NEURAL CONTROL THEORY

A cognitive modeling history

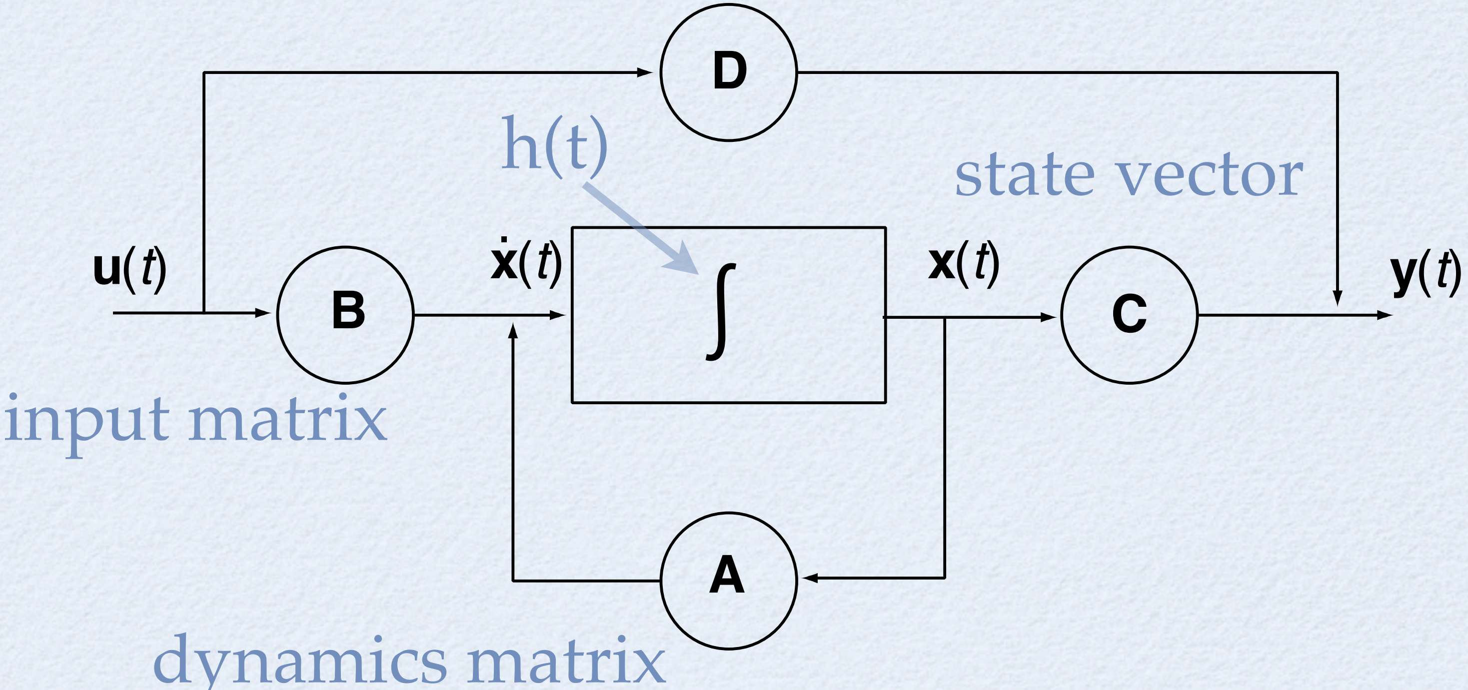
- Cybernetics (40s) Goals. Neurophysiologists (McCulloch & Rosenblueth), mathematicians (Weiner) and engineers (Forrester). Too behaviourist, classical control.

- GOFAI (60s-) Representation and computation. Turing machines, von Neumann architecture, etc. Largely ignores time

- Contemporary (90s-) Time is essential, obvious to neurophysiologists. Idea: reintroduce modern control theory

Standard control system

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)\\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)\end{aligned}$$



Neural control theory

- The synaptic dynamics dominate the overall population dynamics.
- We need to use the *intrinsic* synaptic dynamics to characterize 'neural' control theory

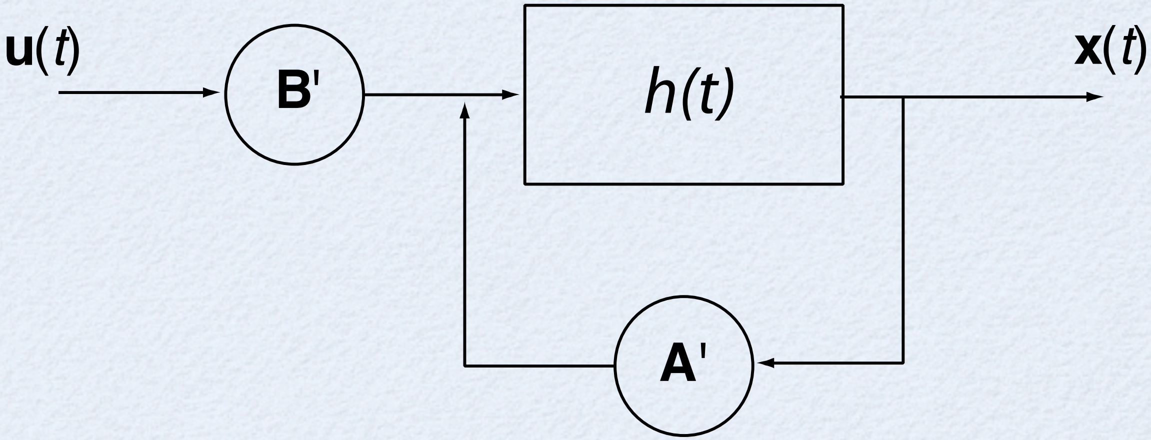
$$h(t) = \frac{1}{\tau} e^{-t/\tau}$$

- Laplace transform is:

$$h(s) = \frac{1}{1 + s\tau}$$

Neural control theory

- Put this into the control diagram (leaving out the **C** and **D** matrices)



$$
\begin{aligned}
\mathbf{x}(s) &= \frac{1}{1+s\tau}\left[\mathbf{A}'\mathbf{x}(s)+\mathbf{B}'\mathbf{u}(s)\right] \\
&= \frac{\tau^{-1}}{\tau^{-1}+s}\left[\mathbf{A}'\mathbf{x}(s)+\mathbf{B}'\mathbf{u}(s)\right]
\end{aligned}
$$

Neural control theory

- So,

$$
\begin{aligned}
(\tau^{-1} + s)\mathbf{x}(s) &= \tau^{-1}\left[\mathbf{A}'\mathbf{x}(s) + \mathbf{B}'\mathbf{u}(s)\right] \\
s\mathbf{x}(s) &= \tau^{-1}\left[\mathbf{A}' - \mathbf{I}\right]\mathbf{x}(s) + \tau^{-1}\mathbf{B}'\mathbf{u}(s)
\end{aligned}
$$

