ohm Technische Hochschule Nürnberg

SPICE Net

Sensorimotor **P**rocessing, **I**ntelligence, and **C**ontrol with **E**mbedded Neural **Net**works

Implementation

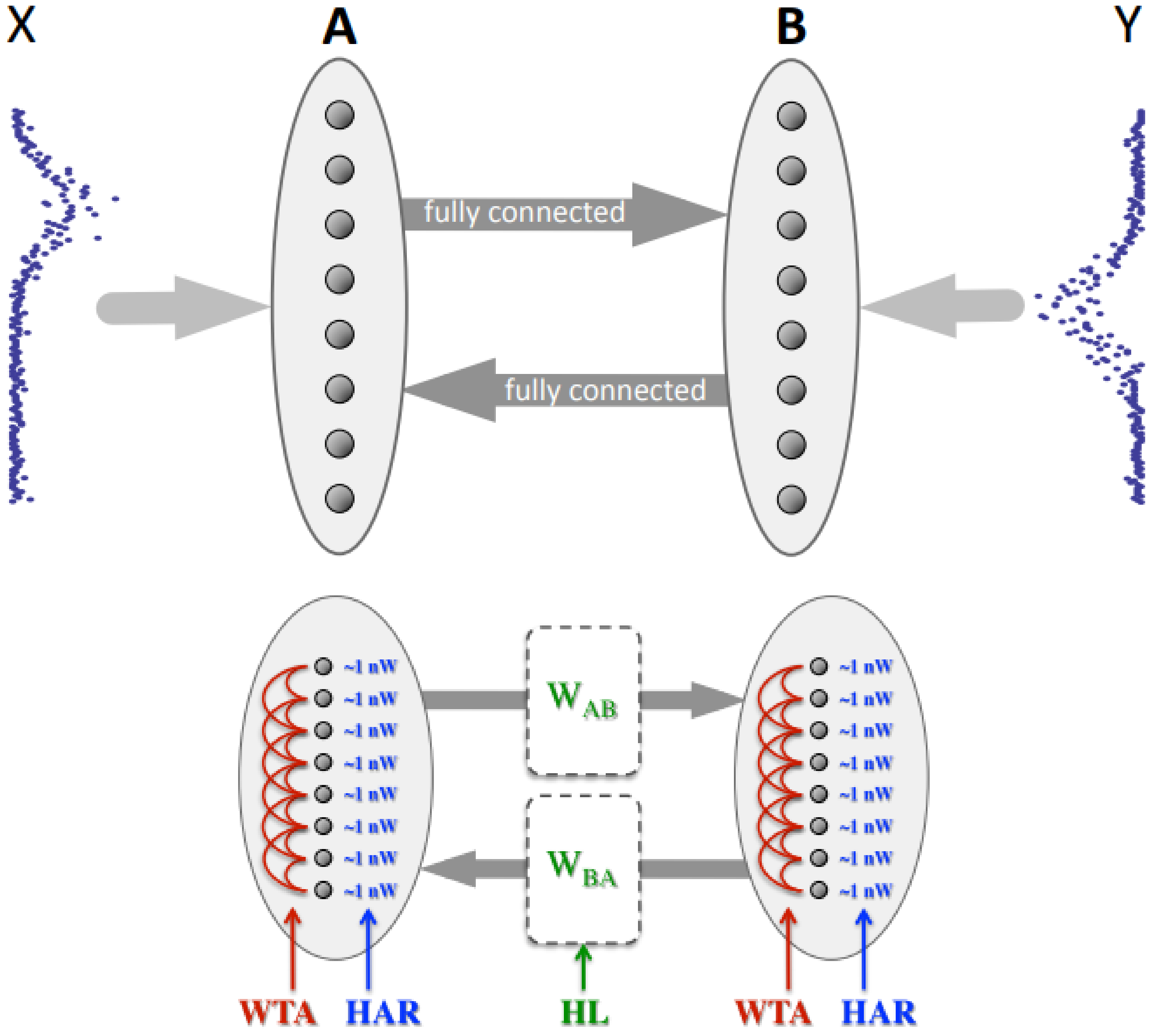
Model architecture described in a **simple example**:

a) Input data resembling **a nonlinear relation**

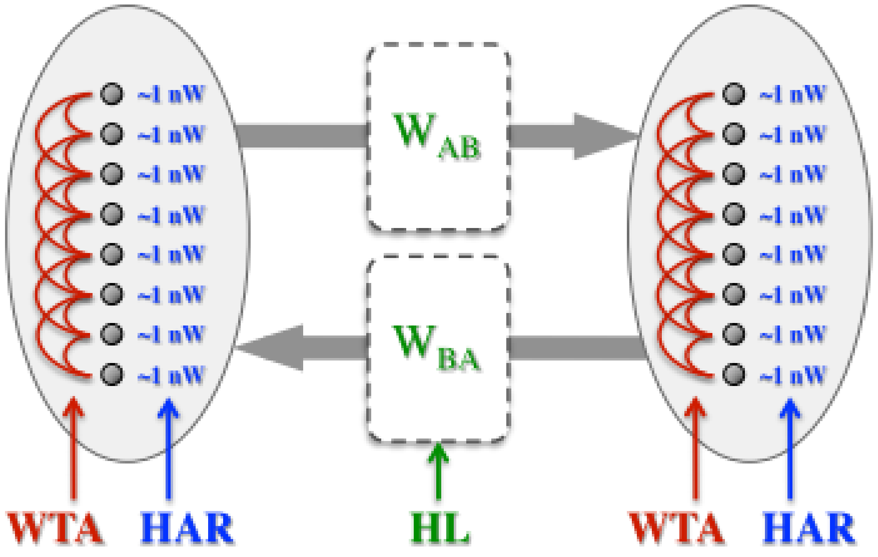
b) Basic **model architecture**;

c) Internal **model architecture**;

d) Computation stages.



Implementation



WTA

$$w_{i,j} = \gamma \cdot e^{-\frac{1}{2}(d(i,j)/\sigma)^2} - \delta$$

$$d(i,j) \; = \; \min\{|i-j|, n-|i-j|\}$$

HL

$$w_{i,j}^{t+1} = (1-\alpha_d) \cdot w_{i,j}^t + \alpha_l \cdot a_i^t \cdot a_j^t$$

