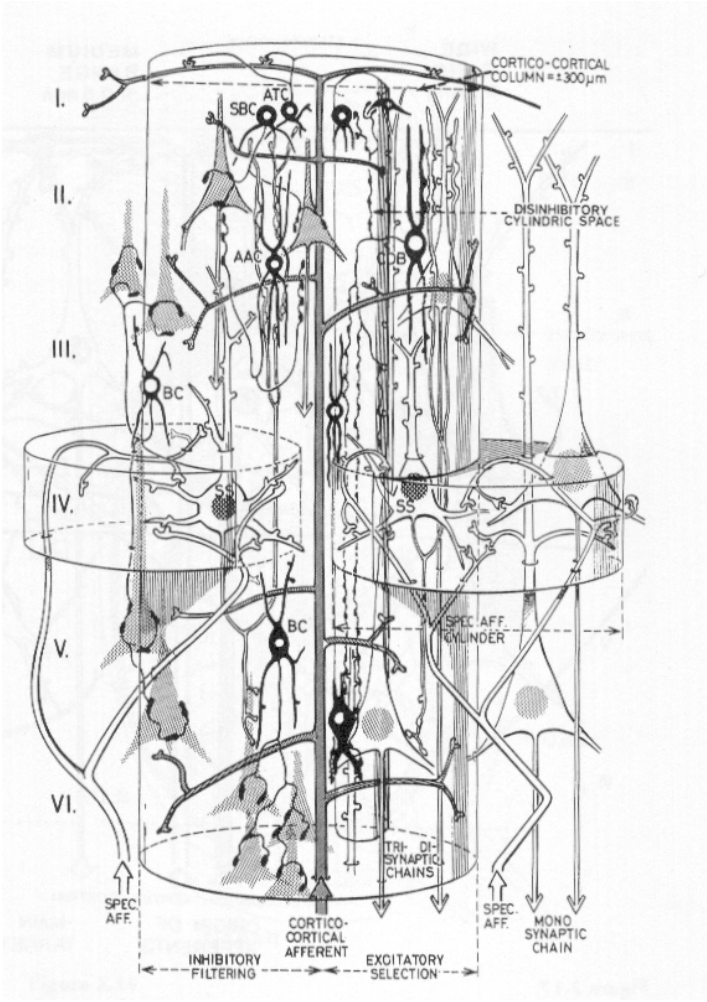
Offline-computation:

Fact: The cortex uses rather stereotypical microcircuits (columns, minicolumns, etc.) for a large variety of complex real-time computation tasks.

Goal: Explain how generic cortical microcircuits

- that consist of diverse types of neurons and diverse types of dynamic synapses
- with sparse recurrent connection patterns ("loops within loops")

could possibly achieve this.



The LSM-approach is based on the following observations:

- If one excites a sufficiently complex dynamical system with a continuous stream of inputs $u(s)$, and looks at the state $x(t)$ of the system at a later time t, then

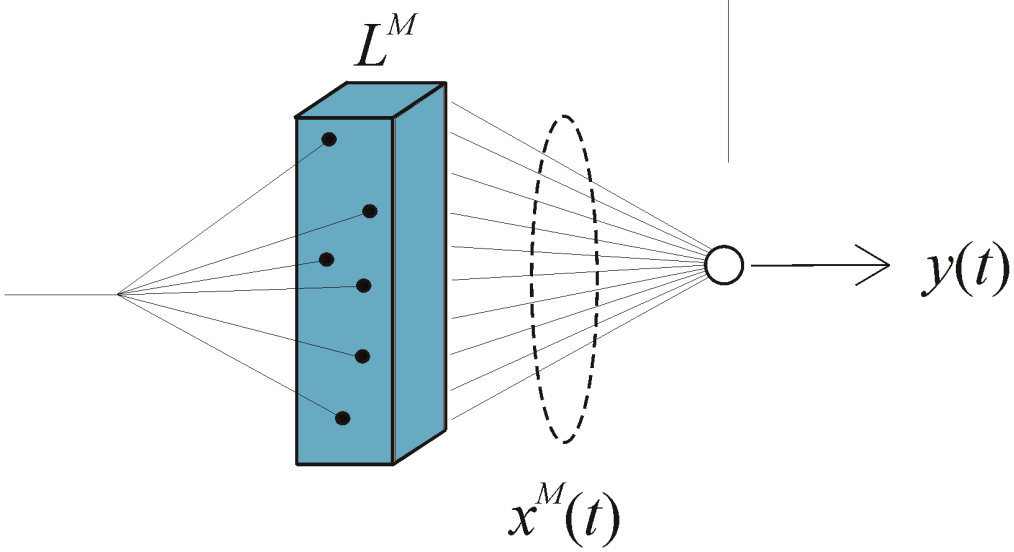
  $x(t)$ is likely to hold a substantial amount of information about recent inputs $u(s)$ for $s < t$,

  especially if the dynamical system consists of diverse dynamical components, and is not chaotic