- Recall also,

$$
s\mathbf{x}(s) = \mathbf{A}\mathbf{x}(s) + \mathbf{B}\mathbf{u}(s)
$$

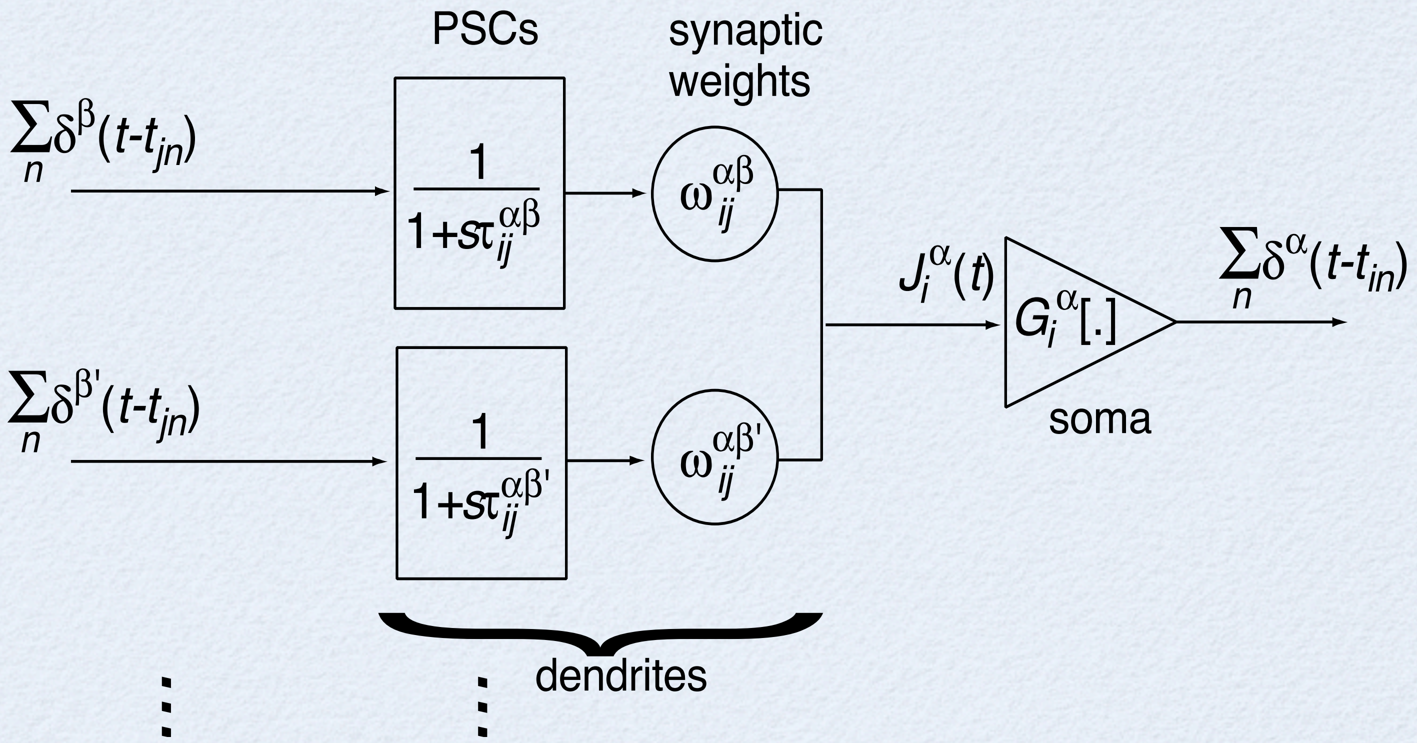
- Equating the two and solving gives the ‘translation’

$$
\begin{aligned}
\mathbf{A}' &= \tau\mathbf{A} + \mathbf{I} \\
\mathbf{B}' &= \tau\mathbf{B}
\end{aligned}
$$

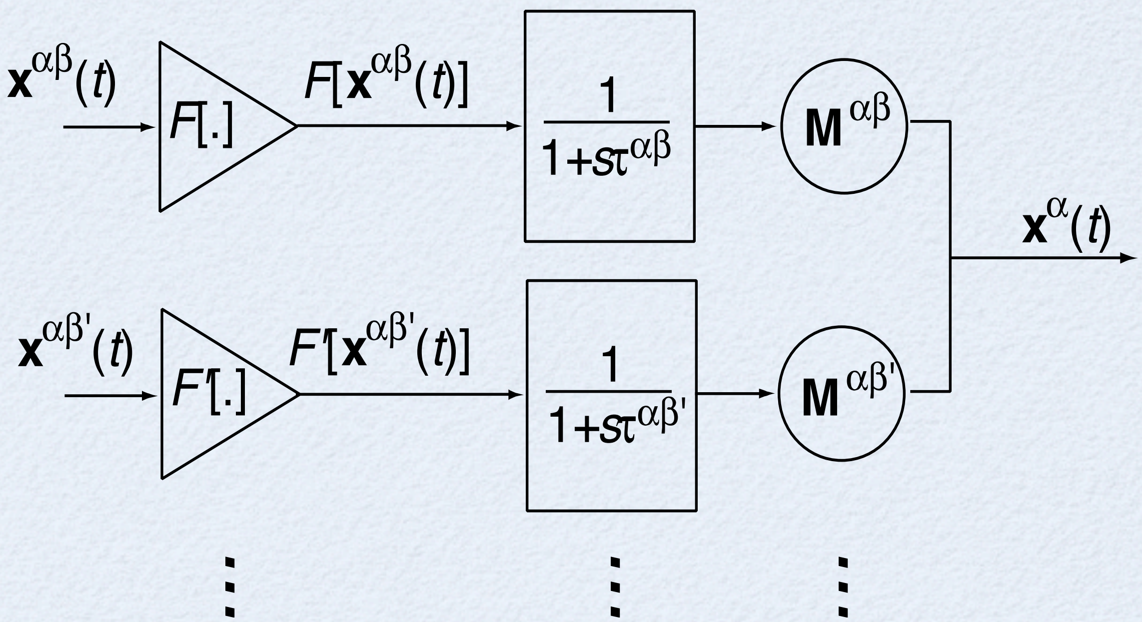
Generic neural subsystem

- GNS: a theoretical subsystem that can be mapped to any spiking neural population involved in some dynamic transformation
- Let's introduce some general notation by looking at different levels of description
- Then we'll write the equations for each principle of the NEF