$$w_{i,j} = \gamma \cdot e^{-\frac{1}{2}(d(i,j)/\sigma)^2} - \delta$$

$$d(i,j) \; = \; \min\{|i-j|, n-|i-j|\}$$

HAR

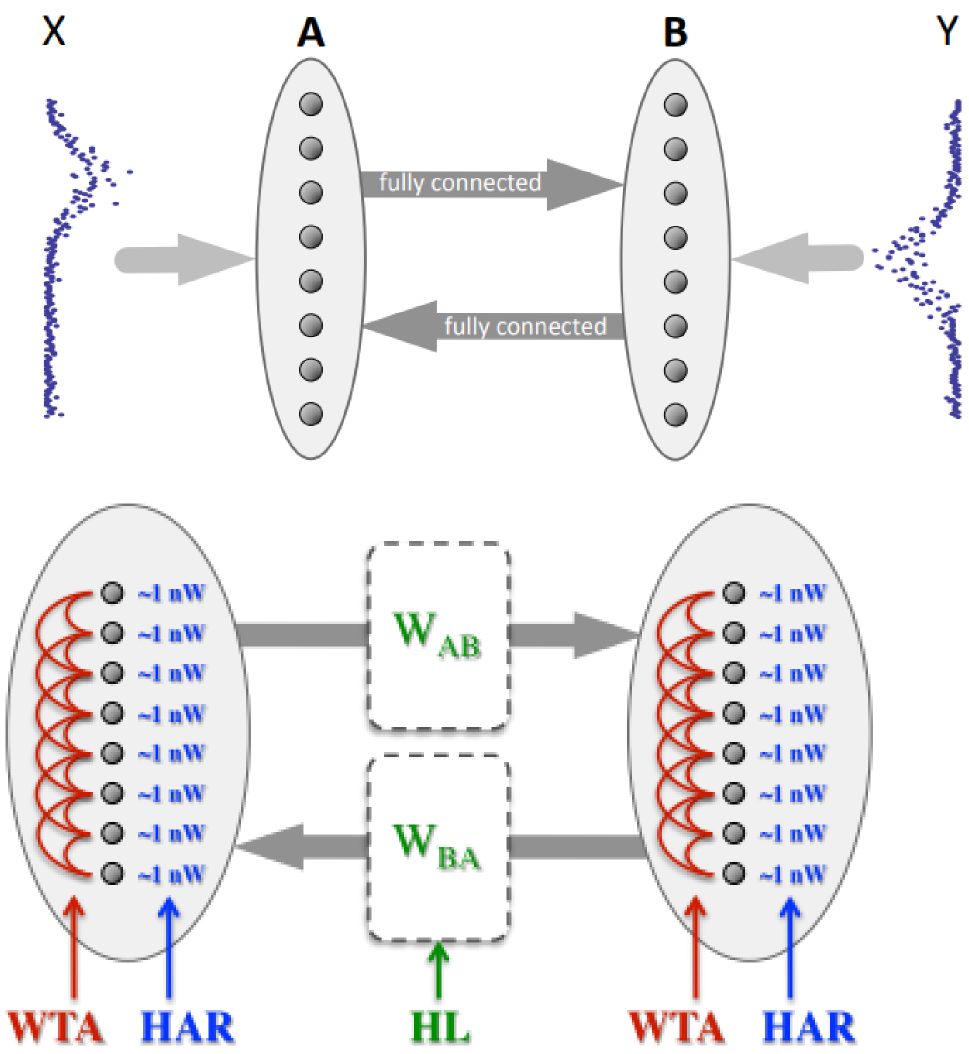
$$h_j^t = -c \cdot (\bar{a}_j^t - a_{\text{target}})$$

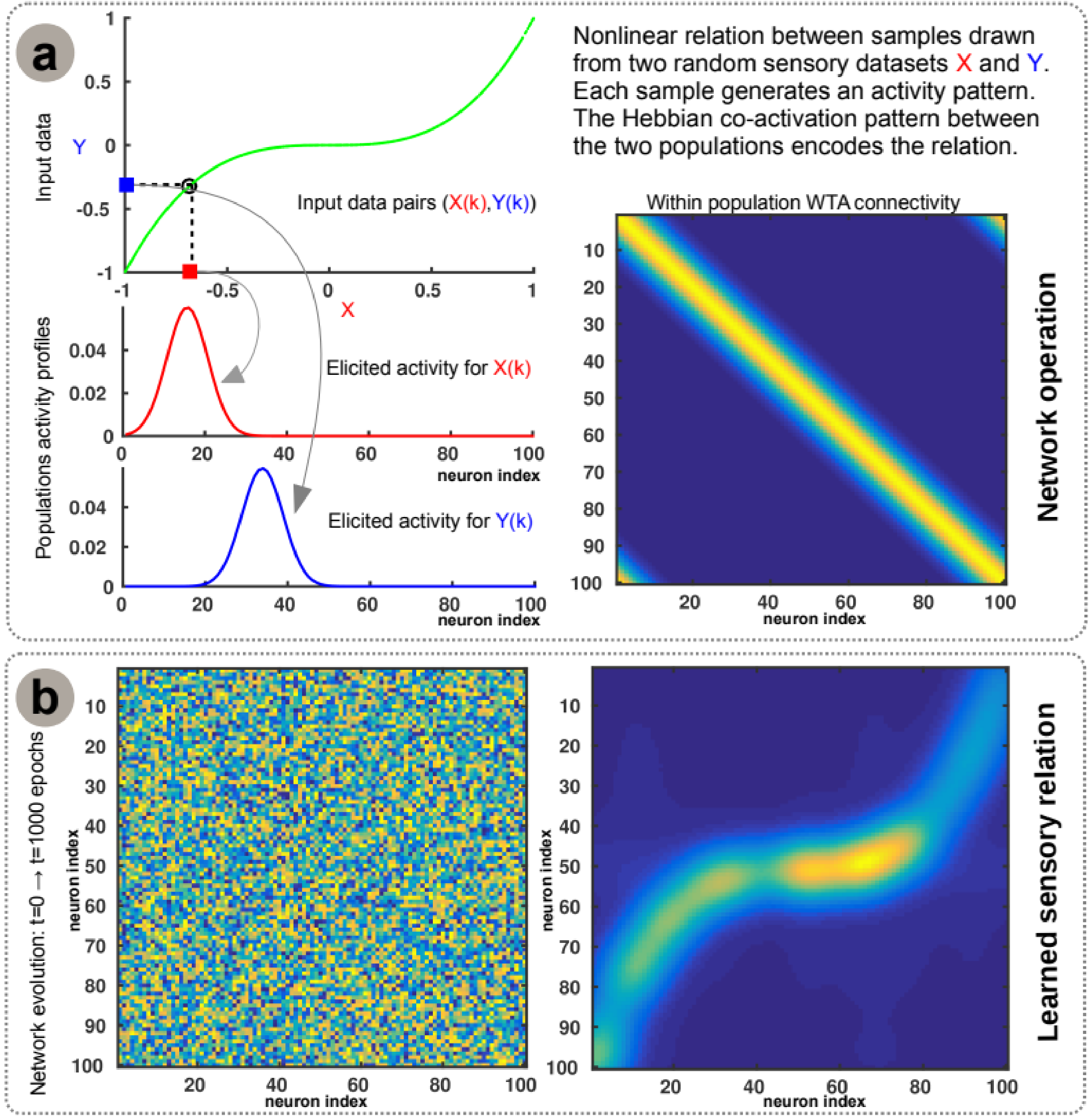
$$\bar{a}_j^t = (1-\omega)\bar{a}_j^{t-1} + \omega a_j^t$$

$$a_j^{t+1} = \theta(h_j^t + \sum_{i \in \Gamma_j^{\text{in}}} w_{i,j}^t \cdot a_i^t)$$