Liquid State Machine

complex dynamical system (viewed as a nonlinear filter)

memoryless readout, trained for a specific task



= liquid state of the Liquid State Machine

$$x^M(t) = (L^M u)(t)$$

$$y(t) = f^M(x^M(t))$$

Mathematical theorems guarantee that any time-invariant fading memory filter that maps input-streams onto output streams can be approximated with any desired degree of precision (and in real-time) by liquid state machines of finite size

- ***if*** there is a rich enough pool **B** of basis filters from which the dynamical system can be composed

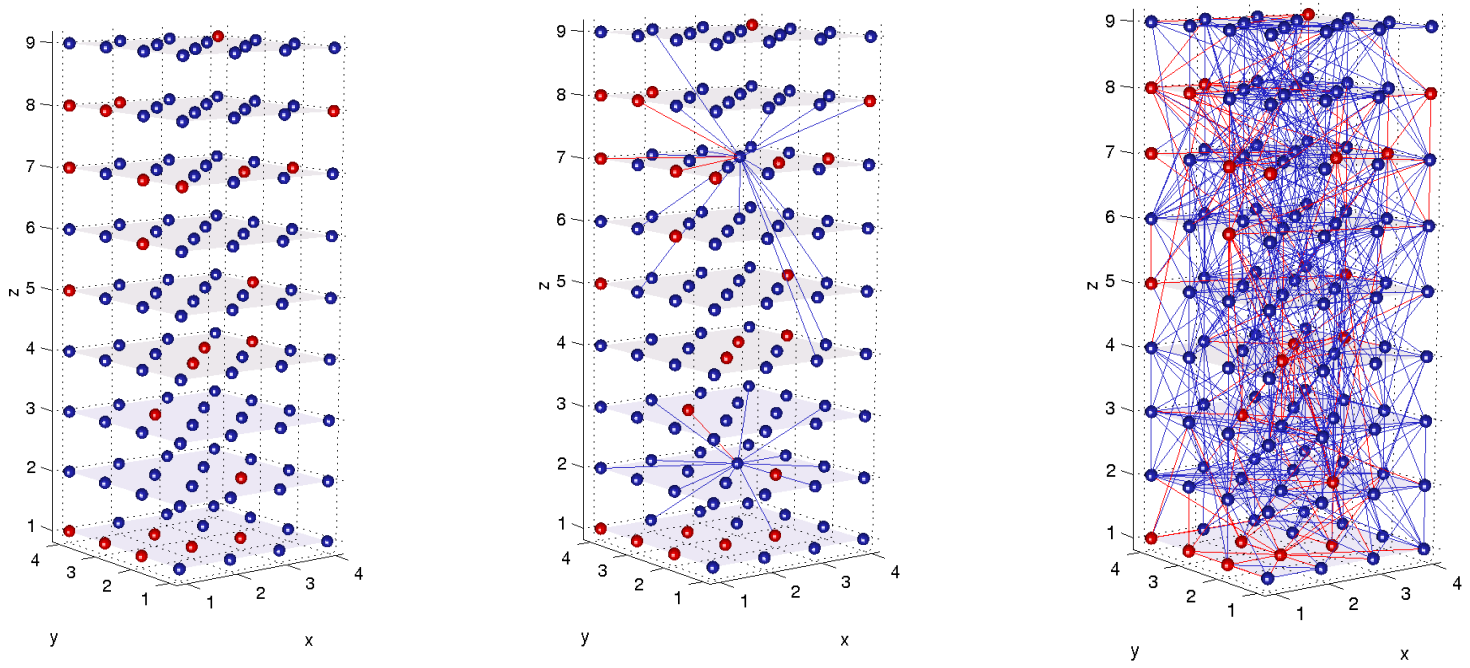
  and

- ***if*** there is a rich enough pool **F** of readout functions.

F has fading memory:

