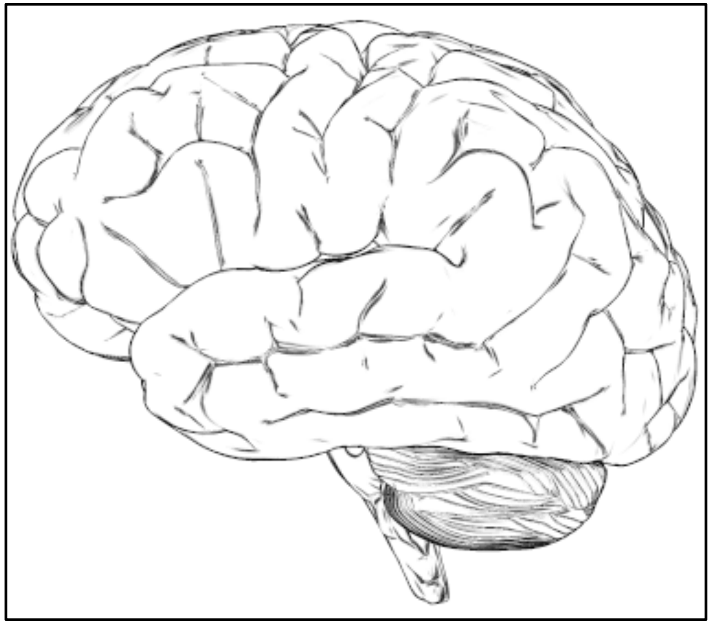
Computing with transient network dynamics

Stefan Klampfl, Wolfgang Maass

Institute for Theoretical Computer Science
University of Technology Graz, Austria

December 11, 2008

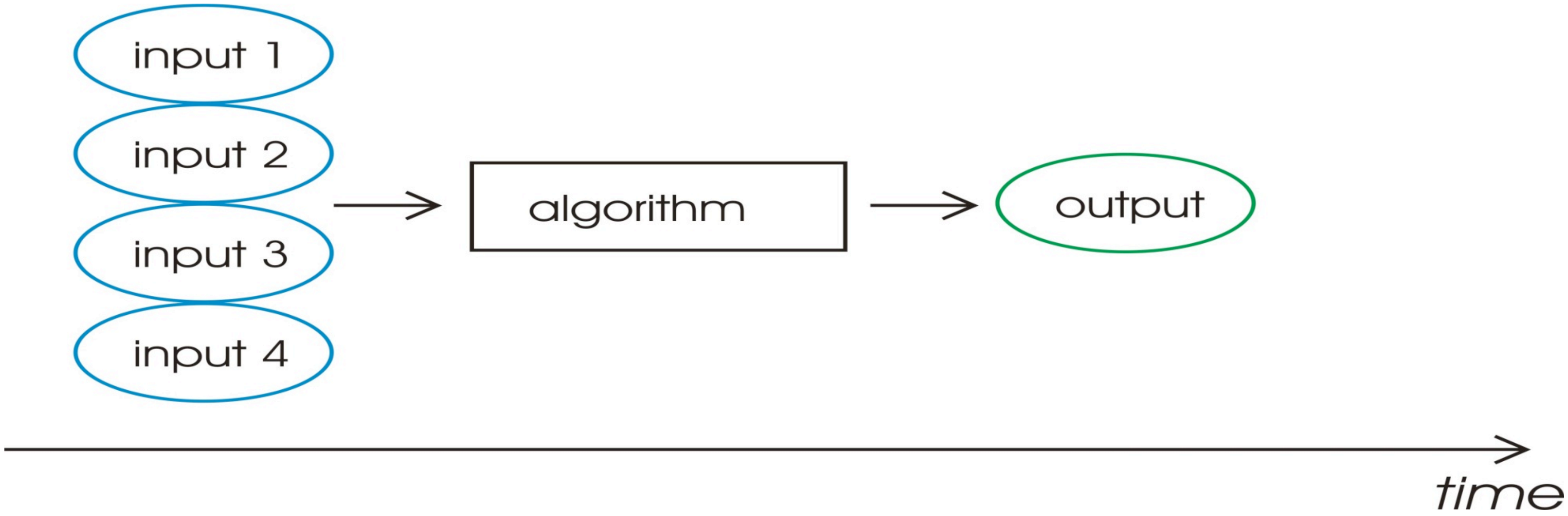
Computation in cortical circuits



Obviously, computation in the brain is different from computation in computers

Computers vs brains

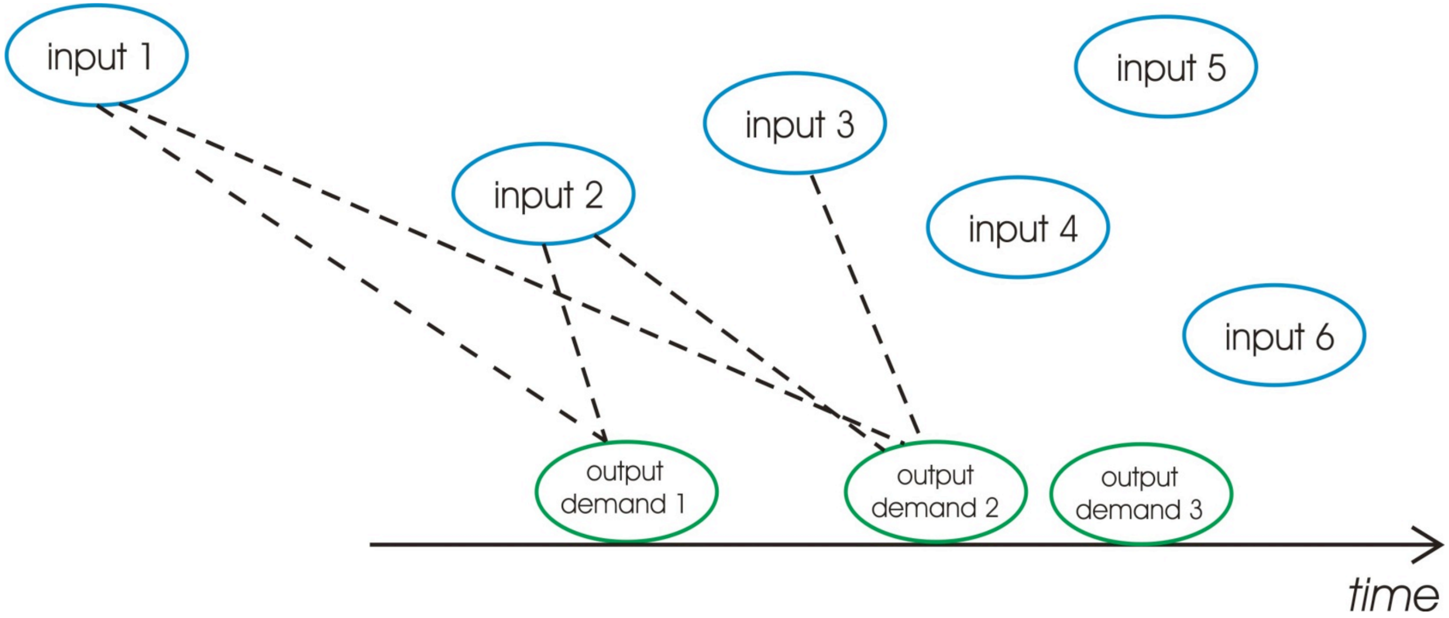
- Computers carry out offline computations



Different input components are presented all at once, and there is no strict bound on the computation time.

Computers vs brains

- Typical computations in the brain are online



In online computations there arrive all the time new input components.
A real-time algorithm has produce for each new input component an output within a fixed time interval D.
An anytime algorithm can be prompted at any time to provide its current best guess of a proper output (which should integrate as many of the previously arrived input-pieces as possible).

Computers vs brains

- Obvious differences:
  - Computations in brains result from learning
  - Brains carry out online computations
  - Neural circuits consist of heterogeneous components

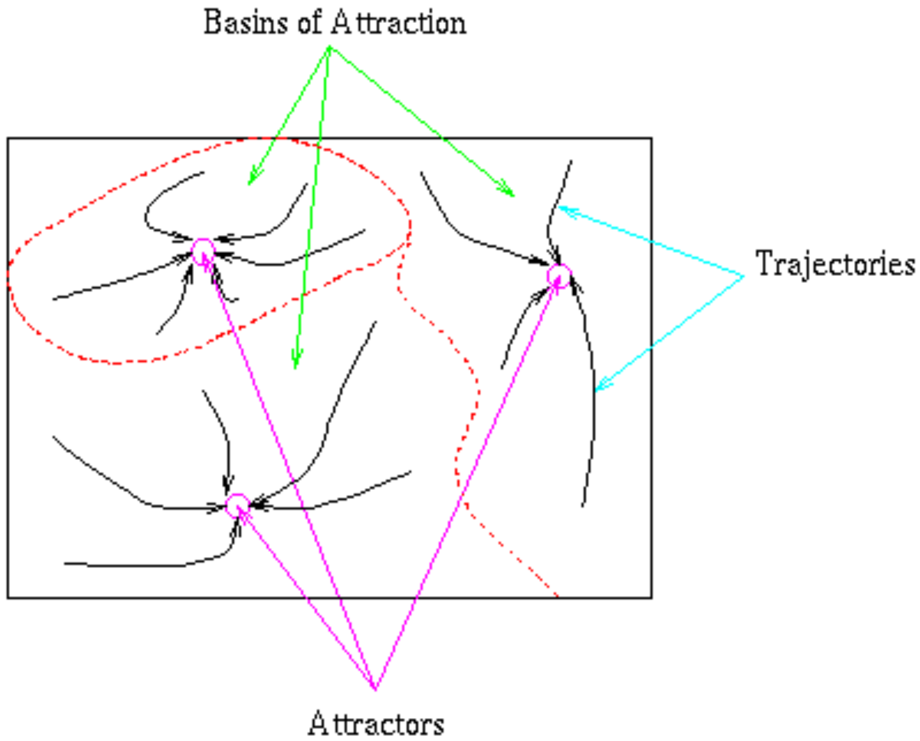
- Standard computational models are not adequate for understanding brain computations

An alternative model

- Computations can be viewed as trajectories to an attractor in a dynamical system

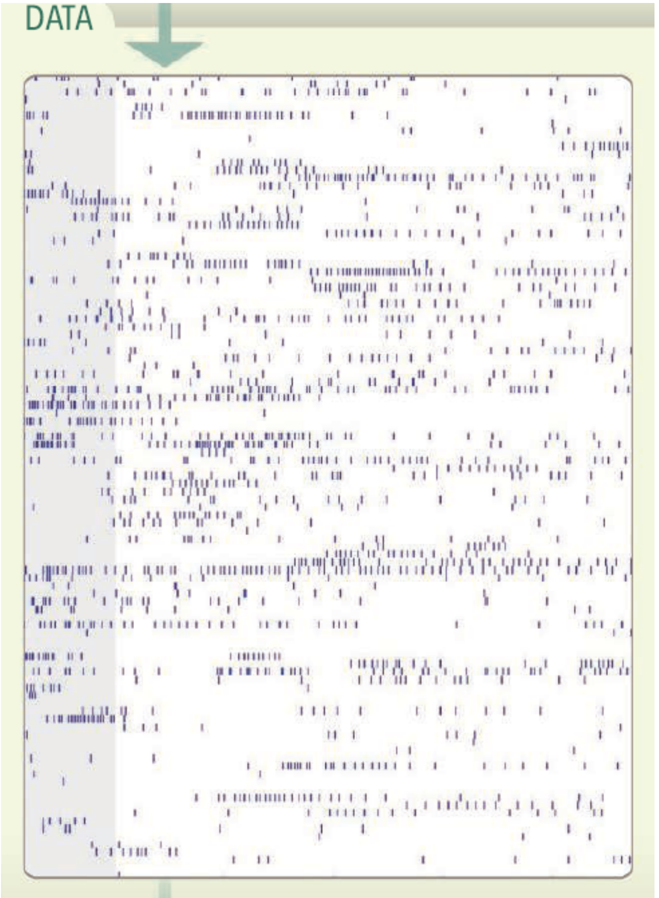
Problems with this model:

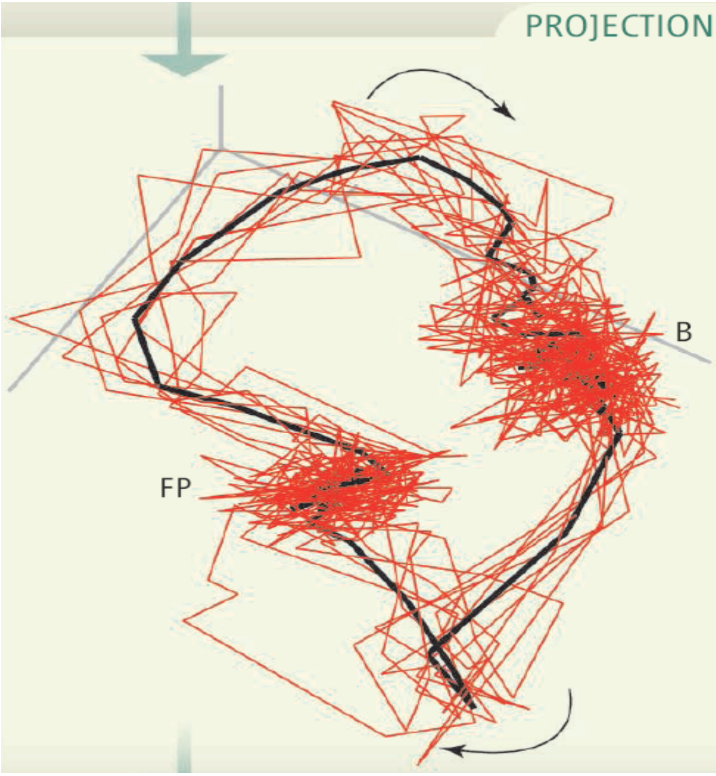
- Not suitable for online computing
- Conditions implausible for biological circuits
- Attractors cannot be reached in short time
- Data rarely show convergence to an attractor



Computing with trajectories

- Is information encoded in the transient dynamics?