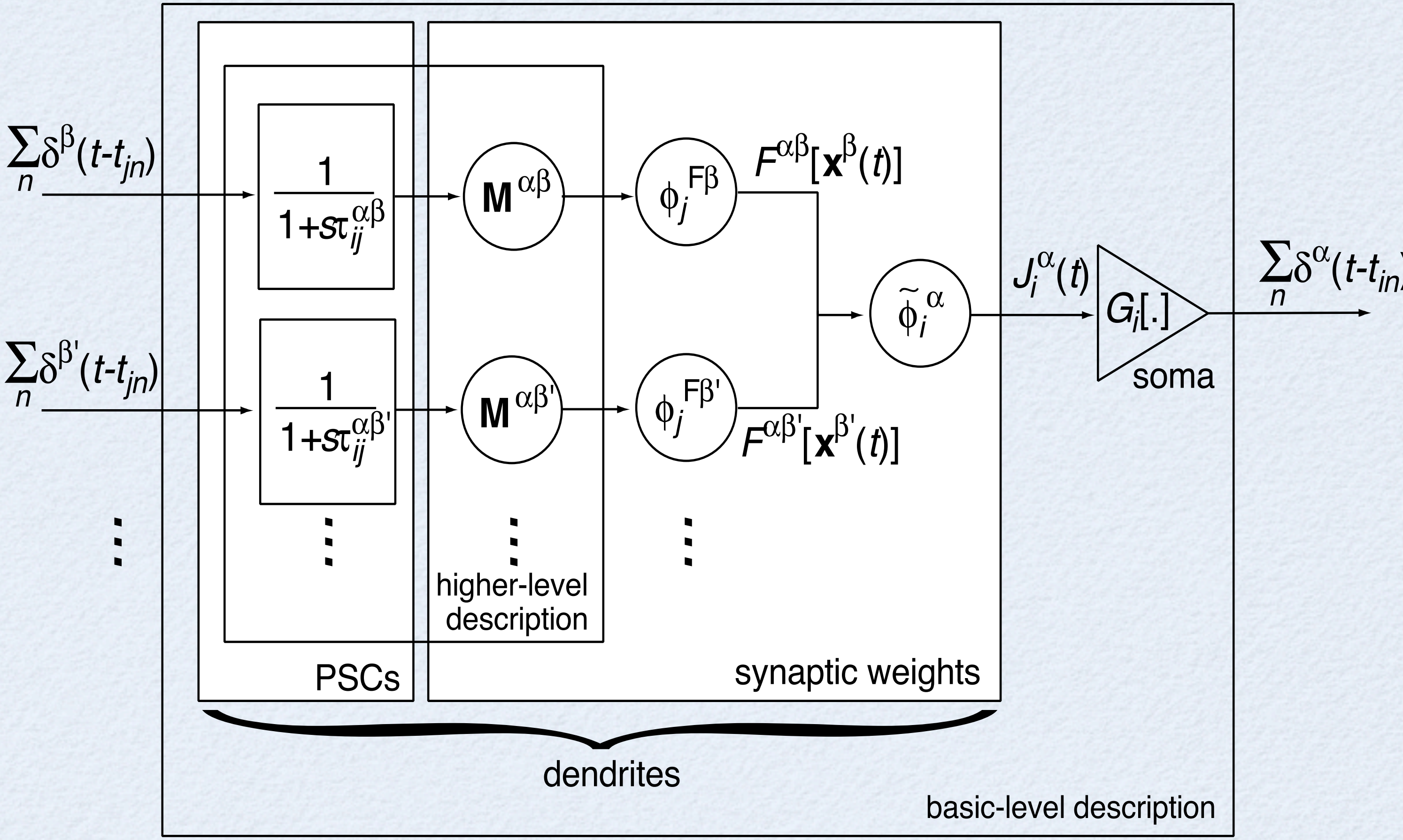
Basic-level description



Higher-level description



Generic neural subsystem



So we can see $\omega_{ij}^{\alpha\beta} = \left\langle \tilde{\phi}_i^{\alpha} \mathbf{M}^{\alpha\beta} \phi_j^{\alpha F\beta} \right\rangle_m$

Principle 1: Representation

- Encoding

$$\sum_n \delta(t - t_{in}) = G_i \left[\alpha_i \left\langle \tilde{\phi}_i \mathbf{x}(t) \right\rangle_m + J_i^{bias} \right]$$

- Decoding

$$\hat{\mathbf{x}}(t) = \sum_i a_i(\mathbf{x}(t)) \phi_i^{\mathbf{x}}$$

- where

$$\begin{aligned} a_i(\mathbf{x}(t)) &= \sum_n h_i(t) * \delta(t - t_{in}) \\ &= \sum_n h_i(t - t_{in}) \end{aligned}$$

Principle 2: Transformation

- Same encoding with a different decoding

$$\hat{f}(\mathbf{x}(t)) = \sum_i a_i(\mathbf{x}(t))\phi_i^f$$

- (where a_i is defined as in principle 1)

Principle 3: Dynamics

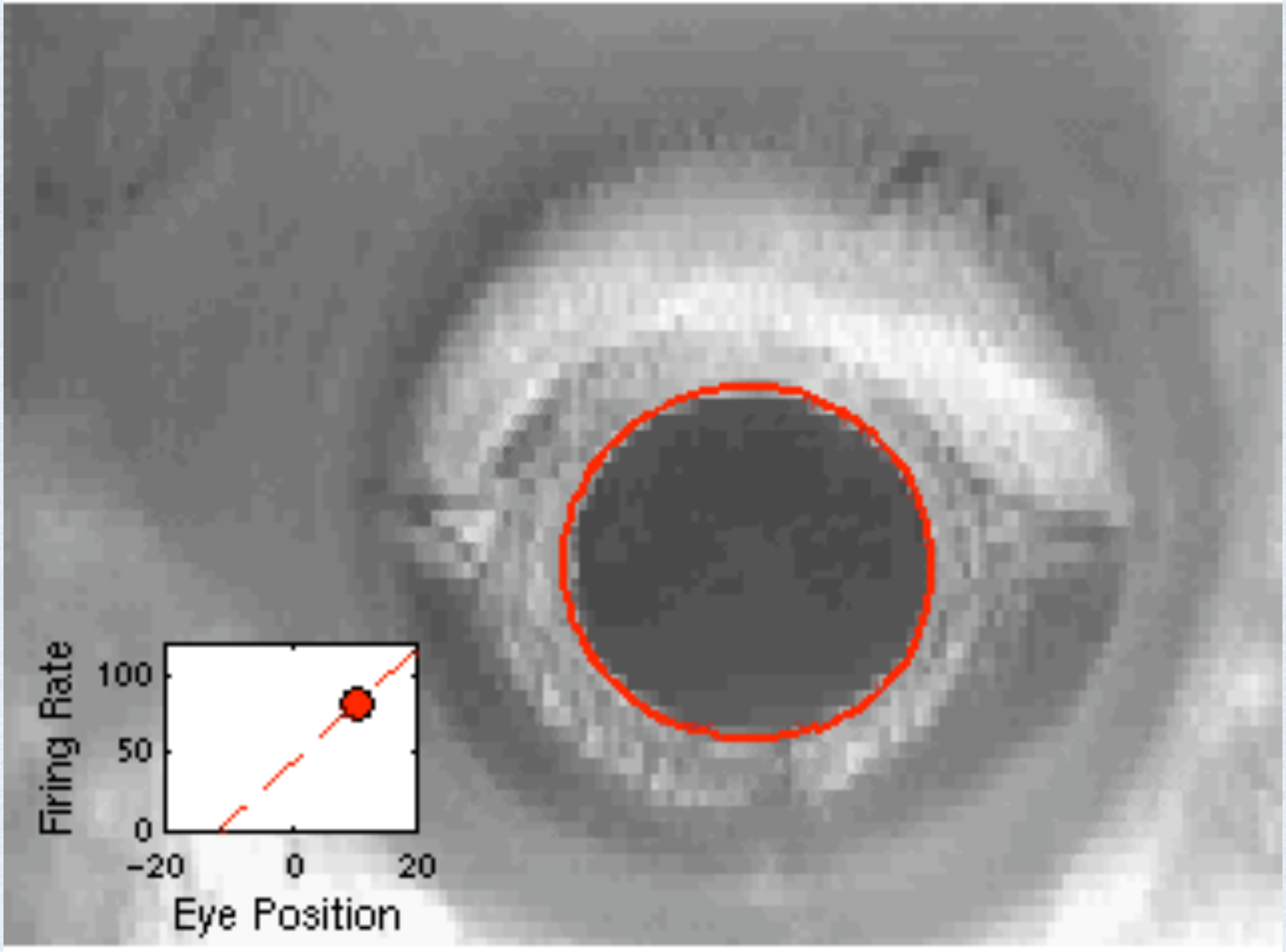
- Allowing **x**(t) to be the neural repn (and hence state variable) and **u**(t) to be the input, we have the following