$$\theta(x) = \frac{1}{1+e^{-m(x-s)}}$$

Implementation





Implementation

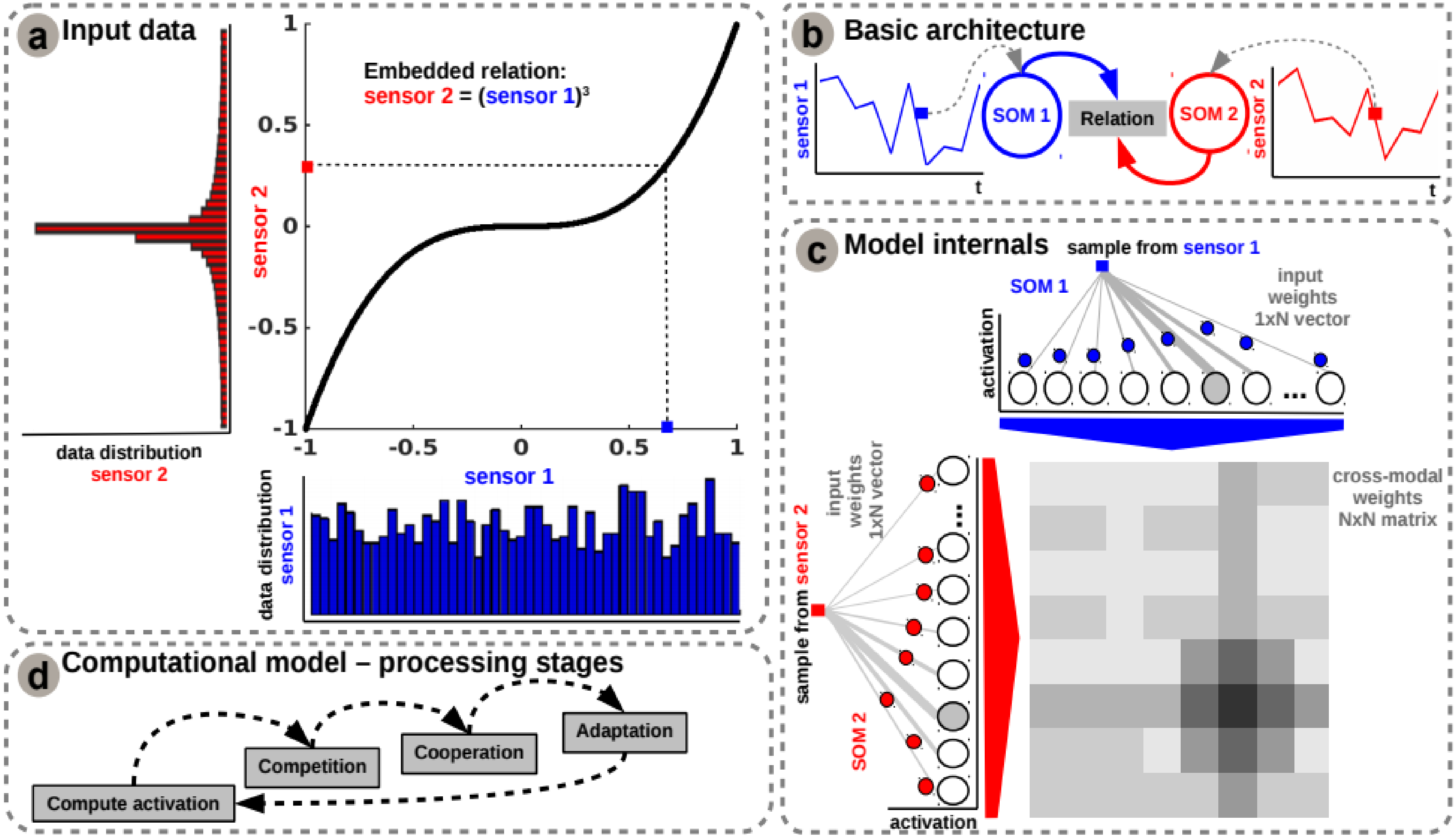
Model architecture described in a **simple example**:

a) Input data resembling **a nonlinear relation** (3rd order power-law) and **input data distributions**;

b) Basic **model architecture**;

c) Internal **model architecture**;

d) Computation stages.

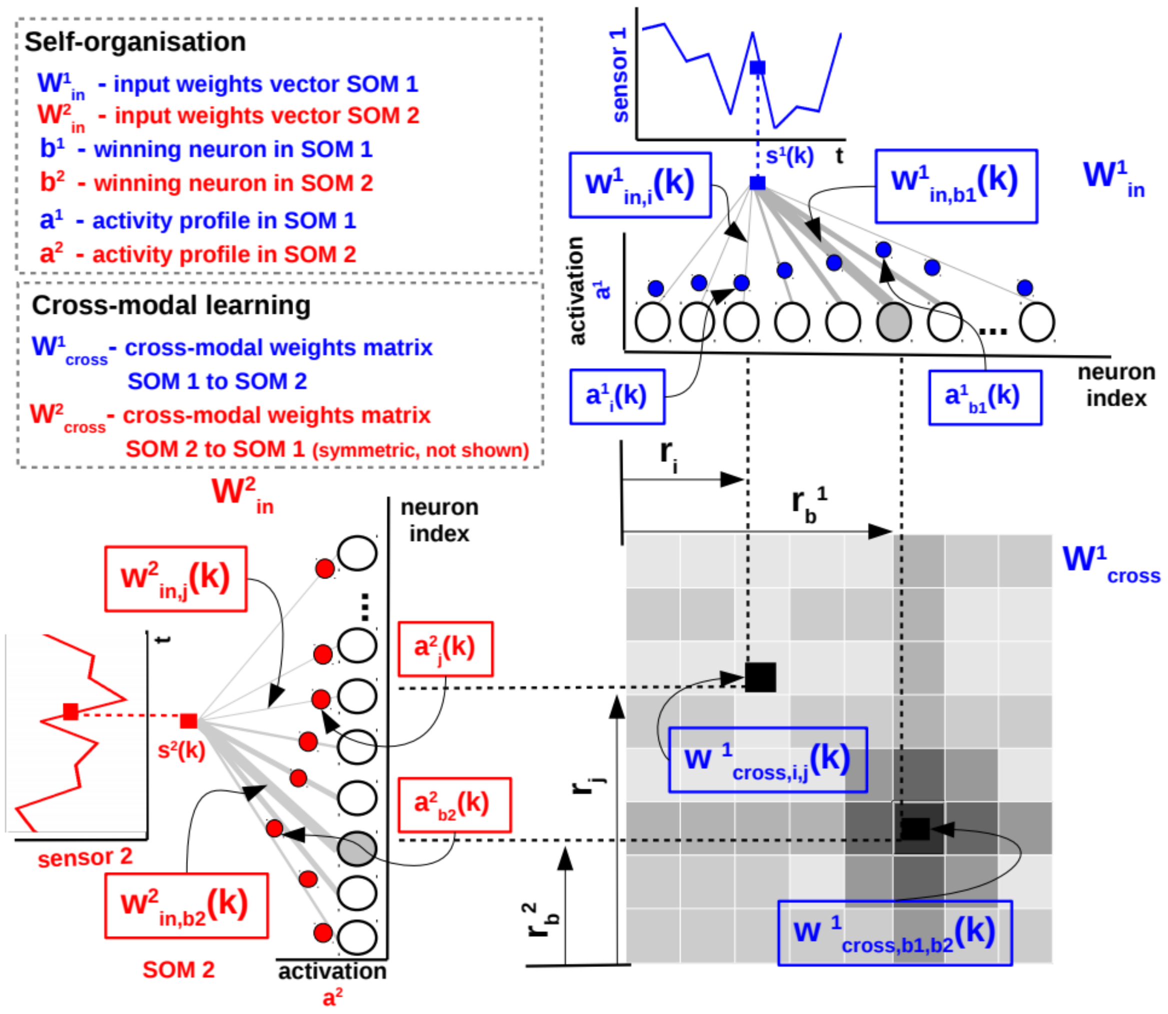


The underlying **neural networks:**

- **Self Organizing Maps (SOM)** to **encode the input data** into **efficient** sparse **representation**
- **Hebbian Learning** to **extract the temporal co-activation patterns** encoding the function

Implementation

Detailed **architecture of the model** and **processing** stages.