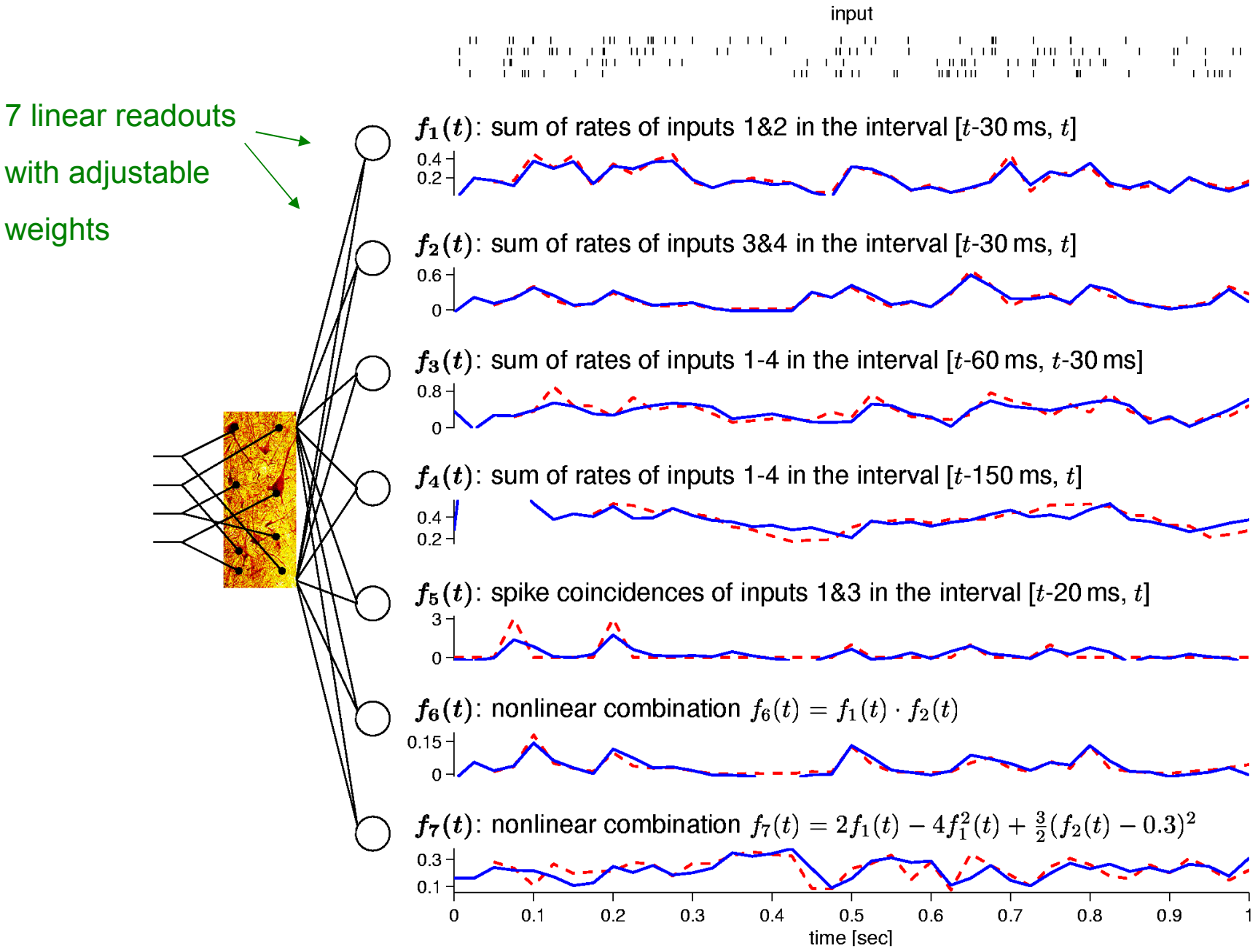
- In order to determine the output $(Fu(\cdot))(t)$ with a given precision ε it suffices to know the values of $u(t-\tau)$ up to some finite precision δ for all *tau* from some finite time interval [0, *T*].

Results of computer simulations:



Networks of leaky-integrate and fire neurons locally connected.

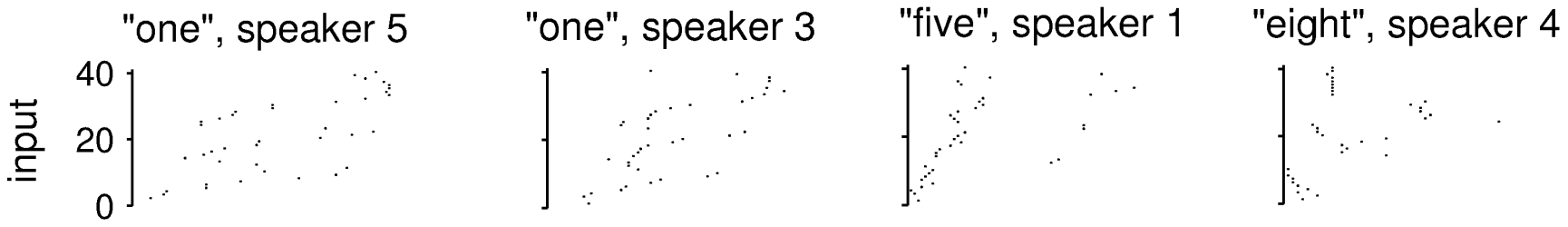
A linear readout neuron.


input
7 linear readouts
with adjustable
weights
$f_1(t)$: sum of rates of inputs 1&2 in the interval [t-30 ms, t]
0.4
0.2
$f_2(t)$: sum of rates of inputs 3&4 in the interval [t-30 ms, t]
0.6
0
$f_3(t)$: sum of rates of inputs 1-4 in the interval [t-60 ms, t-30 ms]
0.8
0
$f_4(t)$: sum of rates of inputs 1-4 in the interval [t-150 ms, t]
0.4
0.2
$f_5(t)$: spike coincidences of inputs 1&3 in the interval [t-20 ms, t]
3
0
$f_6(t)$: nonlinear combination $f_6(t) = f_1(t) \cdot f_2(t)$
0.15
0
$f_7(t)$: nonlinear combination $f_7(t) = 2f_1(t) - 4f_1^2(t) + \frac{3}{2}(f_2(t) - 0.3)^2$
0.3
0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
time [sec]

Testing the generic microcircuit model on the speech recognition task:

- 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)



Results

- the generic neural microcircuit model classifies the spoken word instantly when the word ends (i.e., in real-time)
- linear readouts from the generic microcircuit model can even be trained to do anytime speech classification: