

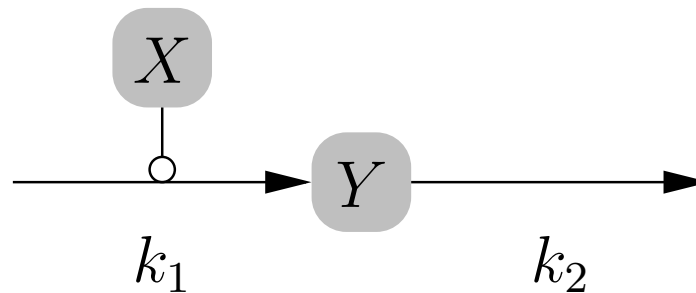
# Intracellular neural networks

TU/e node

ESIGNET meeting, Birmingham, April 2007

# Previously ...

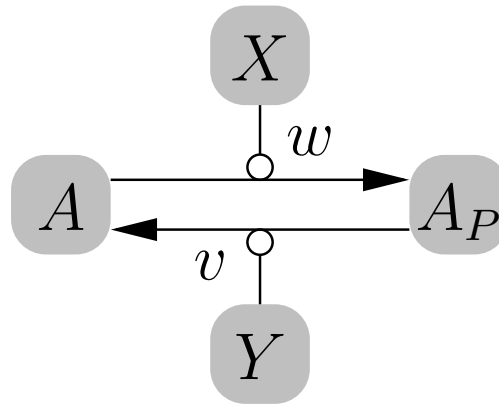
- WP10: A Mathematical Theory for CSNs
- "Create a mathematical understanding of the computational properties of biochemical networks"
- At the Dublin meeting:
  - Hand constructed networks that compute algebraic functions, based on elementary "buffer" motif
  - Accepted in ALife's S.I. on Artificial Chemistries



# ...continued

- Our open questions in Dublin
  - Proper kinetic laws (Michaelis-Menten, ...)
  - Resources = waste particles
  - Phosphorylation cycles
- Last 6 months: Biochemical neural networks
  - Phosphorylation cycle = biochemical neuron
  - Which techniques can we adopt from neural networks that also apply to signal transduction?
  - Study computational power in these neural networks
  - Error-back propagation of these networks
  - Multi-functionality of these networks
  - Stochastic models

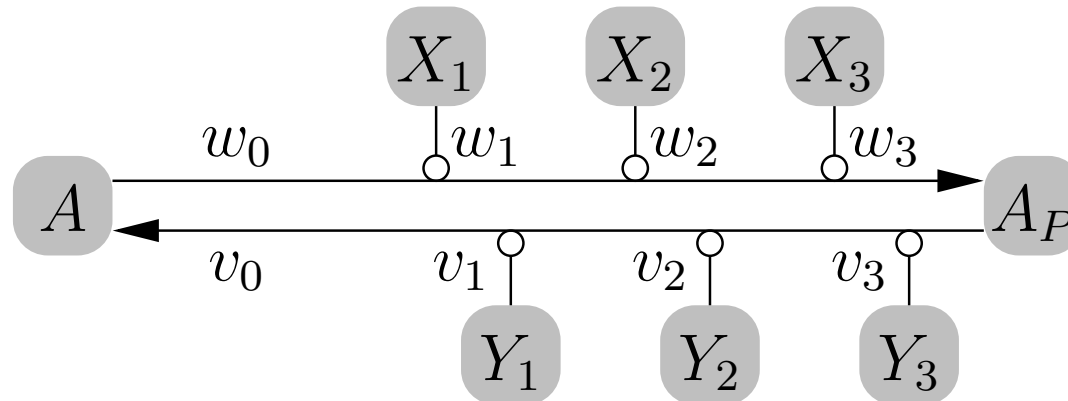
# The phosphorylation cycle



- We consider the steady state (with mass-action):

$$\begin{aligned} \dot{a}_p &= wax - va_p y \\ a + a_p &= c \\ \hat{a}_p &= c \frac{wx}{wx + vy} \end{aligned}$$