Input learning (SOM)

Input elicited activation

$$a_i^p(k) = \frac{1}{\sqrt{2\pi\xi_i^p(k)}} exp\left(\frac{-(s^p(k) - w^p_{in,i}(k))^2}{2\xi_i^p(k)^2}\right)$$

Winner neuron – competition function

$$b^p(k) = argmax\ a^p(k)$$

Neighbourhood function - cooperation function

$$h^p_{b,i}(k) = exp\left(\frac{-||r_i - r_b||^2}{2\sigma(k)^2}\right)$$

Weight update - learning

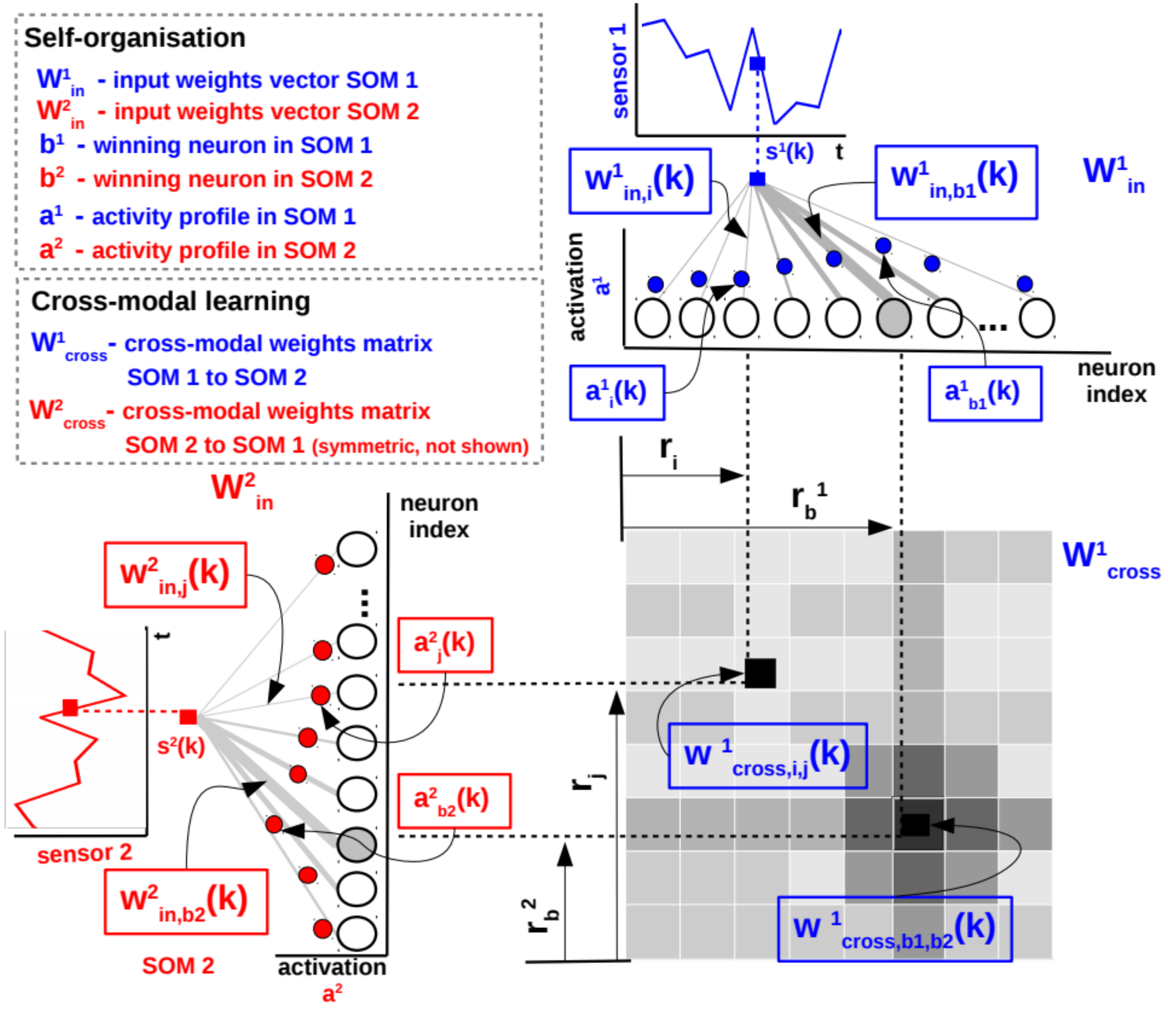
$$\Delta w^p_{in,i}(k) = \alpha(k)\, h^p_{b,i}(k)\, (s^p(k) - w^p_{in,i}(k))$$

Tuning curve update

$$\Delta\xi_i^p(k) = \alpha(k)\, h^p_{b,i}(k)\, ((s^p(k) - w^p_{in,i}(k))^2 - \xi_i^p(k)^2)$$

Implementation

Detailed **architecture of the model** and **processing** stages.



Cross modal learning (Hebbian Learning)

$$a_i^p(k) = \frac{1}{\sqrt{2\pi}\xi_i^p(k)} exp\left(\frac{-\left(s^p(k) - w^p_{in,i}(k)\right)^2}{2\xi_i^p(k)^2}\right)$$

