Unsupervised learning of features from data: a statistical physics approach

R. Monasson
Laboratory of Physics
CNRS & Ecole Normale Supérieure, Paris

http://www.phys.ens.fr/~monasson/Enseignement/MGDS/index.html

Model-guided data science, Como, September 2019

Plan of the lectures

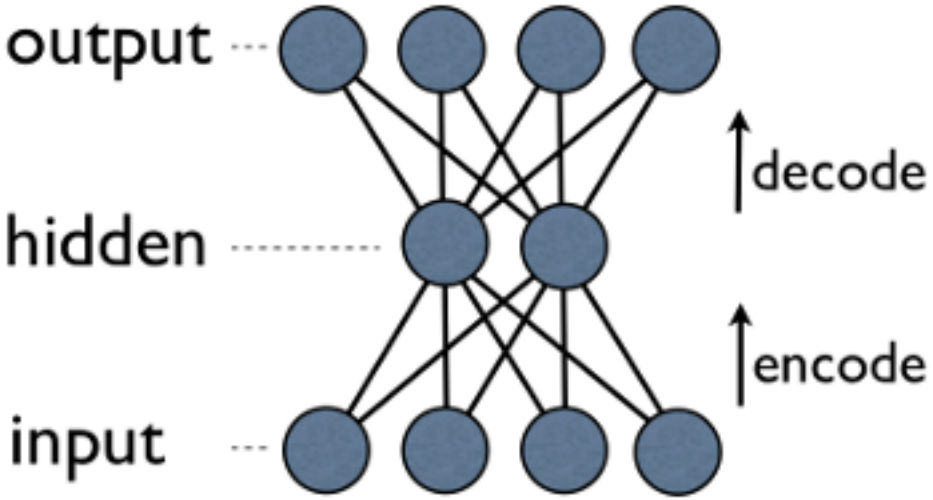
1. Bayesian Inference and dimensional reduction:
   phase transition in principal component analysis

2. **Representations: auto-encoders, Restricted Boltzmann Machines & sparse feature learning**

3. Restricted Boltzmann Machines: connections with graphical models, phase transitions & applications

Autoencoders

Goal: dimensional reduction (similar to PCA)
extract low-dimensional representation of data



V' = G(M'.h) **(dim. p)**

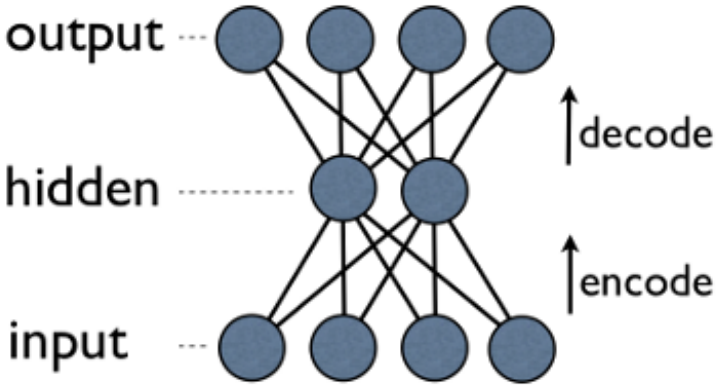
h = F(M.V) **(dim. p'<<p)**

V **(dim. p)**

Cost function: $$D = \sum_s \left| V^{(s)} - G(M' \cdot F(M \cdot V^{(s)})) \right|^2$$

Learning algorithm: gradient descent of D …

Autoencoders: linear case



V' = M'.h (dim p)

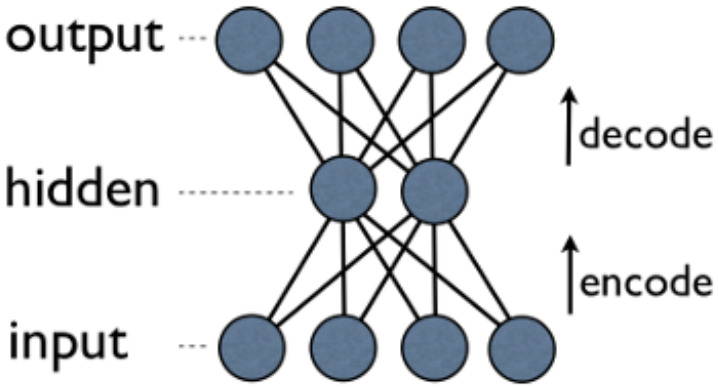
h = M.V (dim p'<<p)

V (dim p)

- Assume V of dimension p with covariance matrix C
- Output V' space spanned by p'-dimensional subspace
- What happens for p'=1?
- Extension to p'>1 case

Autoencoders: linear case

Interesting application: denoising



V' = M'.h (dim p)

h = M.V (dim p'<<p)

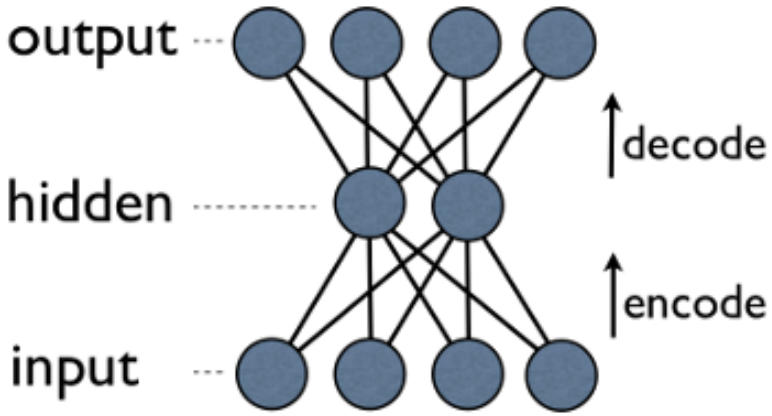
V (dim p)

- Want V' = V – noise, assume noise = multivar. with $C^{noise} = \eta^2$ Id
- Again p' top components of C
- But: $M_\mu = w_\mu,\ M'_\mu = \dfrac{\lambda_\mu}{\lambda_\mu + \eta^2} w_\mu$

Sparse Autoencoders

Problem: trade-off Efficiency (p'<<p) vs. Accuracy (p'=p)

Idea: Allow p' to be large (large pools of representations) but enforce sparsity in hidden layer !!



V' = G(M'.h) **(dim. p)**

h = F(M.V) **(dim. p' ≈ p)**

V **(dim. p)**

Cost function:

$$D = \sum_s \left[\left| V^{(s)} - G(M' \cdot h^{(s)}) \right|^2 + P(h^{(s)}) \right]$$

penalty

with $h^{(s)} = F(M \cdot V^{(s)})$

Images

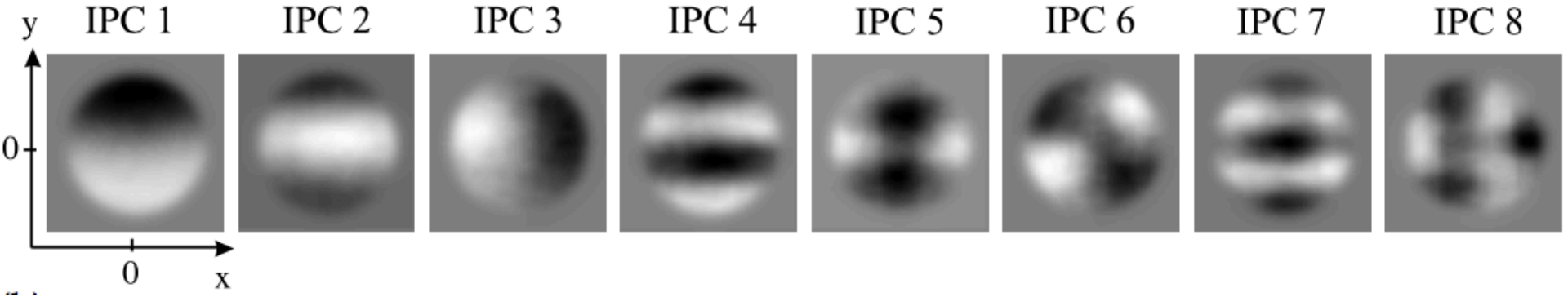
12,000 images
= 6,000 natural
+ 6,000 man-made views

256x256 pixels

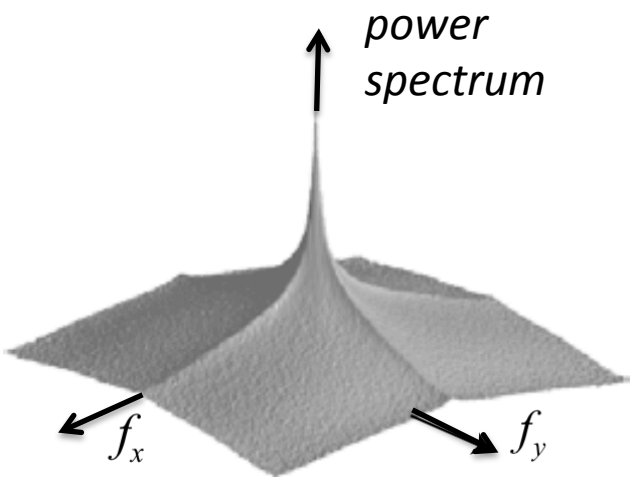
Torralba, Oliva, 2003

Fourier analysis of images

- Compute correlation matrix C(r,r'), where r is 2D vector
- Diagonalize and find top components



- Similar to 2D Fourier modes (plane waves)
- Due to statistical properties of images:
  translation invariance
  approximate rotation invariance

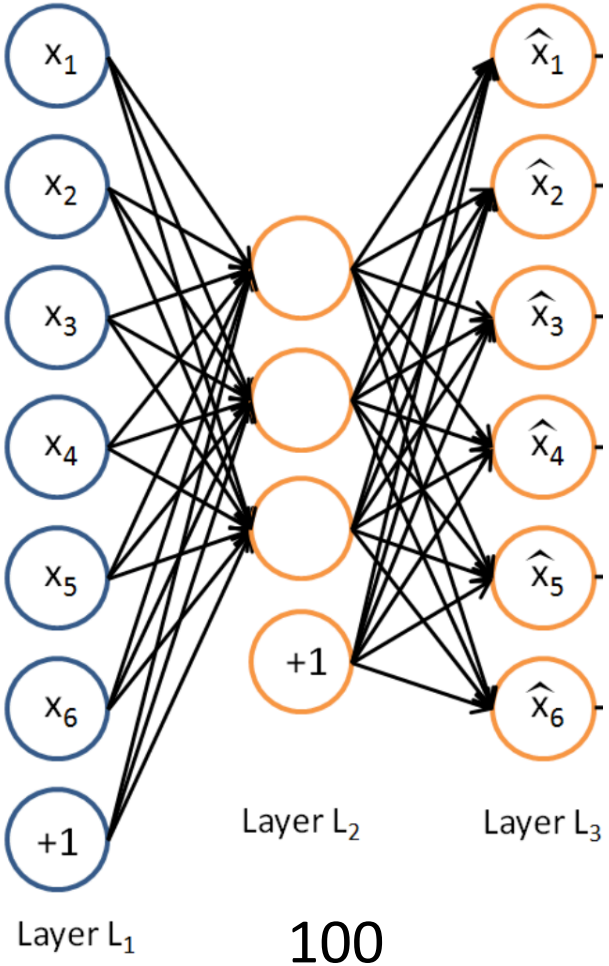


Sparse auto-encoder

100 (+1)
input units

Images =
10x10 pixels

Ng, 2001



100
hidden units

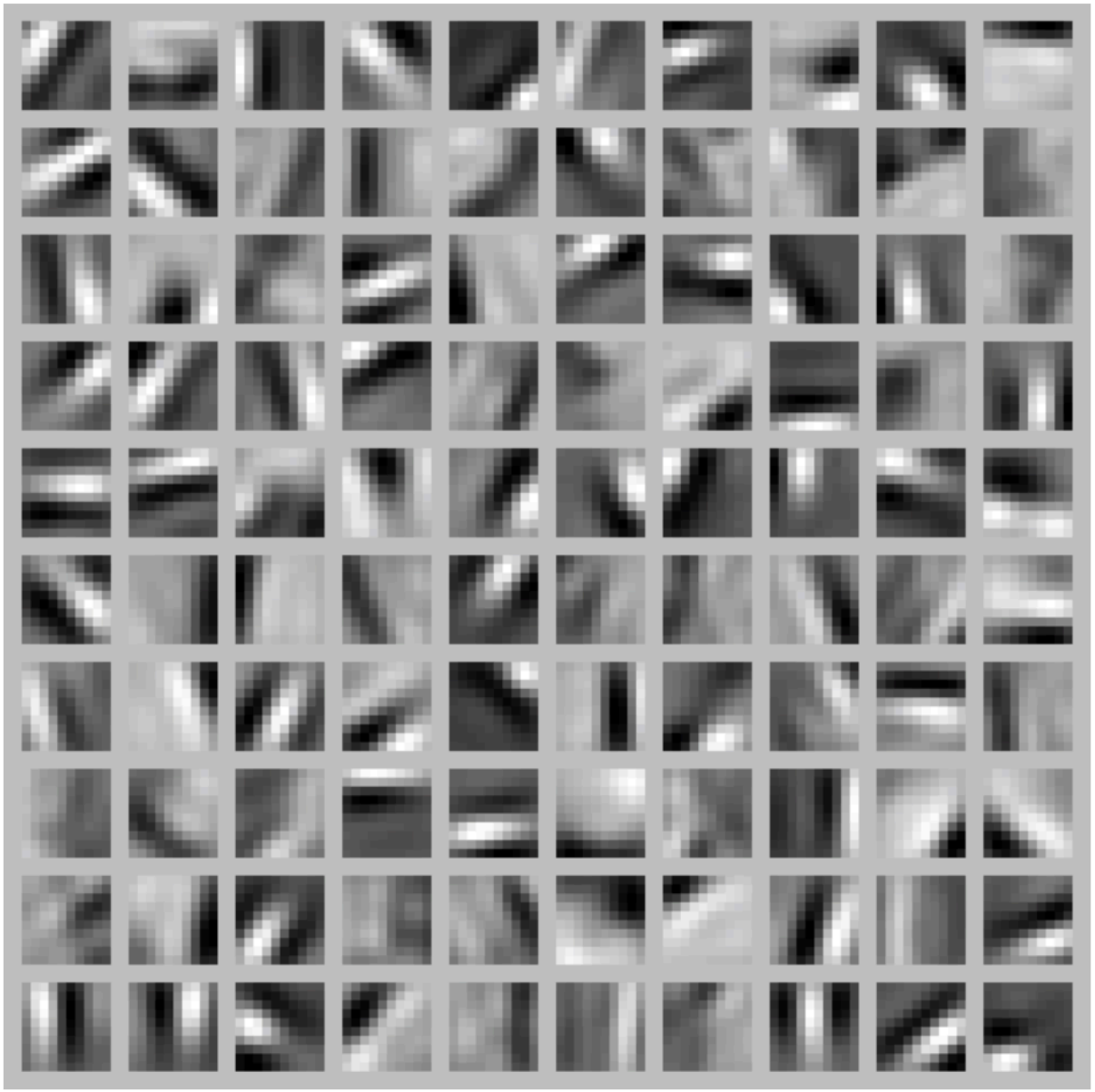
100
output units

Transfer function:

$$\sigma_\mu = \frac{1}{1+\exp\left(-\sum_i w_{\mu i} x_i\right)}$$

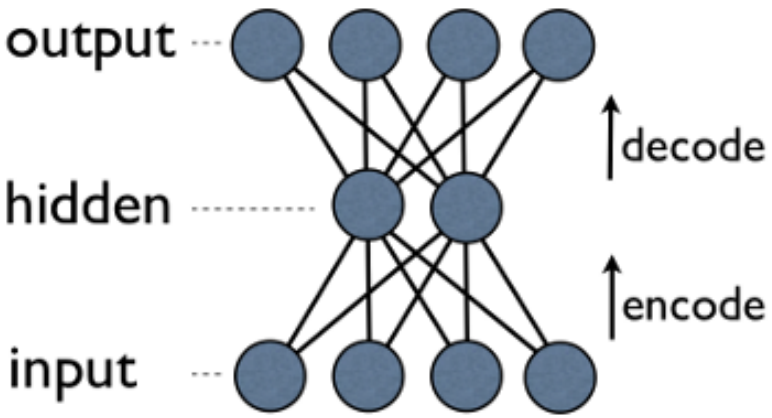
Sparse autoencoder

$$\frac{w_{\mu i}}{\sqrt{\sum_j w_{\mu j}^2}}$$



(trained on 100 natural images)

Sparse Dictionary Learning



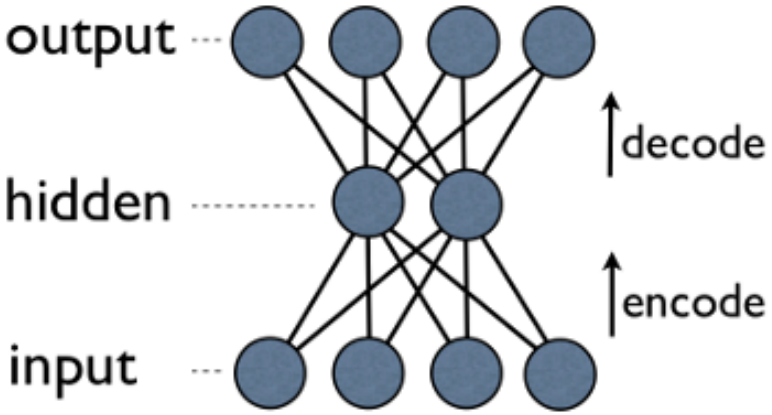
$$D = \sum_s \left[\left| V^{(s)} - M' \cdot h^{(s)} \right|^2 + P(h^{(s)}) \right]$$

with $h^{(s)} = F(M \cdot V^{(s)})$

Minimize over ***M, M'***

M,M' define sets of features and **h** tells us which ones are useful to build **V** (a small number due to sparsity!)

Sparse Dictionary Learning



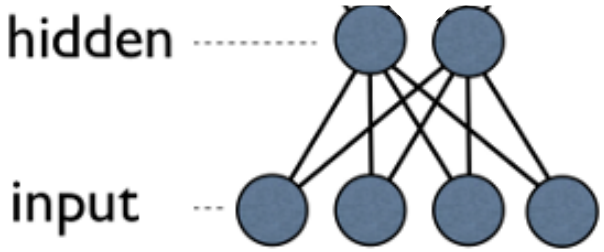
$$D = \sum_s \left[\left| V^{(s)} - M' \cdot h^{(s)} \right|^2 + P(h^{(s)}) \right]$$

with $h^{(s)} = F(M \cdot V^{(s)})$

Minimize over ***M, M'***

Sparse dictionary learning: go for features directly!

Sparse Dictionary Learning



Sparse dictionary learning: go for features directly!

Minimize $$D = \sum_{s} \left[\left| V^{(s)} - M' \cdot h^{(s)} \right|^2 + P(h^{(s)}) \right]$$ over ***M'*** and ***h***

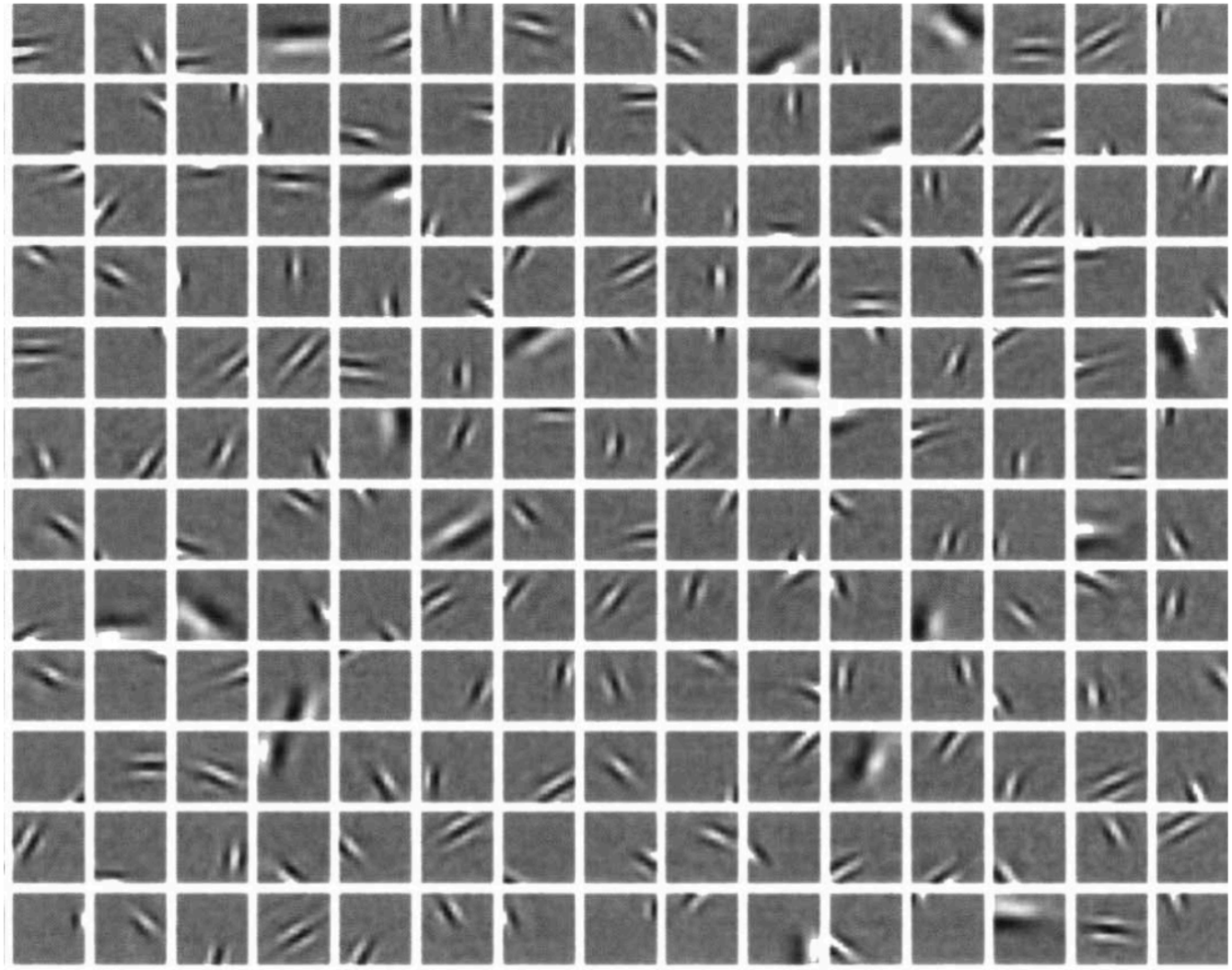
Sparse Dictionary Learning

180 basis functions

12x12 pixel images

10,000 natural images

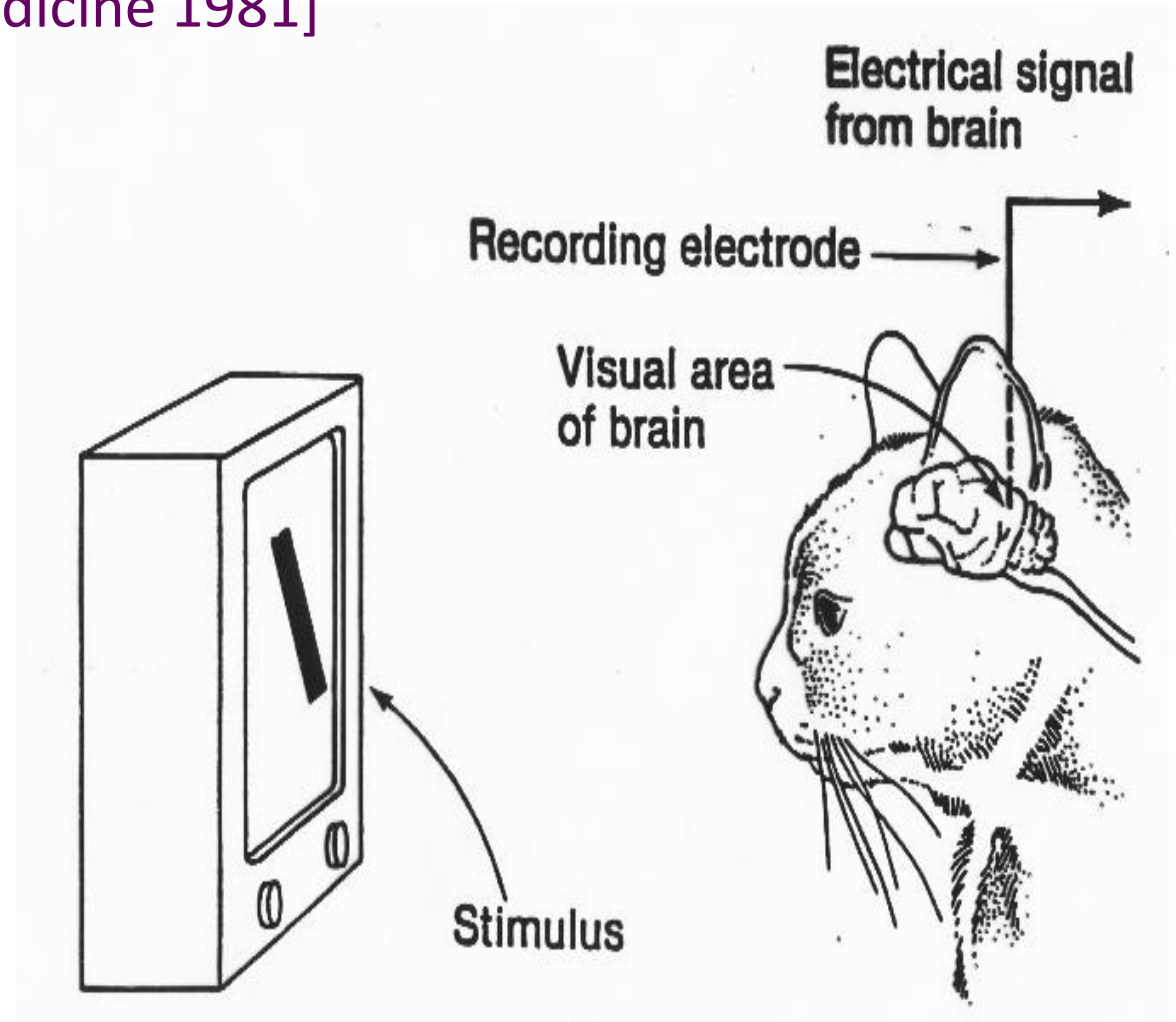
$P(x)=\log(1+x^2)$

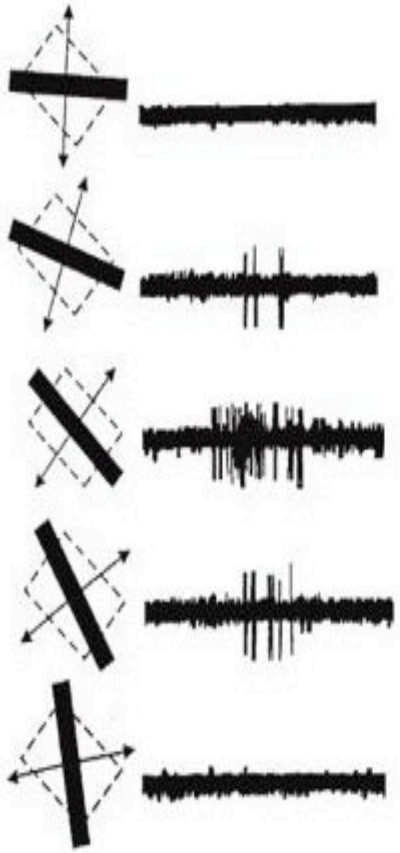


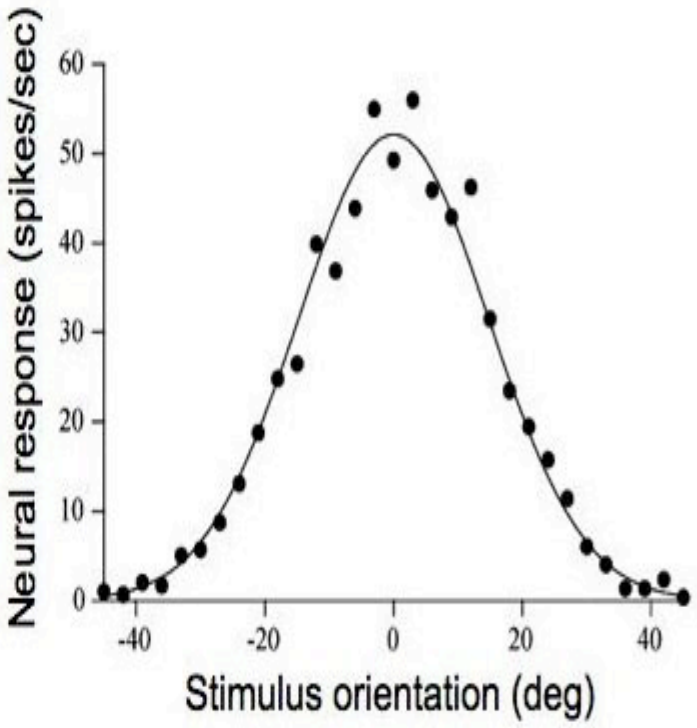
Olshausen, Field 1996

Connection with receptive fields in neuroscience

Hubel-Wiesel experiments (>=1959)
[Nobel Prize in Medicine 1981]



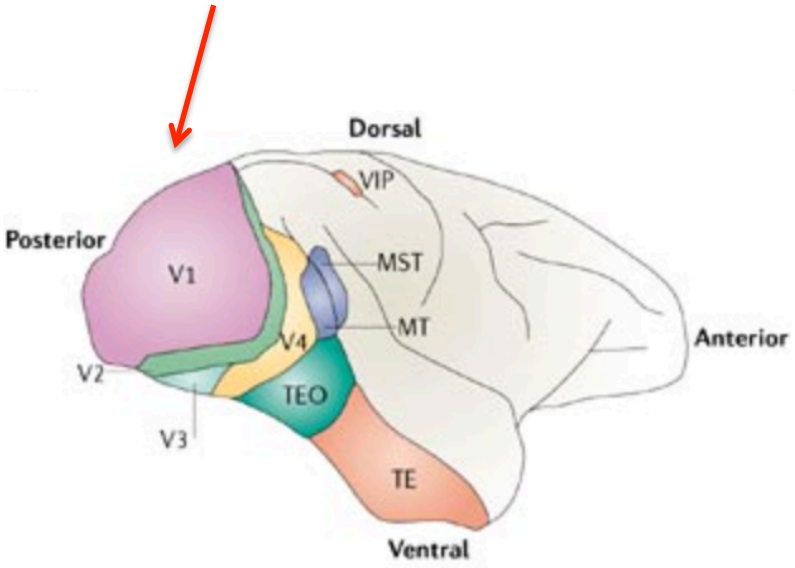


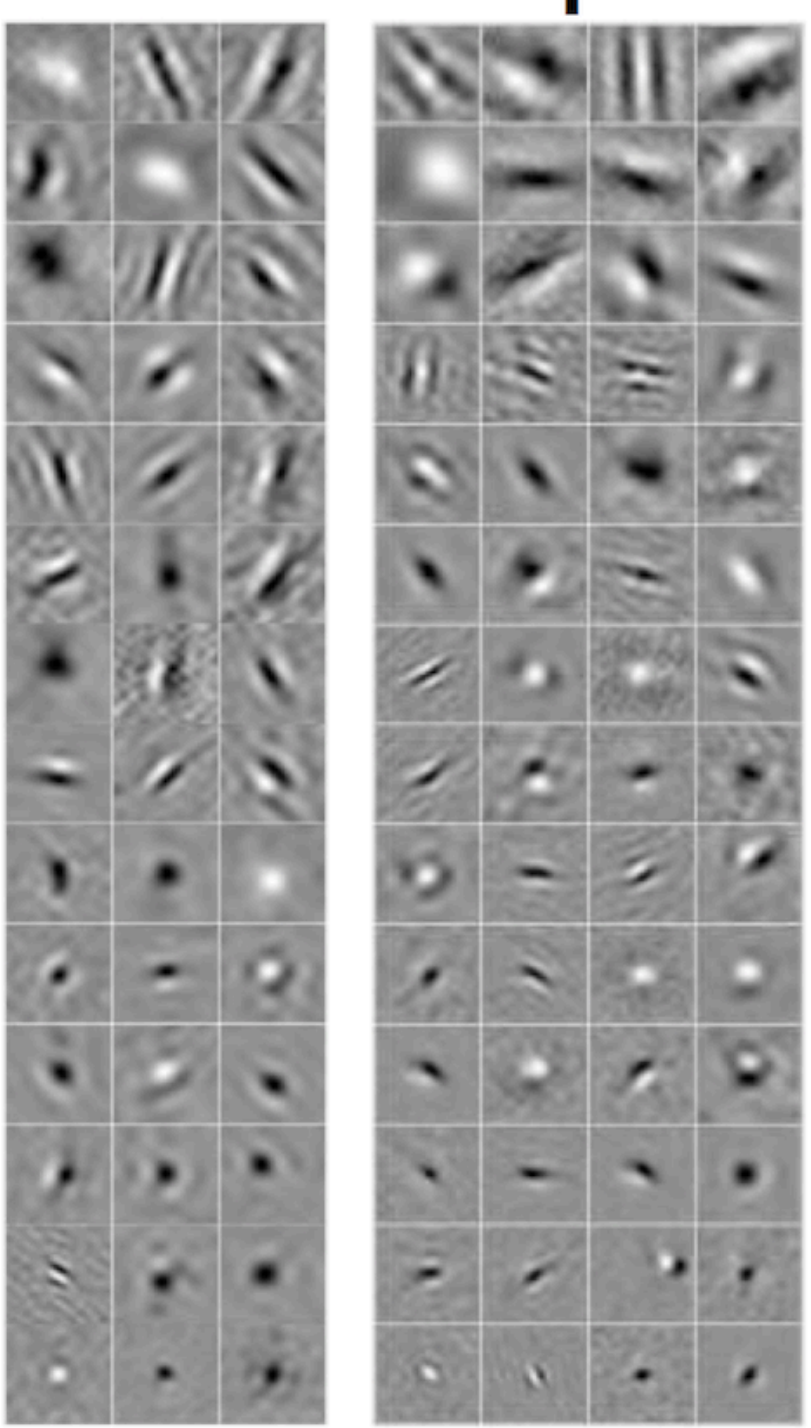


Hubel & Wiesel, 1968

Simple	orientation, position
Complex	orientation, motion, direction
Hypercomplex	orientation, motion, direction, length

Receptive Fields in Macaque V1





Ringach, 2002
Zylberberg, Murphy, DeWeese 2011

Neural implementation of PCA

J. Math. Biology (1982) 15: 267–273

Journal of Mathematical Biology

A Simplified Neuron Model as a Principal Component Analyzer

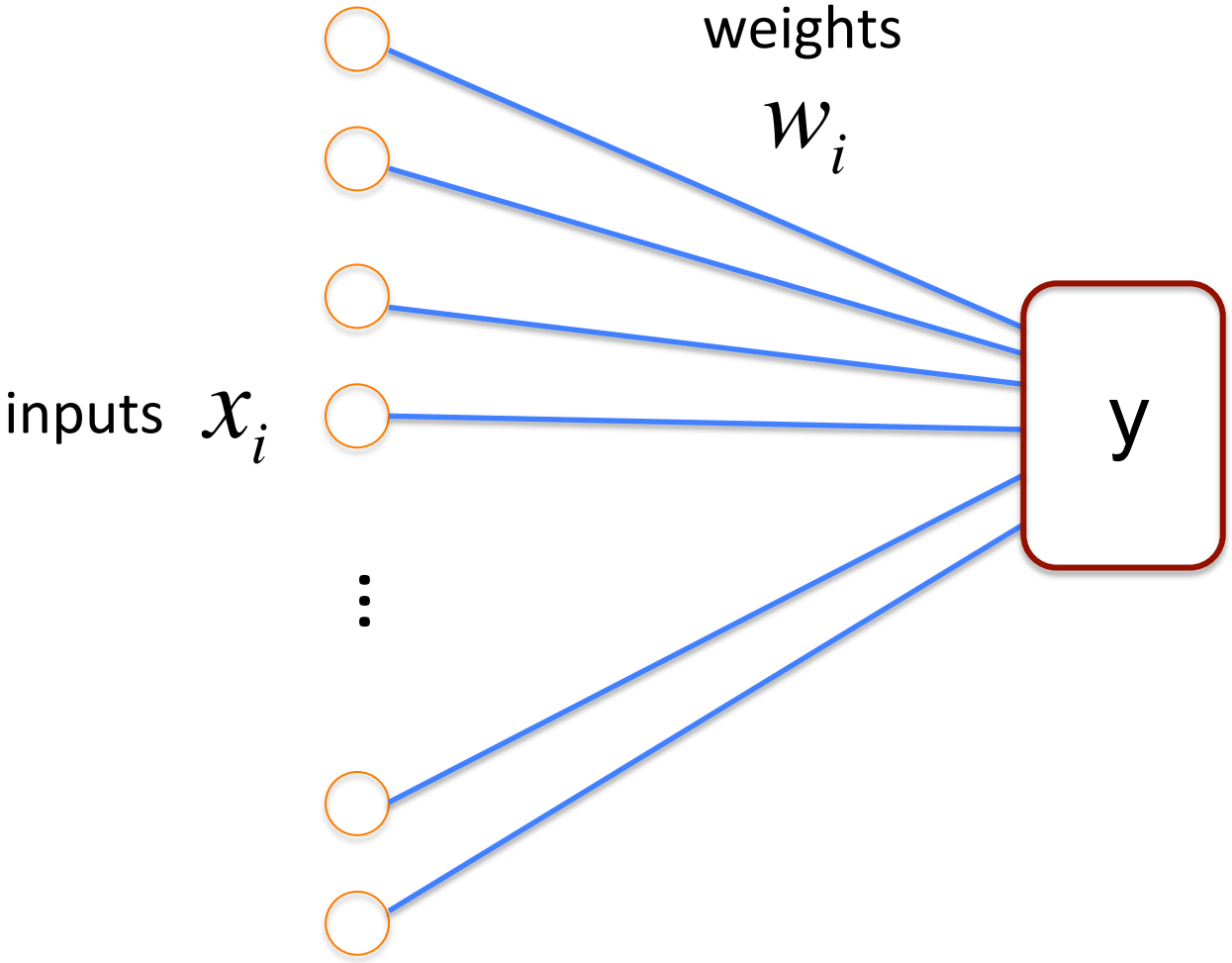
Erkki Oja

University of Kuopio, Institute of Mathematics, 70100 Kuopio 10, Finland

Abstract. A simple linear neuron model with constrained Hebbian-type synaptic modification is analyzed and a new class of unconstrained learning rules is derived. It is shown that the model neuron tends to extract the principal component from a stationary input vector sequence.

Key words: Neuron models – Synaptic plasticity – Stochastic approximation

Neural implementation of PCA



output

$$y = \sum_i w_i x_i$$

$$= \vec{w} . \vec{x}$$

Neural implementation of PCA

Dynamics over the weights?

Hebbian learning …

Donald Hebb (*The organization of behavior*, 1949):

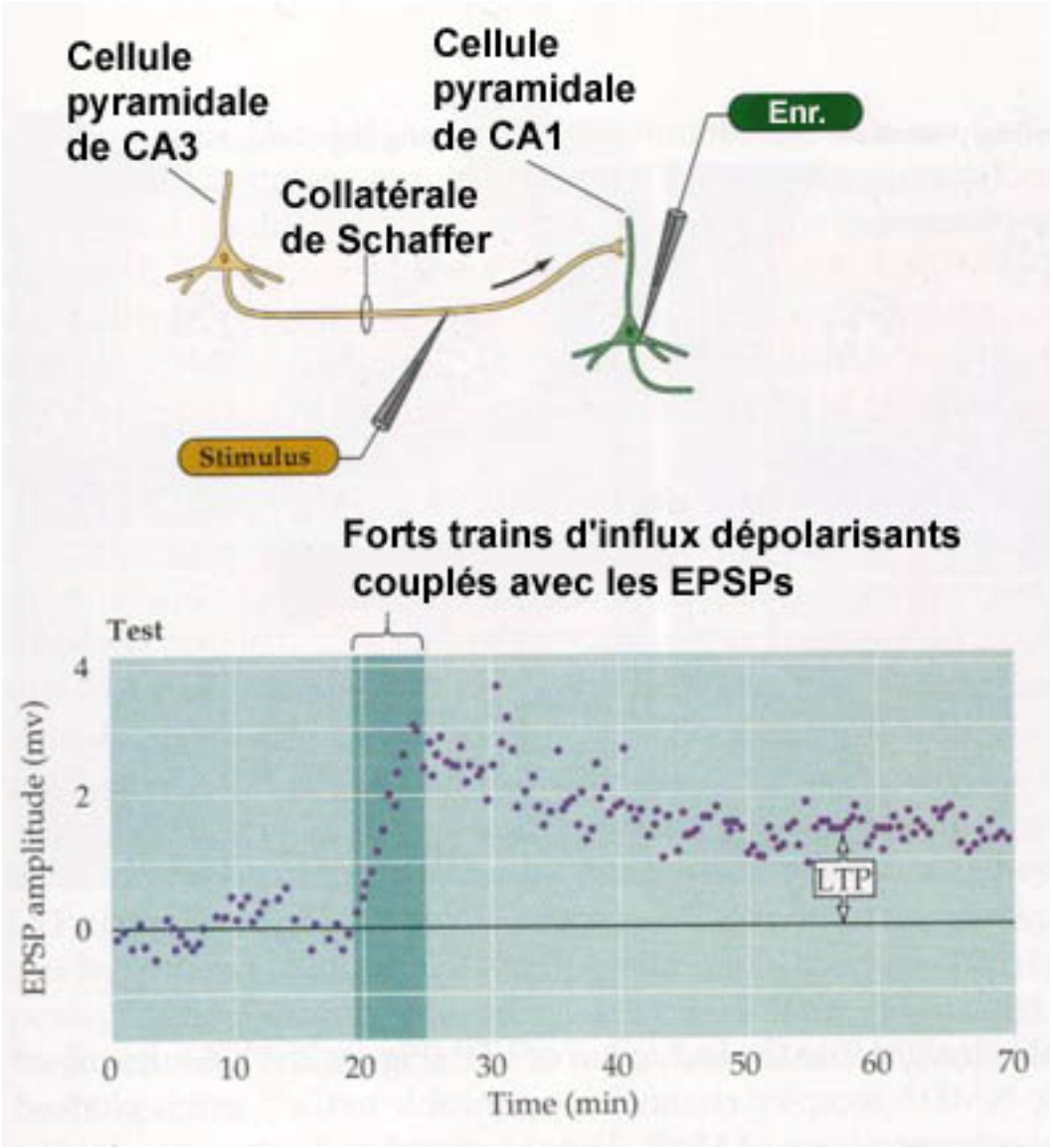
> "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."

i.e. Cells that fire together wire together

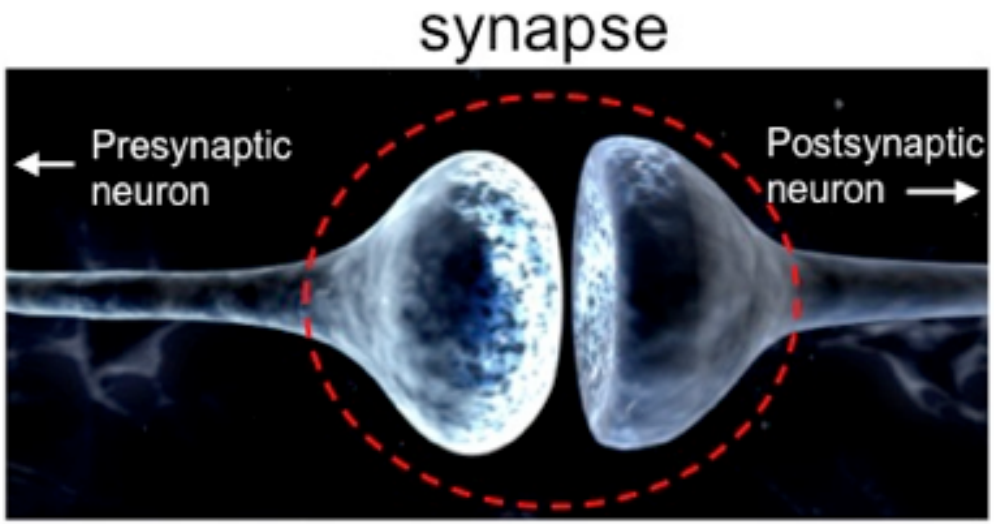
Experimental evidence?

Long term Potentiation (or depletion) (= minutes -> months)

T. Lomo, Recordings in rat hippocampus (1966)

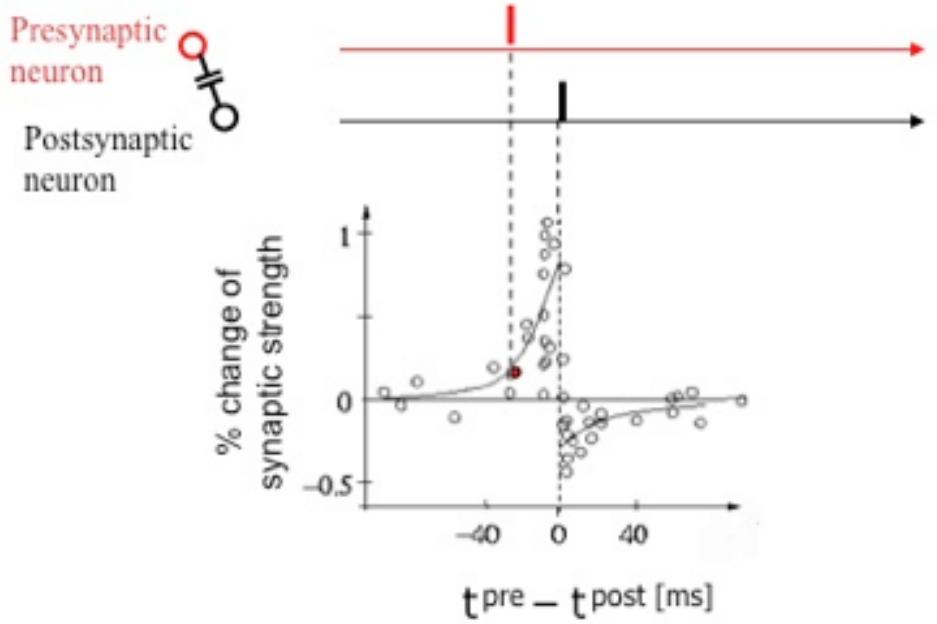


Spike timing dependent plasticity



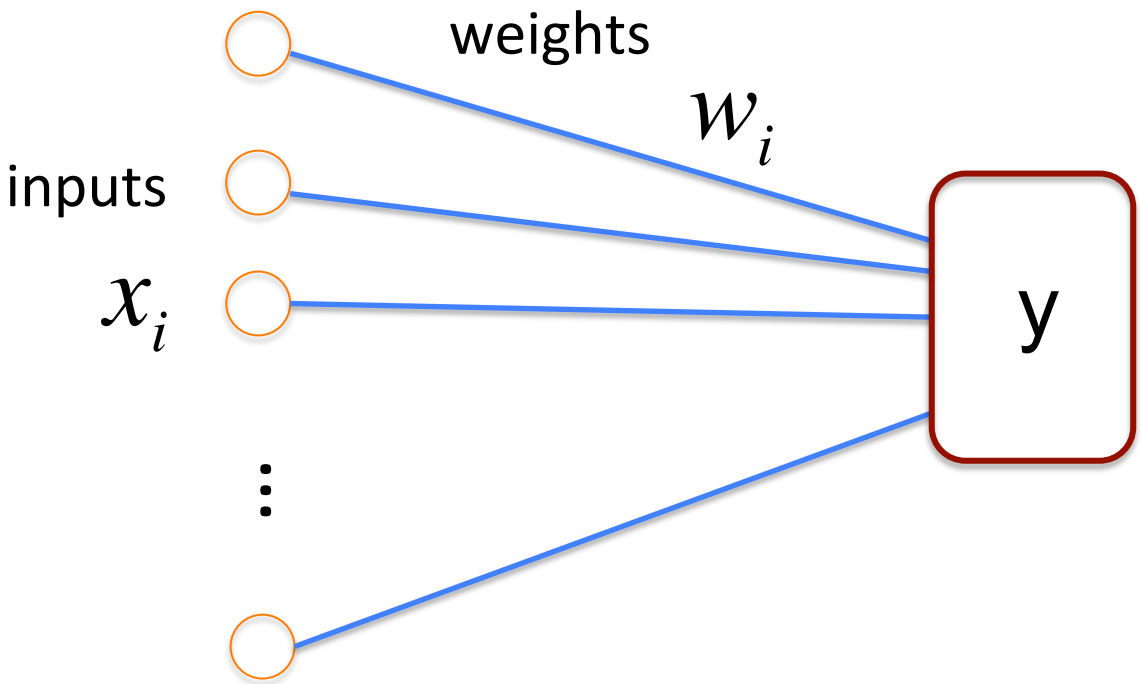
Protocol :

1. Estimation of synaptic efficacy
2. Repeated stimulations at fixed delay (<0 or >0)
3. New estimation of synaptic efficacy



Lasting effect over 10 msec to few minutes

Neural implementation of PCA



output

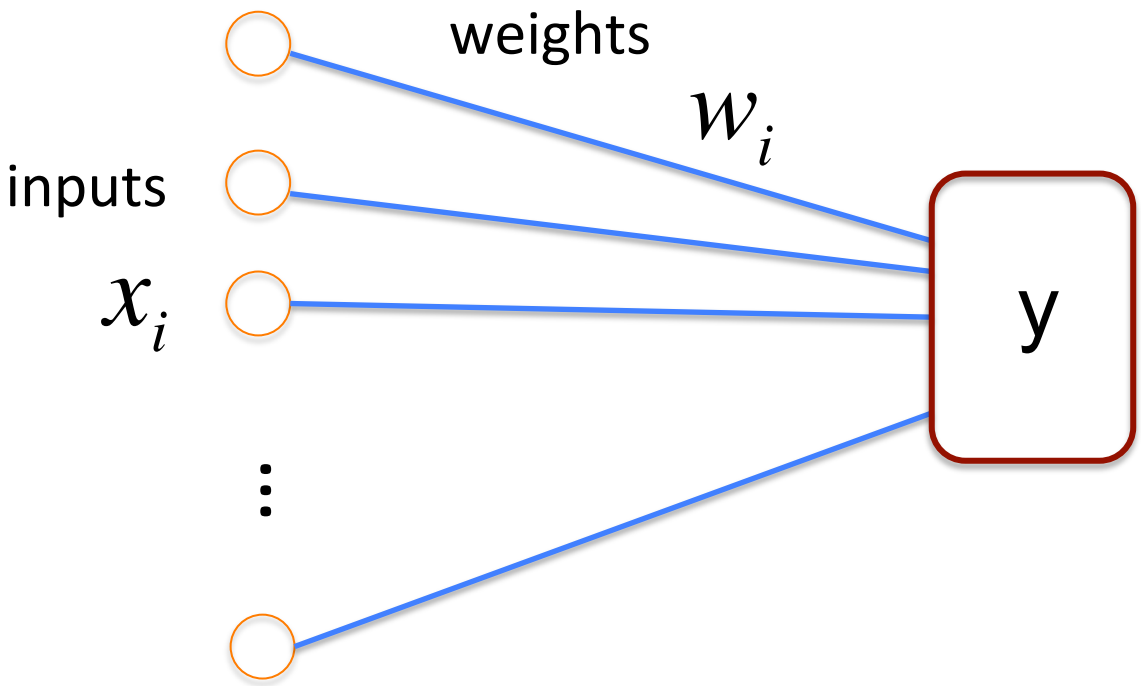
$$y = \vec{w} . \vec{x}$$

Sequential updating of weights (as more and more input vectors are presented):

$$\vec{w}(t+1) = \vec{w}(t) + \eta \; y(t) \left(\vec{x}(t) - y(t) \; \vec{w}(t) \right)$$

Learning rate — Hebbian term — Leak term

Exercise: Anti-hebbian learning



output

$$y = \vec{w} . \vec{x}$$

Sequential updating of weights (as more and more input vectors are presented):

$$\vec{w}(t+1) = \vec{w}(t) - \eta \; y(t) \; \vec{x}(t)$$

?

Non-linear Hebbian learning & sparse features

PCA = maximization of $\left\langle \left(\vec{w} \cdot \vec{x} \right)^2 \right\rangle_{\vec{x}}$

Non-quadratic extensions: $\left\langle V\left(\vec{w} \cdot \vec{x} \right) \right\rangle_{\vec{x}}, \quad V(y) \neq y^2$

e.g. $V(y) = y^4$ to favor large projections

Easy to see that Oja's rule becomes:

$$\vec{w}(t+1) = \vec{w}(t) + \eta \; V'\left(y(t) \right) \left(\vec{x}(t) - y(t) \; \vec{w}(t) \right)$$

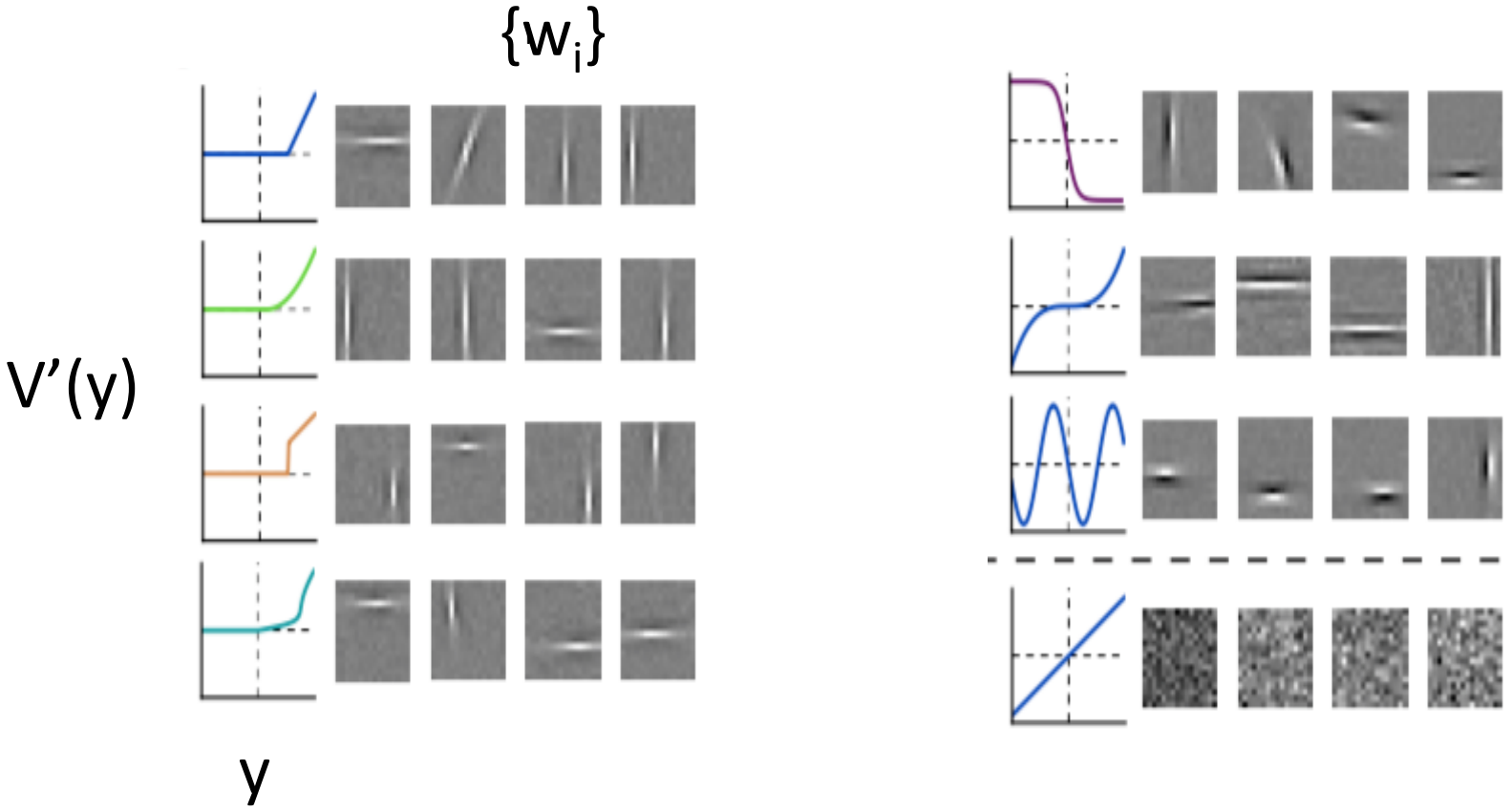
Examples of non-linear Hebbian learning of sparse features

- ~10^6 natural images
- Patches of 16x16 pixels
- Withening procedure: no information in covariance matrix

$$\vec{x} \rightarrow C^{-1/2} \cdot \vec{x}$$

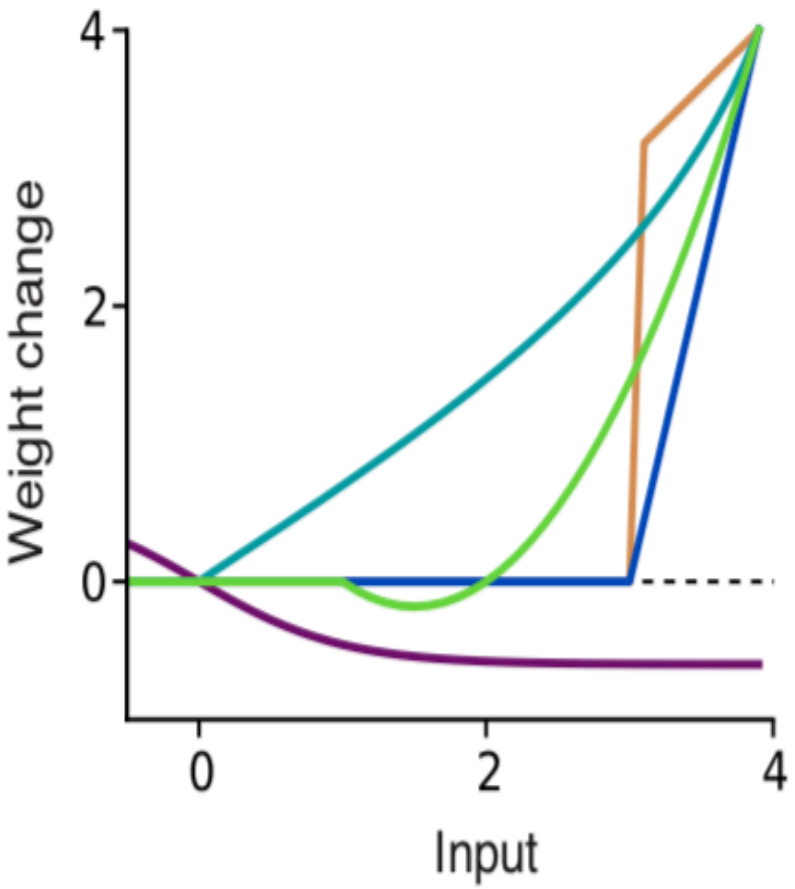
Brito, Gertsner, 2016

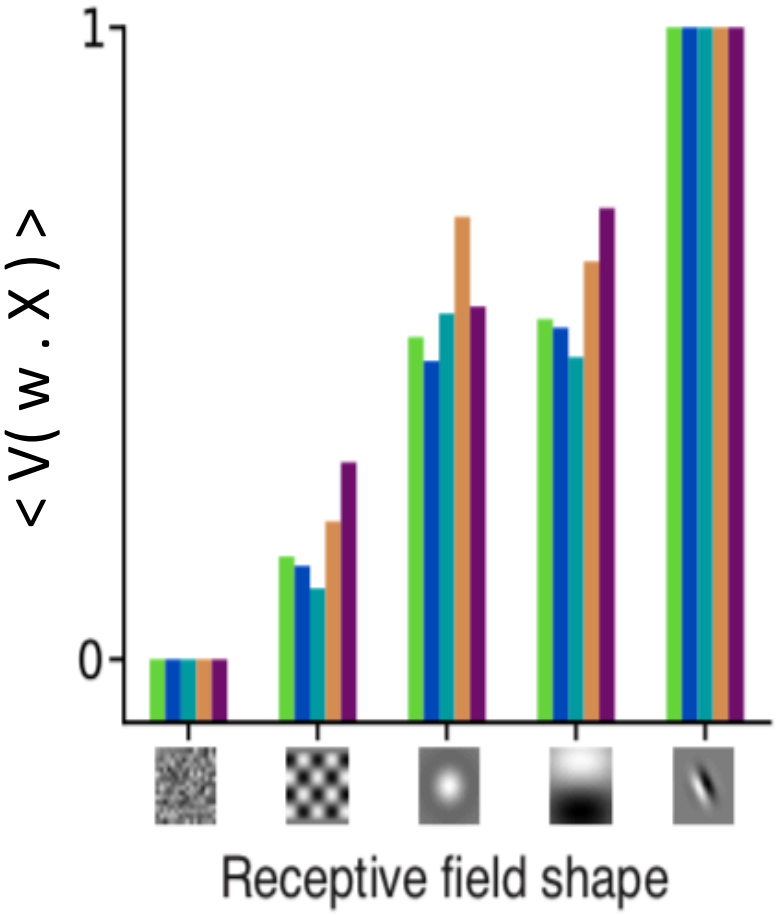
Examples of non-linear Hebbian learning of sparse features



Brito, Gerstner (2016)

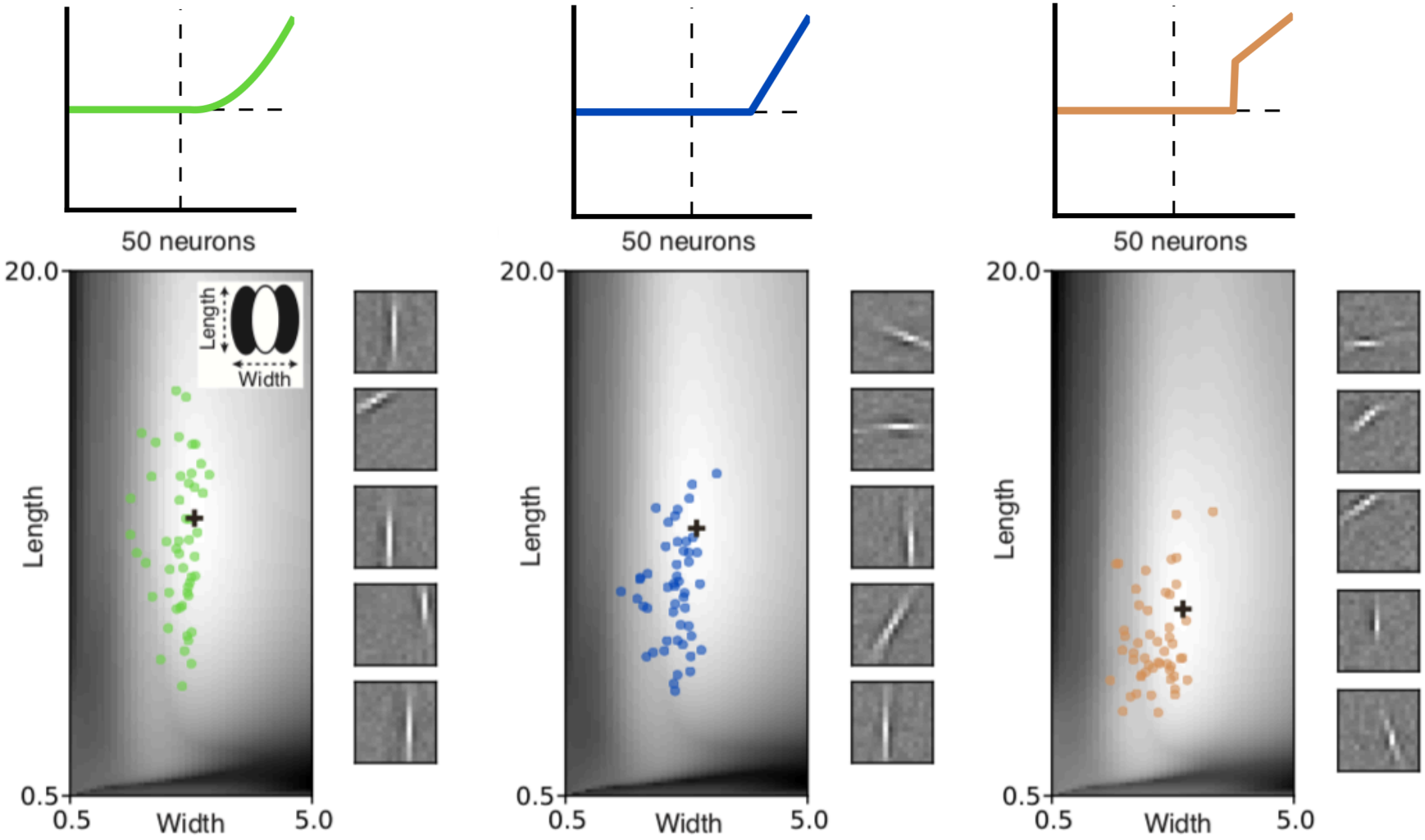
Examples of non-linear Hebbian learning of sparse features





Brito, Gerstner (2016)

Examples of non-linear Hebbian learning of sparse features



Gray level increasing with <V>

Brito, Gerstner (2016)

Restricted Boltzmann Machines

ns. The
ed SHG
in Fig.
we find
ne noise
closely
power
mission
gle with
ns from
Rs (see
etuning
elength
ne SHG
For ex-
vertical
a small
citation
ncident
nificant
ain po-

Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such "autoencoder" networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which finds the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer "encoder" network

Restricted Boltzmann Machines

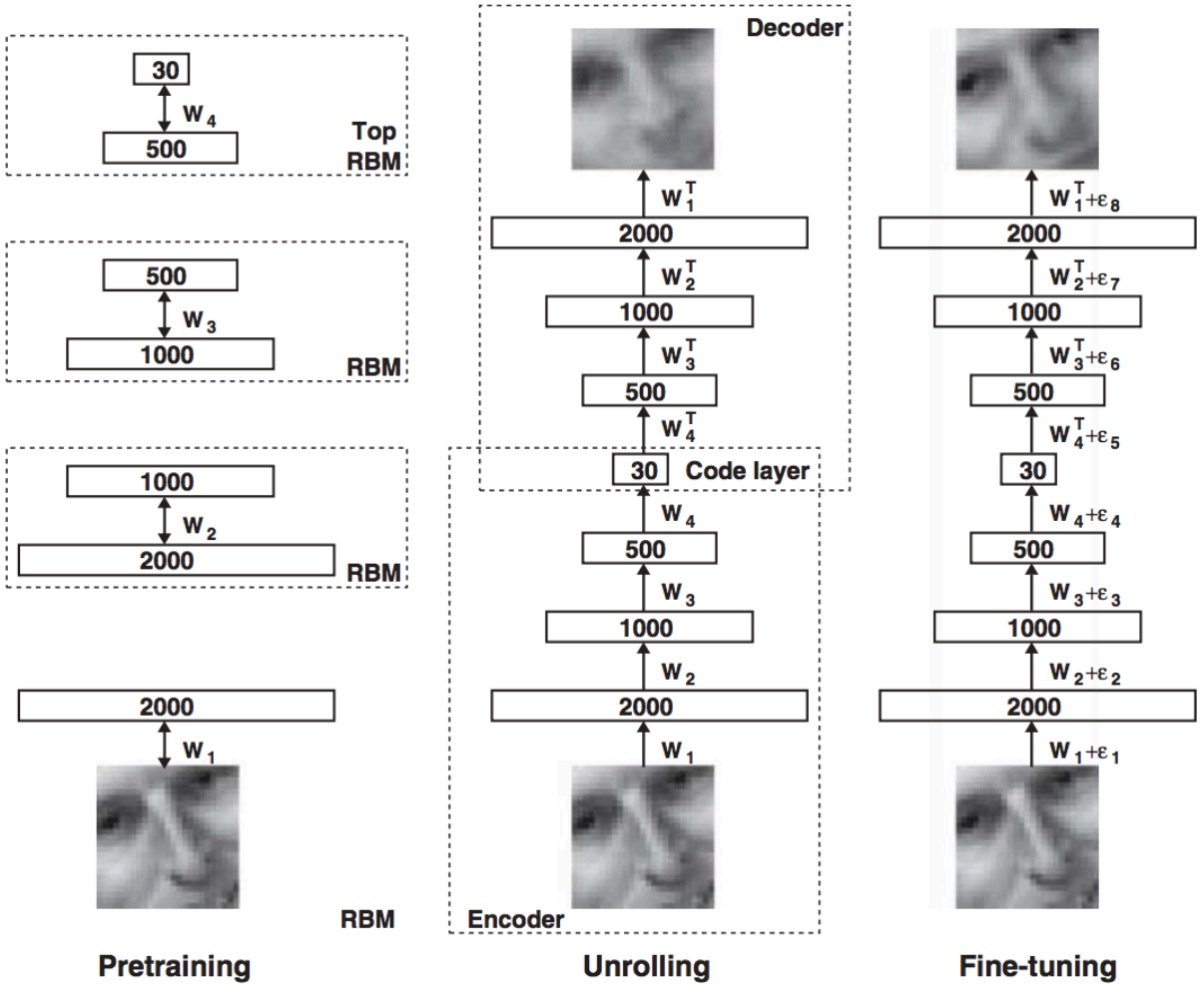


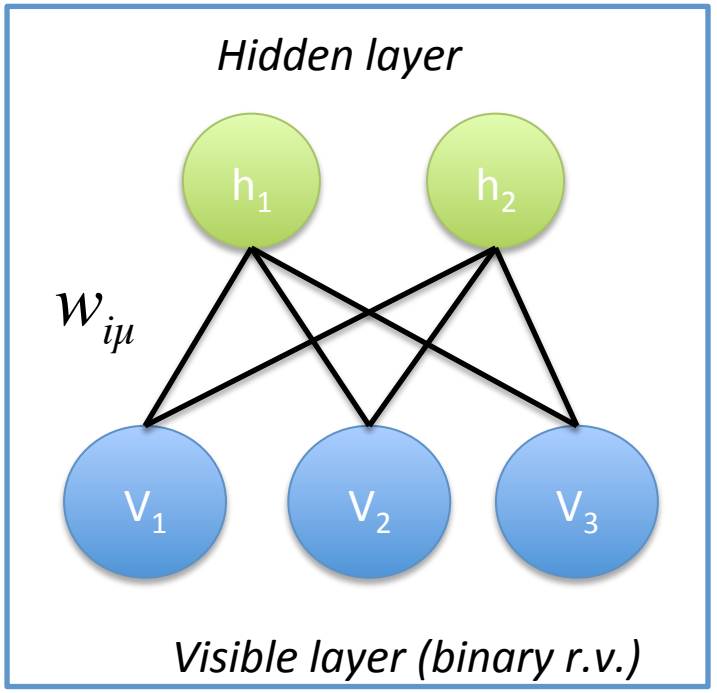
Fig. 1. Pretraining consists of learning a stack of restricted Boltzmann machines (RBMs), each having only one layer of feature detectors. The learned feature activations of one RBM are used as the "data" for training the next RBM in the stack. After the pretraining, the RBMs are "unrolled" to create a deep autoencoder, which is then fine-tuned using backpropagation of error derivatives.

Restricted Boltzmann Machines

- **Graphical model** constituted by two sets of random variables that are coupled together.

$$P(v,h) = \frac{1}{Z}\exp\left[-E(v,h)\right]$$

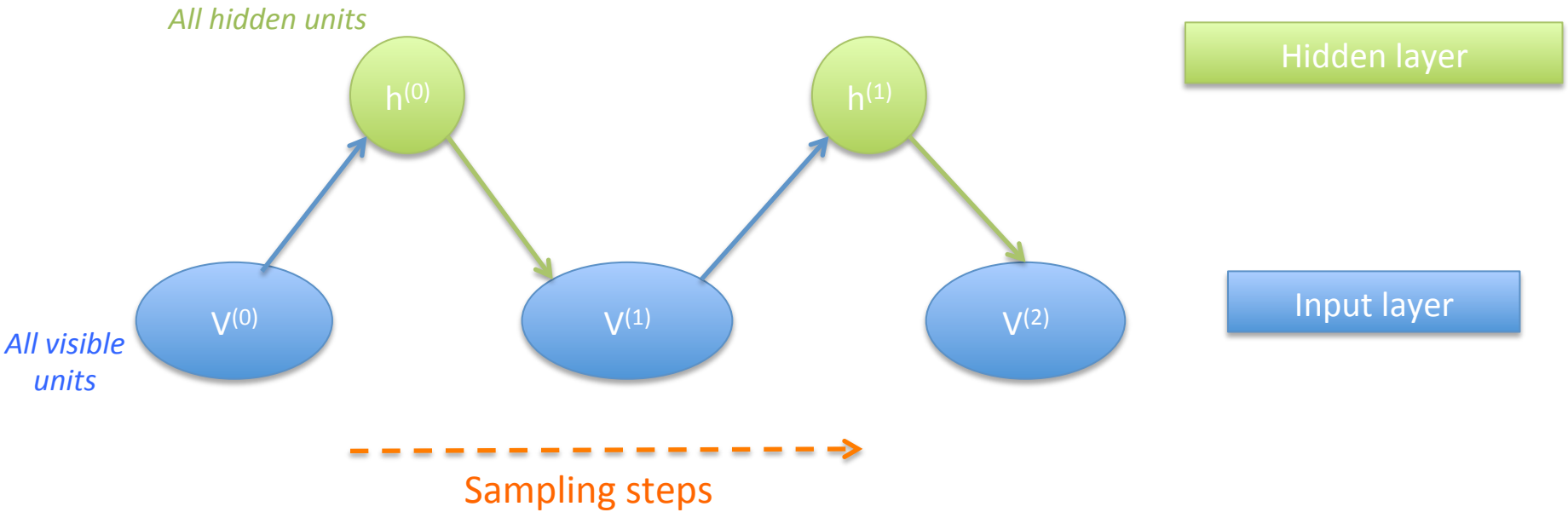
$$E(v,h) = -\sum_i g_i v_i + \sum_\mu U_\mu(h_\mu) - \sum_{i,\mu} w_{i\mu} v_i h_\mu$$



Smolensky 1986

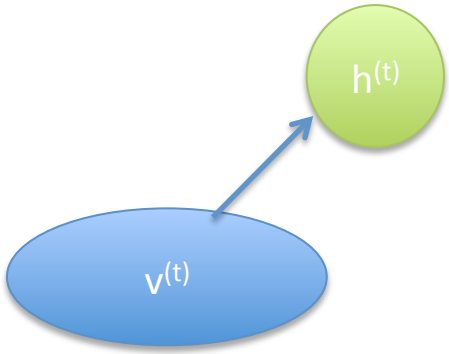
Restricted Boltzmann Machines: Sampling

- Compute Hidden units Inputs $I_\mu^H = \sum_i w_{i\mu} v_i$
- Sample each hidden unit independently $P(h_\mu \left| I_\mu^H \right.) \propto \exp\left[-U_\mu(h_\mu) + h_\mu I_\mu^H \right]$
- Compute the visible layer inputs $I_i^V = \sum_\mu w_{i\mu} h_\mu$
- Sample each visible unit independently $P(v_i \left| I_i^V \right.) \propto \exp\left[(g_i + I_i^V) v_i \right]$



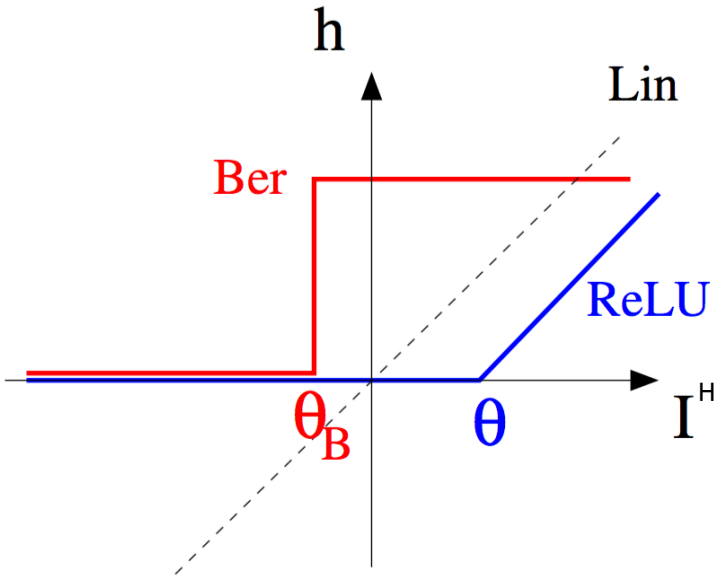
Hidden-Unit Potentials

- Compute hidden units inputs $I_\mu^H = \sum_i w_{i\mu} v_i$
- Sample hidden units $P(h_\mu \left| I_\mu^H \right.) \propto \exp\left[-U_\mu(h_\mu) + h_\mu I_\mu^H \right]$



Most likely value of h given input I^H ?