Principal neurons from locust antennal-lobe in response to a specific odor [Rabinovich et al, 2008]

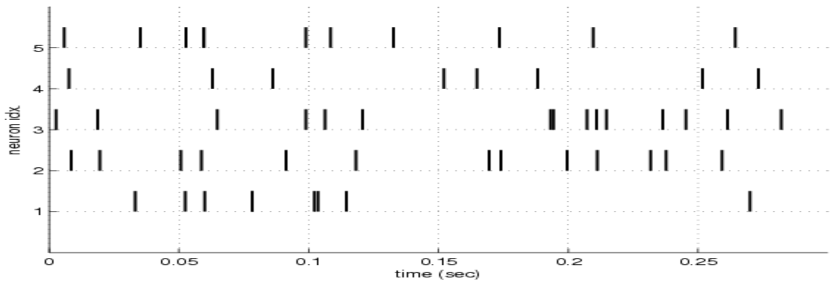
Our approach

- How can temporally stable information be extracted from these transient trajectories?
- We consider the perspective of the “neural user“, i.e., projection neurons that read out information from the circuit
  - e.g., linear discriminator

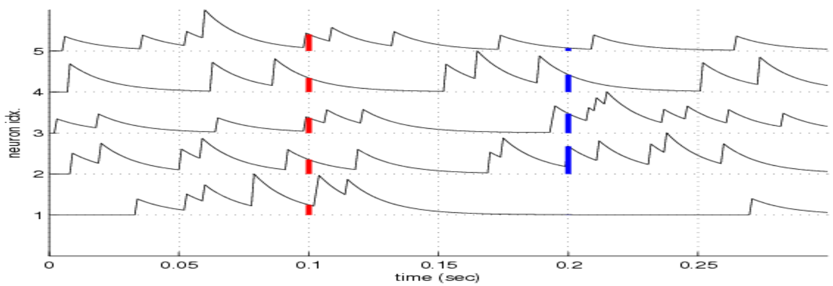
Our approach

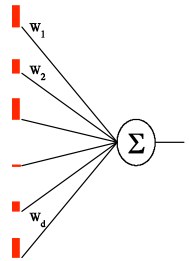
- A linear readout neuron computes the weighted sum of low-pass filtered presynaptic spike trains

Spikes from presynaptic neurons in the circuit



Postsynaptic potentials in the readout neuron caused by these spikes

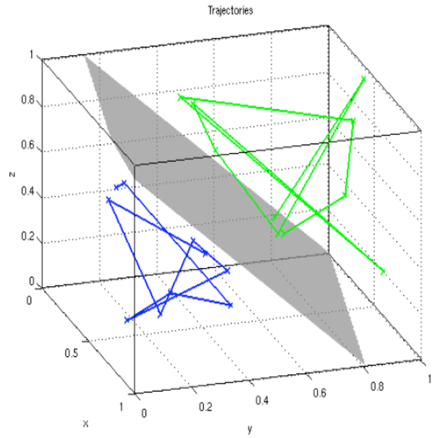




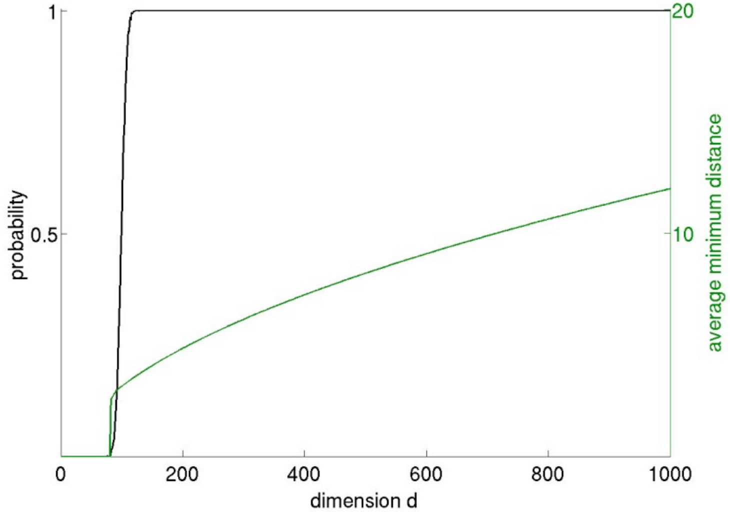
This high dimensional analog input is called the liquid state

Computing with trajectories

- High-dimensionality of state space facilitates the extraction of temporally stable information



Linear separability of 2 randomly drawn trajectories of length 100 in *d* dimensions:



a projection neuron with *d* presynaptic inputs can separate almost any pair of trajectories that are each defined by connecting *less than d randomly drawn points.*

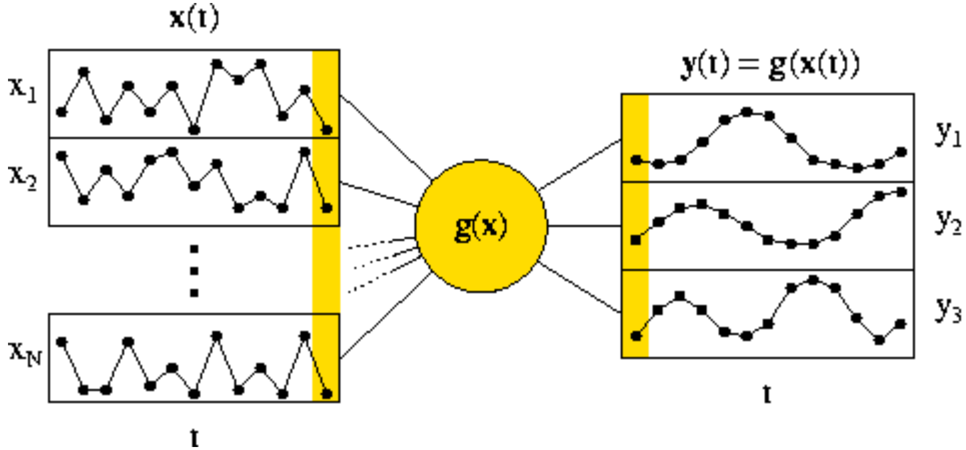
Learning of readout neurons

- How can readout neurons be trained to discriminate between trajectories?
  - Supervised learning (e.g., linear classifier)
  - Reinforcement learning (Reward-modulated STDP [Izhikevich, 2007, Legenstein et al, 2007])
- I will focus on unsupervised learning
  - Slow Feature Analysis (SFA) [Wiskott and Sejnowski, 2002]

Slow Feature Analysis

- Based on slowness as a principle for learning invariances
- Given a multi-dimensional time series **x**(t), it finds those functions g_i which generate the most slowly varying signals $y_i(t) = g_i(\mathbf{x}(t))$
- It minimizes $\langle \dot{y}_i^2 \rangle$ s.t.

$$\langle y_i \rangle = 0$$
$$\langle y_i^2 \rangle = 1$$
$$\langle y_i y_j \rangle = 0 \quad \forall j < i$$



- g_i are instantaneous functions of the input **x**(t)

Slow Feature Analysis

- Consider the special case
  - where the input **x** is whitened: $\langle \mathbf{x} \rangle = \mathbf{0}$ and $\langle \mathbf{x}\mathbf{x}^T \rangle = \mathbf{I}$
  - the set of linear functions $y = g(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$
- Minimize

$$\langle \dot{y}_i^2 \rangle = \mathbf{w}_i^T \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle \mathbf{w}_i$$

  under the constraints

$$\langle y_i \rangle = \mathbf{w}_i^T \langle \mathbf{x} \rangle = 0$$

$$\langle y_i^2 \rangle = \mathbf{w}_i^T \langle \mathbf{x}\mathbf{x}^T \rangle \mathbf{w}_i = \mathbf{w}_i^T \mathbf{w}_i = 1$$

$$\langle y_i y_j \rangle = \mathbf{w}_i^T \langle \mathbf{x}\mathbf{x}^T \rangle \mathbf{w}_j = \mathbf{w}_i^T \mathbf{w}_j = 0 \quad \forall j < i$$

- SFA finds the normed eigenvector of $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle$ corresponding to the smallest eigenvalue

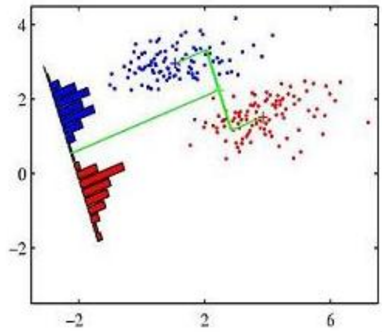
SFA for classification

- Linear SFA can be related to Fisher's Linear Discriminant (FLD)

SFA: $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{w} = \lambda \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}$

FLD: $\mathbf{S}_B \mathbf{w} = \lambda \mathbf{S}_W \mathbf{w}$

$$\mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$$
$$\mathbf{S}_W = \sum_c \sum_{\mathbf{x} \in S_c} (\mathbf{x} - \mathbf{m}_c)(\mathbf{x} - \mathbf{m}_c)^T$$



- Convert the input to the classification problem into a time series for SFA
- At each time select random point from a class chosen by a Markov model

$$\begin{tikzcd} 1 \arrow[loop left, "1-p"] \arrow[r, bend left, "p"] & 2 \arrow[l, bend left, "q"] \arrow[loop right, "1-q"] \end{tikzcd}$$

SFA for classification

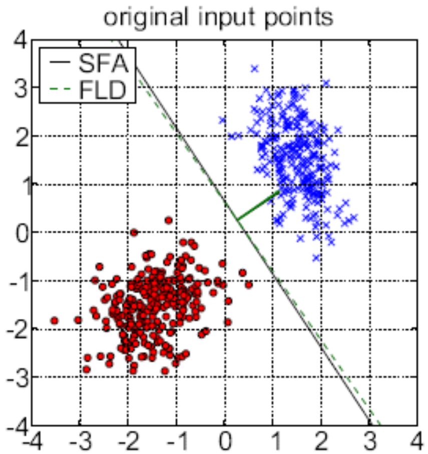
- If the transition probabilities between classes are low, the eigenvalues are the same

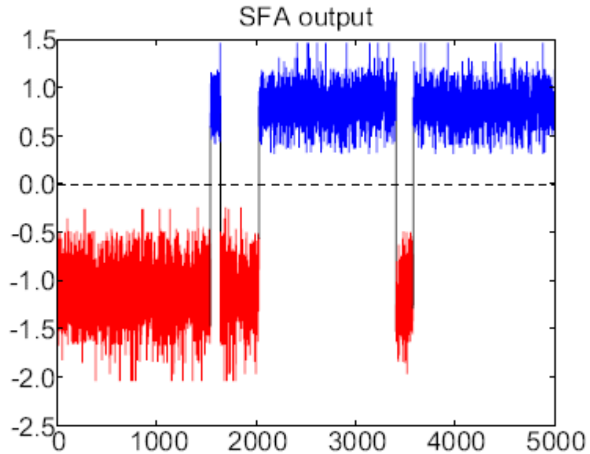
$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \mathbf{S}_W + \mathbf{S}_B$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \approx 2\mathbf{S}_W$$

$$\begin{aligned} \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{W} &= \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{W}\mathbf{\Lambda} \\ 2\mathbf{S}_W\mathbf{W} &= \mathbf{S}_W\mathbf{W}\mathbf{\Lambda} + \mathbf{S}_B\mathbf{W}\mathbf{\Lambda} \\ 2\mathbf{S}_W\mathbf{W}\mathbf{\Lambda}^{-1} &= \mathbf{S}_W\mathbf{W} + \mathbf{S}_B\mathbf{W} \\ \mathbf{S}_B\mathbf{W} &= \mathbf{S}_W\mathbf{W}\left[2\mathbf{\Lambda}^{-1} - \mathbf{E}\right] \end{aligned}$$