Weight update - learning

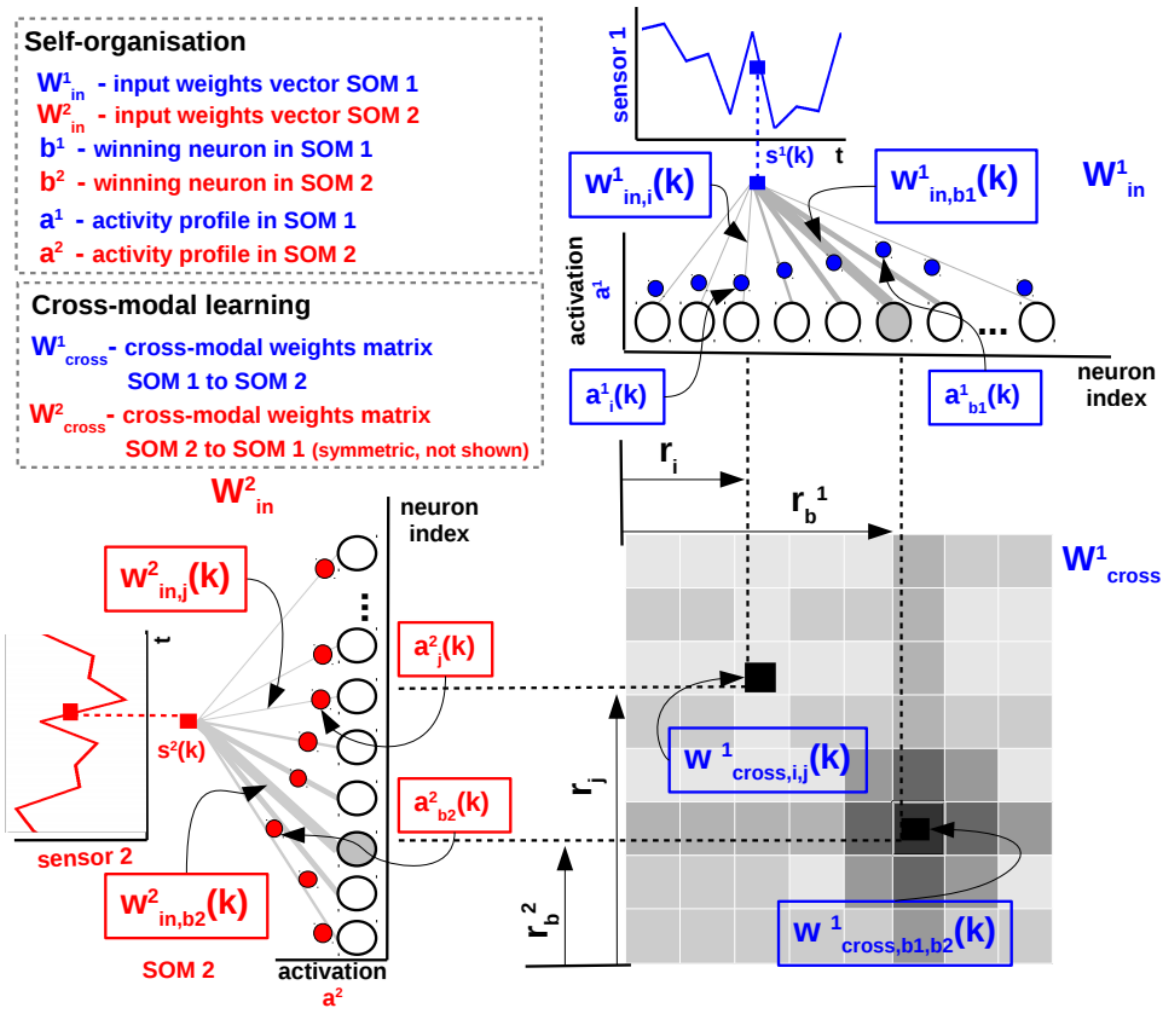
$$\Delta w^p_{cross,i,j}(k) = \eta(k)\left(a_i^p(k) - \bar{a}_i^p(k)\right)\left(a_j^q(k) - \bar{a}_j^q(k)\right)$$

Parameters update

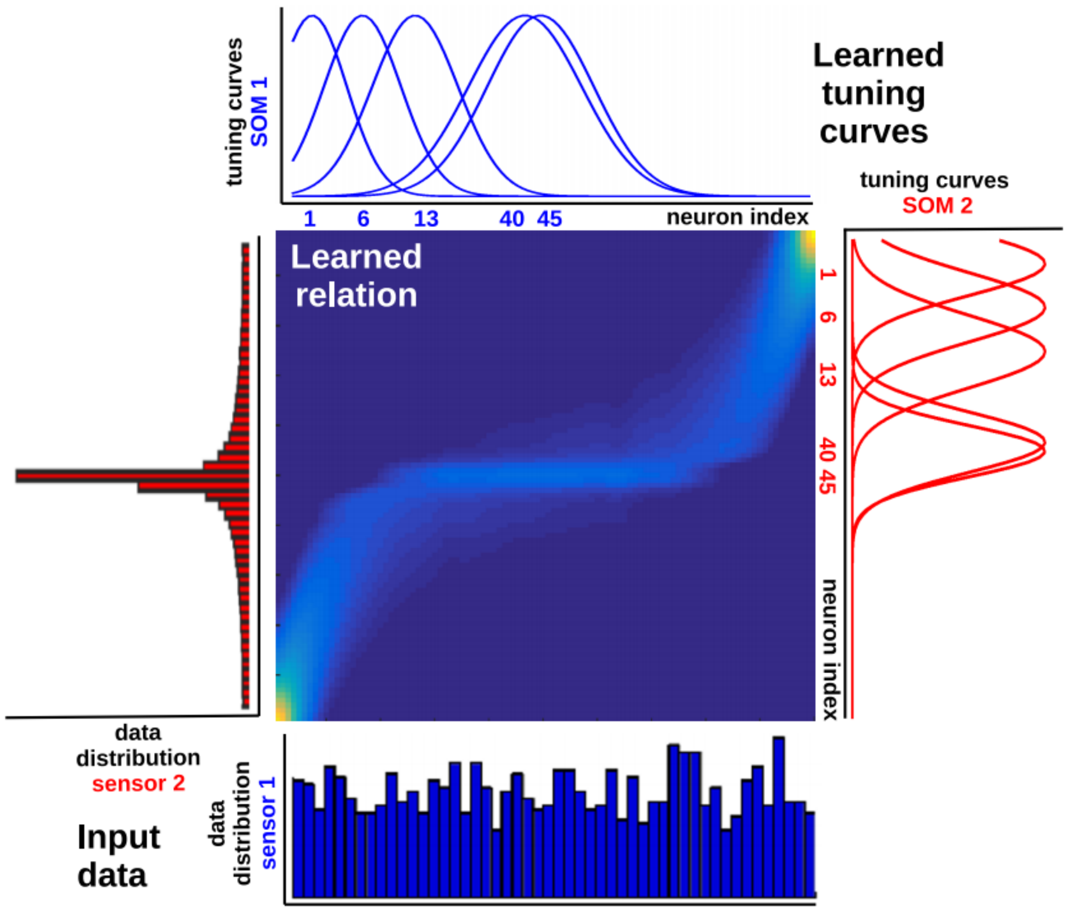
$$\beta(k) = 0.002 + \frac{0.998}{k+2}, \eta(k) = \frac{A}{k+B}, B = \frac{v_f t_f - v_0 t_0}{v_f - v_0}, A = v_0 t_0 + B v_0,$$

Implementation

Detailed **architecture of the model** and **processing** stages.



Learnt sensory relation for the **example** (3^{rd} power-law) and **learnt data statistics** using the proposed model.