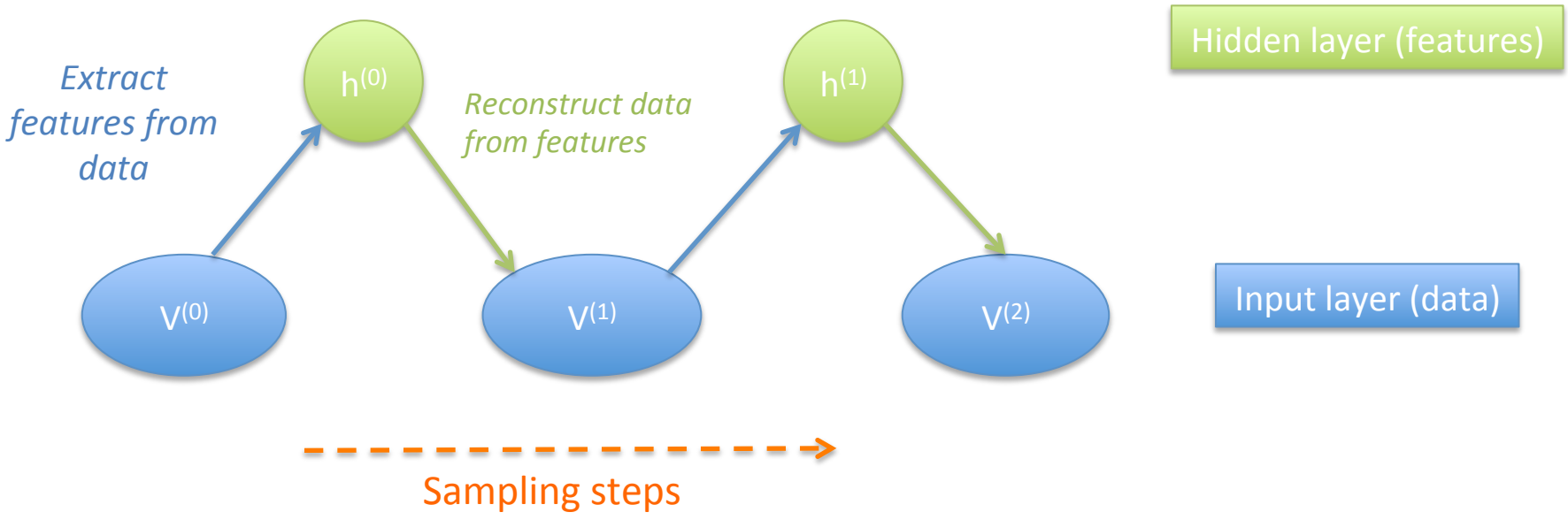
$$\frac{d}{dh} U(h) = I^H \Rightarrow h = \Phi\left(I^H\right)$$



Bernoulli or ReLU empirically known to give better results than linear hidden units …

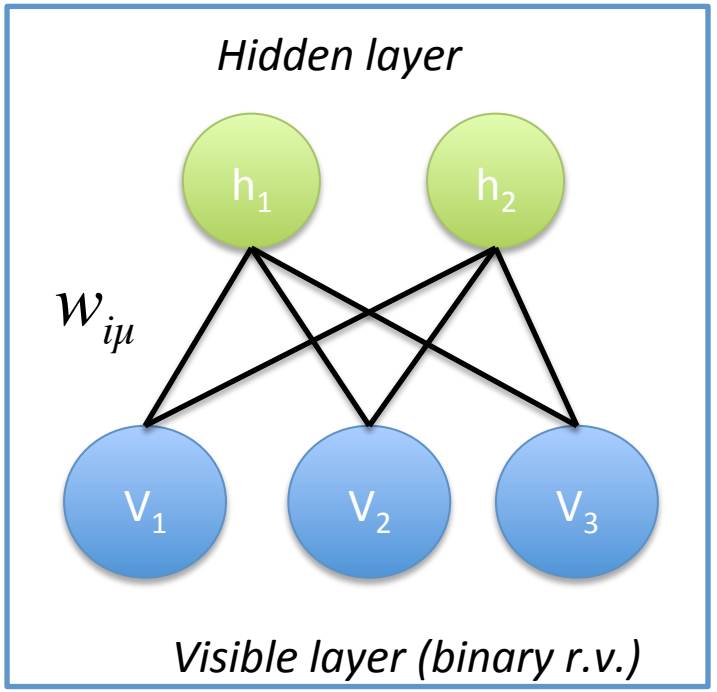
Restricted Boltzmann Machines: Sampling

- Compute Hidden units Inputs $I_\mu^H = \sum_i w_{i\mu} v_i$
- Sample each hidden unit independently $P(h_\mu \left| I_\mu^H \right.) \propto \exp\left[-U_\mu(h_\mu) + h_\mu I_\mu^H \right]$
- Compute the visible layer inputs $I_i^V = \sum_\mu w_{i\mu} h_\mu$
- Sample each visible unit independently $P(v_i \left| I_i^V \right.) \propto \exp\left[(g_i + I_i^V) v_i \right]$



Restricted Boltzmann Machines: Learning

- **Graphical model** constituted by two sets of random variables that are coupled together.



$$P(v,h) = \frac{1}{Z}\exp\left[-E(v,h)\right]$$

$$E(v,h) = -\sum_i g_i v_i + \sum_\mu U_\mu(h_\mu) - \sum_{i,\mu} w_{i\mu} v_i h_\mu$$

- RBM learns a **probability distribution** over the **visible layer**.

$$P(v) = \int \prod_\mu dh_\mu P\left(v,\{h_\mu\}\right) \equiv \frac{1}{Z_{eff}}\exp\left[-E_{eff}(v)\right]$$

- We also have
  - $P(h \mid v)$ extract (distribution of) representations from data
  - $P(v \mid h)$ generate (distribution of) data given a representation

RBM are generative models, trained through unsupervised learning

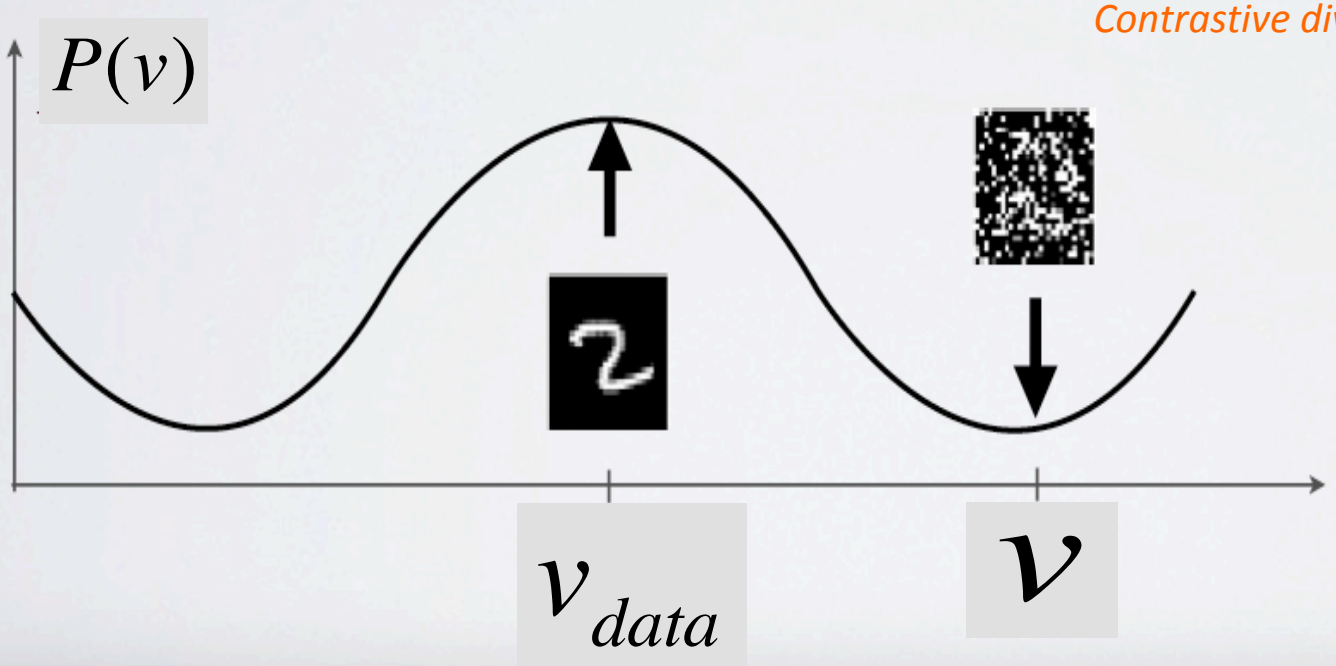
Training algorithm for RBM

Data set: $V = \left\{ v_i^b, i = 1 \ldots N, b = 1 \ldots B \right\}$

Want to maximize log-likelihood $\sum_b \log P\left(v^b \middle| \{w_{i\mu}\}, \{g_i\} \right)$

parameters Θ

Contrastive divergence (CD), Hinton 2002



Training algorithm for RBM

Data set: $V = \left\{ v_i^b, i = 1 \ldots N, b = 1 \ldots B \right\}$

Want to maximize log-likelihood $\sum_b \log P\left(v^b \middle| \{w_{i\mu}\}, \{g_i\} \right)$

parameters Θ

Stochastic gradient ascent:

$$\frac{\partial \log P\left(v^b \middle| \Theta\right)}{\partial w_{i\mu}} \propto \left\langle \frac{\partial E\left(v, h \middle| \Theta\right)}{\partial w_{i\mu}} \right\rangle_{v,h} - \left\langle \frac{\partial E\left(v^b, h \middle| \Theta\right)}{\partial w_{i\mu}} \right\rangle_{h}$$

$$= \quad \left\langle h_\mu v_j \right\rangle_{RBM} \quad - \quad \left[\left\langle h_\mu (v_j) \right\rangle_{RBM} v_j \right]_{data}$$

Hard to compute

Requires MCMC sampling

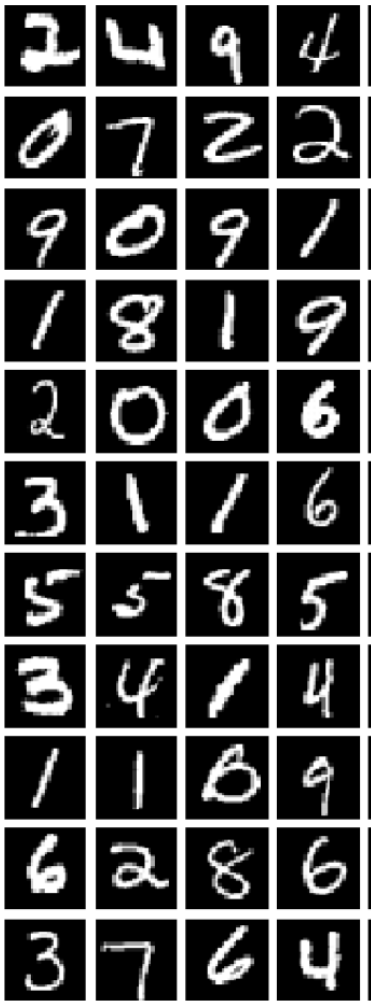
Easier to compute

Computed directly from data

Example : Unsupervised learning of MNIST synthetic digits

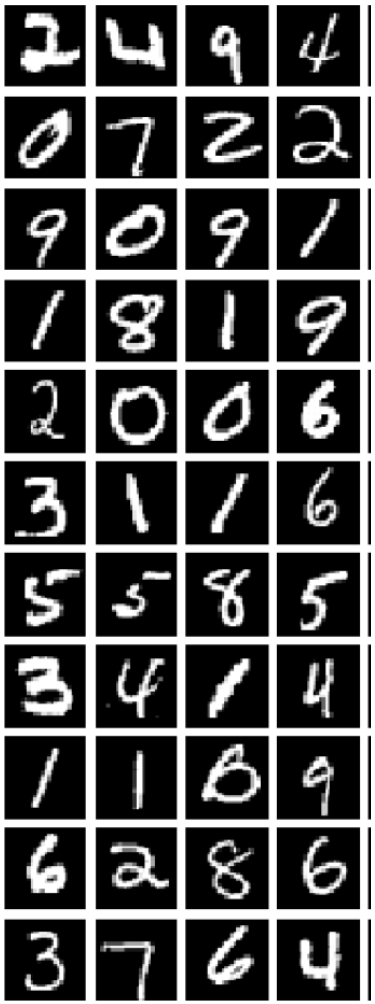
(with J. Tubiana)

60,000 images of digits with 28x28 pixels



Example : Unsupervised learning of MNIST synthetic digits

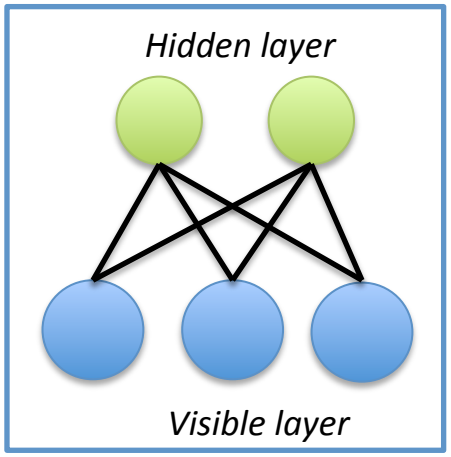
60,000 images of digits with 28x28 pixels



ReLU RBM
M = 400 hidden units

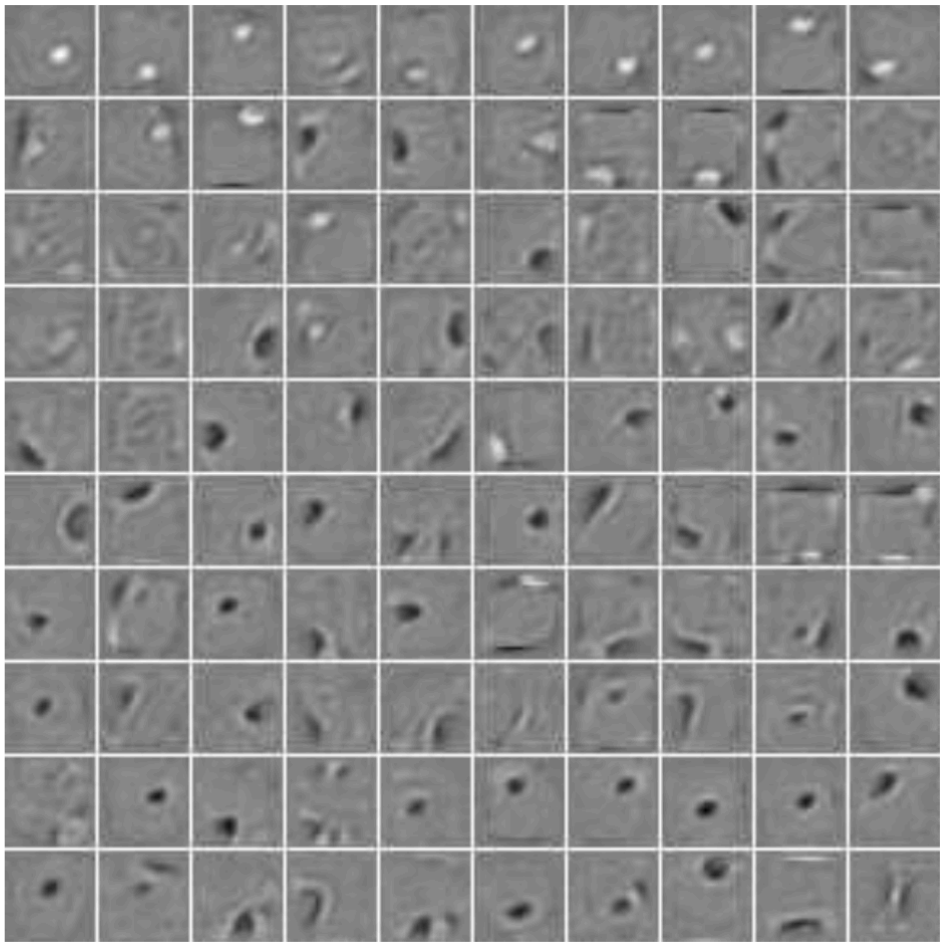
10 independent MCMC simulations with different initial visible configurations

Learning algorithm : PCD, PT (Tieleman 2008, Desjardins 2010)



Learning regimes in RBM

16 hidden units



100 hidden units

Fischer & Igel. Training Restricted Boltzmann Machines: An Introduction, 2014.

Learning with RBM and receptive fields

(with M. Harsh)

- Features reflect the data distribution in a non-trivial way …
- What happens for particularly well controled data?