$$\sum_n \delta(t - t_{in}) = G_i \left[\alpha_i \left\langle \tilde{\phi}_i \left(h_i(t) * [\mathbf{A}'\mathbf{x}(t) + \mathbf{B}'\mathbf{u}(t)] \right) \right\rangle_m + J_i^{bias} \right]$$

- In fact this could be written more generally by having the control system be any f(**A**,**B**,**x**,**u**,t)

- Principle 2 then tells us how to compute f

Neural integrator



Neural integrator

- NPH & VN turn velocity signals into eye position commands
- Difficult problem to solve, but simple to formulate:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

$$\mathbf{A} = 0$$

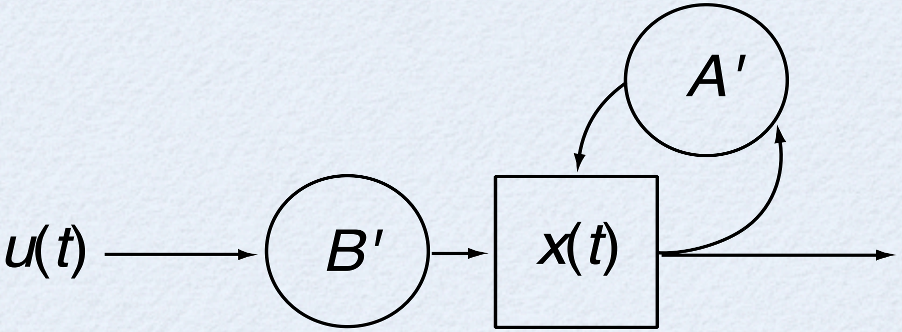
$$\mathbf{B} = 1$$

Neural integrator

- So, in 'neural control' we have (assuming input and recurrent time constants are equal)

$$\mathbf{A}' = 1$$

$$\mathbf{B}' = \tau$$



Neural integrator

- Substitute this into the encoding equation:

$$a_j(t) = G_j\left[\alpha_j\left\langle x(t)\tilde{\phi}_j\right\rangle + J_j^{bias}\right]$$

- Gives

$$\begin{aligned} a_j(t) &= G_j\left[\alpha_j\left\langle h(t) * \tilde{\phi}_j\left[A'\sum_i a_i(t)\phi_i^x + B'u(t)\right]\right\rangle + J_j^{bias}\right] \\ &= G_j\left[h(t) * \left[\sum_i \omega_{ji}a_i(t) + B'\tilde{\phi}_j u(t)\right] + J_j^{bias}\right] \end{aligned}$$

- Where

$$\omega_{ji} = \alpha_j A'\phi_i^x\tilde{\phi}_j$$

Neural integrator

- If we expect any error in $x(t)$, we won't be able to build a perfect integrator... and we expect error
- We can think of error as being captured by some gain, k, in the circuit (which is a function of x)
- This gain effectively acts to modify A, and any deviation of A from 0 (or 1 in the neural case) moves the circuit over time, even with no input

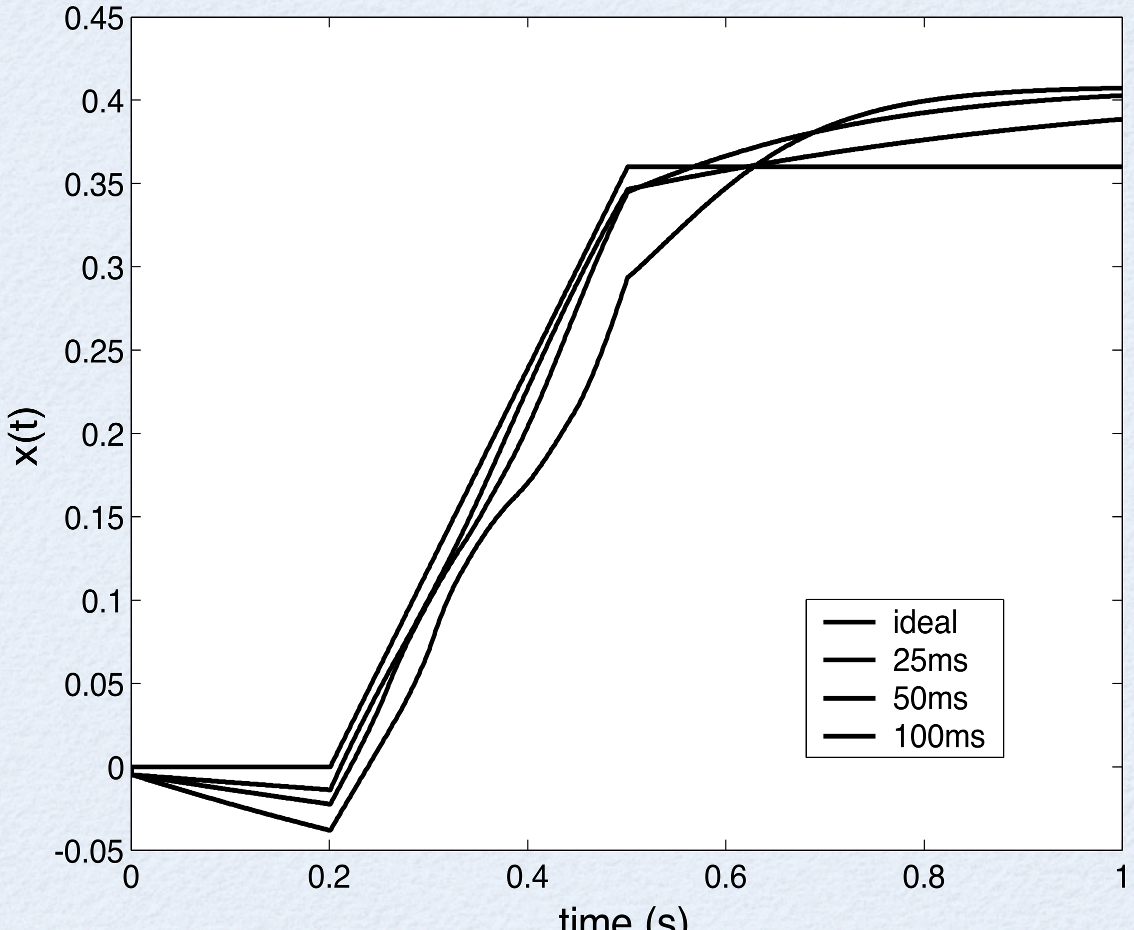
Effective time constant

- So, A acts like a rate constant: $A = -\frac{1}{\tau_{eff}}$
- Or, in neural terms:

$$\begin{aligned} -\frac{1}{\tau_{eff}} &= \frac{A' - 1}{\tau} \\ \tau_{eff} &= \frac{\tau}{1 - A'} \end{aligned}$$