- If the class is slowly varying, SFA finds the same subspace as FLD



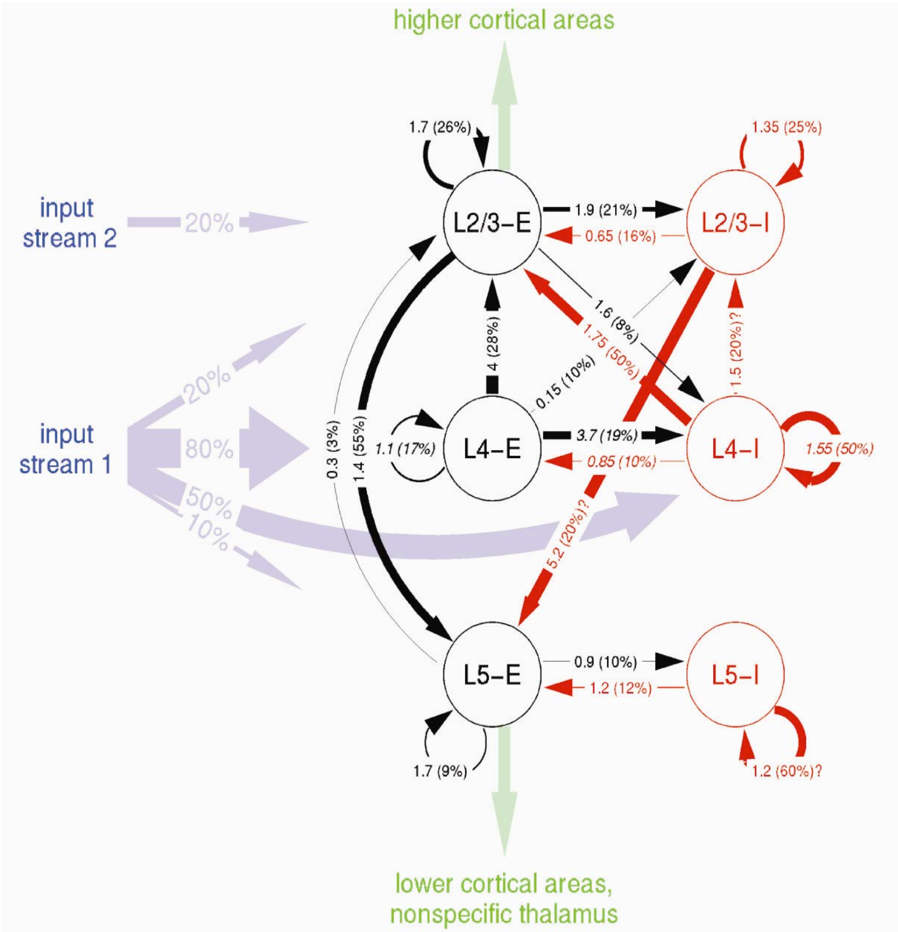


Training readouts with SFA

- Our model of a cortical microcircuit:

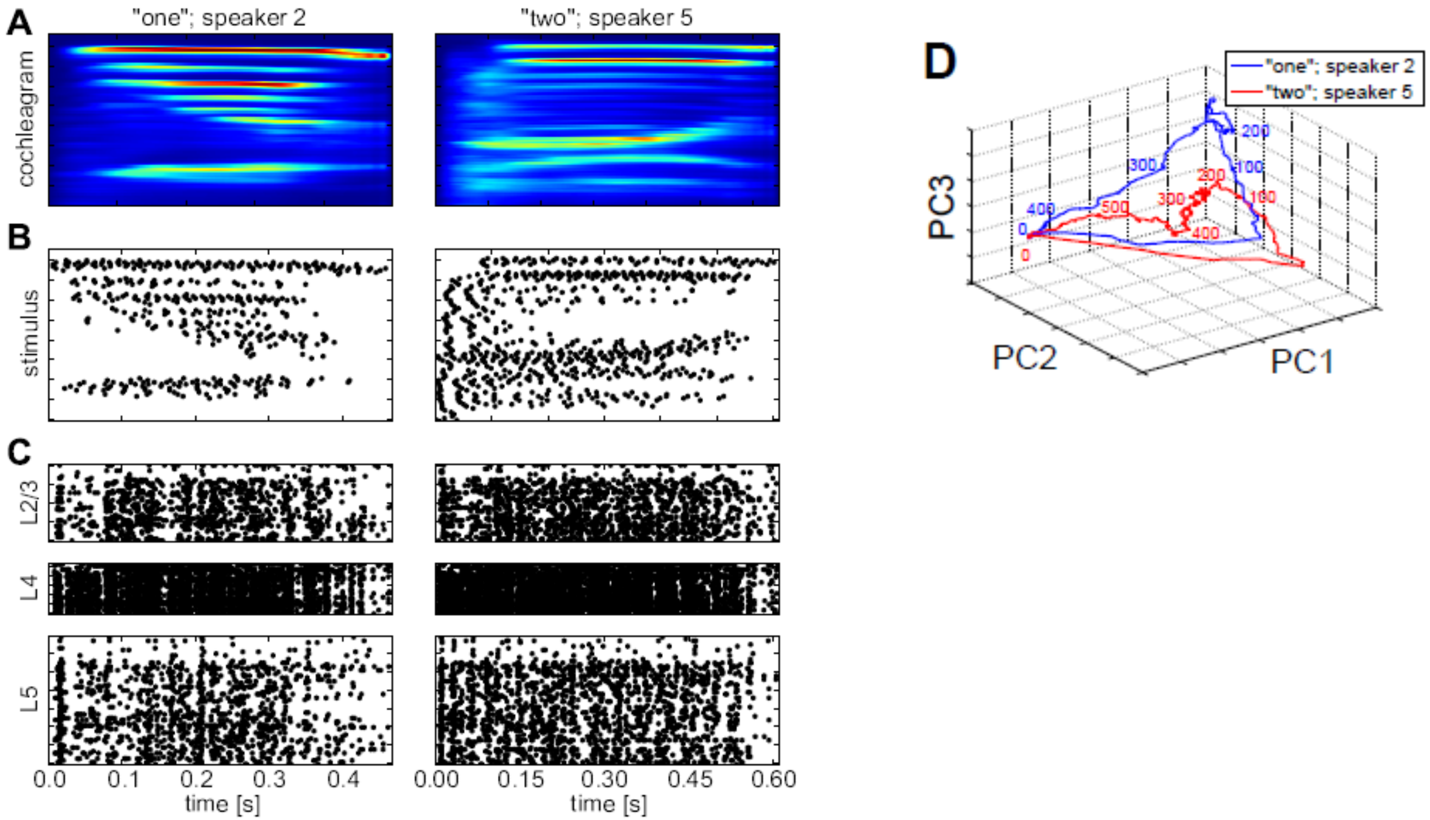
Computer model (560 neurons) from [Haeusler and Maass, 2007], based on data from the Labs of Alex Thomson, Henry Markram, simulated with single compartment HH-neurons, conductance based synapses with short-term plasticity with noise reflecting in-vivo-conditions according to Destexhe et al.

We have applied long-term plasticity so far only to projection neurons in this model.