# Multiple inputs

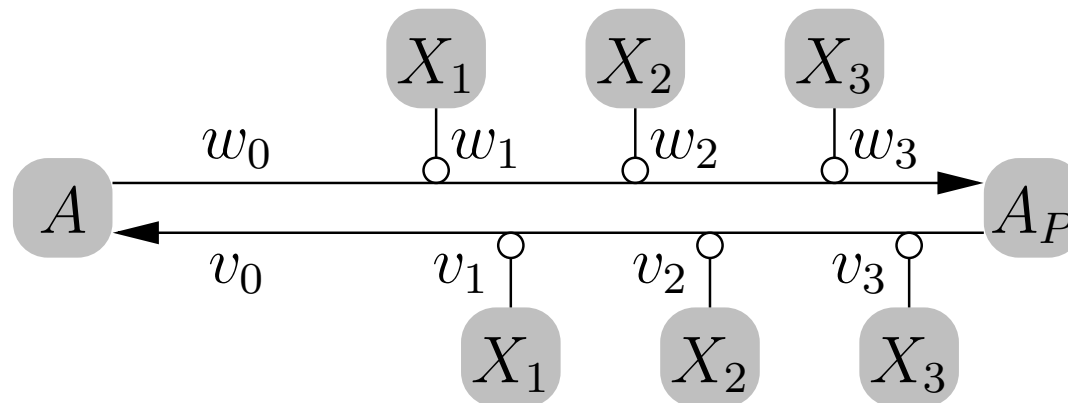


- Output is a function of the weighted sums of inputs

$$\hat{a}_p = c \frac{w_0 + \sum_{i=1}^n w_i x_i}{w_0 + \sum_{i=1}^n w_i x_i + v_0 + \sum_{i=1}^{n'} v_i y_i}$$

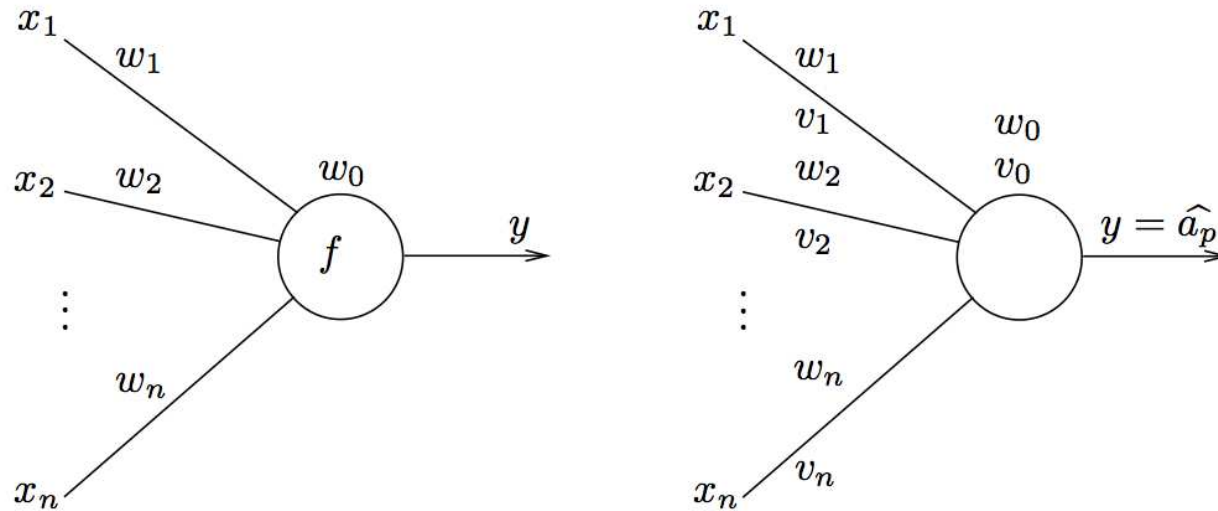
# Our biochemical neuron

- Theoretically, we can assume kinases = phosphatases, each with specific reaction rates:



$$\hat{a}_p = c \frac{w_0 + \sum_{i=1}^n w_i x_i}{w_0 + \sum_{i=1}^n w_i x_i + v_0 + \sum_{i=1}^n v_i x_i}$$