Implementation

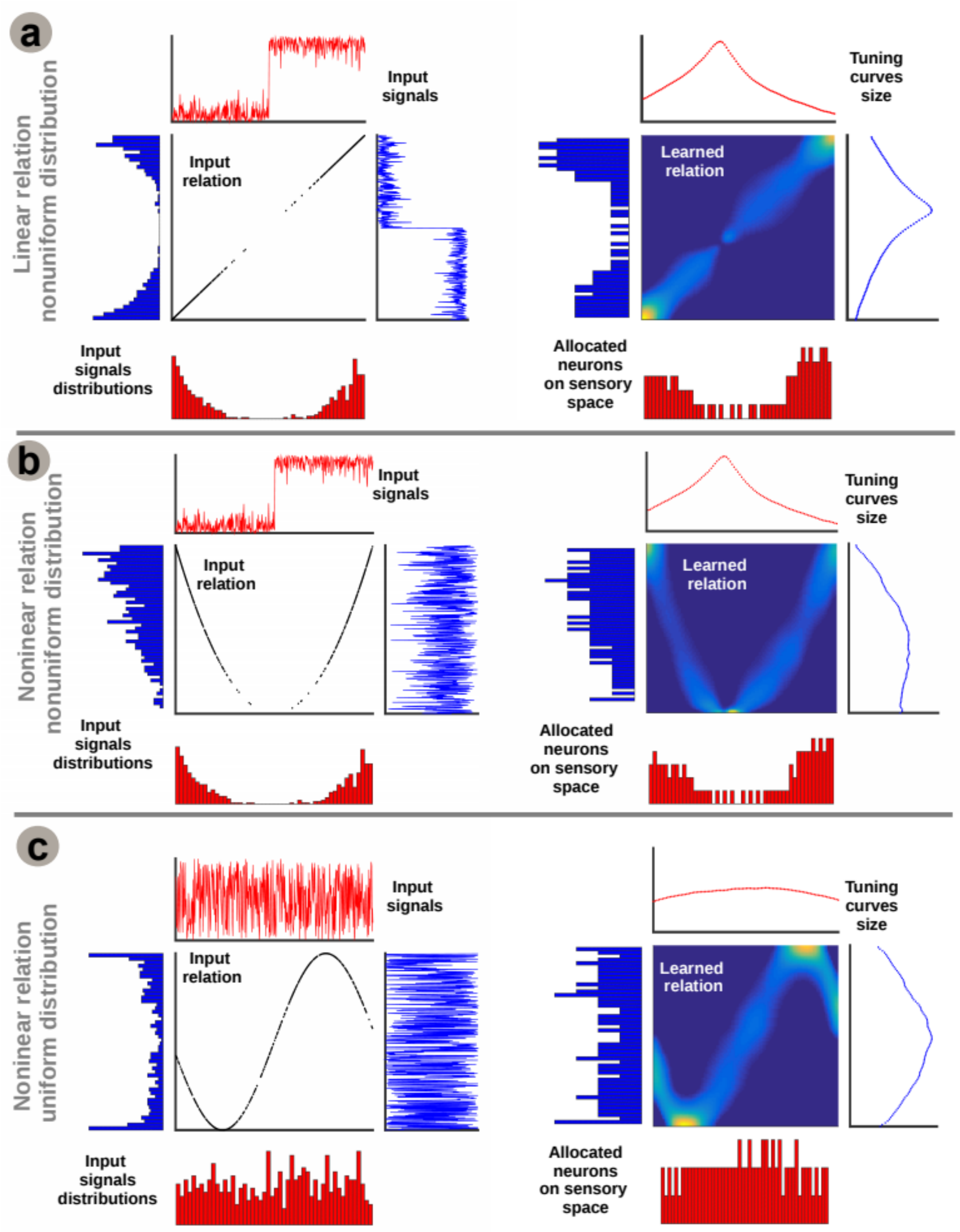
Analysis of the **basic model** in a **bimodal** scenario **(2 sensors)**.

Different **(hidden) sensory relations** and **data distributions**: **input data** and its probability distribution (left); **learned relation** and **allocated resources** (i.e. neurons) according to input distributions (right).

a) **Linear sensory relation** with **non-uniform** data **distribution**;

b) **Nonlinear sensory relation** with **non-uniform** data **distribution**;

c) **Nonlinear sensory relation** with **uniform** data **distribution**.



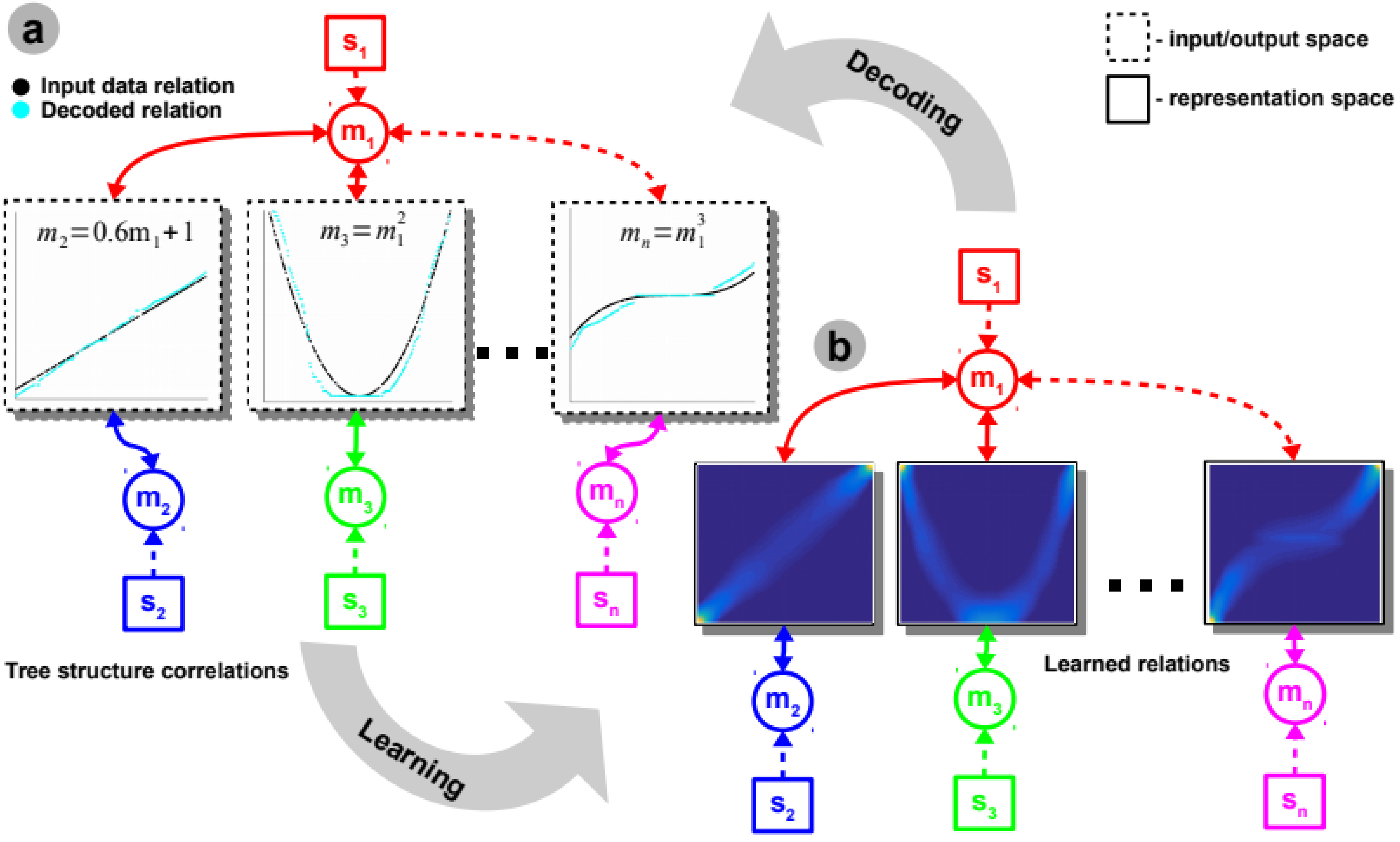
Implementation

Analysis of the **extensibility** capabilities of the **neural network**.

Sample scenario with a **4-dimensional network (sensors: s1, s2, s3, s4)** with a tree shaped **correlation structure**.

a) **Input data** and **decoded** learned **representation encoded in maps (m1, m2, m3, m4)**;

b) **Learned sensory relations** encoded in the **neural network weights**.