Training readouts with SFA

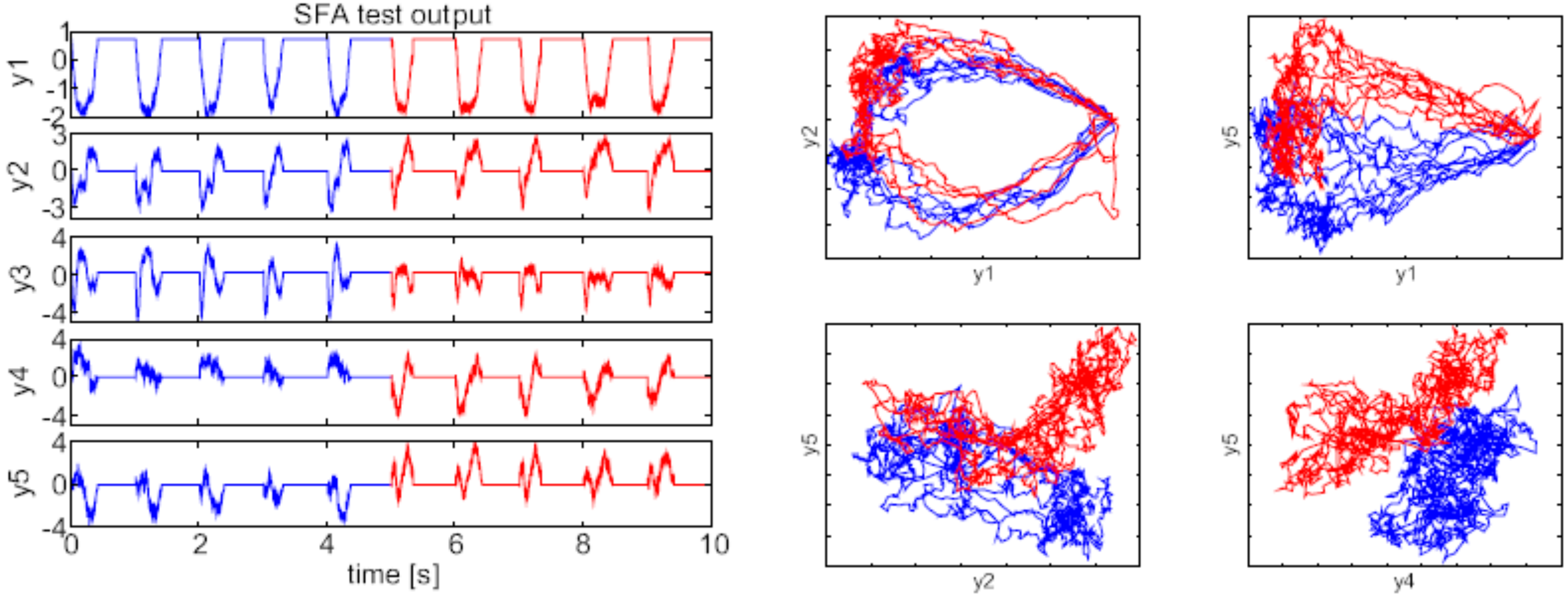
- Isolated spoken digits task [Hopfield and Brody]
- Inputs preprocessed with cochlea model [Lyon, 1982]



Training readouts with SFA

- Linear SFA applied to a long random sequence of circuit trajectories in response to different versions of words "one" and "two" (of a single speaker)

SFA output in response to 5 test trajectories of each class:



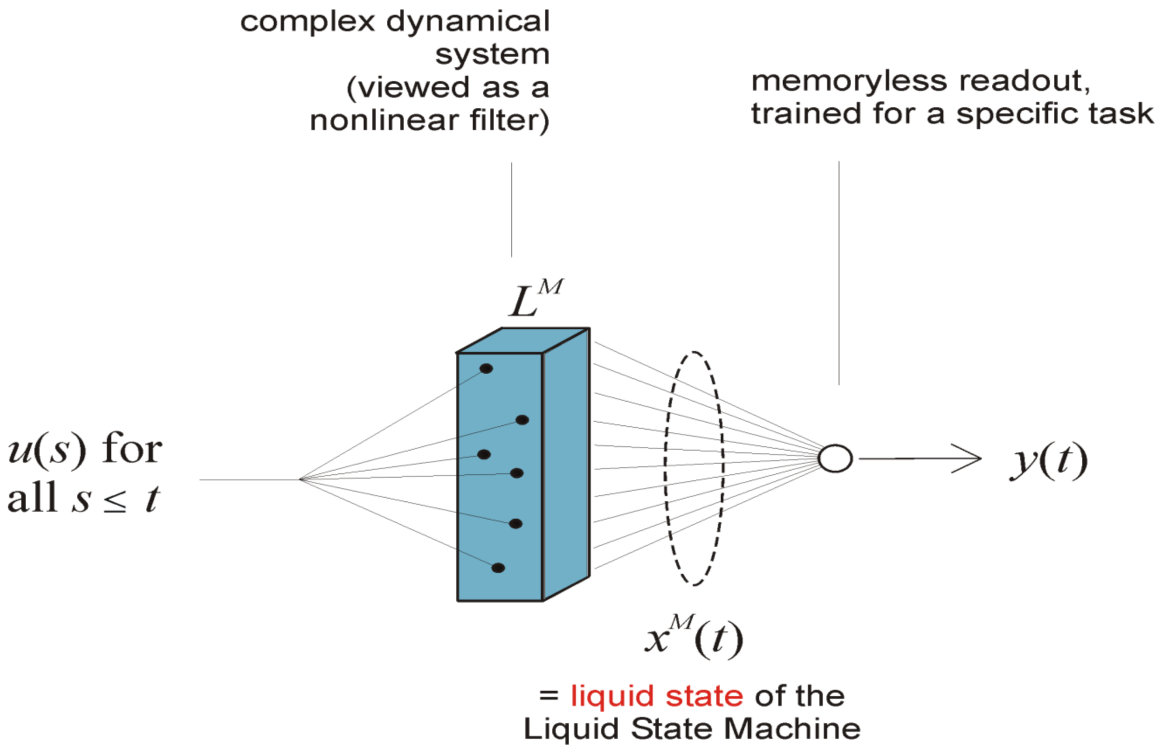
Slow features encode "pattern location" (Where-information) and pattern identity (What-information)

Liquid Computing model

- The computational performance of a cortical microcircuit should be judged on the basis how well it supports the task of readout neurons
- A microcircuit could support the learning capability of linear projection neurons by providing:
  - analog fading memory (to accumulate information over time in the state)
  - nonlinear projection into high-dimensional space (kernel property)