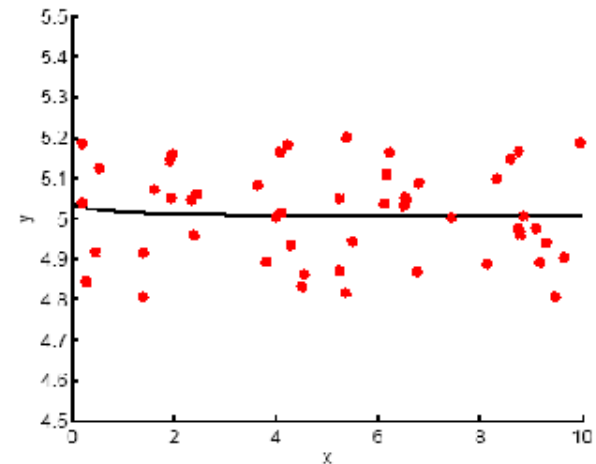
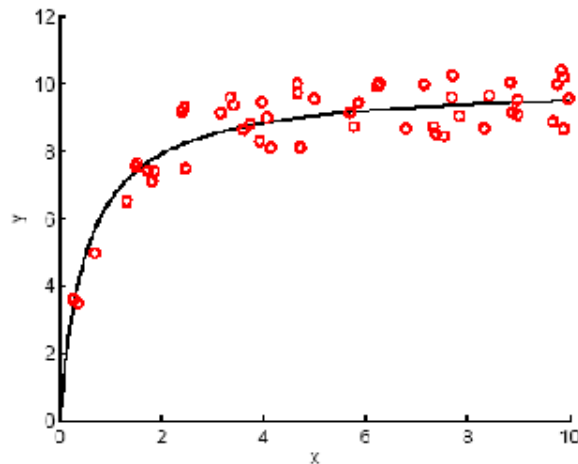
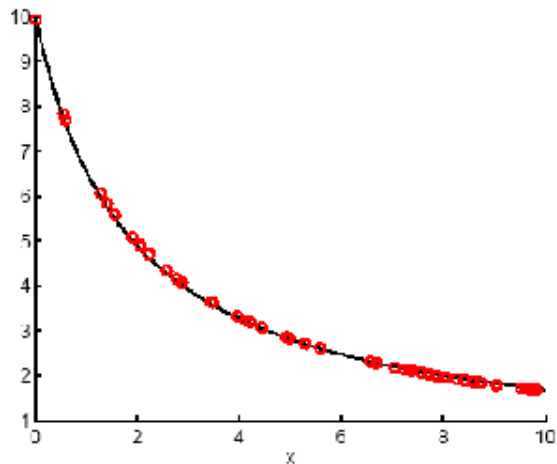
# Biochemical vs standard artificial neuron



- 2x parameters, strictly positive values and weights
- Activation function is function of quotient of linear combinations of inputs
- Total mass  $c$ , which can be assumed to be 1

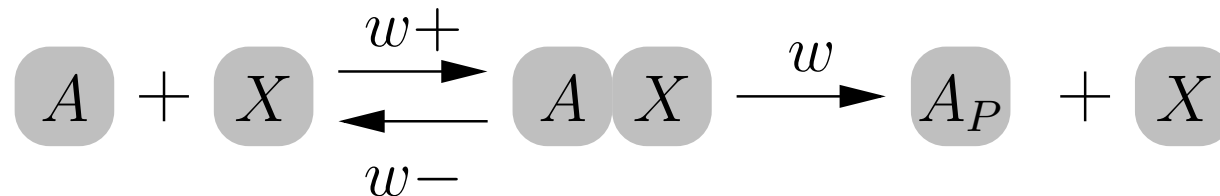
# Parameter fitting with gradient descent

- Chemical neuron with one input computes function of form  $C \frac{x+a}{x+b}$ , with  $C, a, b > 0$
- With gradient descent, we can optimize weights of neuron to fit a given function, e.g.:



# Michaelis-Menten neuron

- Kinetics with substrate-enzyme complex



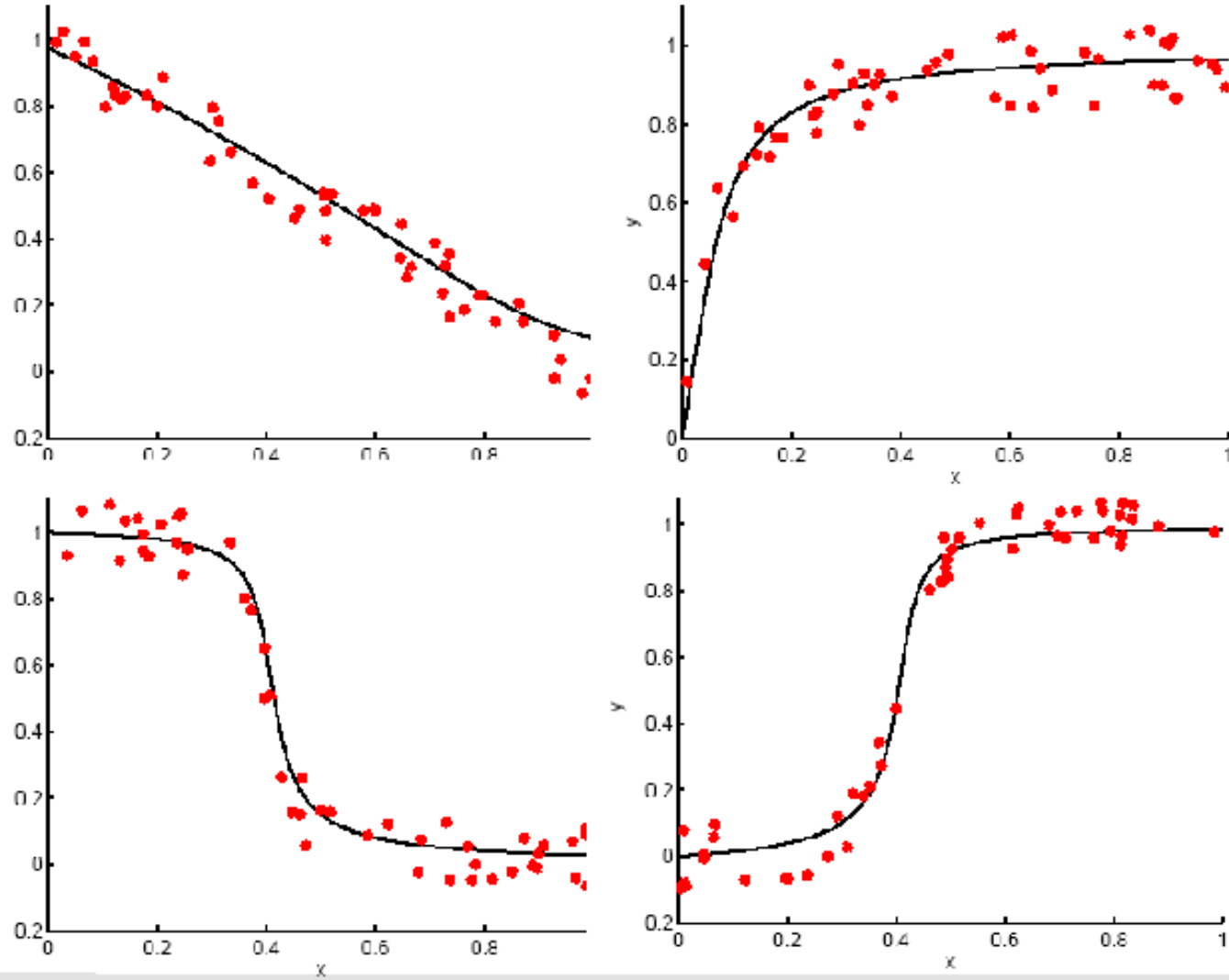
- MM kinetics:  $\frac{da_p}{dt} = \frac{axw}{k_m+a}$  with  $k_m = \frac{w_-+w}{w_+}$

- Steady state of MM neuron (Goldbeter-Koshland):