- So, an increase in synaptic time constant will lengthen the effective time constant

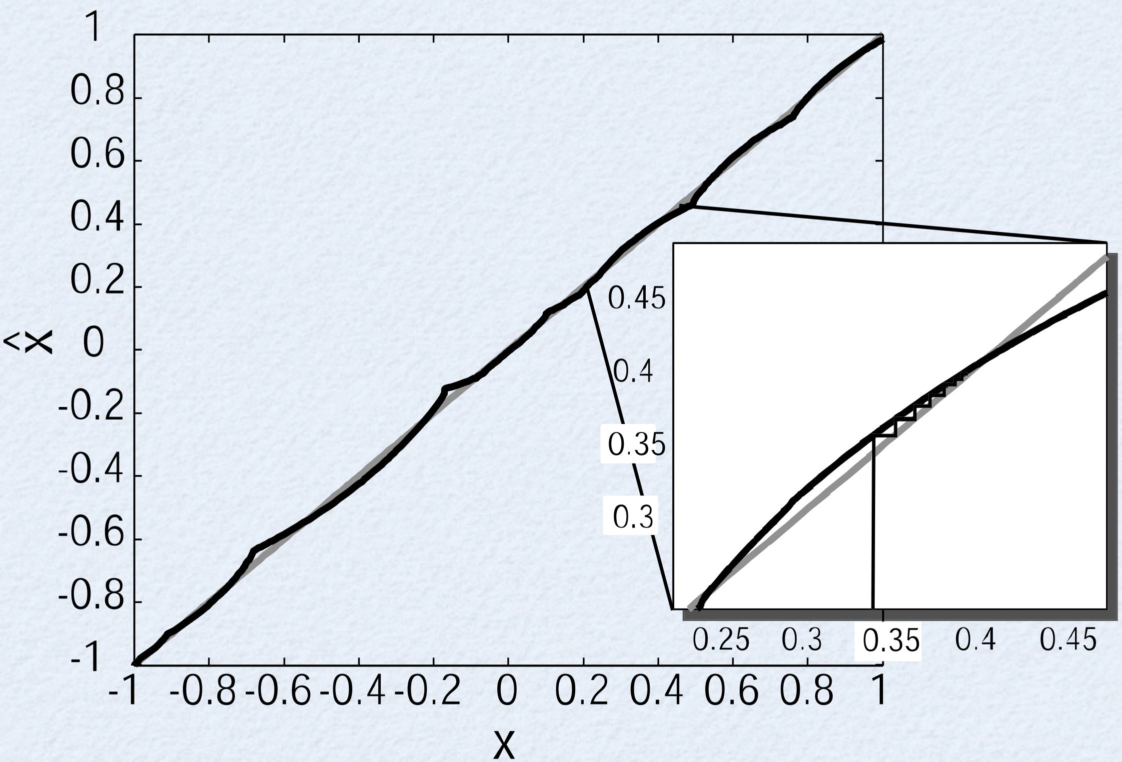
Integrator and synaptic TC

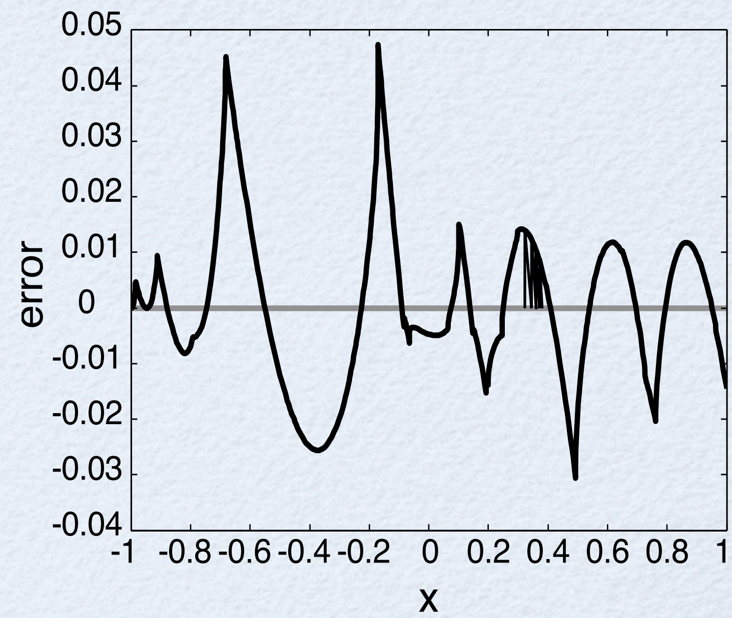


Representation error

- Since we can set A′, and setting it to 1 gives an infinite time constant, the only remaining source of error is representational
- So, looking at the deviation from identity (as we've graphed many times) gives insight into the dynamics of the system
- You can see stable points and drift speed directly from this graph