Liquid Computing model

- Liquid State Machine (LSM) generalizes finite state machines to continuous input values u(s), continuous output values y(t), and continuous time t



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

[Maass, Natschläger, Markram, 2002]

Liquid Computing model

- What is the computational power of this model?
  - If the dynamical system *L* is sufficiently large and consists of sufficiently diverse components, the readout function *f* can be trained to approximate any Volterra series. [Maass, Markram, 2004]
  - If one allows feedback from the readout into the dynamical system, then this model becomes already for rather simple dynamical systems *L* universal for analog (and digital) computation on input streams (in particular it can simulate any Turing machine). [Maass, Joshi, Sontag, 2007]

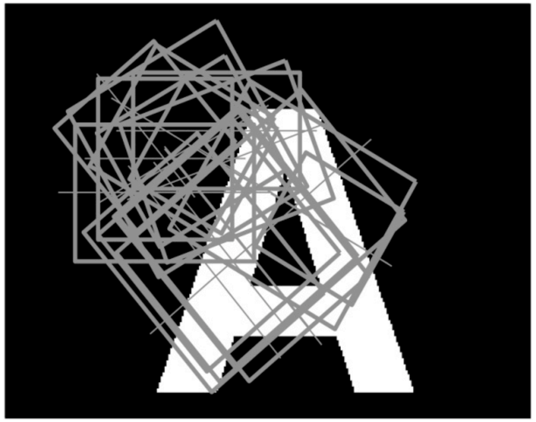
Liquid Computing model

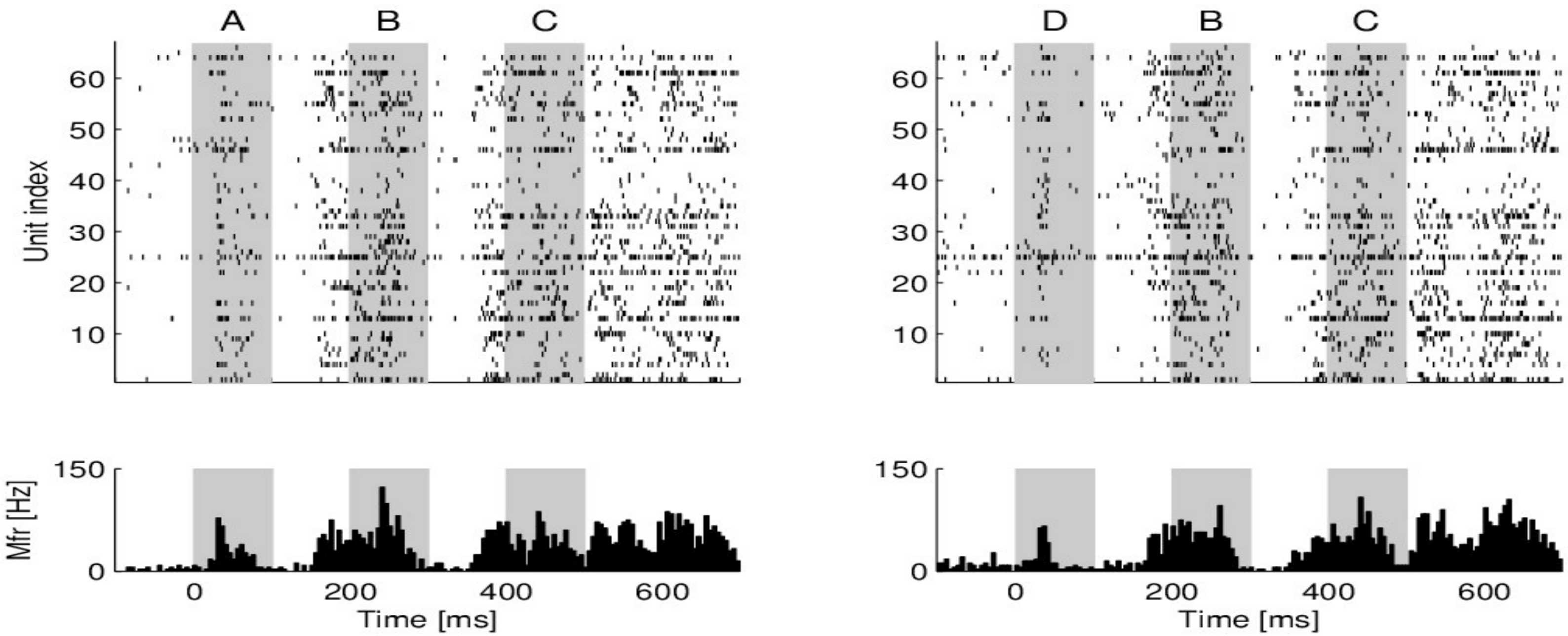
- Experimentally testable predictions of the liquid computing model for cortical microcircuits:
  - Temporal integration of information (fading memory)
  - General purpose nonlinear preprocessing (kernel property)

Data from visual cortex

Recordings by D. Nikolic from primary visual cortex of anaesthesized cat

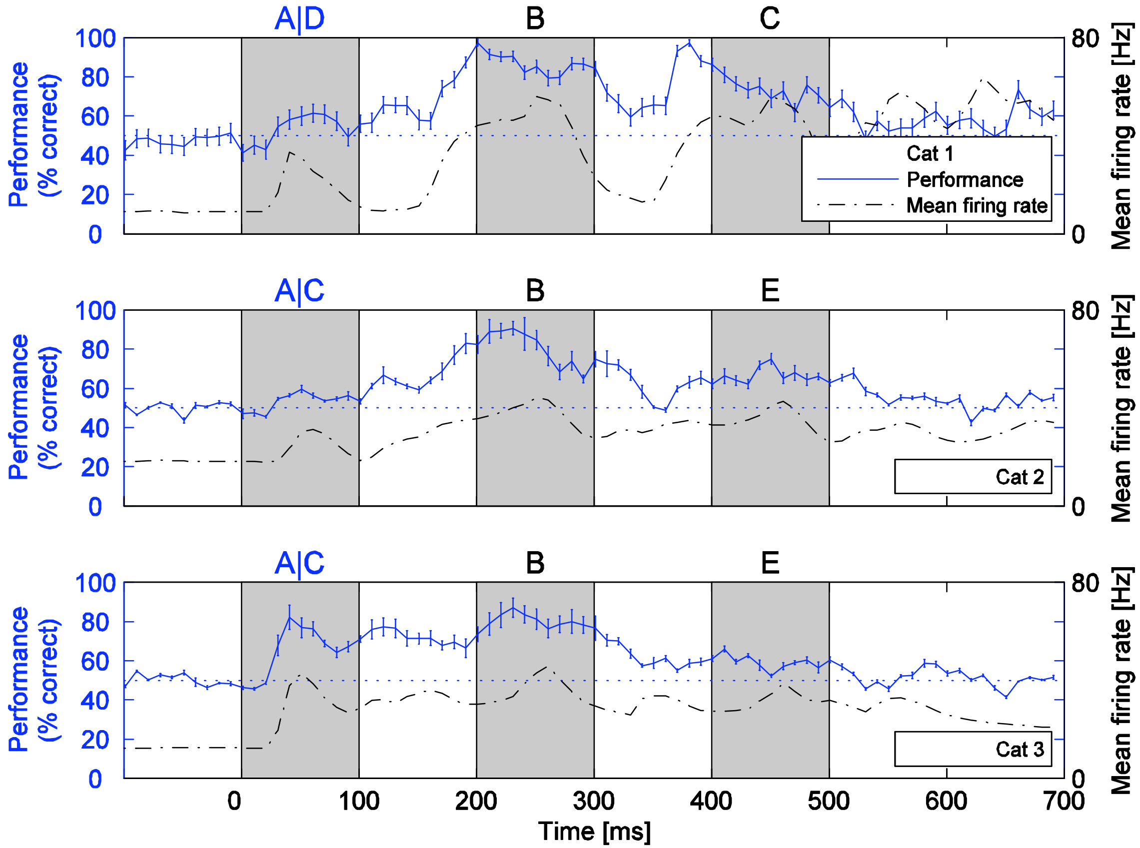
[Nikolic, Haeusler, Singer, Maass, 2007]