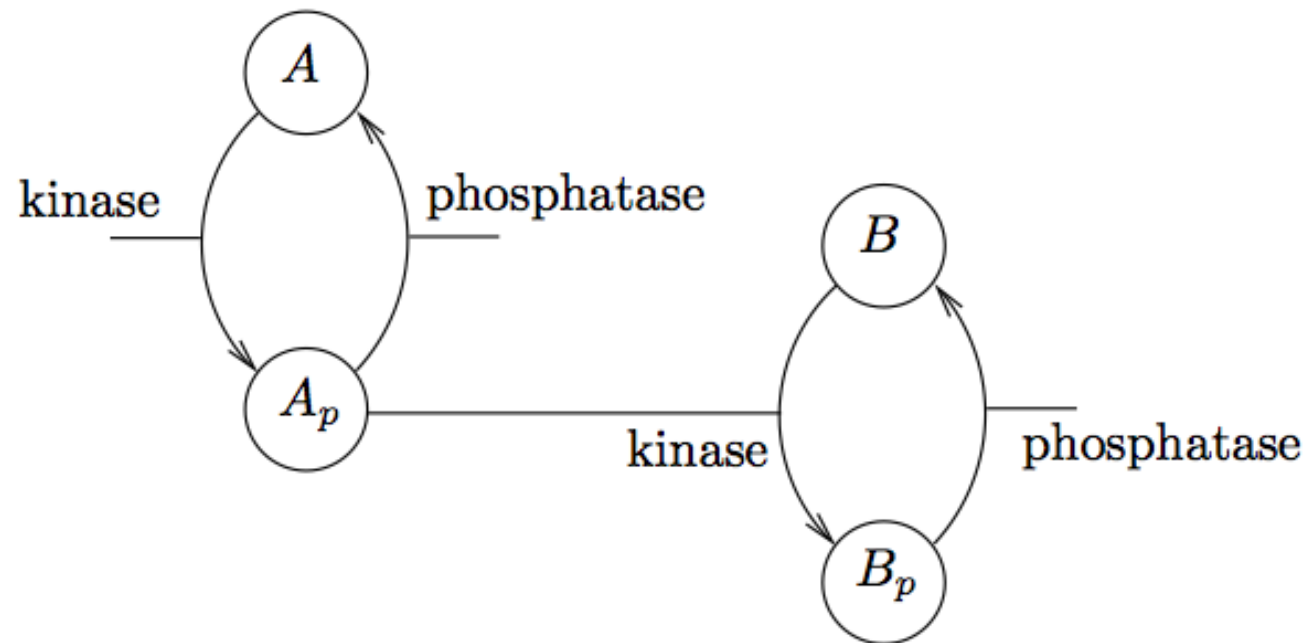
$$\hat{a}_p = \frac{p - Kp - q - Jq + \sqrt{4Kp(p - q) + ((K - 1)p + (J + 1)q)^2}}{2(p - q)}$$

where  $p = w_0 + \sum_i w_i x_i$ ,  $q = v_0 + \sum_i v_i x_i$ ,  $J = \frac{k_{m+}}{c}$ ,  $K = \frac{k_{m-}}{c}$