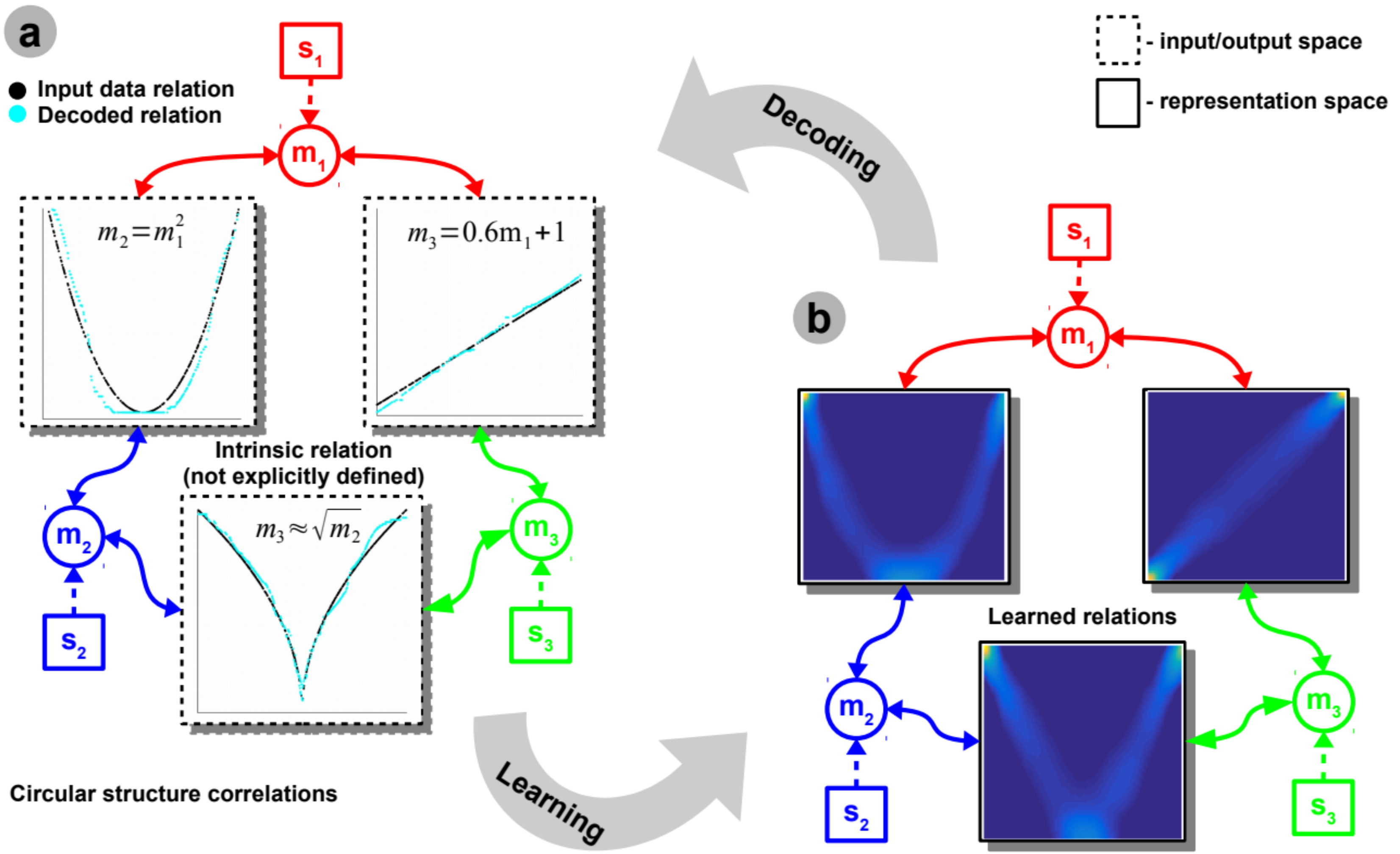
Implementation

Analysis of the **extensibility** capabilities of the **neural network**.

Sample scenario with a **4-dimensional network (sensors: s1, s2, s3, s4)** with a circular shaped **correlation structure**.

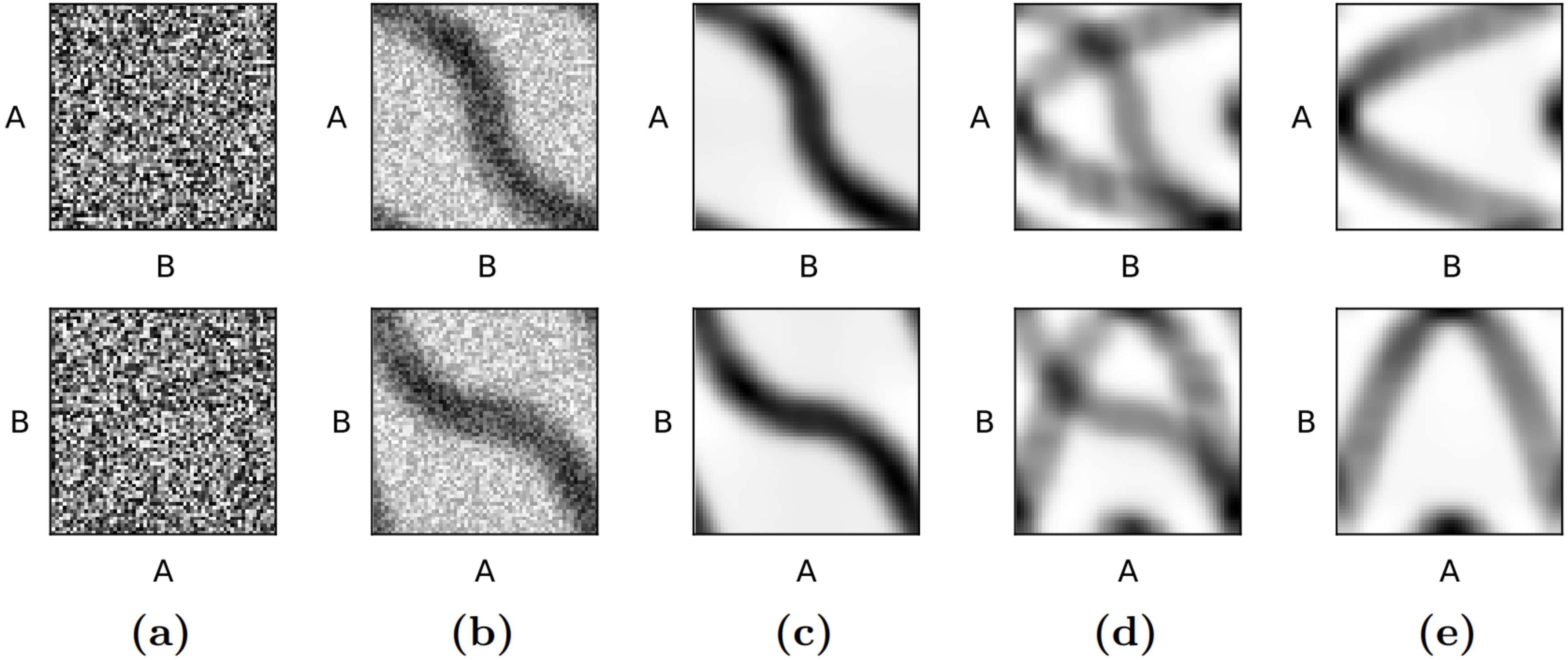
a) **Input data** and **decoded** learned **representation encoded in maps (m1, m2, m3, m4)**;

b) **Learned sensory relations** encoded in the **neural network weights**.