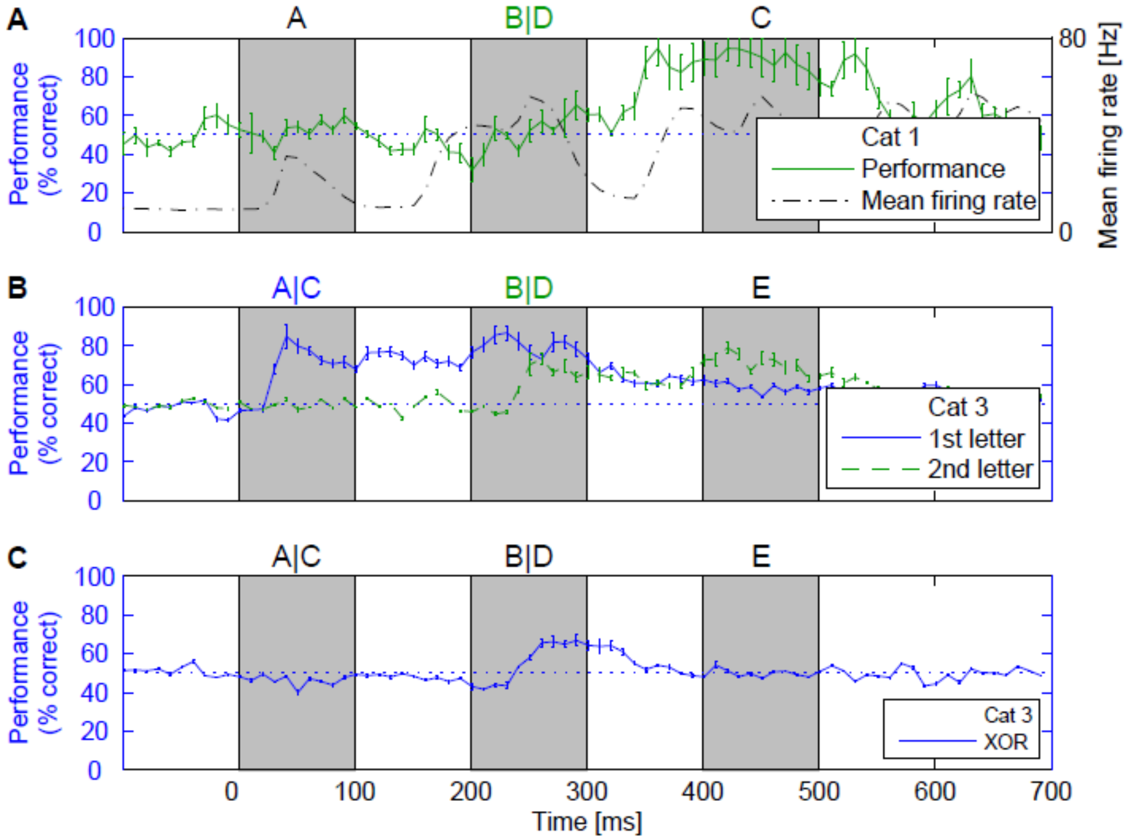
Data from visual cortex

- Information about previously shown letters is maintained during presentation of subsequent letters (temporal integration)



Data from visual cortex

- Information from two subsequent letters is nonlinearly combined in the circuit (kernel property)



Data from auditory cortex

- Recordings with 4 electrodes from area A1 in awake ferrets (unpublished data from the lab of Shibab Shamma)

unpublished experimental data was removed from this publicly available version

Data from auditory cortex

- Information is maintained during presentation of the next tone (temporal integration)

unpublished experimental data was removed from this publicly available version

Data from auditory cortex

- Almost all information contained in the spike trains can be extracted by linear classifiers (kernel property)

unpublished experimental data was removed from this publicly available version

Summary

- I have presented a model for online computing with trajectories in cortical microcircuits
- It views cortical computations from the perspective of generic preprocessing for learning
- Slow Feature Analysis (SFA) as an unsupervised mechanism for training readouts
- Predictions of this model (temporal integration of information, kernel property) have been tested by experimental data

Thank you for your attention

References

Rabinovich, Huerta, and Laurent. Transient dynamics for neural processing. *Science* 321 (2008), pp. 48-50

D. Nikolic, S. Haeusler, W. Singer, and W. Maass. Temporal dynamics of information content carried by neurons in the primary visual cortex. In *Proc. of NIPS 2006, Advances in Neural Information Processing Systems*, volume 19, pages 1041-1048. MIT Press, 2007.

E.M. Izhikevich. Solving the distal reward problem through linkage of STDP and dopamine signaling. *Cerebral Cortex* 17:2443-2452, 2007.

R. Legenstein, D. Pecevski, and W. Maass. A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS Computational Biology*, 4(10):1-27, 2008.

L. Wiskott L, T.J. Sejnowski. Slow feature analysis: unsupervised learning of invariances. *Neural Computation* 14:715-770, 2002.

S. Häusler and W. Maass. A statistical analysis of information processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, 17(1):149-162, 2007.

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.

W. Maass, P. Joshi, and E. D. Sontag. Computational aspects of feedback in neural circuits. *PLoS Computational Biology*, 3(1):e165, 1-20, 2007.