Goal: try to learn invariant distribution with simple (few hidden units) RBM

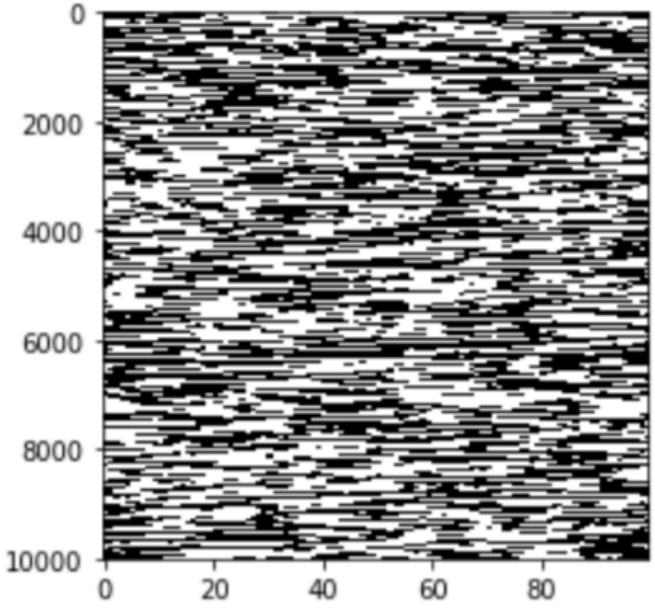
Here, 1D Ising model configurations over 100 sites:

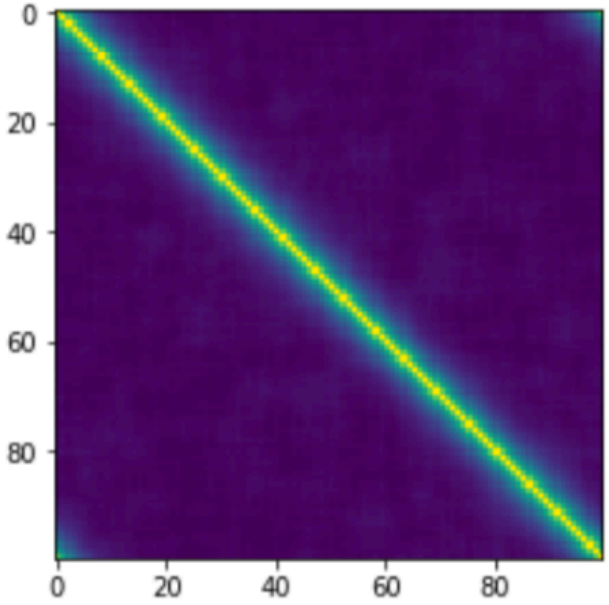
$$P\left(\sigma_1,\sigma_2,\ldots,\sigma_N\right)=\frac{1}{Z}\exp\left(\beta\sum_i\sigma_i\sigma_{i+1}\right)$$

$$\left\langle\sigma_i\right\rangle=0,\ \left\langle\sigma_i\sigma_{i+d}\right\rangle=(\tanh\beta)^d=\exp\left(-d/L(\beta)\right)$$

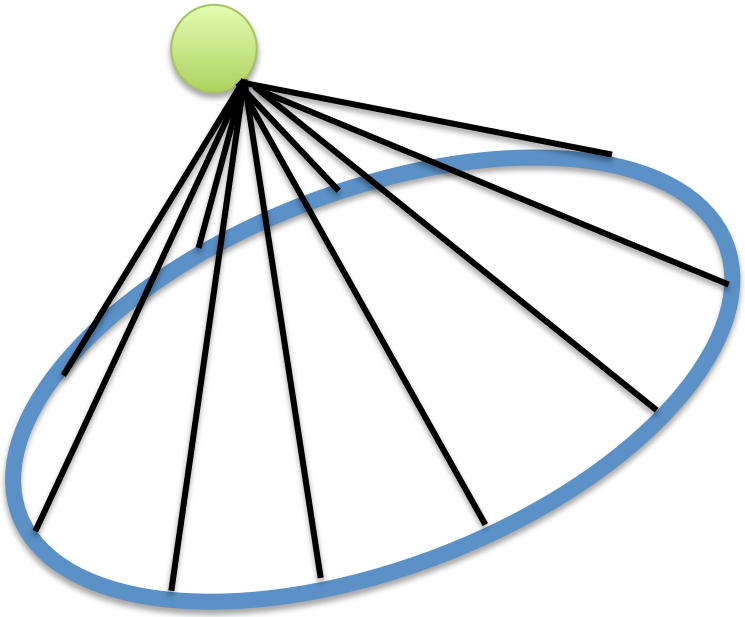
Learning with RBM and receptive fields

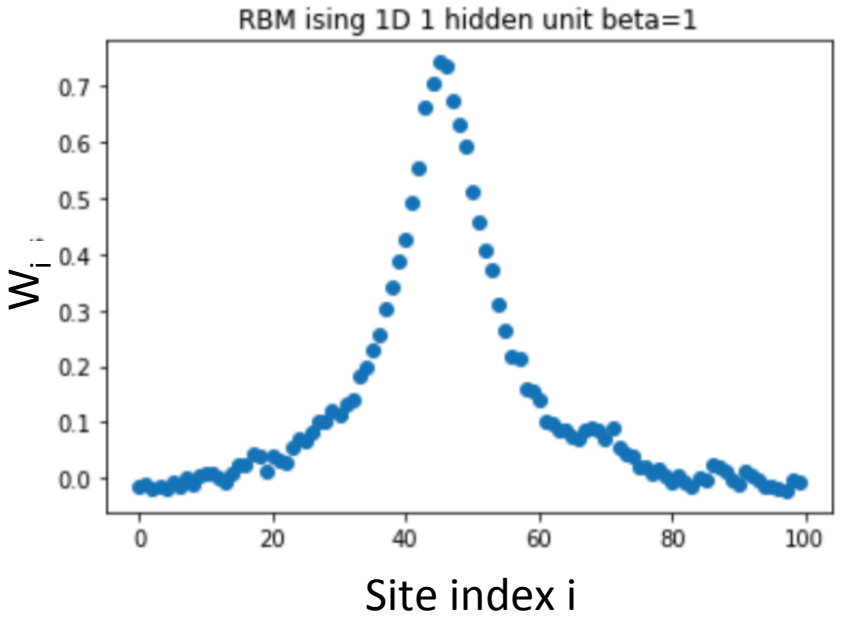
10,000 configurations over 100-site rings; β=1

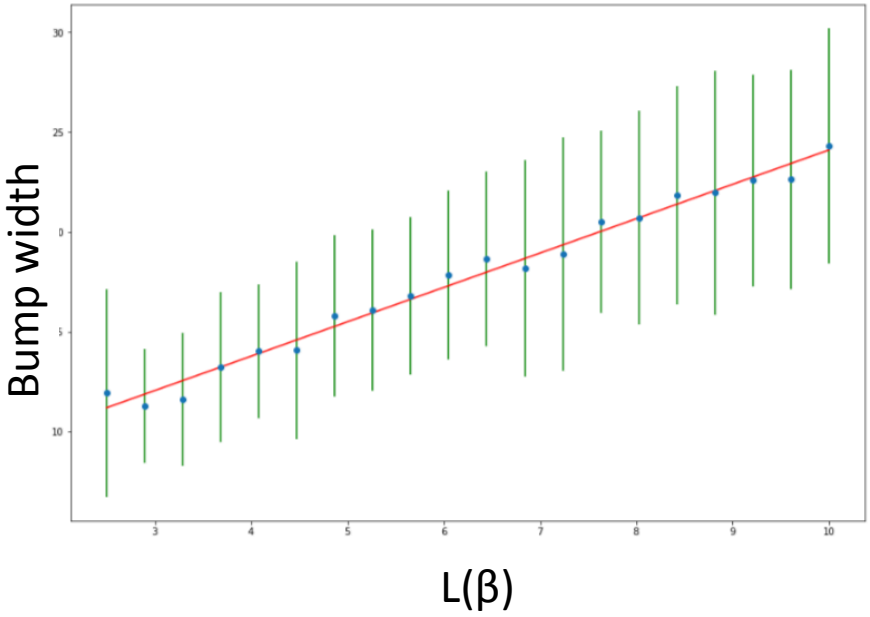




Learning with RBM: dynamical symmetry breaking







Learning with RBM: Mechanism for bump formation

$$\frac{dW_i}{dt} = \eta \frac{\partial L}{\partial W_i} = \eta\left(\langle h\ W_i\rangle_{RBM} - \langle h\ W_i\rangle_{data}\right)$$

$$= \eta\left(\beta W_{i+1} + \beta W_{i-1} + W_i^3 - W_i \sum_k W_k^2 + O(W^5, \beta W^3)\right)$$

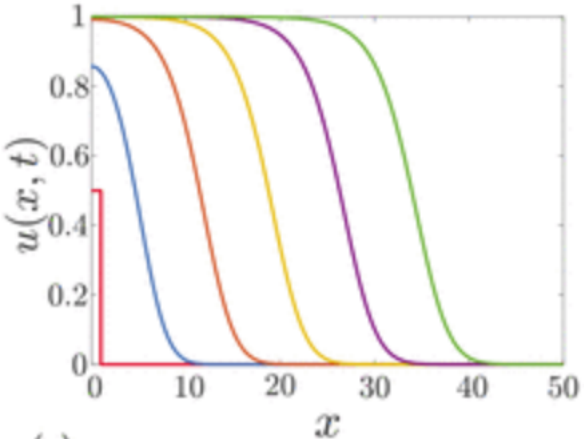
small β expansion

$$W \to W \cdot \beta^{1/2}$$
$$t \to t \cdot \beta\eta$$

$$\boxed{\frac{dW_i}{dt} = W_{i+1} + W_{i-1} - 2W_i + W_i\left(2 - \sum_k W_k^2\right) + W_i^3}$$

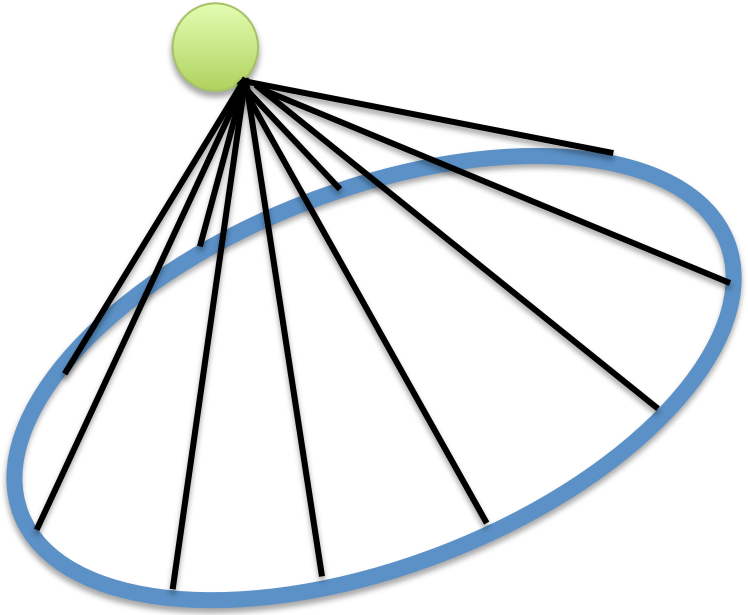
- **Discrete-like variant of Fisher-Kolmogorov-Petrovsky-Pistunov equation**

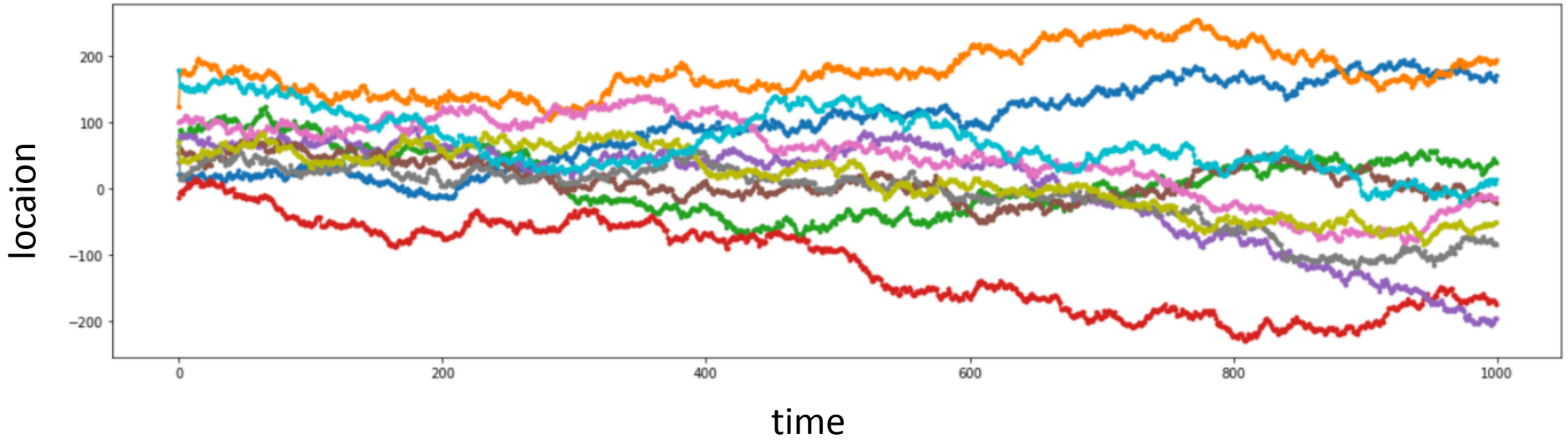
$$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = \frac{\alpha}{k} u(1 - u^q)$$



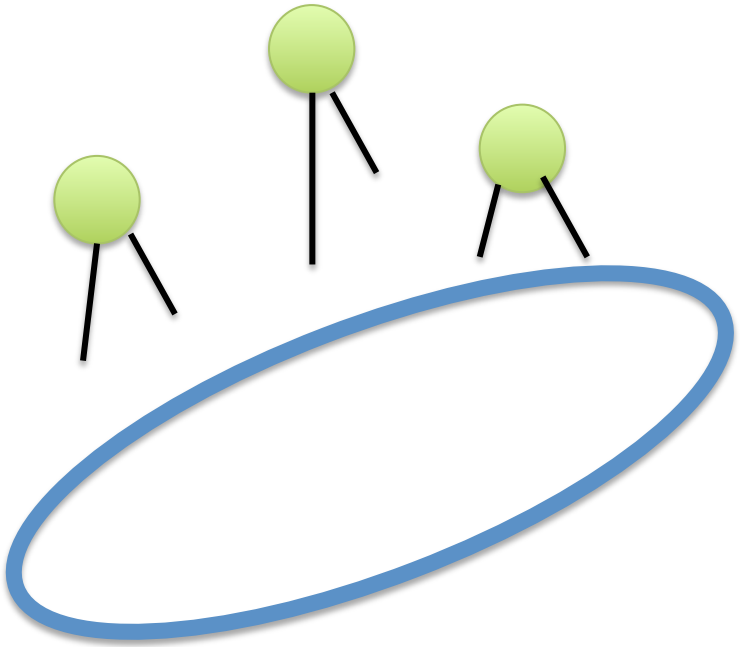
- **But growth stops due to long-range « inhibitory » term**