Capabilities

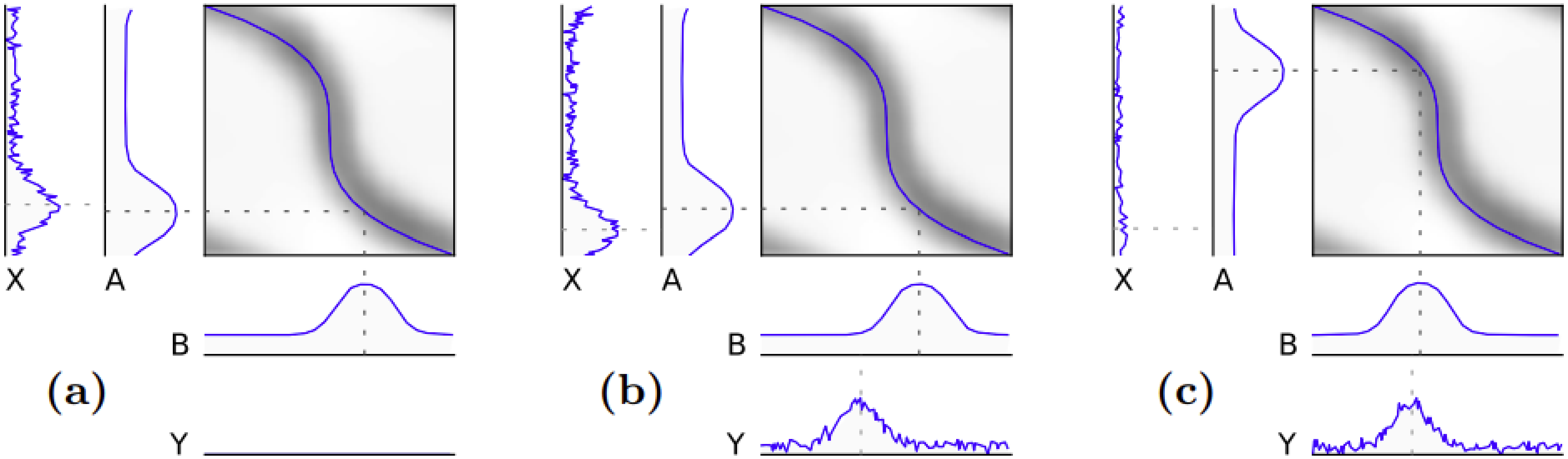
Learning and relearning



The time course of **learning and relearning** in the sample network. Each plotted subfigure shows a snapshot of the connection weights W_{AB} (top row) and W_{BA} (bottom row) for different times during learning and relearning. The weights are color coded (black for strong, white for weak connections). (a) Initial random weights. (b) Weights captured during learning. (c) Weights after the relation $y = x^3$ was learned. (d) Weights captured during relearning. (e) Weights after the relation $y = x^2$ was learned.

Capabilities

Inference and de-noising

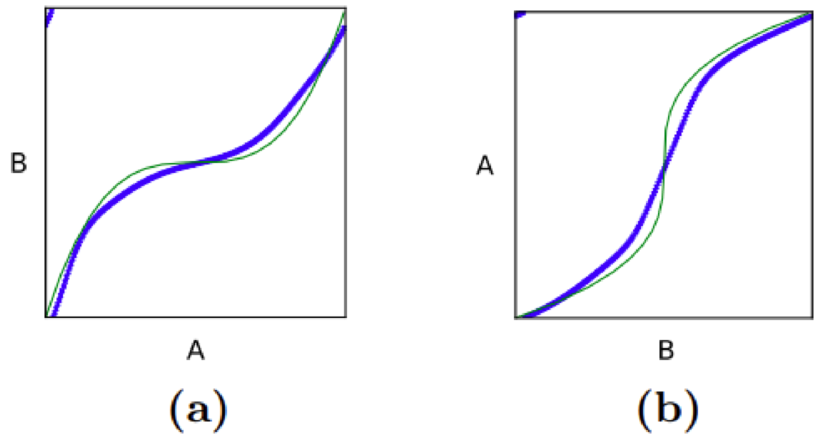


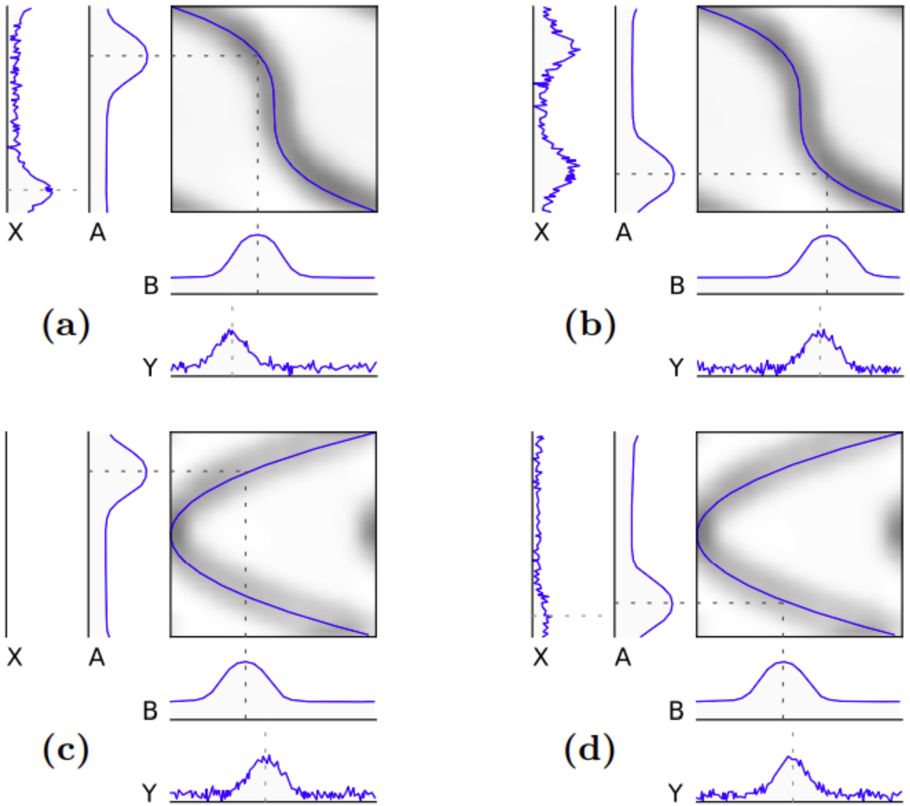
Inference and de-noising: (a) An example of inference from X to B which shows also the de-noising properties of the network with respect to noise in the firing rates of the units, (b) when two inputs are presented which are inconsistent with the learned relation the network shifts both peaks until their positions are in accordance with the relation, (c) same as (b) but with unequal reliability of the inputs (unequal input strength); note that the larger (more reliable) peak is much less shifted than in (b).

Capabilities

Inference and decision making

Simple inference in sample network after the network has learned the relation y = x^3 (green thin line in both plots). (a) shows the results of the inference tasks (thick blue line) for a set of population codes fed to A (horizontal axis). (b) like (a) but for the opposite inference direction.





Decision task. (a) when the peaks of the inputs are not close to the learned relationship the network uses one of the inputs and infers the other one, (b) when in X there are two contradicting inputs present, while one is being supported by the input in Y , the network decides for that combination of peaks., (c) in the case of a non-invertible function like y = x^2 there exist two possible peak positions and the system decides for one of the two, (d) same as (c) but the network's decision is biased by a very small input fed to X.