# Parameter fitting with MM neuron



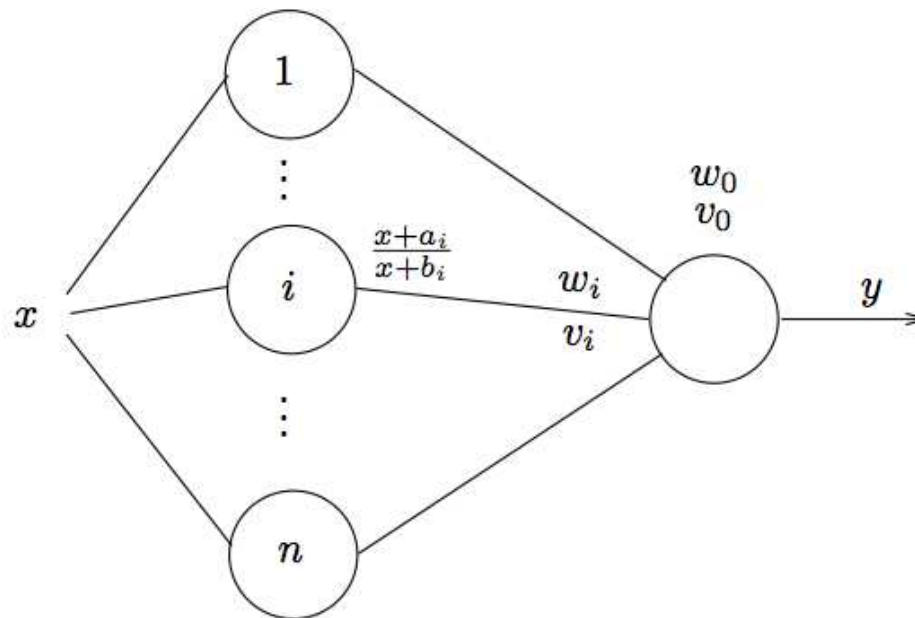
# Coupling of neurons



**Fig. 1.** Two coupled phosphorylation cycles,  $A_p$  acts as kinase for the  $B - B_p$  cycle

# Networks of neurons

- For now, we consider layered feed forward networks
- What functions can be computed with one layer of  $n$  mass-action neurons, with one input  $x$ ?



# Computable functions in 1 MA layer

$$y = c \frac{w_0 + \sum_{i=1}^n w_i \frac{x+a_i}{x+b_i}}{v_0 + w_0 + \sum_{i=1}^n (v_i + w_i) \frac{x+a_i}{x+b_i}}$$
$$= c \frac{w_0 \prod_{i=1}^n (x + b_i) + \sum_{i=1}^n w_i (x + a_i) \prod_{j=1, j \neq i}^n (x + b_j)}{v_0 \prod_{i=1}^n (x + b_i) + \sum_{i=1}^n v_i (x + a_i) \prod_{j=1, j \neq i}^n (x + b_j)}$$

- $y$  is the quotient of two  $n$ th degree polynomials
- Not all quotients of  $n$ th degree polynomials can be written as a layer
  - Polynomials have positive coefficients
  - Polynomials have roots with negative real parts

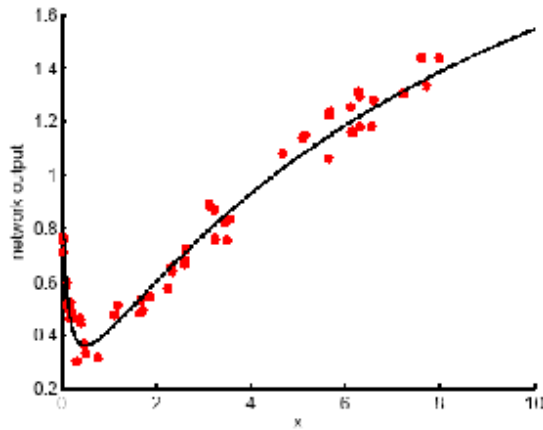
# Computable functions of networks

- Generalization to multiple hidden layers:
  - Again, one input  $x$
  - $r$  consecutive layers, with  $n_i$  nodes in layer  $i$
  - output is again quotient of polynomials  $P(x)/Q(x)$
  - $P(x)$  and  $Q(x)$  are polynomials of degree  $\prod_{i=1}^r n_i$
  - $P(x)$  and  $Q(x)$  have positive coefficients
- Submitted to ICANN07 (*Computing with Feedforward Networks of Artificial Biochemical Neurons*)

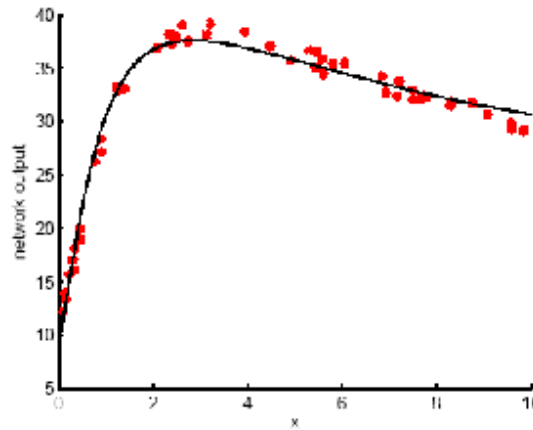
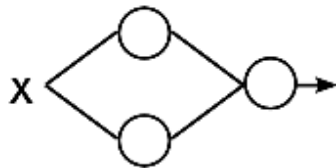
# Error back propagation

- Given 1 network topology (layered & FF), we can design various output functions, by tuning the parameters
- In NN: error backpropagation
  - Provide input-target pair
  - Compute error on output
  - Find partial derivative of parameters to decrease error
  - Propagate to preceding layers
- This is also possible with networks of biochemical (mass-action and Michaelis-Menten) neurons