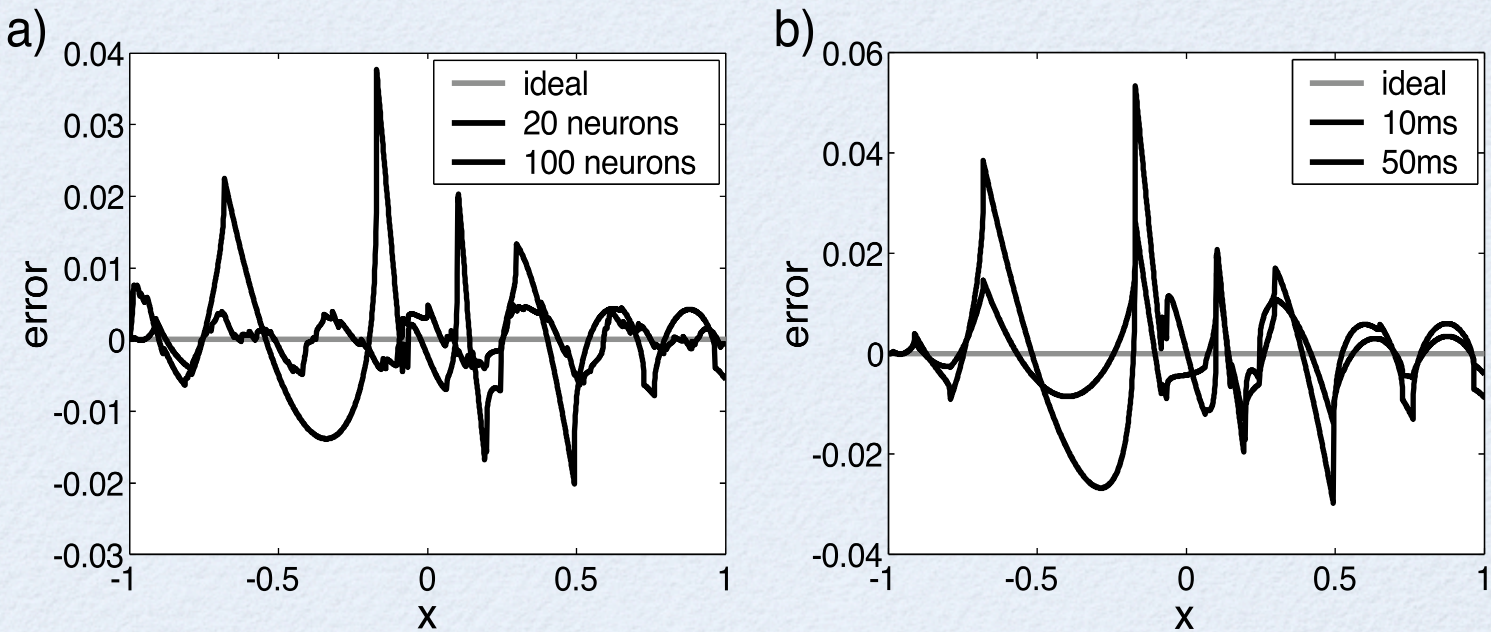
Dynamics and error





Fixed points in integrator

- Effect of number of neurons and membrane time constants



Tunable filter

- We can take this a step further and tune A′ directly with a neural ensemble $A'(t) = \sum_l b_l(t)\phi_l^{A'}$
- Direct substitution gives

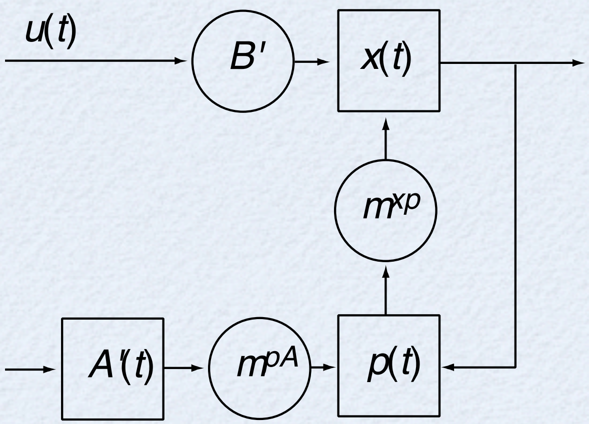
$$a_j(t) \quad = \quad G_j\left[\alpha_j\left(h(t) * \tilde{\phi}_j\left[\sum_l b_l(t)\phi_l^{A'}\sum_i a_i(t)\phi_i^x + B'u(t)\right]\right) + J_j^{bias}\right]$$

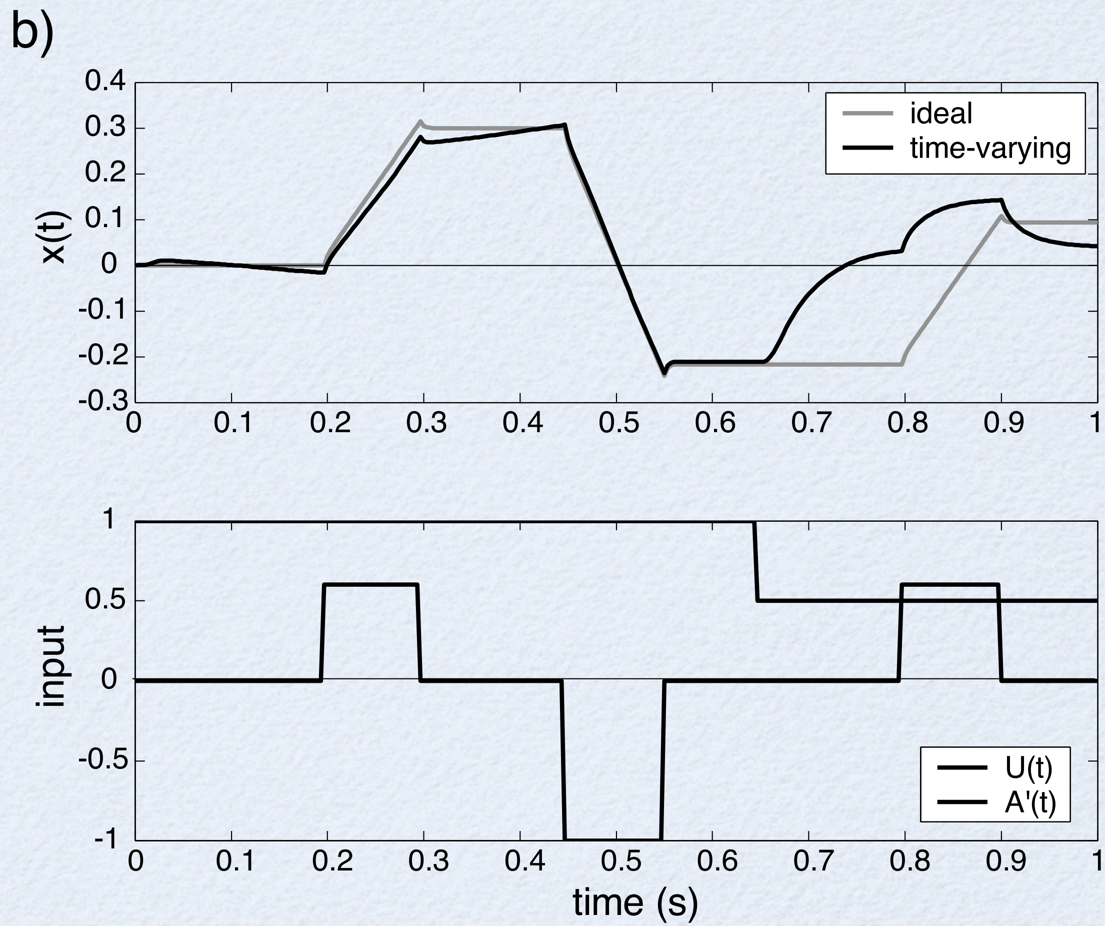
- Using an intermediation population gives

$$a_j(t) = G_j\left[\alpha_j\left(h(t) * \tilde{\phi}_j\left[\sum_m c_m(t)\boldsymbol{\phi}_{\boldsymbol{m}}^{\mathbf{p}} + B'u(t)\right]\right) + J_j^{bias}\right]$$