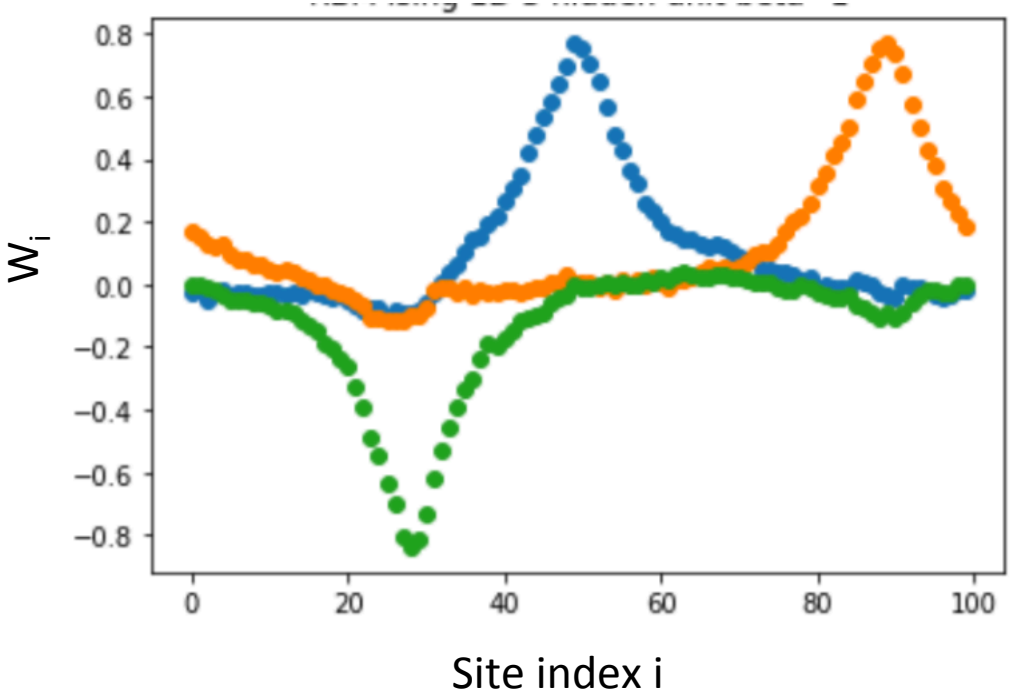
Learning with RBM: dynamical symmetry restoration

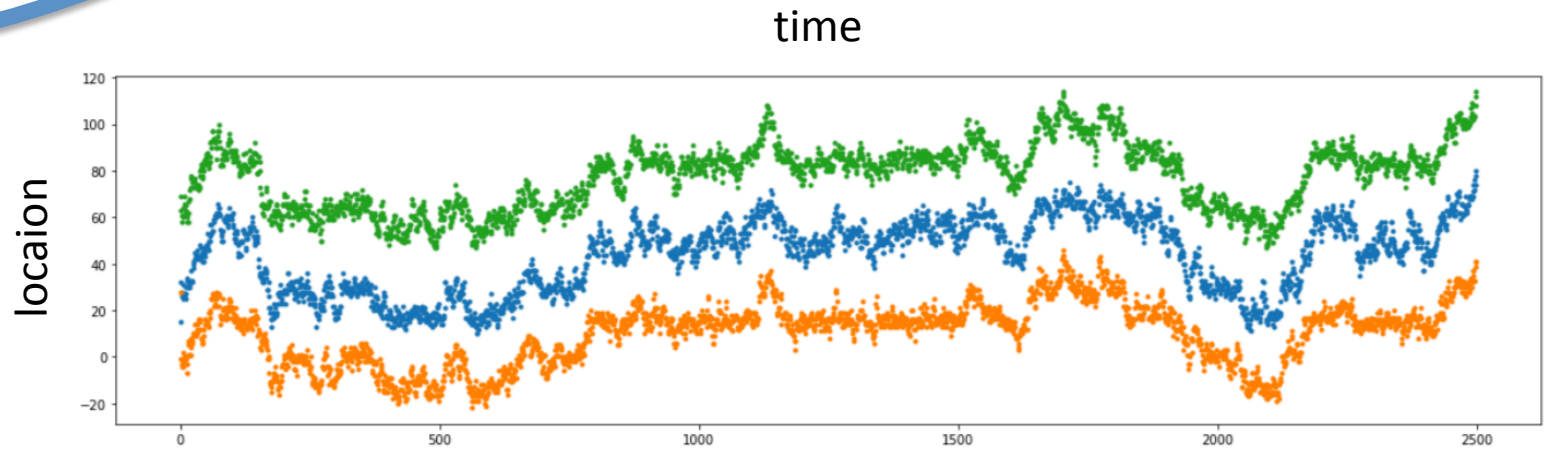




Learning with RBM: Multiple hidden units

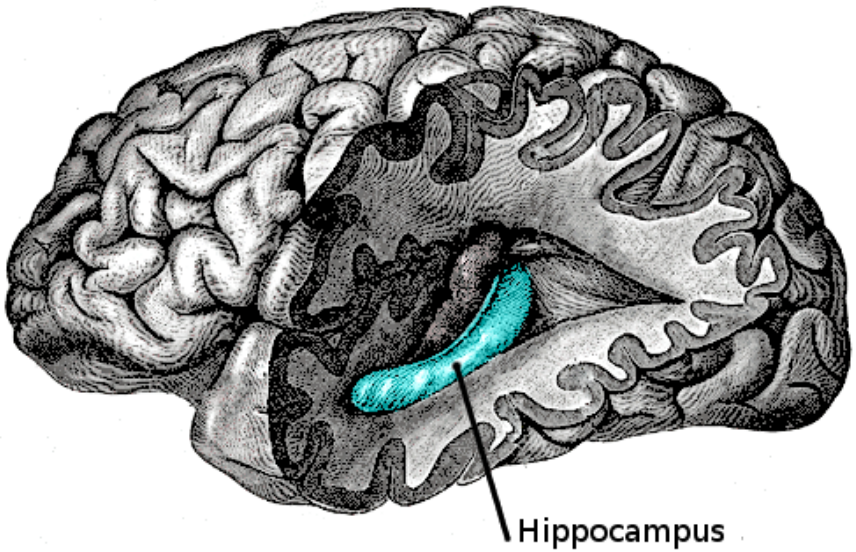


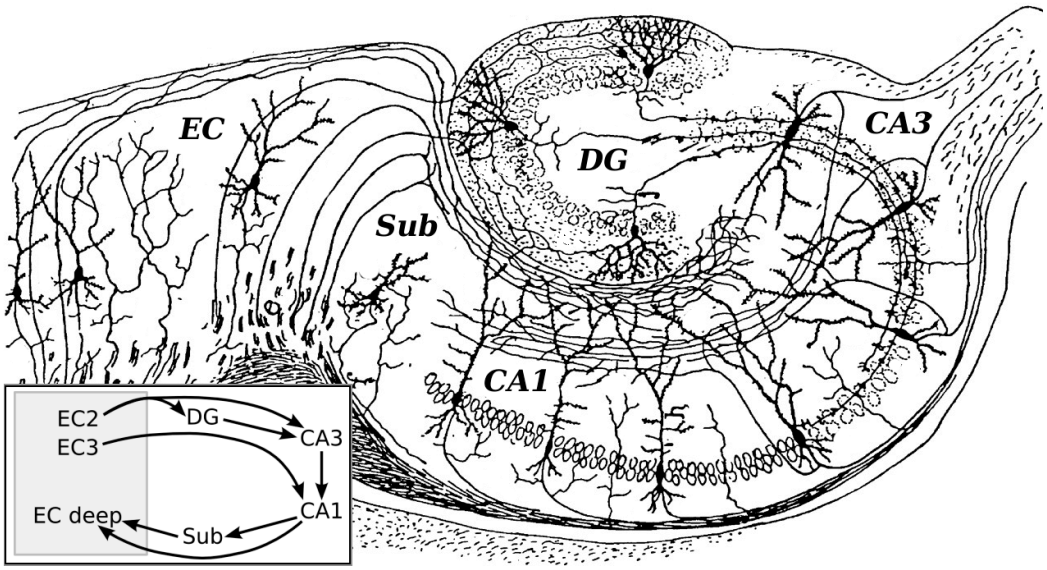




Locking of bumps (relative phases are maintained through time)

Representation of space in the brain





- necessary to form and retain new memories
- deeply intra-connected and connected to neighboring cortical regions, e.g. EC
- Hippocampus and EC fundamentally involved in the representation of space

O'Keefe, Dostrovsky (1971)

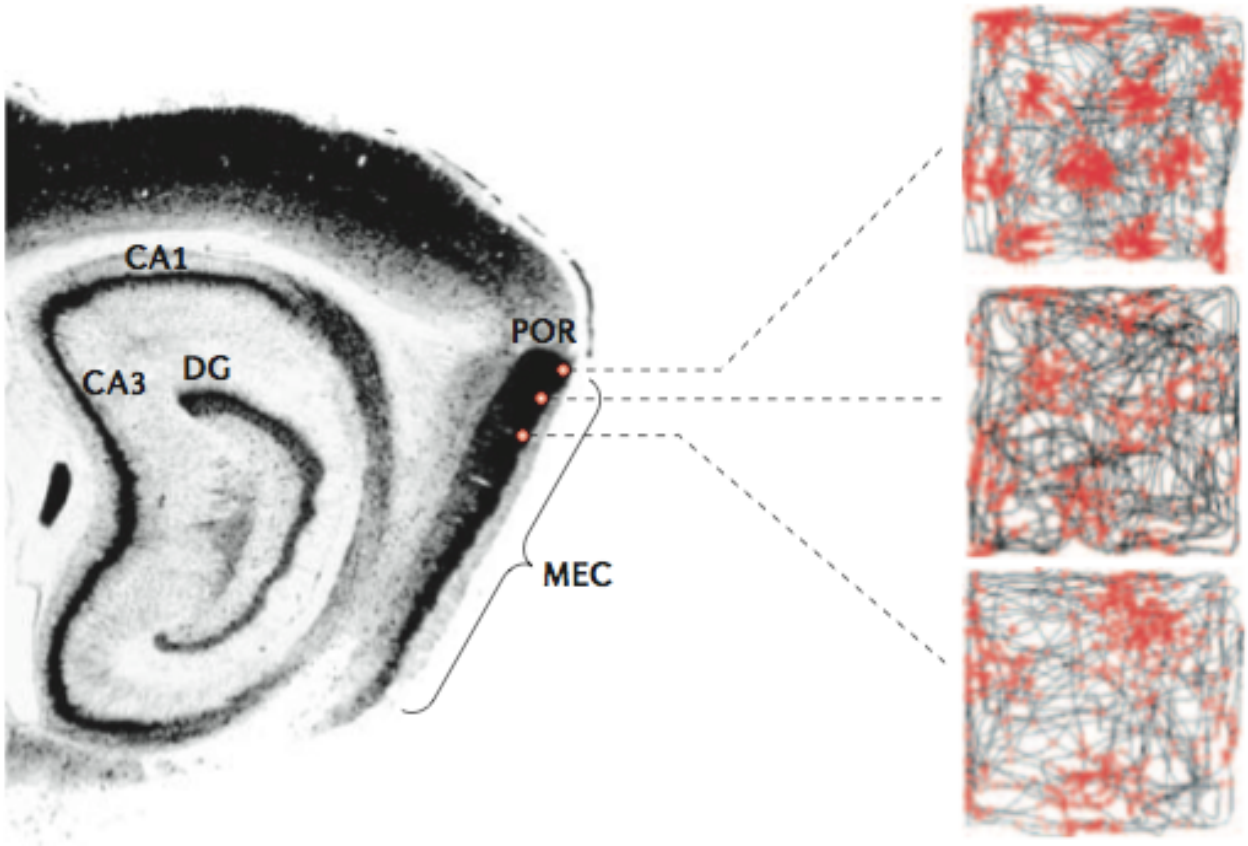
Prix Nobel 2014

Place cells

Movie

Grid cells

Hafting, Fyhn, Molden, Moser & Moser(2005); Nobel Prize 2014



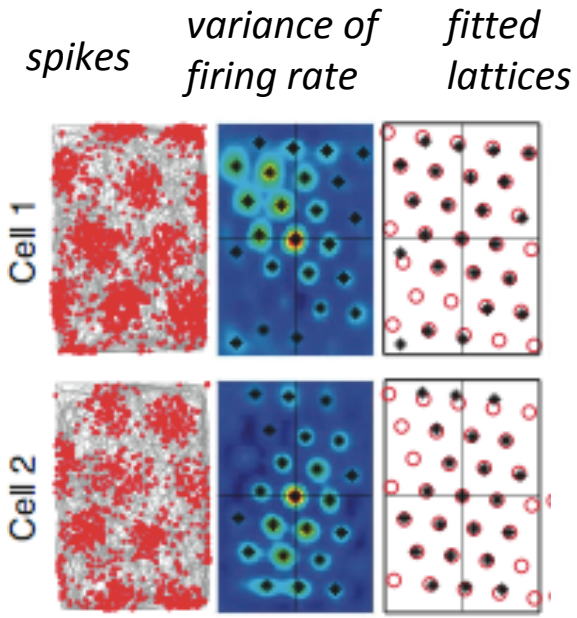
Trajectory of a rat through a square environment is shown in black. Red dots indicate locations at which a particular entorhinal grid cell fired.

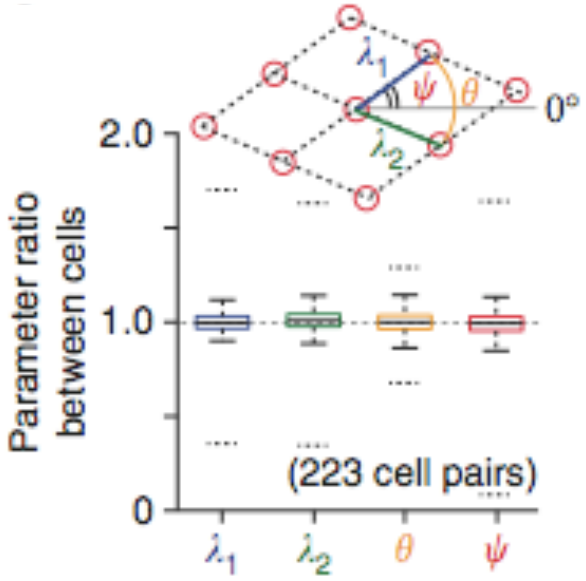
Grid cell properties:

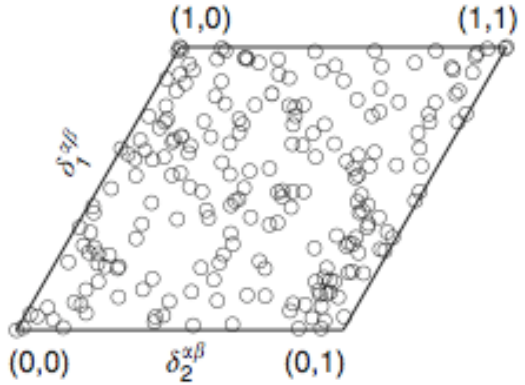
- fire on triangular lattice
- neighbouring cells differ by translation of their grids
- 'far away' cells also differ by grid rotation
- mesh sizes vary with recording depth in MEC
- geometric organization of grids (5 sizes, ratio 1.4)
- establish very fast in a new environment and stabilize over days
- found in rodents, monkeys, bats
- 2D continuous attractor models (with local inhibition)

Grid cells and phase locking

Yoon et al., Nature Neuroscience (2013)

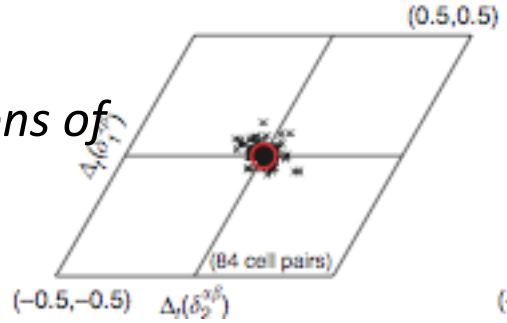


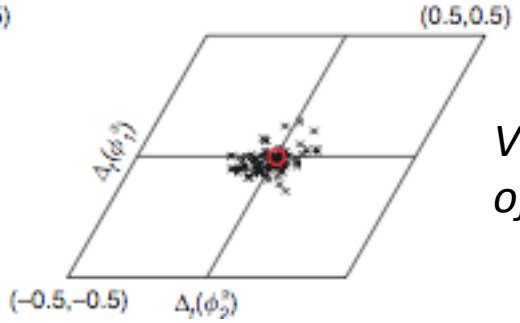




- neighbouring cells define identical 2D lattices, up to a 2D translation
- relative values of translation parameters are more stable over long periods of time than parameters themselves

Variations of relative phases





Variations of phases

- stability against moderate pertubations e.g. environment reshaping …