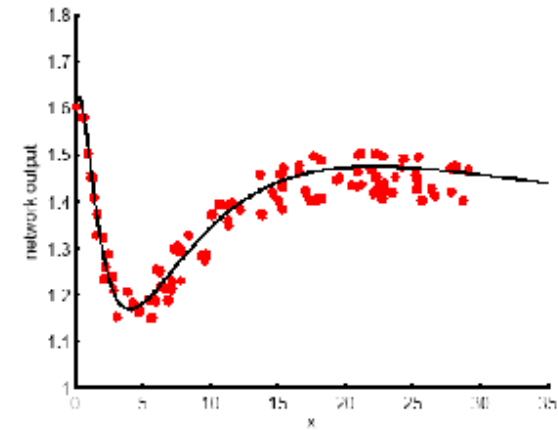
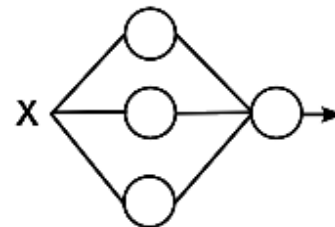
# Backpropagation with MA neurons



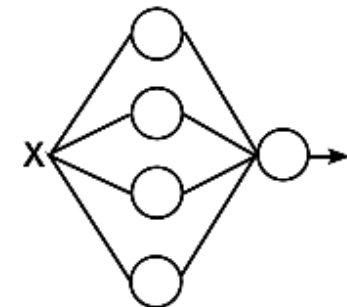
$$\frac{3x^2 + x + 1}{x^2 + 10x + 1}$$



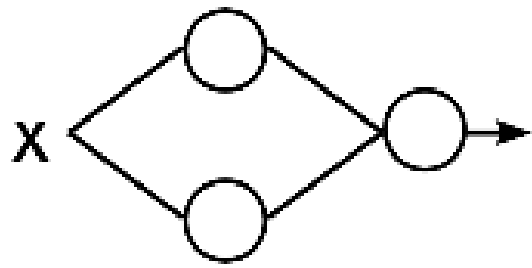
$$\frac{2x^3 + 48x^2 + 2x + 20}{\frac{1}{10}x^3 + 1\frac{1}{5}x^2 + \frac{21}{100}x + 2}$$



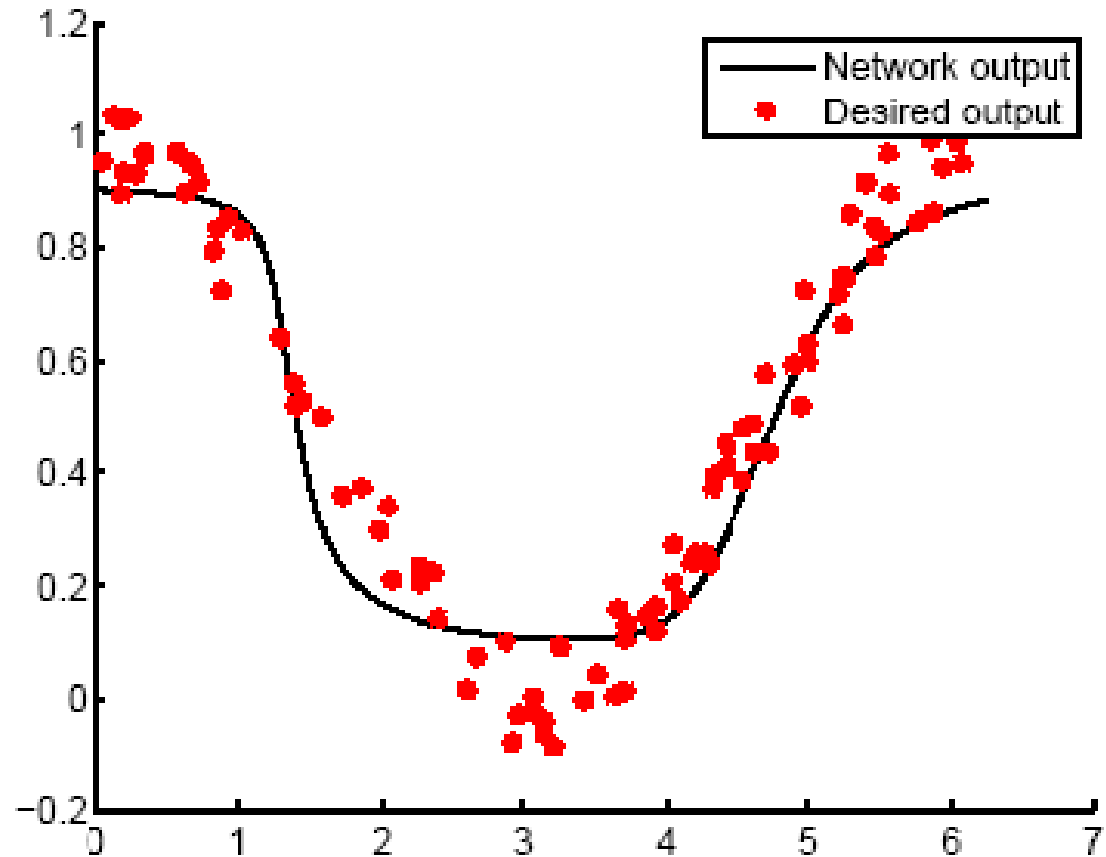
$$\frac{x^4 + 15x^3 + 140x^2 + 100x + 1000}{\frac{9}{10}x^4 + x^3 + 200x^2 + 10x + 620}$$