- Using a 2D population is most efficient

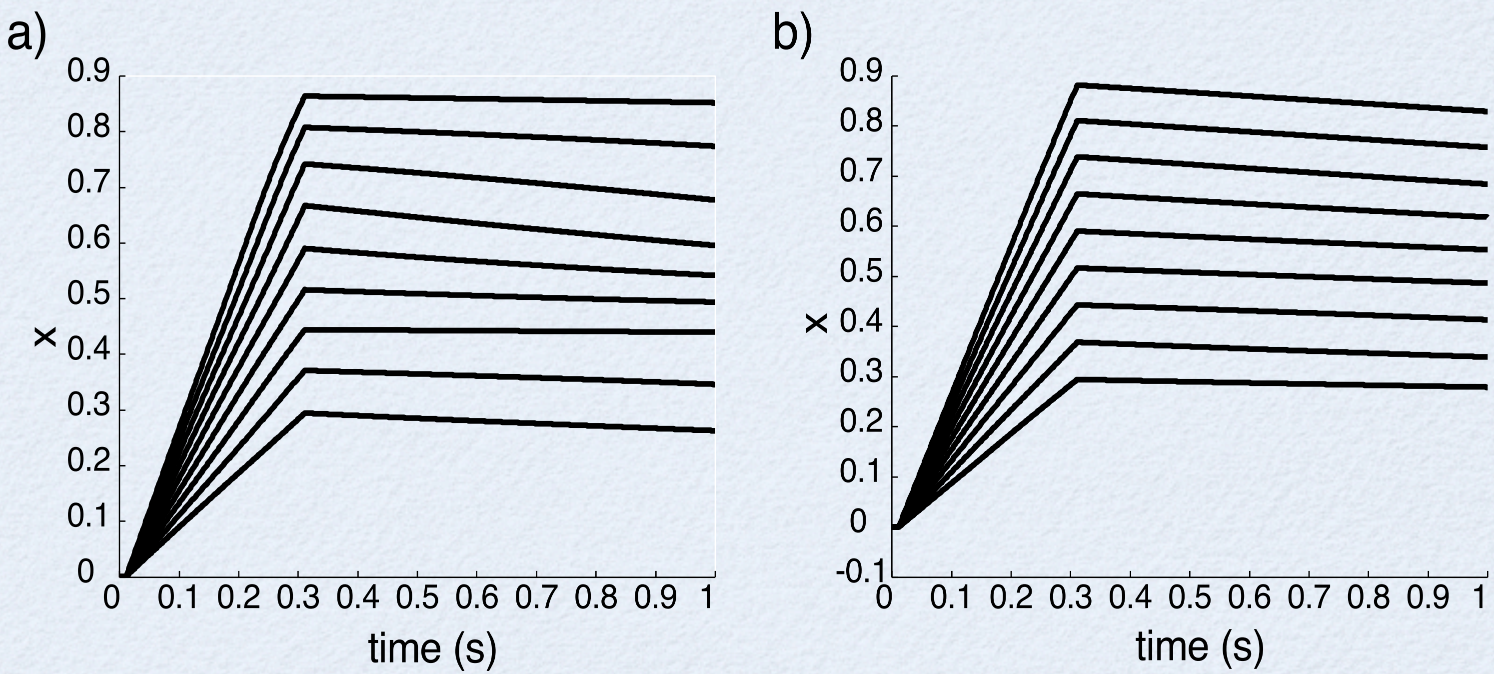
Tunable filter





Humans and goldfish

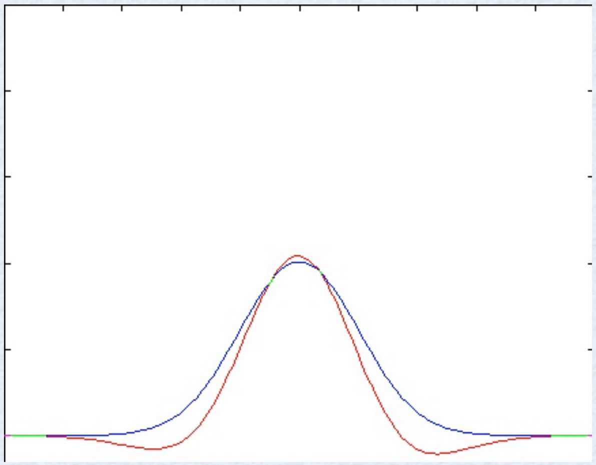
- Humans (left) have more neurons, and hence slower drift (70s vs 10s)
- Both have centripital drift (i.e. $A'<1$)

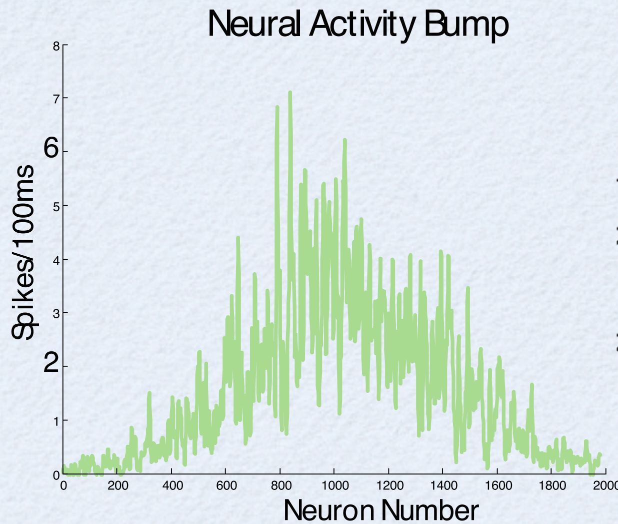


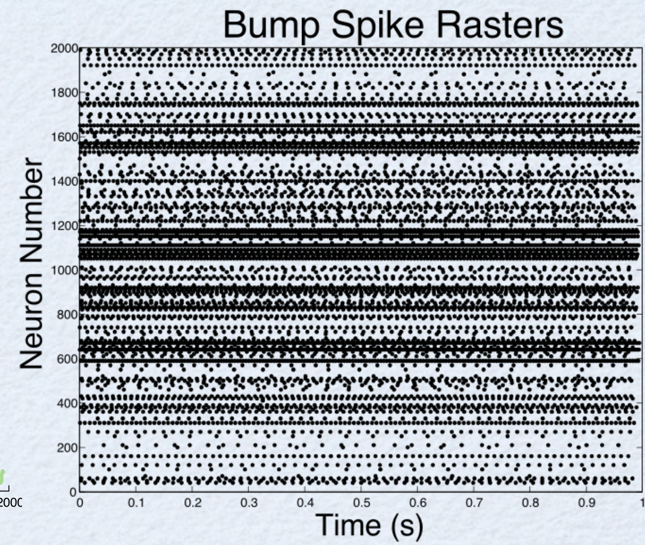
Working memory

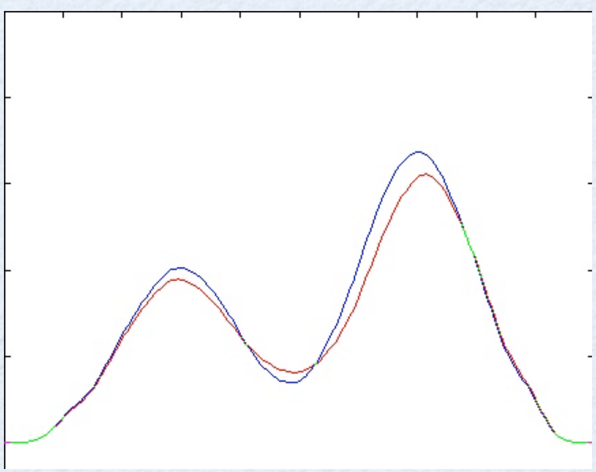
- Just another integrator...
- Defined over the co-efficients in the high-dimensional vector space representing functions
- This model was one of the first successful ones of parametric working memory (observed e.g., in LIP)