# Backpropagation with MA neurons



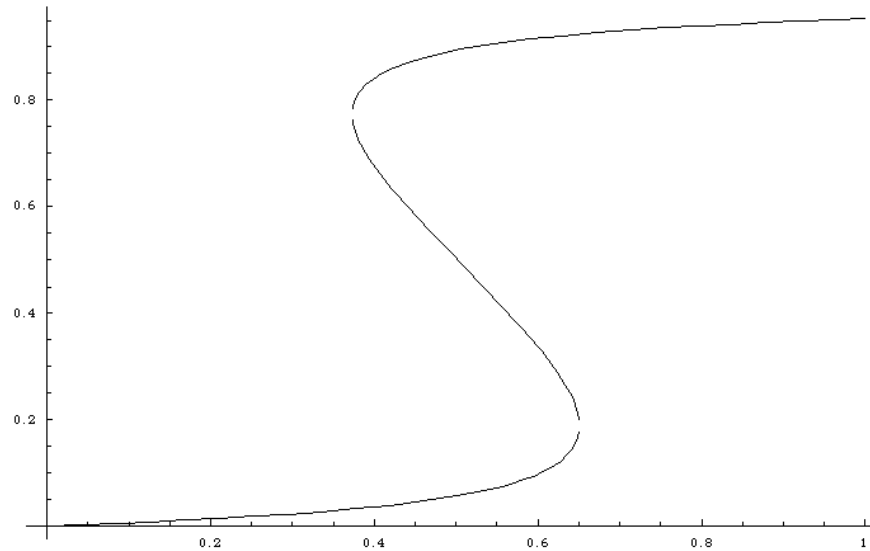
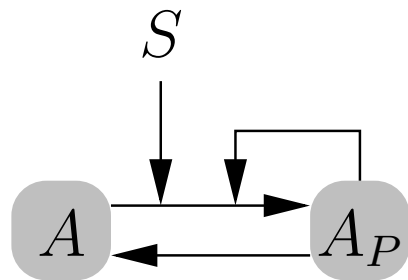
$$0.5 + \cos(x)/2$$



# Stochastic modelling

- Finite Markov models with discrete timesteps
  - Each possible combination of molecules = state (with phosphorylation cycles: finite)
  - With small enough time step: determine probability of going from each state to each other
  - Study long term behavior (eigenvectors) of these models
  - Transition matrices are HUGE: Parallel implementation

# Stochastic Stability of Hysteresis



- Replace the ODE model with finitely numbered species
- For specific signal, what is the eq. distribution?
- ...

# ... and ...

- Other topologies
  - Elementary recurrent motifs (bistable switches)
  - Recurrent networks (structured networks, like Hopfield etc?)
  - Random networks?
  - Liquid state machines?
- Evolution of networks with these "neural" nodes
- Integration with other levels, e.g., gene transcription