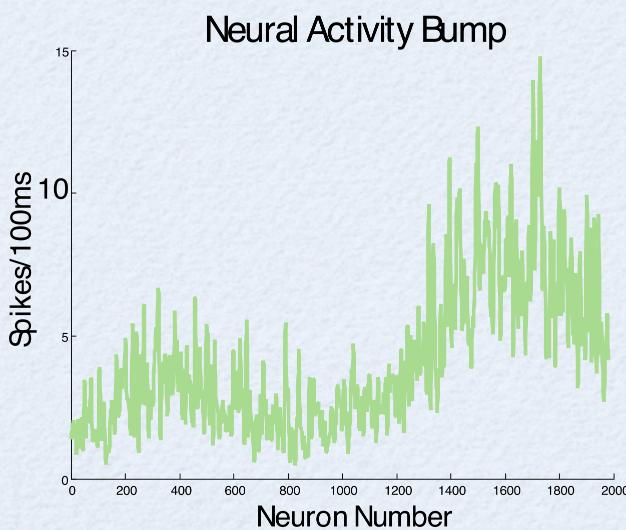
Implementation (2000 neurons, 5000 iterations)

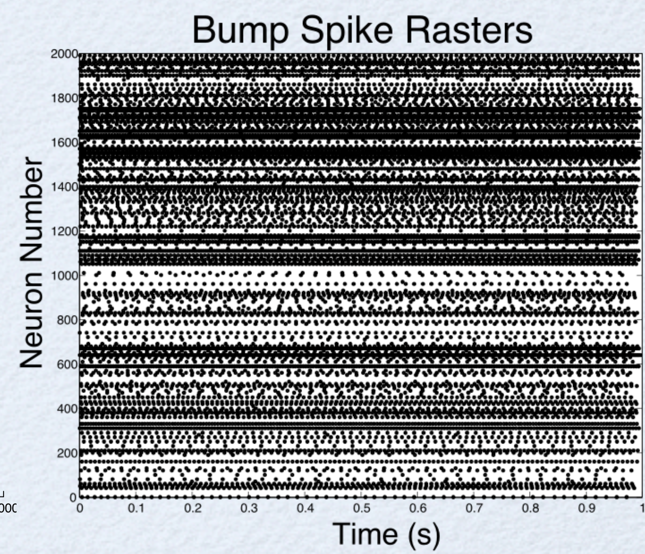






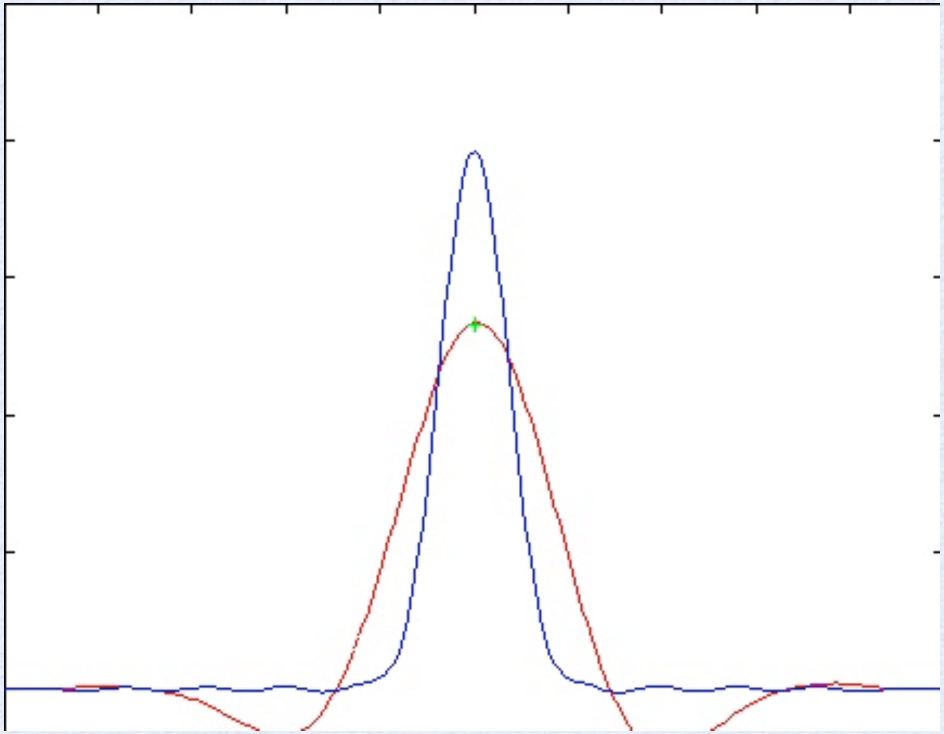






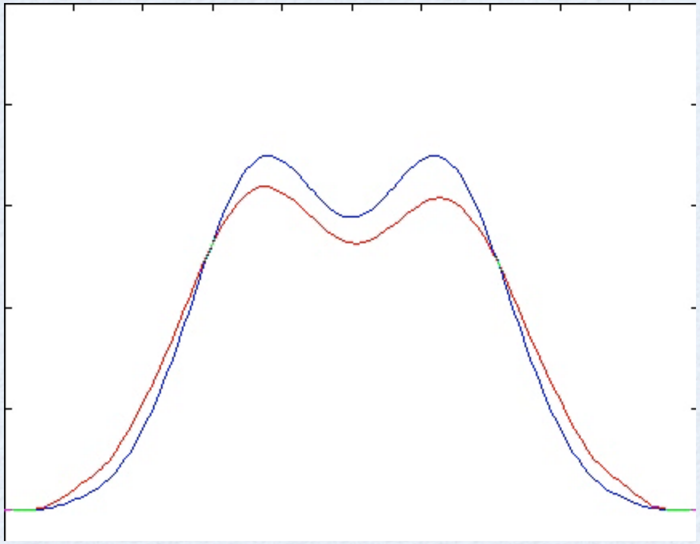
Predictions

- For narrow stimuli central cells (+) have decreased firing rate but peripheral cells (o) have increased firing rate.

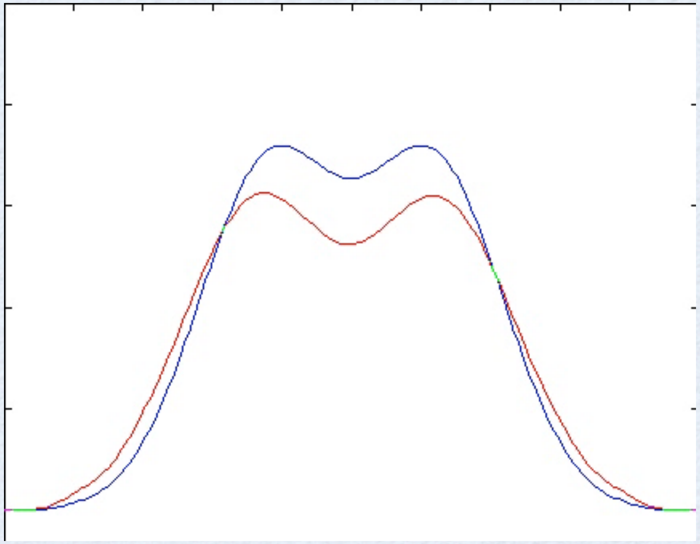


Predictions

- For multiple stimuli whose centers are too close together, there are two kinds of error, *forgetting* and *blending*.



Forgetting



Blending